5. Iot開発を容易にする技術

第４項で紹介されているようにIoTのサービス・製品のための技術は様々で揃ってきました。しかしながら、実際にいざ始めようとするとどこから始めたらよいのか戸惑われることも多いかと考えます。
そこでこの章ではIoT開発を始めるのに役立つ、容易に始めることのできる開発方法をご紹介します。

ここでご紹介する技術の一部は福岡県内の企業、団体が提供するものです。

- **IoTに適した開発言語mrubyとIoTフレームワークPlato（プラトン）**
  SCSK九州（株）、九州工業大学、㈱ネットワーク応用通信研究所（NACL）まつもとゆきひろ様
  有明高等専門学校
- **仮想IoTデバイス　MockMock**
  Fusic㈱
- **機械学習ビッグデータ処理　MAGELLAN BLOCKS**
  ㈱グルーブノーツ

1）開発言語 mruby と IoT フレームワーク Plato（プラトン）

IoT開発はセンサーなどのデバイスを用い、それらをハードウェアに組み込みそこからのデータをサーバに送信するという手法が一般的です。これらの開発には組み込み開発でよく使われるC言語などが用いられます。C言語は少ないメモリなどのリソースで動作する組み込み向けの言語で幅広く用いられています。ただしIT企業では一般的ではないかもしれません。IoT開発を始めるのにまずC言語の習得から始めるというのではなかなか大変です。そこでmrubyをご紹介します。mrubyはRubyというWeb系言語を組み込み開発でも使えるように小さいリソース
でも動作するように提要した言語です。

mrubyは経済産業省のサポートを得て「平成22年度　地域イノベーション創出研究開発事業」「軽量Rubyを用いた組込みプラットフォームの研究・開発」において2年の歳月をかけて開発されました。

Rubyそのものは1995年にまつもとゆきひろ氏が発表したオブジェクト指向言語です。まつもと氏はMatz（マツ）という愛称で世界中で親しまれています。同様にRubyも世界で最も人気のあるプログラミング言語トップ10に選ばれるほど人気の高い言語です。

Rubyの特長は書きやすく読みやすく、習得しやすいところにあります。書きやすい、すなわちコード量が少ないため、生産性が高く（開発工数はJavaの1/5、コード数は1/2と言われていています）、さらにバグなどの不具合の軽減も期待できます。

このRubyの書きやすさそのままに組み込み開発で使えるようにしたのがmrubyです。mrubyはミニマムRubyの略です。日本語名で軽量Rubyと呼ばれることもあります。mrubyは以下のような特徴を持っています。

① mrubyの特徴

◆ ISO, JIS規格のRubyに準ずる言語仕様

- 本家Rubyと同様に使えます

◆コンパクトな処理系「mrubyVM」

・mrubyコンパイラが出力するバイトコードを実行

・VMさえ動作すればどんな環境でも動作可能 Windows, Mac, Linux, ITRON, Android, iOS ...

・Non-OSでも動作

◆C/C++言語との高い親和性 – 組込みシステムの資産が再利用可能 – アプリケーションにmrubyを組込み可能

◆オープンソース　商用利用しやすいMITライセンスでソース公開の必要がありません

＜高い生産性＞

例）サーバに文字列を10回送信するコードの比較

一見してmrubyの方がC言語に比べてコード数が少ないことが見ていただけると思います。

この例ではTCP/IPで通信するためにTCPSocketというクラスを使用しています。RubyでTCPSocketを使用するためにはrequire 'socket' を呼び出す必要がありますが、mrubyでは事前にライブラリを追加することでrequireなしで使用することができます。また、rescueはRubyと同様に例外処理であり、簡潔に記述できます。

C言語
(35行)

```c
#include <stdio.h>
#include <sys/socket.h>
#include <arpa/inet.h>
#include <string.h>

int main(void)
{
  int sock;
  int i;
  struct sockaddr_in svaddr;
  const char msg[] = "Hello!!";

  if ((sock = socket(PF_INET, SOCK_STREAM, IPPROTO_TCP)) < 0) {
    puts("socket() failed.");
    return 1;
  }
  memset(&svaddr, 0, sizeof(svaddr));
  svaddr.sin_family = AF_INET;
  svaddr.sin_addr.s_addr = inet_addr("192.168.1.1");
  svaddr.sin_port = htons(30000);
  if (connect(sock, (struct sockaddr*)&svaddr,
      sizeof(svaddr)) < 0) {
    puts("connect() failed.");
    exit(2);
  }

  for (i=0; i<10; i++) {
    if (send(sock, msg, strlen(msg), 0) !=
        strlen(msg)) {
      puts("send() failed.");
      exit(3);
    }
  }
close(sock);
  return 0;
}
```

・コードが長くなりがち
・処理が複雑になりがち
・危険なポインタ操作
・メンテナンス性が低い

MRUBY
(9行)

```
begin
  sock = TCPSocket.open("192.168.1.1", 30000)
  10.times {
    sock.write("Hello!!")
  }
  sock.close
rescue => e
  p e
end
```

・短いコード
・簡潔な記述
・ポインタ操作なし
・メンテナンス性が高い

② mrubyを使ってみる　インストール

mrubyのインストールには以下のものが必要です。

・Cコンパイラ（gcc推奨）
・Bison（yacc上位互換のパーサジェネレータ）
・Ruby（本家Ruby）

Windowsユーザーならば、Cコンパイラは無償版のMicrosoft Visual Studio 2013 Expressが利用できます。
「Express2013 for windows desktop」をダウンロードしておきましょう。
https://www.visualstudio.com/ja-jp/products/visual-studio-express-vs.aspx

mrubyは現在安定板としてmruby　v1.2.0デバッガ対応版が公開中（2017年3月現在）で、

「mrubyコミュニティー」と http://mruby.org
「軽量Rubyフォーラム」、http://forum.mruby.org
それに「GitHub] https://github.com/mruby/mruby
からダウンロード可能です。

◆インストールの仕方

Gitツールをインストールしていれば

$ git clone　https://github.com/mruby/mruby.git

Zip形式でのインストラーも用意されていますのでそちらでもインストール可能です。

上記コマンドでmrubyディレクトリが作成されますのでmrubyディレクトリに移動して、makeを実行します。
$ cd mruby $ make

Visual Studioを利用するWindowsユーザーはVisual Studioツールから“開発者コマンドプロンプト for VS2013”を開き、下記を実行します。

>ruby minirake.
ビルドが完了すると下記の構成で作成されます。
bin/mrbc　mrubyコンパイラ
bin/mruby　mrubyインタプリタ
bin/mirb　Interactive mruby
bin/mrdb　mrubyデバッガ
build/host/lib/libmruby.a　mrubyライブラリ

詳しくはMonoistに掲載されているmruby概論をごらんください。使い方など分かりやすく説明されています。
http://monoist.atmarkit.co.jp/mn/series/2056/

2）IoT　フレームワーク　Plato（プラトン）

IoT開発にはセンサーなどのデバイス知識、ネットワーク、クラウドでのデータ蓄積、分析など様々なスキルが必要です。

このフレームワークはIoT開発の最初の部分、センサーからの値を取ってネットワークにあげるまでをシナリオに沿って答えていくことでmrubyのプラグラムコードを自動生成します。これによってIoT開発の取りかかりを素早く行え、製品試作をすぐに始めることが可能です。

Plato（プラトン）はNEDO助成事業「組込みシステムの高効率開発を可能とする開発 フレームワークの研究開発」にて開発されました。(~2017/2) (九州工業大学、有明工業高等専門学校、SCSK九州による産学連携プロジェクト)

①“Plato”で提供しているもの

・マイコンボードとシームレスに繋がる統合開発環境の提供
　オープンソースで提供
・組込み向けIoTセキュリティ機能の提供
　ライセンス形式で提供
・その他：オプションボード
　White-Tiger（右下写真）センサと通信モジュール搭載済み

オプションボード White-Tiger

◆ mrubyによるIoT製品開発の効率化

組込み向けデバイスクラスライブラリの提供
BLE、Wi-Fi、ZigBeeなど様々な通信方式にも対応
典型的なアプリケーションの雛形を自動生成 (Railsのように)
シミュレータによるPC上でのシミュレーション

◆ 使えるセンサ、NWモジュール、I/O

- 汎用IO　デジタル/アナログIO、UART、I2C、SPI
- センサ　温度、湿度、照度
- 通信デバイス
  - BLE
  - Wi-Fi
  - ZigBee
  - Ethernet

これらのセンサと通信デバイスはWhite-Tigerに乗っているものを使うと一段と容易に開発できます。

- ファイルシステム
- RTC

◆ Plato IDE （開発環境・エディター）

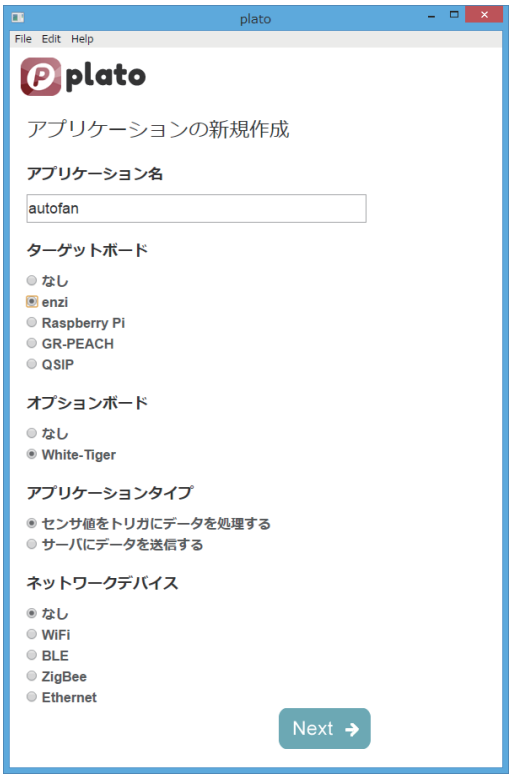
Visual Studio Codeベースの開発環境 (マルチ環境に対応)
アプリ構成、使用ライブラリをGUIで指定
アプリケーションのコンパイル
アプリケーションのシミュレーション実行
マイコンへのプログラム書き込み

② Platoのメニュー構成について

Platoのメニュー内容をご紹介します。

Platoのメニューはシナリオに沿って答えていくだけで簡単にアプリが自動生成できる”シンプル”メニューと自分でライブラリなどの追加が行える”カスタム”メニューから成り立っています。

シンプルメニューを起動します。



各項目の説明です。

・アプリケーション名

作成するアプリの名称

・ターゲットボード

作成したアプリを動作させるマイコンボードを選択します。

現在の対応ボードは

- enzi(Manycolors社製　mrubyがそのまま動作するボード、福岡）
- Rasoberry Pi
- GR-PEACH　ルネサス製ボード
- QSIP　富士通九州ネットワークネクノロジーズ

・オプションボード

White-Tigerはセンサなどが搭載されたボードです。

・アプリケーションタイプ

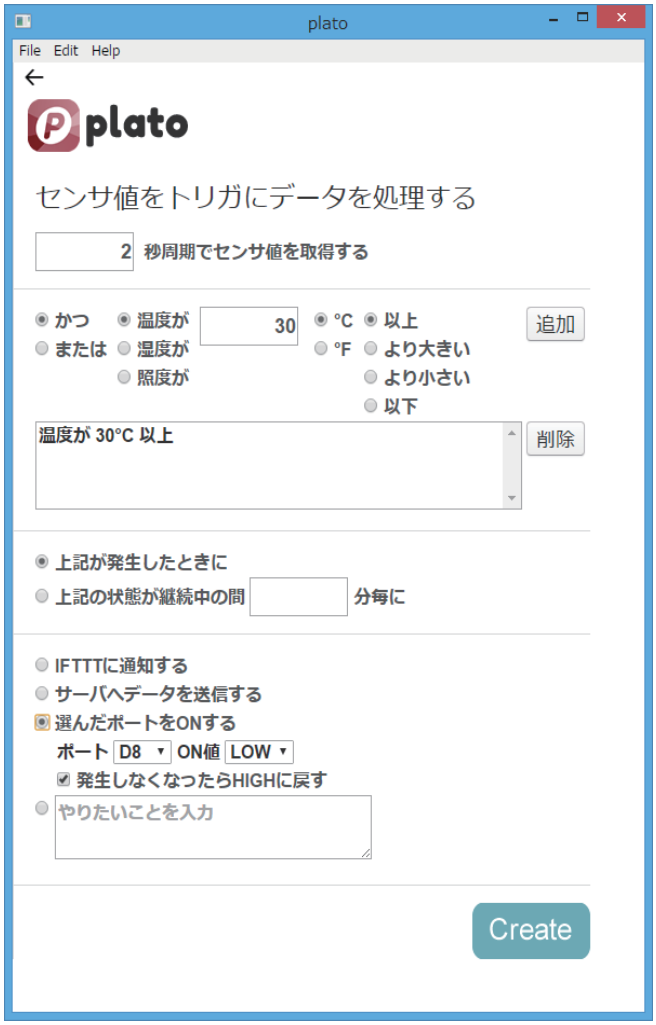
IoTでよく使われる内容を選べるようになっています。

・ネットワークデバイス

標準的な通信モジュールが利用可能です。

◆「センサ値をトリガにデータを処理する」メニューついて

「センサ値をトリガにデータを処理する」を選択すると次のような設定画面になります。



各項目の説明です。

・◯秒周期でセンサ値を取得する

センサから値を読み取る周期（間隔）を1秒単位で指定します。

・トリガ条件

処理を実行するきっかけ（トリガ条件）を指定します。

例では温度が30℃以上になった場合 を設定しています。

温度、湿度、照度を様々な条件で組み合わせる事が可能です。

・アクション

トリガ条件が成立した時に実行する内容を指定します。

例ではWhite-Tigerボードの8番ピン（enziボードのD8ピン）がONになるように 「選んだポートをONする」を選択しています。

これは例えばトリガー条件になった場合にこのポートを「ON」にすることでFANの電源を入れたりすることができます。

その他IFTTT(Webサービス）に連携したり、トリガーをきっかけにサーバにデータを送ったり、後でコードを追加できるようにテキストボックスも用意されています。

またトリガー条件の発生中に何分か起きで実行できるようなアクションも用意されています。

・Createボタンでプログラムが自動生成されVisual Studio Codeがが立ち上がります。

◆「サーバにデータを送信する」メニューついて

「サーバにデータを送信する」を選択すると次のような設定画面になります。

各項目の説明です。

・◯秒周期でセンサ値を取得する

センサから値を読み取る周期（間隔）を1秒単位で指定します。

・◯分周期でデータを送信する

各種センサから読み取ったデータをネットワークに送信する周期（間隔）を 1 分単位で指定します。

・送信データ

送信するデータを指定します。 ここでは、温度、湿度、照度それぞれの平均値を送信することにします。 以下の３項目を順に選択して「追加」ボタンをクリックして下さい

温度 ℃ の平均値
湿度 % の平均値
照度 lx の平均値

・アクション

データ送信時に実行する内容を指定します。 無料で利用可能なIoTサービス「IFTTT」やMAGELLAN BLOCKS (グルーブノーツ社）のIoTボードにも接続可能です。

右図はBlocks接続用の必要項目入力画面です。

・Createボタンでプログラムが自動生成され Visual Studio Codeがが立ち上がります。

③ Platoの入手について

Platoはオープンソース　ソフトウェアです。以下のサイトよりダウンロード可能です。

http://plato.click　　Plato専用サイト
http://forum.mruby.org　NPO軽量Rubyフォーラム

Platoのインストールに関しては各サイトの内容に従ってください。

対応OS
- Windows: Windows7,.8,10　　*Vista以前は対応外
- Mac　OS X
- Linux

④ Platoの体験できるファブラボ

Platoは福岡県内のファブラボにて実際に触ってその使いやすさを実感していただけます。

各ファブラボでは以下の内容を体験いただけます。

- **温度が30℃以上になったらUSB 扇風機を回す　M2M体験**
- **センサの値をインターネットに送信し、グラフ表示をする　IoT体験**

場所：

- FABBIT 小倉
  福岡県北九州市小倉北区浅野2丁目14−3 あるあるCity 2号館3階
  http://fabbit.in/

- タカハイノベーションパーク（飯塚）
  http://www.tipiizuka.com/
  問い合わせ： 0948-80-5396（受付時間：平日9時〜17時）

- グルーブノーツ　福岡天神
  福岡県福岡市中央区今泉1丁目19番22号天神CLASS 3F
  http://www.groovenauts.jp/

展示時間等、詳細については各ファブラボにお問い合わせください。

3）疑似データでIoT開発を加速するmockmock

IoT開発においてWebアプリケーションの動作検証や負荷検証をする際、「デバイスが開発中で存在しない」「デバイスの大量確保が難しい」などの理由で思いどおりに作業が進まないことがあります。

mockmockでは、クラウド上に仮想デバイス（mock）を作成し、サーバーに疑似データを送信します。

実際のデバイスとは違い、短時間で作成可能で、数や動作も思いのままに操作できるのが特徴です。

mockmockを利用することで、動作検証や負荷検証が手軽に実施できるようになり、開発効率の向上やシステムの信頼性向上が見込めます。

特に実際の運用時を想定した大規模なテストを仮想的に行えるメリットがあります。

mockmockの開発にはmrubyが使われています。

① mockmockの使い方

心拍数や呼吸数の遠隔監視を行うという仮想案件での設定例を示します。

mockmockは通信機能のある仮想デバイスを作成するサービスです。送信されたデータの表示や蓄積にはクラウドサービスの利用が便利です。ここでは、クラウドサービスであるAmazonWebServicesのIoTプラットフォーム「AWS IoT」に向けて、HTTPS通信でデータを送ります。（なお、操作画面はβ版のものです。正式版リリースの際に変わる可能性があります。2017.3現在）

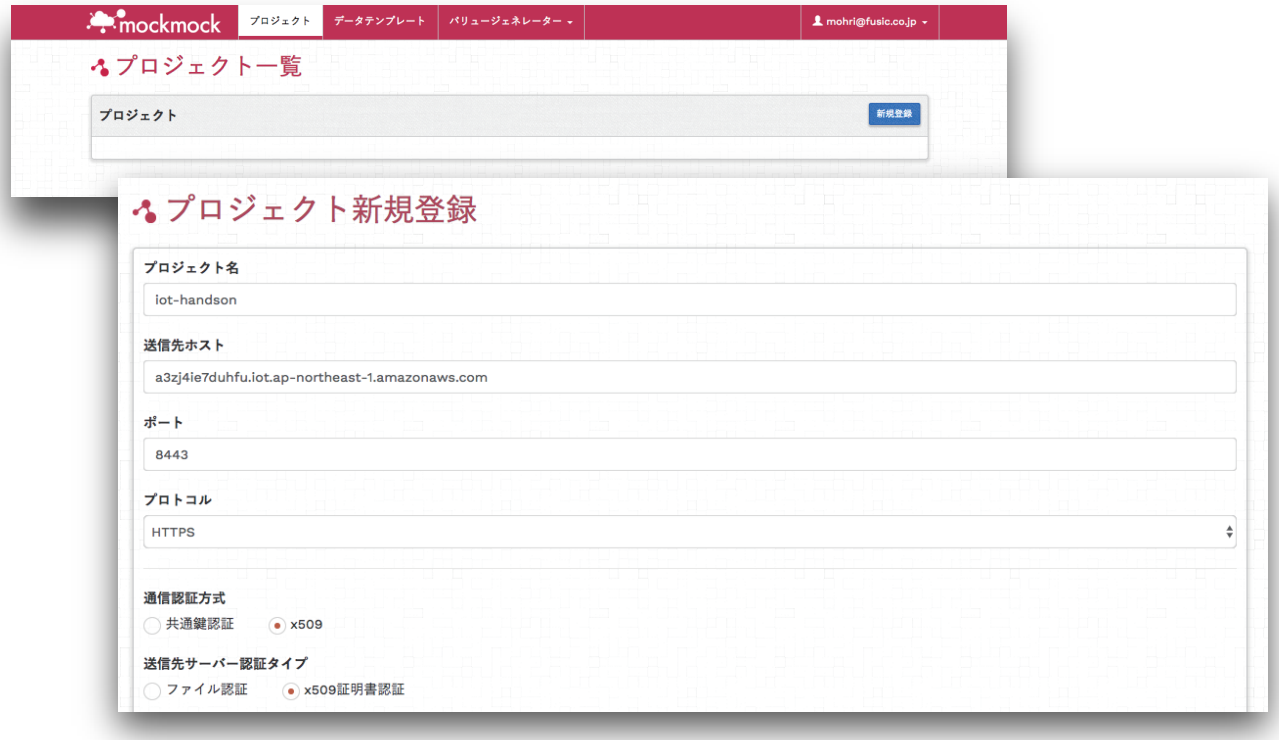
1.アカウント登録

mockmockのサイトよりアカウント登録をします。登録自体は無料です。無料利用枠も用意されています。

https://mock-mock.com/ja

2.プロジェクト登録

上記のWebサイトにログインし、プロジェクトを作ることでmockmockを使えます。例のように入力します。

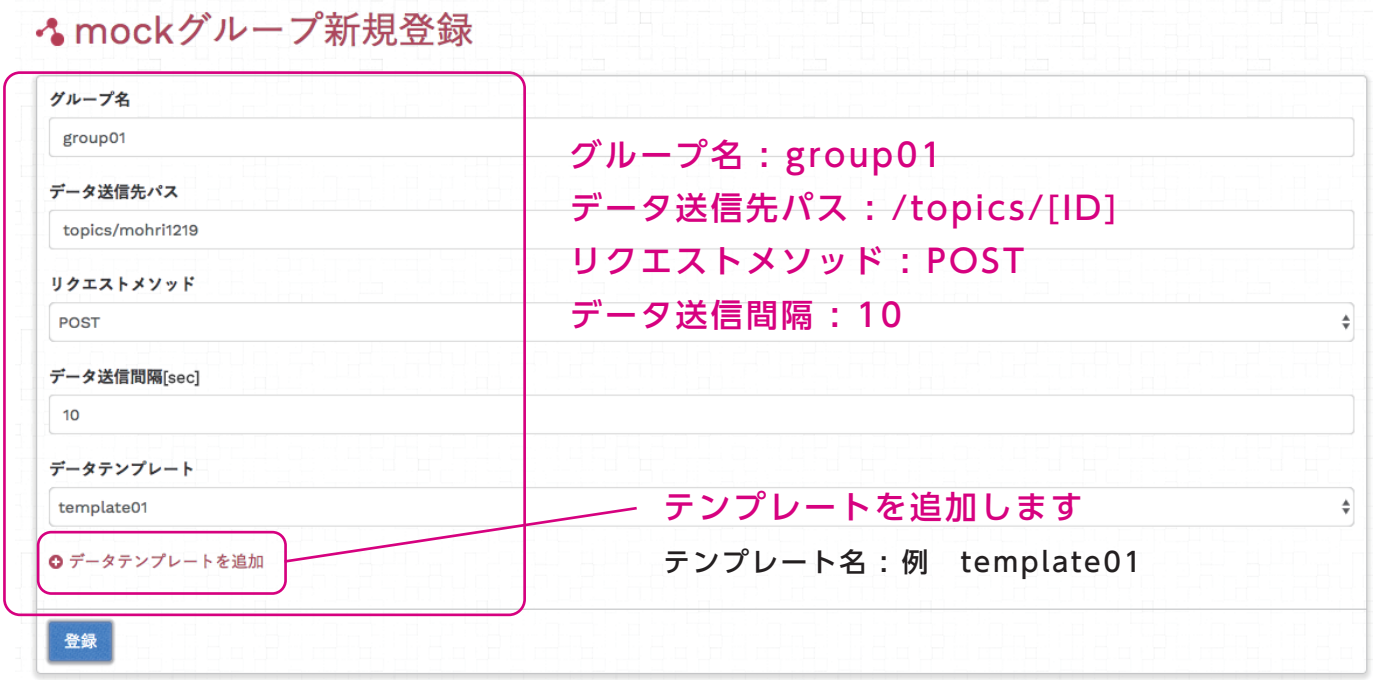


プロジェクト登録手続き

AWS IoTに向けてデータを送る想定なので、AWS IoTで取得した証明書を設定します。

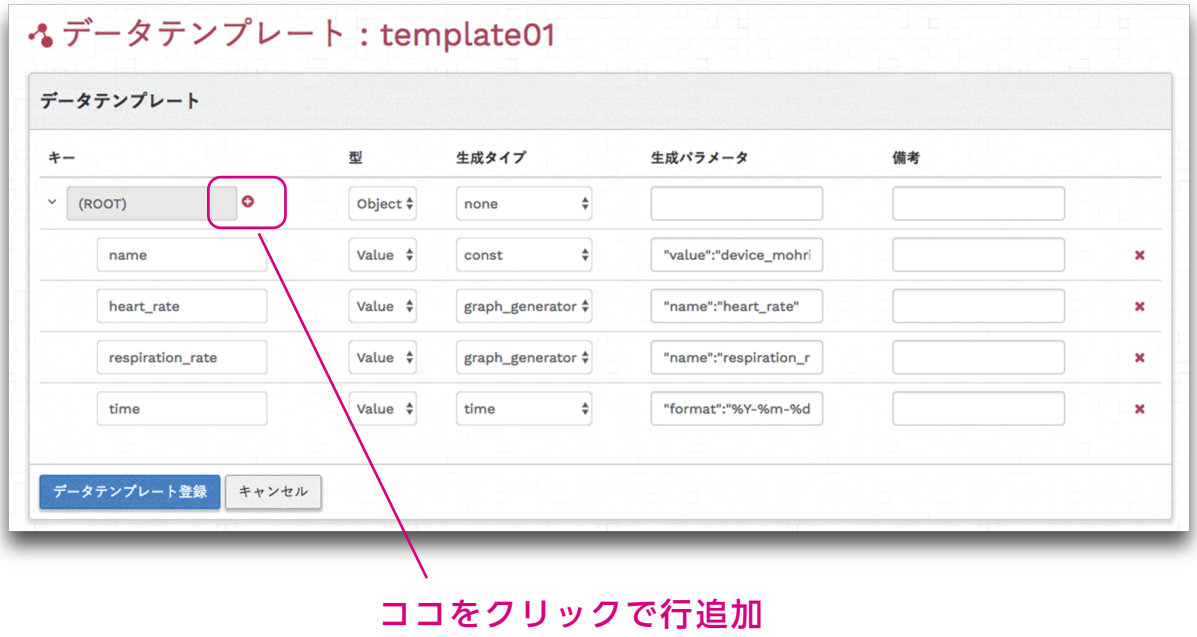
3.グループ登録

mockグループの登録をします。さらにこのグループで使うデータの雛形を作成します。



4.データテンプレートを編集して必要なデータを送れるようにします。

詳細ボタンでデータタイプなどの追加を行います

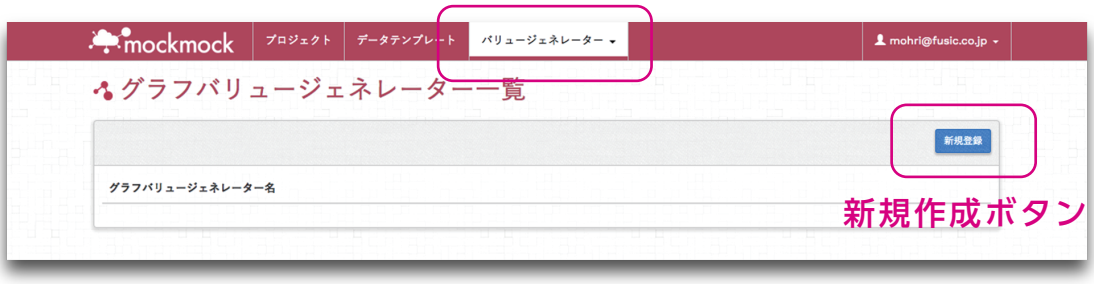


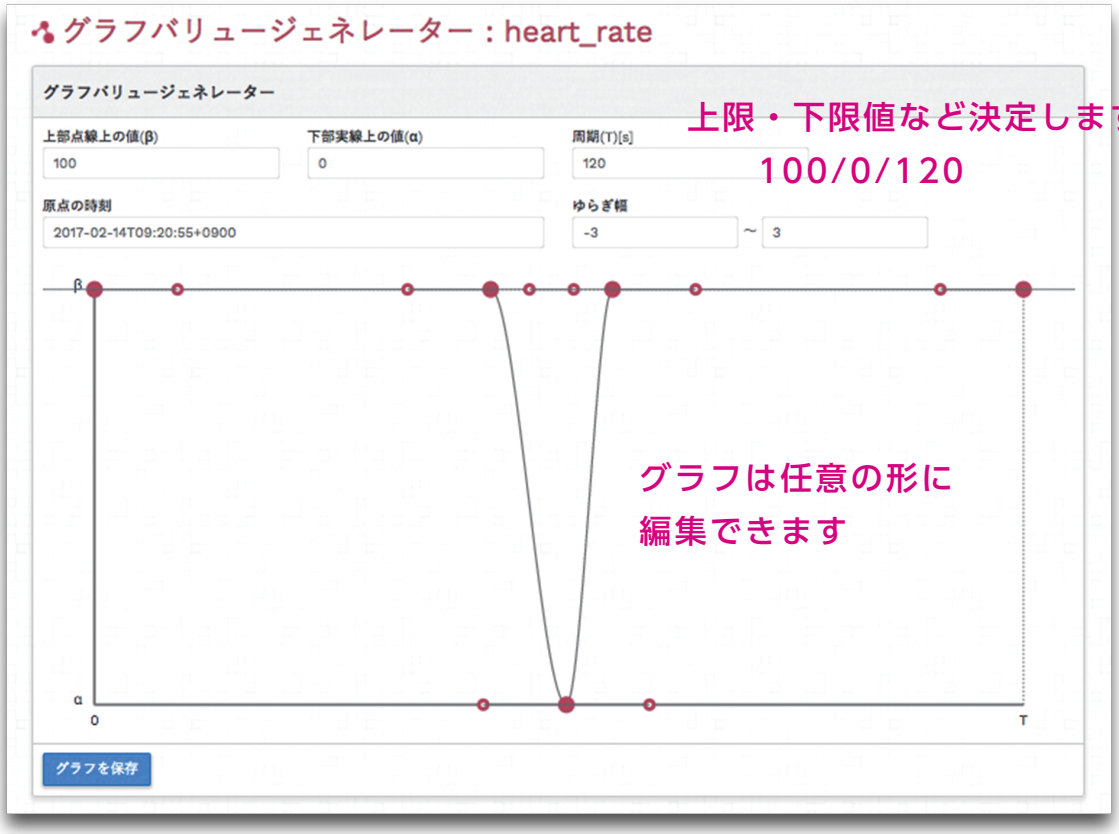
入力値例です

(ROOT)	Object	none	
name	Value	const	″value″:″device_[ID]″
heart_rate	Value	graph_generator	″name″:″heart_rate″
respiration_rate	Value	graph_generator	″name″:″respiration_rate″
time	Value	time	″format″:″%Y−%m−%dT%H:%M:%SZ″

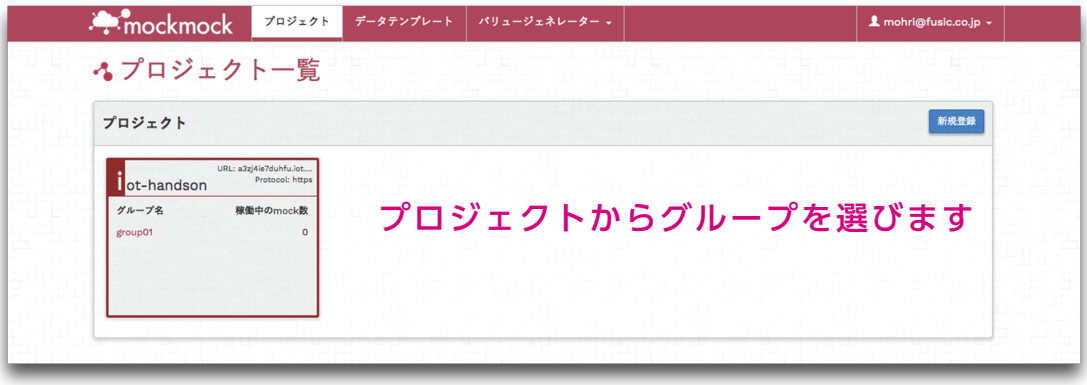
5.バリュージェネレータを作成します。

mockmockでは、バリュージェネレータによって時系列で変化する値を定義できます。現在はグラフによる定義のみですが、今後追加されていく予定です。





6.mock（仮想デバイス）を作成します。



② mockmockデータの表示・蓄積について

mockmockは通信機能のある仮想デバイスです。データの表示や分析にはAWS IoTなどの利用をお勧めします。
以下のブログにAWS IoTを使った例が有りますので参考にして下さい。

http://dev.classmethod.jp/cloud/aws/introduce-of-mockmock-to-aws-iot/

4）機械学習・ビッグデータ処理　MAGELLAN BLOCKS(グルーブノーツ社)

BLOCKSは専門知識やプログラミング知識を必要としないで、Google Cloud Platform (GCP) で提供される各種サービスを簡単に使えるようにしたものです。BLOCKSで提供されるサービスは以下の３つです。

- **IoTボード：様々なデバイスからデータを集める機能が実現できるサービス**
- **MLボード：専門知識を必要とせずノンプログラミングで手軽に機械学習が利用できるサービス**
- **BigDataボード：ノンプログラミングで、さまざまなデータを処理するシステムが構築・実行・運用できるサービス**

●BLOCKS利用にはGoogle Cloud Platformの登録が必須です。
https://www.magellanic-clouds.com/blocks/engineersblog/gcp/magellan-blocks-gcp-entry1/

●BLOCKSの始め方
各ボードの使い方が説明されています。
この項目の２で紹介されているIoTフレームワークPlatoを使ってセンサー値をIoTボードに送ることが可能です。
https://www.magellanic-clouds.com/blocks/guide/blocks/

5）Iot インフラ構築を容易にする技術の紹介

① AWS IoT

IoTデバイスからのデータをAWSクラウドに安全に送信し、AWSクラウド上の機能を使って収集・処理・分析・実行することを容易に実現するクラウドサービス。AWSがもつ70以上のサービスと組み合わせることが可能です。
（サービスの概要は本レポートの４.IoTの要素技術を参照してください）

AWS IoTを素早く始めるためのSDKやスターターキットが用意されています。

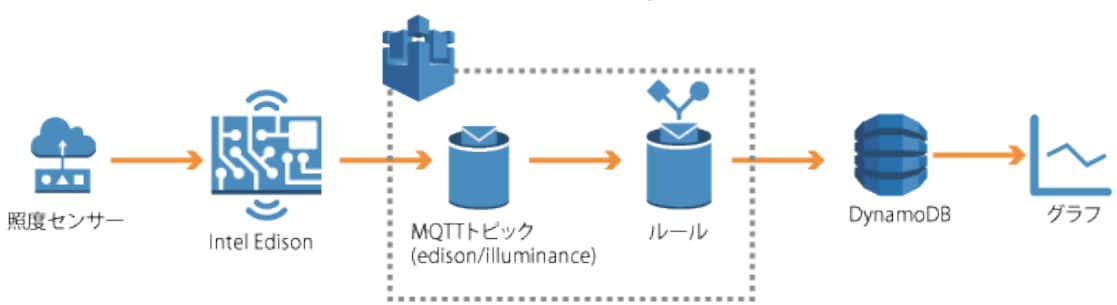
◆ AWS IoTデバイスSDK

Embedded CやJavaScript、Aruduino Yun, Python, Android, iOSを使ったSDKです。
ソース取得ができ、Developerガイドも豊富です。
https://aws.amazon.com/jp/iot/sdk/

◆ AWSスターターキット

マイコンボード、センサー、アクチュエーター、SDKと入門ガイドが含まれたキットです。
https://aws.amazon.com/jp/iot-platform/getting-started/#kits

上記のEdisonスターターキットを利用して照度センサーのデータをAWSへ送信、AWS IoTでルールに基づいてNoSQLデータベースDynamoDBへデータ保存、さらに閾値を下回った場合E-mailアラートを送信する仕組みのハンズオンが公開されています。
http://awsiot-handson-fundamentals.readthedocs.io/ja/latest/01.html



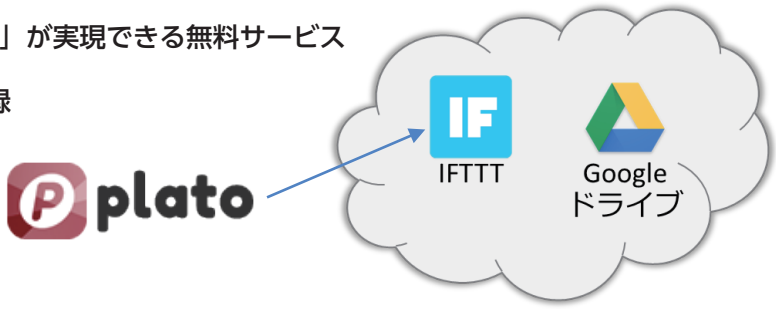
② IFTTT（イフト）

IoT向けWebサービスで、概要は以下のとおりです。

●IF This Then That 「もしコレになったらアレする」が実現できる無料サービス

●This (コレ) と That (アレ) を「レシピ」として登録
　様々なWebサービスを利用可能

●iOS, Androidでも利用可能

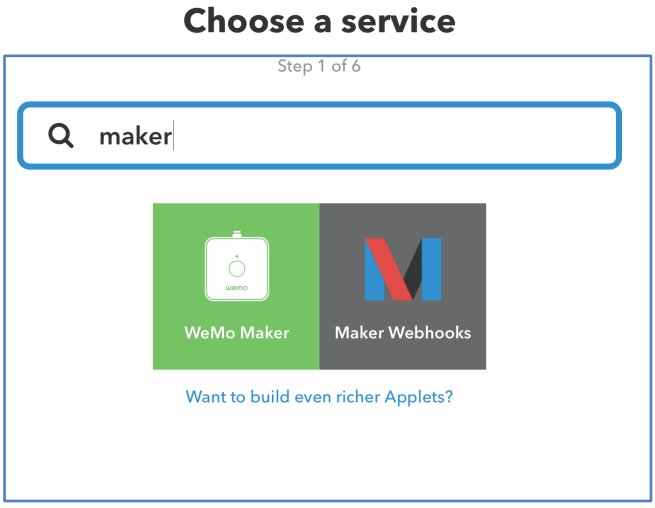


実際にIFTTTを使いGoogleドライブを利用してセンサーからのデータをグラフ表示する設定をご紹介します。
フレームワークPlatoからIFTTTにデータを送ります。IFTTTは受け取ったデータをGoogleドライブに転送します。
Googleドライブでは転送されたデータをスプレッドシートに保存し、グラフ表示します。

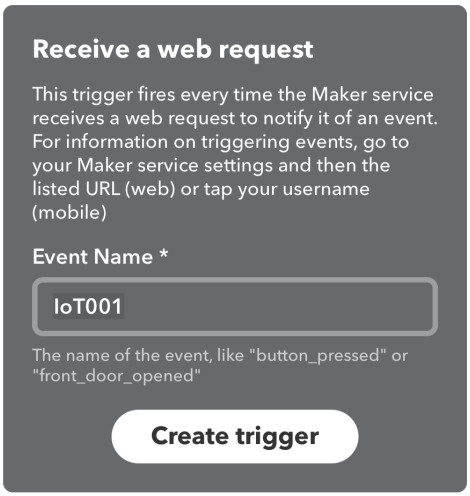
1. IFTTTのユーザ登録は必須ですのでまず登録を
　完了しておきます。

2. アプレットの新規作成をします。

3. レシピを登録します。
　レシピとはIFTTTの用語で行いたい処理を指します。
　Maker Webhooksを使います。
　検索フィールドを使うと探しやすいです。



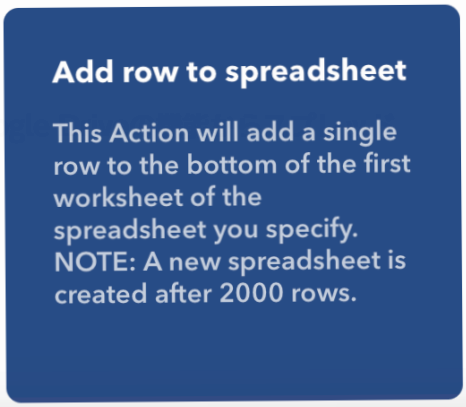
4. Maker Webhooksをクリックし、イベント名を入力し、トリガーを作成します。



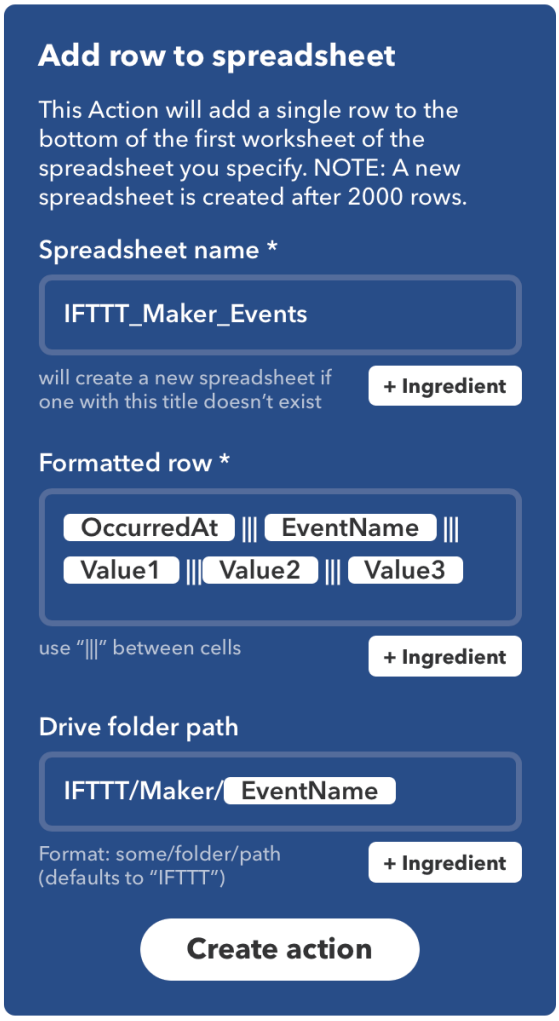
5. 次にアクション（何をする）を登録します。

Google Driveを選びます。

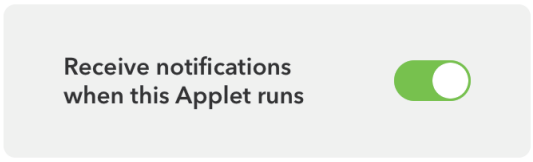
6. Googleへの接続を許可し、
　シートの設定を選びます。



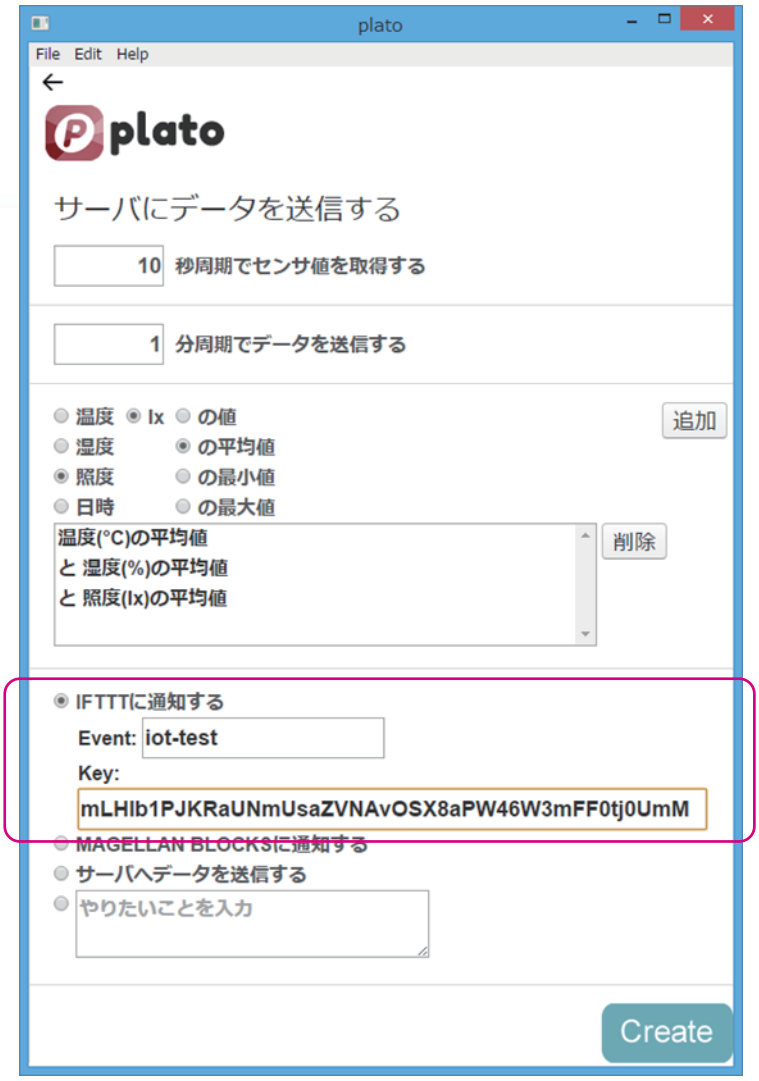
7. スプレッドシートの設定をし、アクションの作成を行います。



8. アクションが作成されます。



9. PlatoのIFTTT接続メニューを使います。



③ Azure IoT Suite

汎用的な IoT シナリオを実現するクラウド ベースの事前定義済みソリューションで、リモート監視や資産管理、予兆保守といったシナリオが用意されています。
（サービスの概要は本レポートの４．IoTの要素技術を参照してください）

特徴として
・既存のデバイスや機器が利用できる。
・拡張可能なIoTシナリオにより、すぐに開始できる

Azure IoT 認定デバイスが紹介されています。
BeagleBone、Inrtel Edison, Rasoberry Piといった認定デバイスのリストです。
https://azure.microsoft.com/ja-jp/suites/iot-suite/

実際の導入に関するチュートリアルが多く用意されています。

・Internet of Things キット　ハンズオン　トレーニング
本物のセンサー搭載ハードウェア（.NET Micro Framework）やWindows 10 IoT Coreデバイスなどとクラウド（Microsoft Azure）で、ソフトウェアをVisual Studioで開発しながら、体系的に開発スキルを獲得
http://ms-iotkithol-jp.github.io/
・Azure IoT Suite 自習書シリーズ初級編　ー　リモート監視の事前校正済みソリューション
https://www.microsoft.com/ja-jp/cloud-platform/products-Microsoft-Azure-IoT-Service.aspx
・オンライン学習コンテンツ
http://ms-iotkithol-jp.github.io/LetsBegin.htm
・事前構成済みソリューション利用チュートリアル
https://docs.microsoft.com/ja-jp/azure/iot-suite/iot-suite-getstarted-preconfigured-solutions

④ IBM Bluemix

BluemixはIBMのCloud Foundryをベースにしたpaasです。開発環境と実行環境がクラウド上に用意されています。対応している開発言語はJava, JavaScript, Swift, Ruby, Python, Perl, PHPなどで主だった開発言語はほとんど対応しています。
その上で使えるサービスは140種類以上、Watsonなども利用可能です。
Bluemix開発者向けのDeveloperWorksが用意されています。そこではテーマ別のチュートリアルが用意されています。

例として
・IoTデバイスからGPSデータを取得して使用するPHPアプリを作成する
・携帯電話をIoTデバイスに変身させる、など
https://www.ibm.com/developerworks/jp/bluemix/tutorial.html

料金体系など詳しい解説はこちら。
http://qiita.com/zuhito/items/86e7ad47d14937d3e9b6