

デザイン工学：プロトタイピング論

秋田純一（金沢大）

akita@is.t.kanazawa-u.ac.jp
@akita11

(準備)マイコンとコンピュータ

マイコンは、
演算性能は低めだが、
実世界(物理世界)との
連携が得意

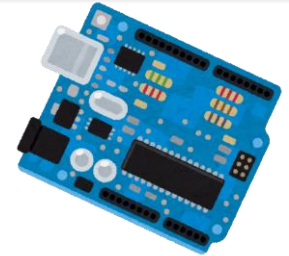


PC

(Personal Computer)



スマホ



マイコン

(Micro-controller)

演算性能



インターネット
接続



省サイズ



低価格



実世界連携



(準備)使用するマイコン:M5stack

- 中国M5stack社の製品
 - ESP32 (WiFi/BluetoothLEつきマイコン)
(240MHz Dual Core, 600MIPS)
 - センサ等の接続ができるコネクタ
 - 小型液晶モニタ
 - プッシュスイッチ(3個)
 - 単体で「完成品」に近いプロトタイプを作成可能
 - プログラミング環境: MicroPython / ブロック型言語等
 - 積み重ねて機能拡張が可能



(準備) M5stackのプログラミング



UI Flow DesktopIDE (USB版)

<https://m5stack.com/pages/download>

UI Flow (Web版)

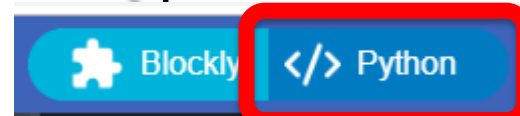
<http://flow.m5stack.com/>

<http://flow01.m5stack.com/>

MicroPythonでコーディング
(プログラミングに慣れている
人は便利かも)



Blockly→MicroPythonへの変換が可能(※逆は×)

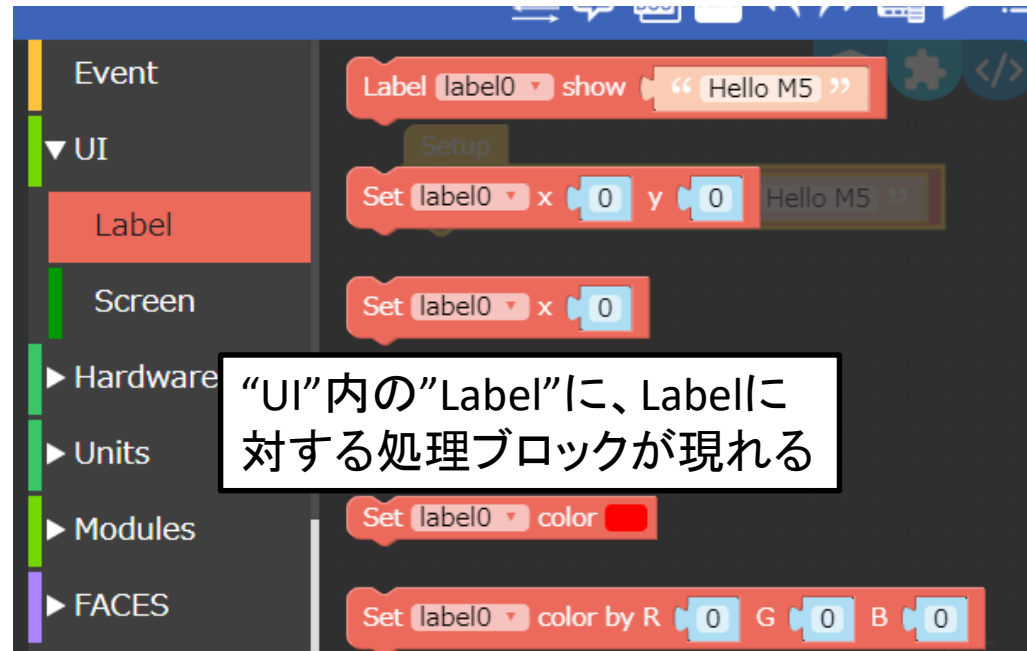


1.お約束の”Hellow World”

- UI Flow DesktopIDEを開く



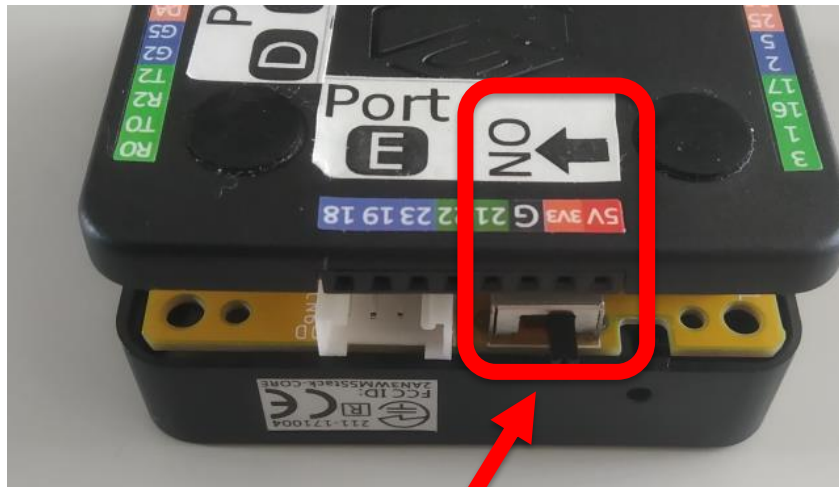
※不要なLabel等はドラッグ時に現れる「ゴミ箱」へ入れれば消去



※”Setup”は、起動後に実行される処理を「はめる」場所

1.M5stackの電源のON/OFF

- PC等にUSB-Cケーブルで接続して給電
 - 使用後はケーブルを抜けば電源OFF(ブチ切OK)
 - 内蔵バッテリーもある(この授業では普段は使わない)



主電源
(普段はOFFにしておく)
※ONにするとバッテリーから給電&充電



本体電源SW(赤)
・1回押し→リセット(※よく使う)
・2回押し→電源ON/OFF(バッテリー動作時)

1.プログラムの転送・実行

初回のみ行う設定



本体ディスプレイの上部に「USB Mode」と出ているのを確認

なっていなかったらリセット
→すぐ右ボタンを押し続け
→メニューから「USB Mode」を選択

Setting

COM :

Language : English

Device : Core

Win: COM1 など

Mac: /dev/tty.SLAB_USBtoUART など

Cancel

OK

これらが選べなかったら、
左上の「Driver Installation」からドライバを入れる

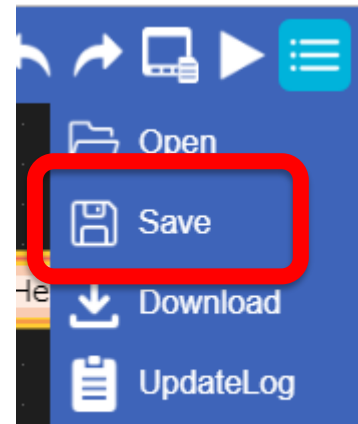
Hide UI この部分が↑で指定した名称になる

COM: Unknown

[Disconnected]



“Run”ボタンで、M5stackに
プログラムを転送し、実行



“Save”でプログラムを保存できる
※「1a.m5f」のように、作業ごとに名前を
変えて保存しておくこと
※“Load”で読み込むこともできる
※画面キャプチャもあわせて
保存しておくと便利

2.テキストの属性を変更する

置いたLabel(テキスト)をクリック
→Labelの属性(位置、フォント等)が変更できる

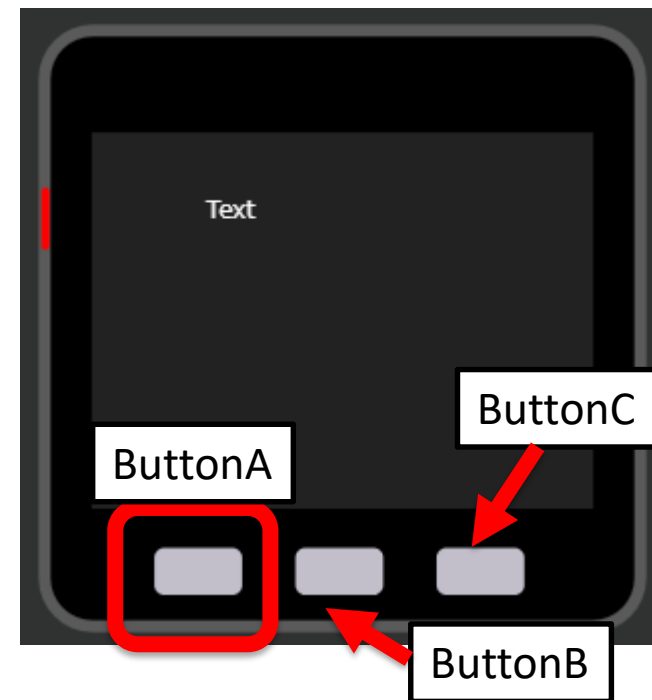
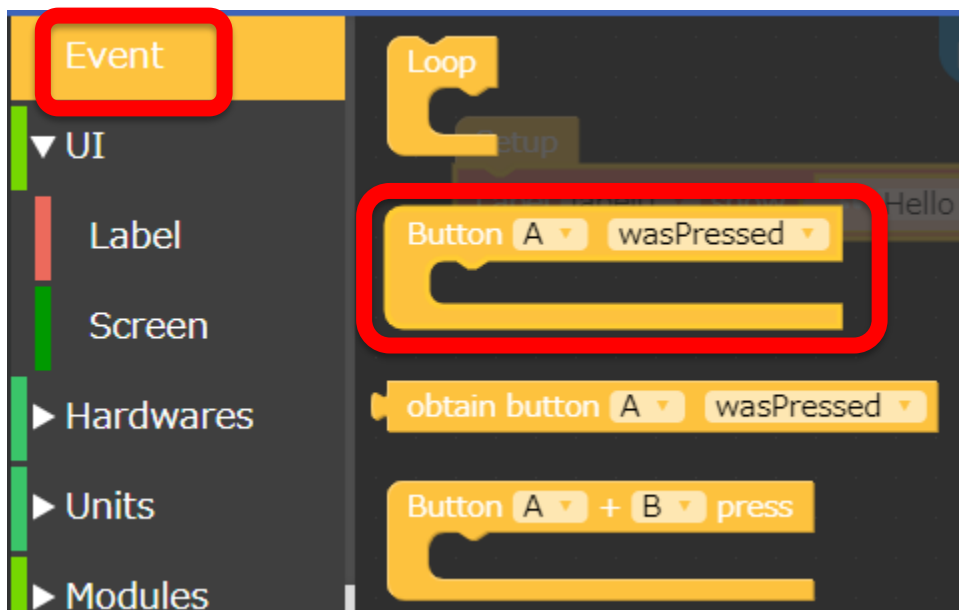


黒枠白字の内容は各自で実施すること
(結果(写真等)・プログラムも保存しておくとうい)

※この"text"属性で、画面上に表示される文字列を設定できるが、プログラムの"Setup"内で設定してもよい(結果は同じになる)

位置や色、フォント(文字サイズ)を変更して実行してみる
また、背景の色の変更や、適当な図形(矩形・円)なども描画してみる

3.本体ボタンの動作



ButtonAを押した時に実行される処理を書くブロック
※検出するボタンの種類・押し方などを選ぶ



いずれかのボタンを、いろいろな押し方で押して
Labelを変更するプログラムを実行してみる

4. 本体ボタンの動作 : Wait

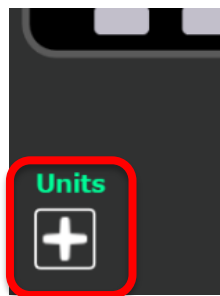


Timer内の"Wait"で、指定時間だけ待機する

Waitで指定した時間待つ動作を確認する

5. センサの使い方 : Button

- M5stack純正の各種センサ等の「Unit」を接続し、UI Flowの「Unit」から使用できる



Unitを追加



DualButtonを選択
(接続するPortをBとする)



DualButtonユニットが追加された

やや見にくいですが、スクロールバーでスクロール

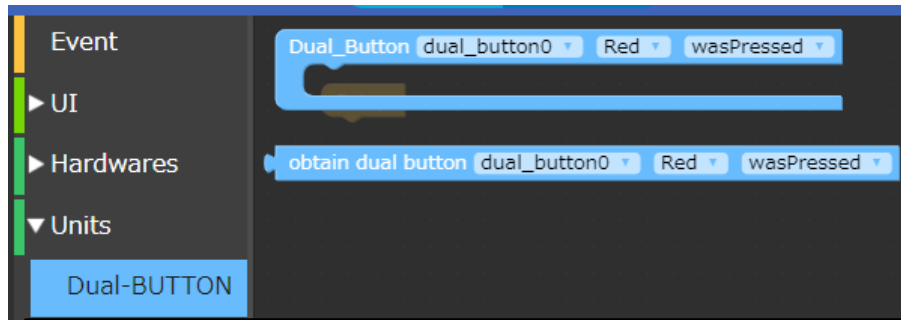


実際にDualButtonユニットを、指定した「PortB」に接続する

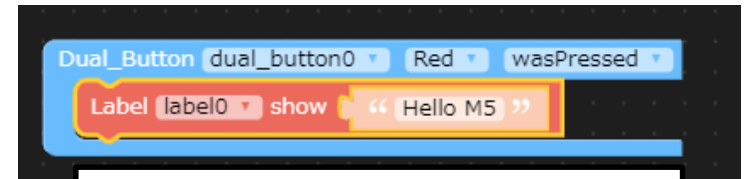


コネクタの片面の両側にヤマがあるので、ソケットにあわせる

5. センサの使い方 : Button



“Units”内に、追加したUnitに関するブロックが現れる



接続したButtonユニットの赤ボタンを押した時にlabel0を“Hello M5”とするプログラム

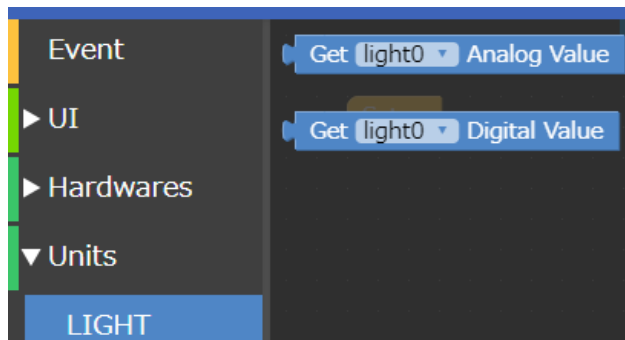
これを参考に、以下の仕様のプログラムを作成して動作させる

- 動作仕様
 - ラベルを1つおく
 - 初期状態では、label0=""とする
 - 赤ボタンを押したら、label0を1秒間だけ“Red”に変更し、その後""に戻る
 - 青ボタンを押したら、画面中央(160,120)に半径30の円を描画し、1秒後に消す

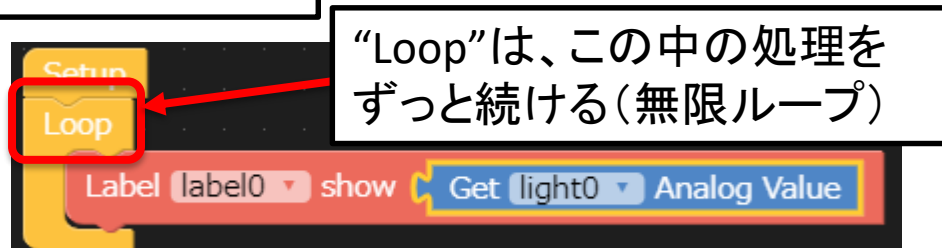
6. センサの使い方 : Light (明るさ)



Lightユニットは、センサ面の明るさに応じた電圧が出力され、それを取得できる



Lightユニットで使える処理
("Analog Value"でアナログ値)



Lightsユニットで取得した明るさの値を表示する例

Lightユニットの値(明るさ)をディスプレイに表示し、明るさに応じて値が変化することを確認する

6.条件分岐

Logic(論理)から持ってくる

Math(数学)から持ってくる

設定したい条件にあわせて変更



If以下の条件が成り立つ場合に実行

If以下の条件が成り立たない場合に実行

6.変数 (Variables)

- 値を一時的に保管しておく「箱」= 変数

変数を作成

Do you want to create variables?
light

変数に名前をつける (例: "light")

その変数を使う
・変数の値を設定
・変数の値を変更
変数の値を読む

明るさセンサの値を変数に入れる

その変数の値を使う(条件分岐)

```
Setup
Loop
  set light to Get light0 Analog Value
  if light < 300
  do
    Label label0 show "Bright"
  else
    Label label0 show "Dark"
```

7. センサの使い方 (ToF)

- “ToF”ユニット＝センサ面から対象物までの距離を計測する
 - ToF = Time of Flight
※光の往復時間から距離を計測する
 - Unitから”TOF”を追加 (PortAを選択)
 - PortA (複数あるがいずれでもOK)に接続する



ToFセンサから取得した値(単位:mm)をディスプレイに表示するプログラムを作成し、実行させる

8. センサの使い方 (Color)

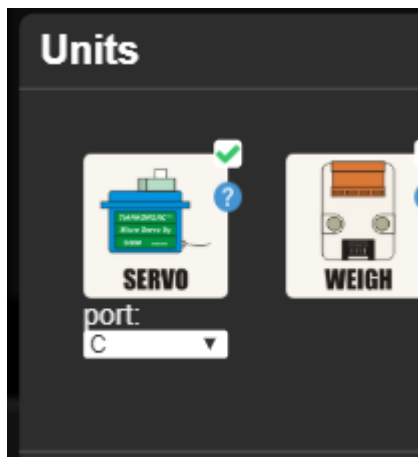
- “Color”ユニット＝対象物の色 (RGB値) を計測
 - Unitから”COLOR”を追加 (PortAを選択)
 - PortA (複数あるがいずれでもOK) に接続する



Colorセンサから取得した値 (R, G, Bそれぞれ) をディスプレイに表示するプログラムを記述し、実行させる

9. アクチュエータの使い方 (サーボ)

- 指定した位置にアームを移動できる
サーボモータ
- Unitから追加し、指定したPortに接続

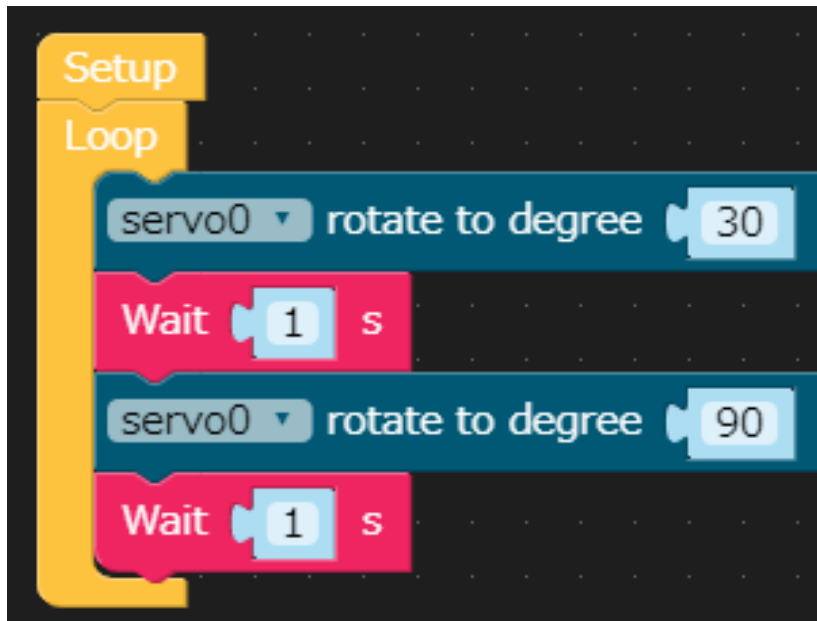


※アクチュエータ (actuator)
= 環境に作用を与えるもの
(センサ (sensor) の逆の機能)



9. アクチュエータの使い方(サーボ)

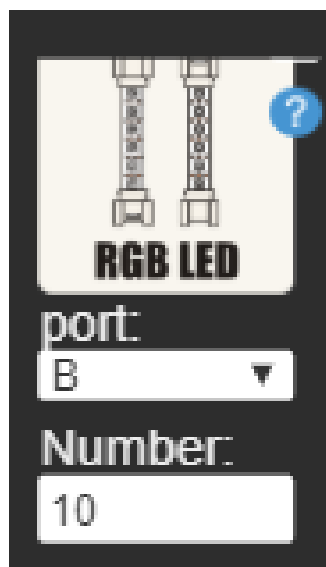
- サーボの角度は”角度[度]”で指定できる



このプログラムの動作を確認する。
またサーボの移動順序を自分で決め、それを実現する
プログラムを記述して実行する

10. アクチュエータの使い方 (LEDテープ)

- コマンドで複数LEDのRGB値を個別に指定できるLED (NeoPixel)



接続するPortと
LED数(この実験の
機材では10個)を指定



指定したPortに接続

10. アクチュエータの使い方 (LEDテープ)

Blocklyのブロックを参考に、自分でLED点灯の仕様を決め、それを実現するプログラムを作成して実行する

Set neopixel0 index 1 RGB color

indexで指定した番号のLEDの色を設定

Set neopixel0 from 1 to 5 RGB color

indexで指定した範囲のLEDの色をまとめて設定

Set neopixel0 from 1 to 5 R 0 G 0 B 0

indexで指定した範囲のLEDの色をRGB値で設定

Set neopixel0 all RGB color

すべてのLEDの色をまとめて設定

Set neopixel0 brightness 20

すべてのLEDの色を変えずに明るさのみを設定

Set neopixel0 show lock true

Show neopixel0 RGB LED

設定した各LEDの色を実際に表示する

※これを実行した時点で、実際のLEDの色が変わる

アイデアの具体化

- ものをつくるのためには「仕様」が必要
 - 「仕様」を決めてからものを作る？
 - 作って、使ってみて、「仕様」を決める？
 - 「完璧な完成品」を作ってから試す？
 - 「とりあえず動く完成品」を作り、試して、「アイデアが正しかったか」を検証し、改良して作って・・・を繰り返す？

プロトタイプ
(Prototype)



デザイン思考
(Design Thinking)

作る＝思考の道具

アイデアの具現化(レポート)

- 「あったら便利なもの、生活がよくなるもの」を1つ考え、M5stackで試作する
 - 実用レベルの完成度である必要はない
 - あくまでも試作品・動作検証ができればよい
- 以下の内容を含むこと
 - 作ったものの概要
 - それを作ろうと思った理由
 - 試作結果(実機の写真、プログラム(できれば))
 - 実際に使ってみての感想・改善点
- 例: ※これと全く同じものは×、ヒントにするはOK
 - 部屋に誰か入ってきたらスマホに通知
(ToFセンサ・Remoteを使えば作れそう)
 - 視覚障害者向けの、対象物の色を音で表現する機器
(Colorセンサ・スピーカを使えば作れそう)
 - 暗くなったら部屋の電灯をつける
(Lightセンサとサーボモータ(吊り紐を引く)でできそう)

成果発表について

- 以下の内容を含むこと
 - 作ったものの概要
 - それを作ろうと思った理由
 - 試作結果(実機の写真、Blocklyのプログラム)
 - 実際に使ってみての感想・改善点

プロタイピングの心得

- 機能やコンセプトを検証するための最低限の実装
 - つくって触ってみることで初めてわかることがある
- プロトタイピングに必要なもの
 - 検証したい機能やコンセプト
 - 実装の迅速さ
- プロトタイピングに必ずしも必要ないもの
 - 耐久性
 - バグのないコード



プロタイピングの心得



「似たものが既にある」は
気にしなくてOK！
(むしろ改善・差別化のチャンス)



おまけ: さらにやってみたくなった人は



SWITCHSCIENCE

OPEN SOURCE HARDWARE SHOP

送料 0 円 ~ 500 円 (税込)
3,000 円 または 10,000 円 以上で送料無料



商品を探す M5stack

マイページ ログアウト カート



All Products | 93 products



並び順 公開日が新しい 表示件数 10 20 50 100 200

PAGE: 1 2 >>

Category すべて開く

新商品 (135)

- > SSX (Switch Science eXperiment) (12)
- スイッチサイエンス製品 (294)
- スイッチエデュケーション製品 (70)
- スイッチサイエンスマーケットプレイス (委託商品) (645)
- Rapiro(26)
- MakerBot(82)
- Arduino(297)
- SparkFun(539)
- Seeed(217)
- Adafruit(295)
- Pololu(160)
- Pimoroni(46)
- Kitronik(27)
- Digi International(21)
- Raspberry Pi(274)
- Mbed(75)
- Intel(38)
- > SPRESENSE(16)
- micro:bit(127)

 <p>M5Stack Basic 3,575 円 在庫: 多数 1 カートに追加</p>	 <p>M5Stack Commuモジュール 1,441 円 在庫: 0 1 カートに追加</p>	 <p>M5Stack用LANモジュール 2,770 円 在庫: 多数 1 カートに追加</p>	 <p>M5Stack用電池モジュール 2,077 円 在庫: 多数 1 カートに追加</p>	 <p>M5Stack FIRE 6,325 円 在庫: 0 1 カートに追加</p>
 <p>M5Stack PLUS 1,628 円 在庫: 多数 1 カートに追加</p>	 <p>M5Stack用オッチバンド (オレンジ) --在庫限り 1,815 円 在庫: 0 1 カートに追加</p>	 <p>M5Stack用PLCモジュール 2,770 円 在庫: 多数 1 カートに追加</p>	 <p>M5Stack Gray (9軸 IMU搭載) 4,290 円 在庫: 多数 1 カートに追加</p>	 <p>M5Stack用ミニプロトボードユニット 379 円 在庫: 多数 1 カートに追加</p>
 <p>M5Stack THERMAL 32x24P 40C~100C 115x70P 1 カートに追加</p>	 <p>M5Stack PROTO Module ESP32 Core2P 800mAh Battery Arduino micro:python 1 カートに追加</p>	 <p>M5Stack LIME BUTTON 1 カートに追加</p>	 <p>M5Stack GREEN BUTTON 1 カートに追加</p>	 <p>M5Stack BLUE BUTTON 1 カートに追加</p>



マルツ(金沢西インター店)

<https://www.switch-science.com/>

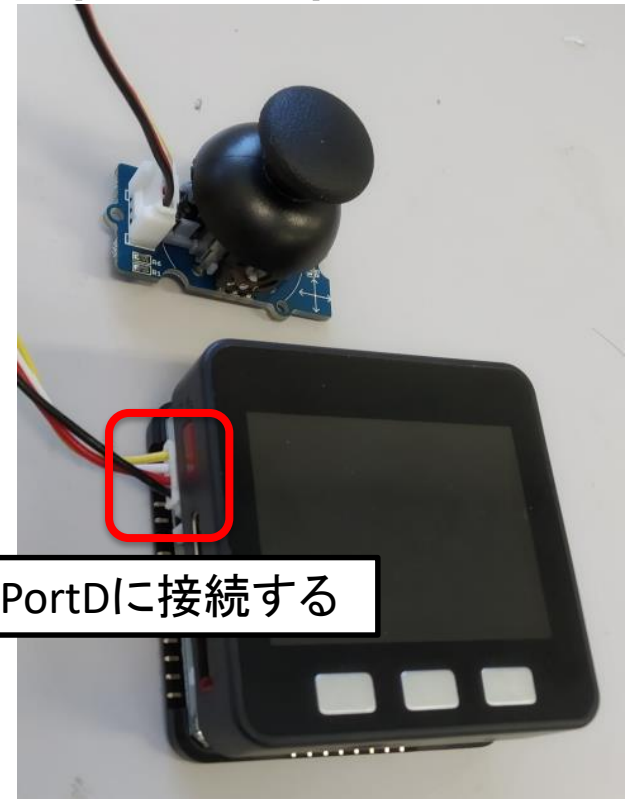
以降、オプションの発展的内容
(興味のある人、余裕のある人向け)

実装上の注意

- PortA～Eには、以下の通りIOピンが接続されている。
- 以下のIOピンは、PortA～Eコネクタと本体コネクタに接続されていて、同時に使えないので注意
 - PortA: IO21, IO22
 - PortB: IO26, IO36
 - PortC: IO17, IO16
 - PortD: IO35, IO34
 - PortE: IO13, IO5

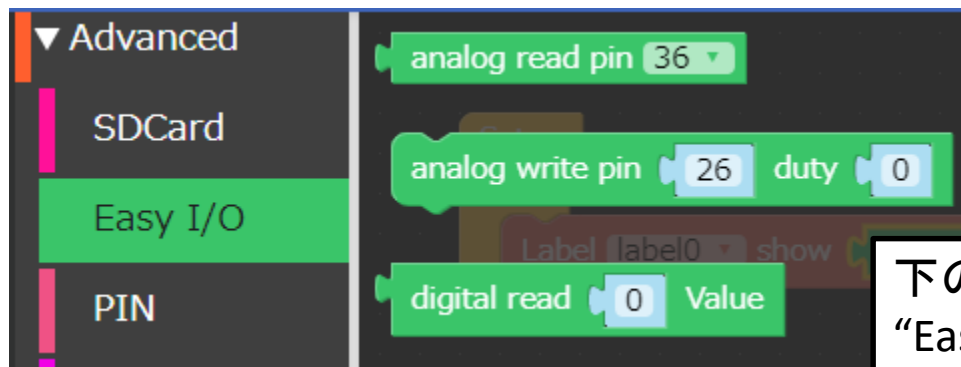
E1. センサの使い方 (ジョイスティック)

- ジョイスティックは、Unitにないので、信号を直接読み取る
- PortD=2本の信号線 (IOピン34番と35番)
(IO=Input/Output, 入出力のこと)
- スティックの傾き (X軸・Y軸) に応じて、それぞれの電圧が変化する

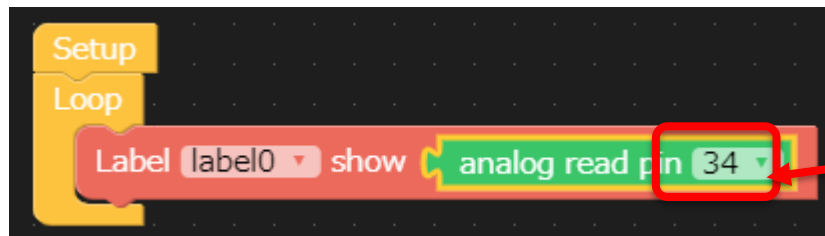


E1. センサの使い方 (ジョイスティック)

- IOピンの電圧は、“analog read”で取得できる



下のほうの“Advanced”内の
“Easy I/O”にある



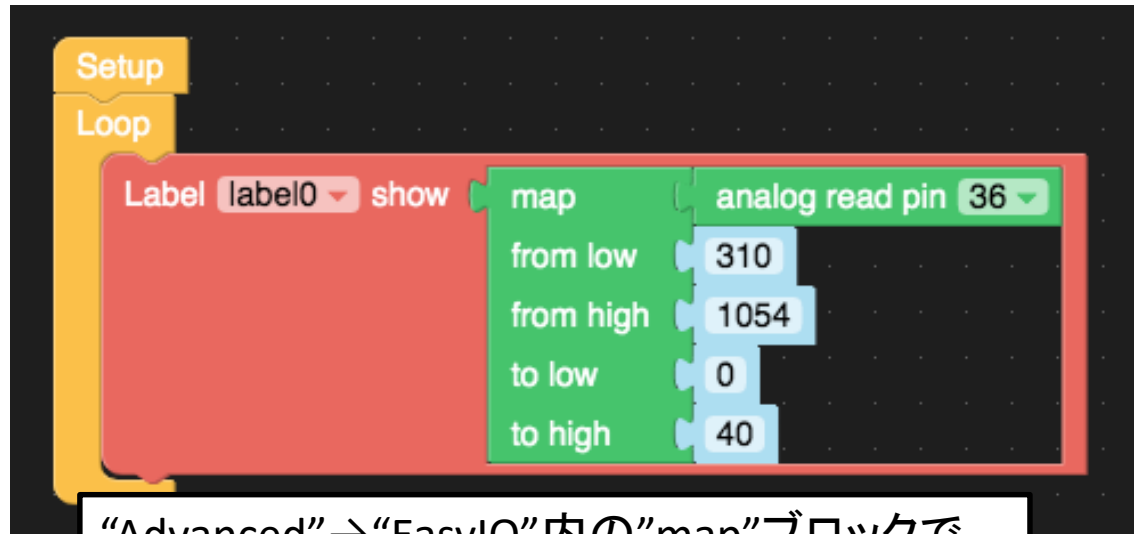
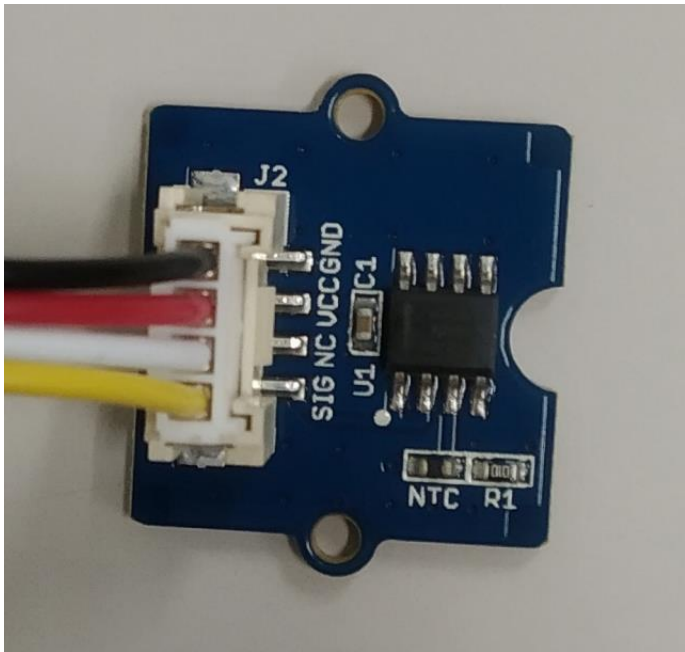
ジョイスティックのX軸は、
IOピン34番につながっているので
そのピン番号の電圧を取得する

このプログラムを実行させて、スティックを傾けたときに表示される値の変化を確認する。
またY軸 (IOピン35番に接続されている) の値もあわせて表示させてみる。

E2.温度センサ

- 温度に応じた電圧が出力される
- 理論計算は複雑だが、以下の値の範囲を変換すればよい

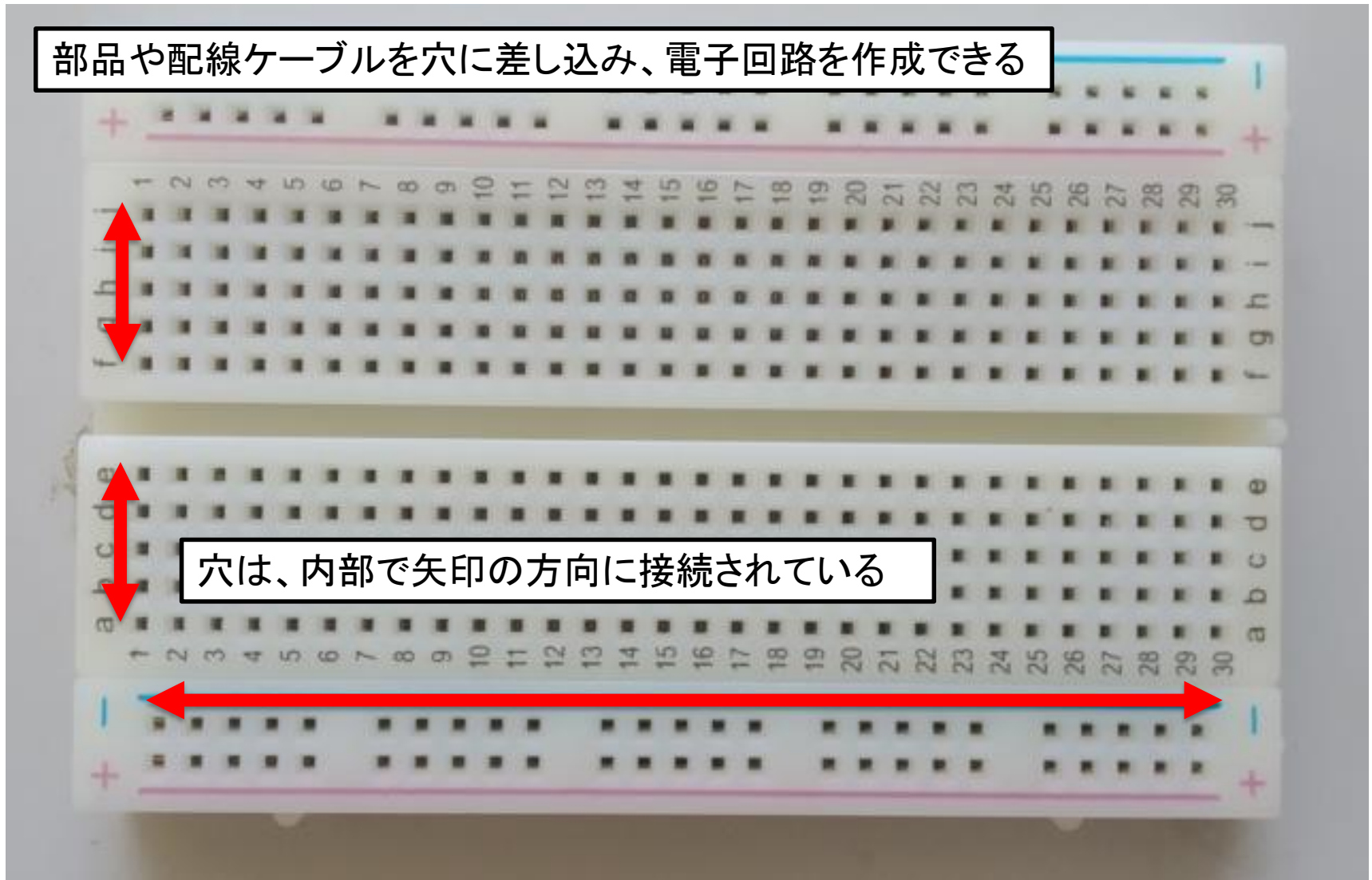
analog readの値 : 310 ~ 1054 \leftrightarrow 温度 : 0°C ~ 40°C



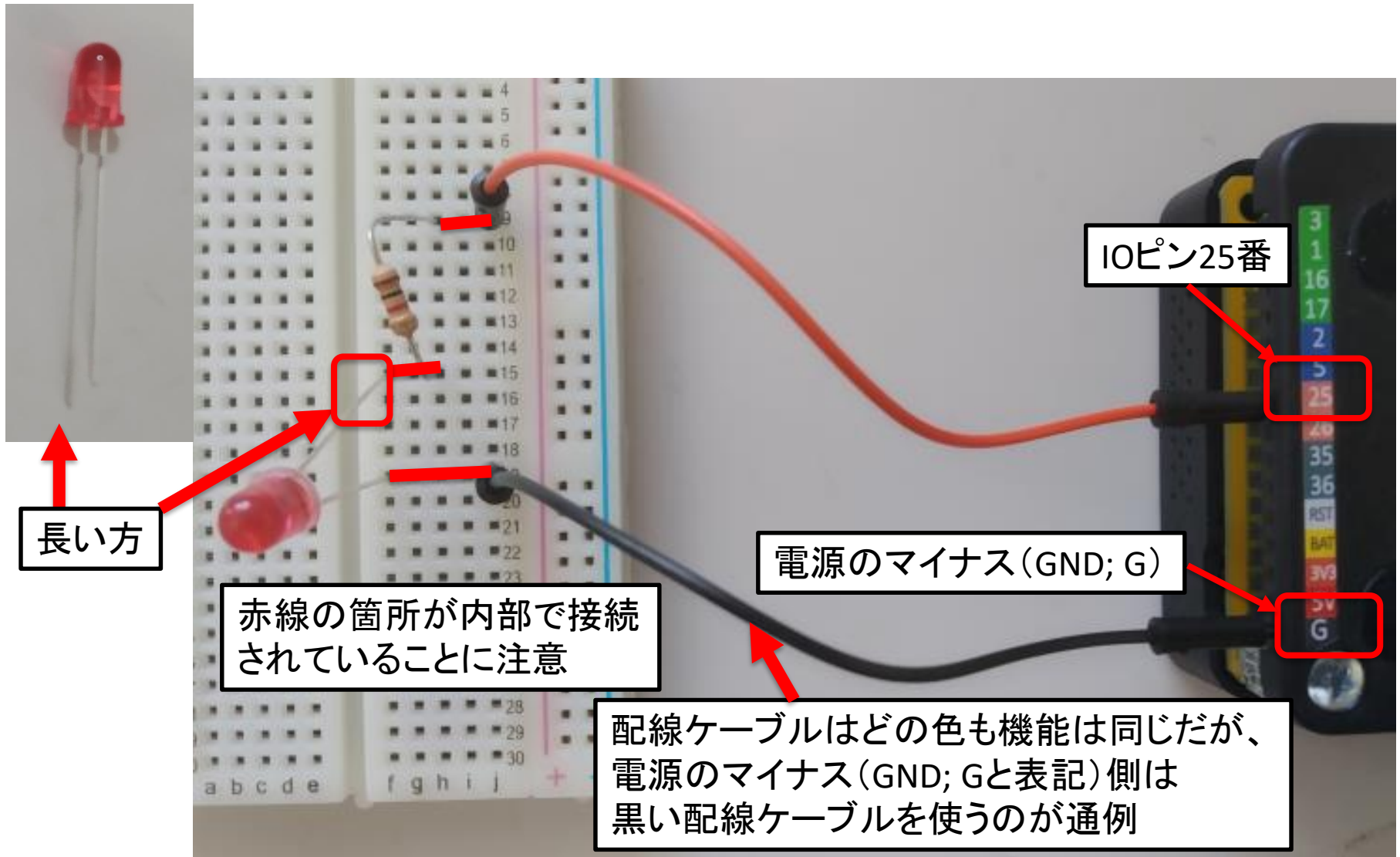
“Advanced” → “EasyIO”内の“map”ブロックで、この変換を行える

E3.ブレッドボードの使い方

部品や配線ケーブルを穴に差し込み、電子回路を作成できる

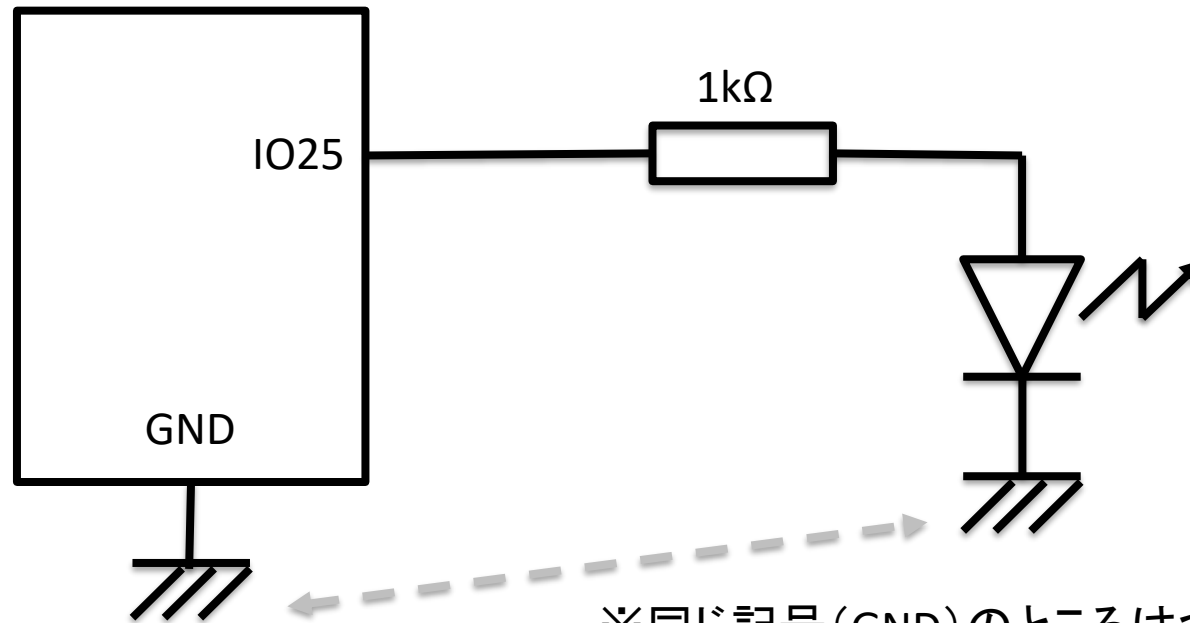


E3.LEDの接続と回路



E3.LEDの接続と回路

2b.でブレッドボード上に作成した回路の回路図



※同じ記号(GND)のところはつながっている

2b.で作成した回路が、上記の回路図となることを理解する

E3.LED点滅(Lチカ)

```
Setup
Loop
  digital write pin 25 value 1
  Wait 0.5 s
  digital write pin 25 value 0
  Wait 0.5 s
```

IOピン25番に"1"を出力=3.3Vが出力され、LEDが点灯

IOピン25番に"0"を出力=0Vが出力され、LEDが消灯

このプログラムを実行させ、LEDを点滅させる。
またwait()を変更してLEDの点滅周期を変える。

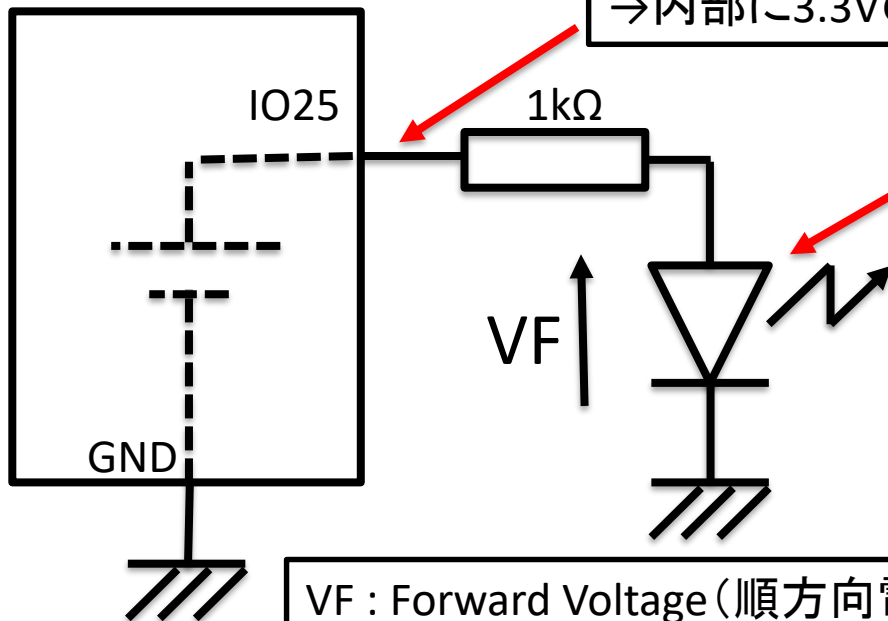
E4.複数のLチカ

直接、以下の動作をするPythonプログラムを記述し、実行させる

- 動作仕様
 - 赤LEDを、抵抗を通してIO25に接続
 - 緑LEDを、抵抗を通してIO26に接続
 - 以下の順序でLEDを点滅させる
 1. 赤のみ点灯(0.5秒間)
 2. 緑のみ点灯(0.5秒間)
 3. 赤と緑を点灯(0.5秒間)
 4. いずれも消灯(0.5秒間)

E5.LEDに流れる電流を求める

“1”のときは、電源電圧の3.3Vとなる
→内部に3.3Vの電圧源(点線)があると考えればよい



V_F : Forward Voltage (順方向電圧)
→この電圧の電圧源と考えればよい

※英文のデータシートだが、得るべき情報は一部なので、慣れれば楽(になるはず)

OptoSupply社OSDR5113A

<http://akizukidenshi.com/catalog/g/gi-00624/>

※データシートはここからダウンロードできる

■Electrical -Optical Characteristics

Item	Symbol	Condition
DC Forward Voltage	V_F	$I_F=20\text{mA}$
DC Reverse Current	I_R	$V_R=5\text{V}$
Domi. Wavelength	λ_D	$I_F=20\text{mA}$
Luminous Intensity	I_v	$I_F=20\text{mA}$
50% Power Angle	$2\theta_{1/2}$	$I_F=20\text{mA}$

LEDのデータシートをダウンロードし、 V_F を読み取り、それをもとにLEDに流れる電流を求める

E6.明るさセンサ (CdS) の接続と回路

※CdS(硫化カドミウム)は明るさに応じて電気抵抗が変わる特性がある

明るさセンサ (CdS)

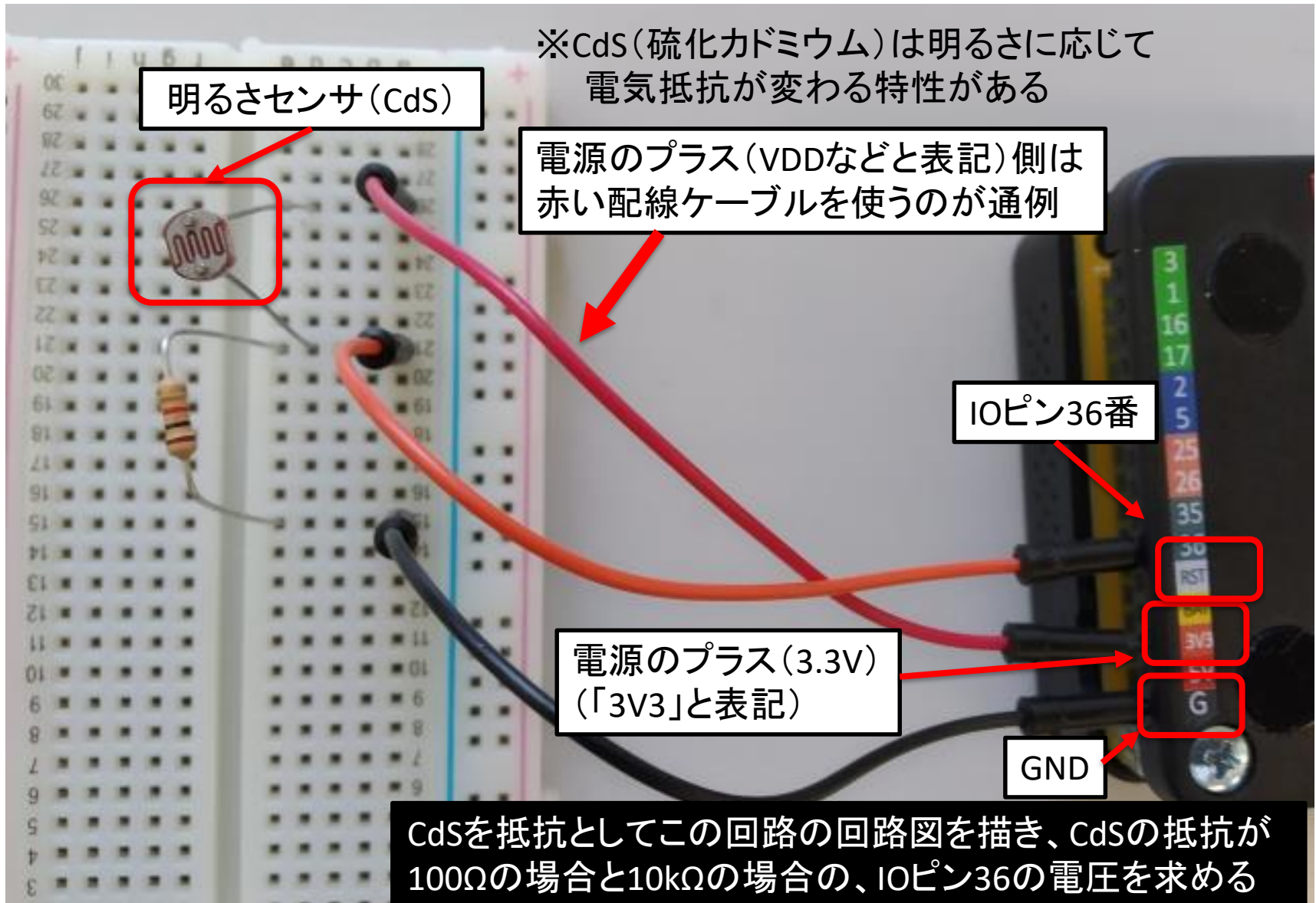
電源のプラス (VDDなどと表記) 側は赤い配線ケーブルを使うのが通例

IOピン36番

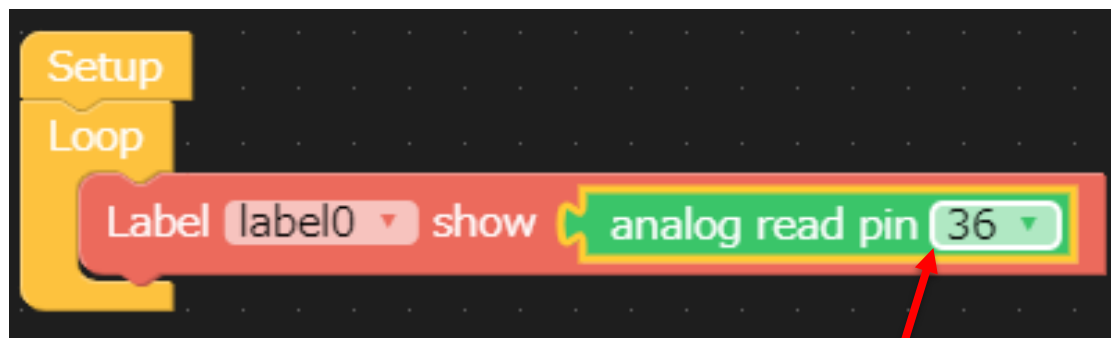
電源のプラス (3.3V) (「3V3」と表記)

GND

CdSを抵抗としてこの回路の回路図を描き、CdSの抵抗が100Ωの場合と10kΩの場合の、IOピン36の電圧を求める



E6.明るさセンサ (CdS) の読み取り

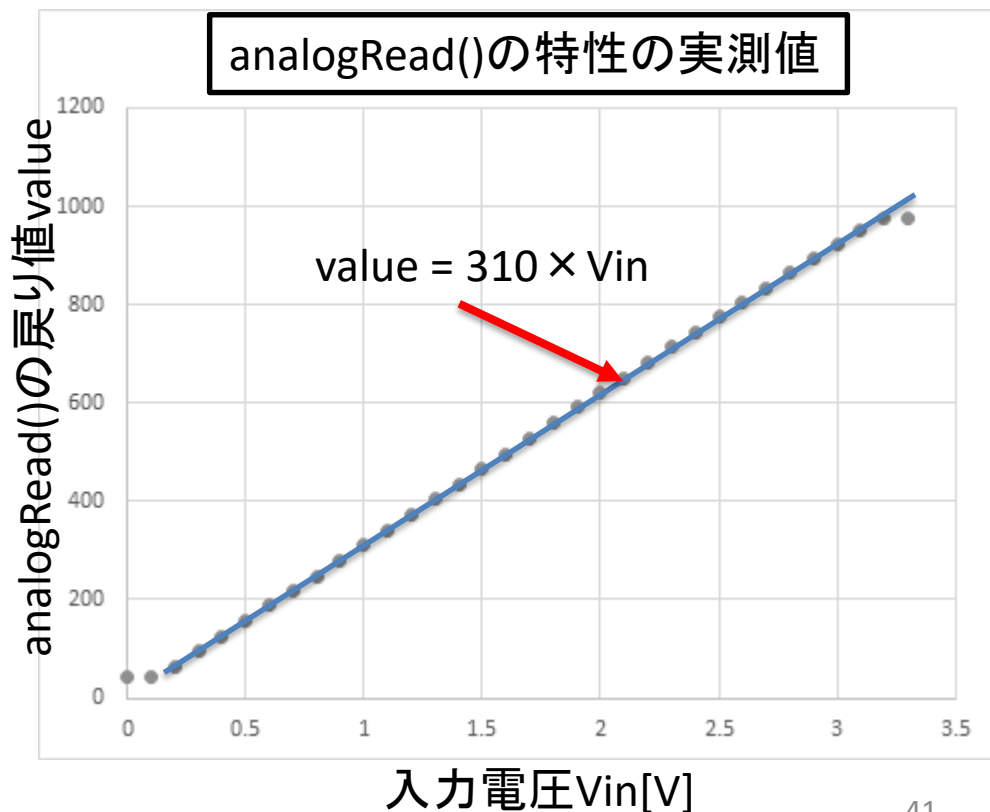


IOピン36番の電圧を読み取り、label0に設定

このプログラムを実行させ、以下の3つの場合で値を読み取り、IOピン36の電圧を求める。
※電圧の計算は次項を参照
(a)手で覆って暗くした場合、(b)部屋の照明下、(c)スマホのライト等の照明を直接照射

E7.明るさセンサ (CdS) の電圧の計算

- “analog read”で取得される値は、加える電圧 [V]に(ほぼ)比例
 - 0V付近と3.3V付近は、やや線形性が崩れるが、ほぼ無視できる
- 今回の回路では、抵抗分圧から関係が求められる



E8.温度センサの接続と回路

- 使用する温度センサ「LM61」のデータシートを型番で検索するなどしてダウンロードする
- データシートから、動作に必要な回路接続を読み取る(TO-92パッケージ)
- 電源、GND、出力(VOUT)をブレッドボードでM5stackに接続



<http://akizukidenshi.com/catalog/g/gi-11160/>

- 電源は5Vを供給
- VOUTはIO35へ

※データシートの読み取りに便利な主な英単語

- power Supply = 電源
- output = 出力
- ground = 接地 (電源のマイナス(negative)側、「G」と表記)
- bottom view = 底面図 (部品を裏側(足側)から見た図)

TEXAS INSTRUMENTS

LM61 2.7-V, SOT-23 or TO-92 Temperature Sensor

1 Features

- Calibrated Linear Scale Factor of 10 mV/°C
- Rated for Full Temperature Range (-30° to 100°C)
- Suitable for Remote Applications
- UL Recognized Component
- ±2°C or ±3°C Accuracy at 25°C (Maximum)
- ±3°C Accuracy for -25°C to 85°C (Maximum)
- ±4°C Accuracy for -30°C to 100°C (Maximum)
- 10 mV/°C Temperature Slope (Maximum)
- 2.7-V to 10-V Power Supply Voltage Range
- 125-µA Current Drain at 25°C (Maximum)
- ±0.8°C Nonlinearity (Maximum)
- 800-Ω Output Impedance (Maximum)

2 Applications

- Cellular Phones
- Computers
- Power Supply Modules
- Battery Management
- FAX Machines
- Printers
- HVAC
- Disk Drives
- Appliances

3 Description

The LM61 device is a precision, integrated-circuit temperature sensor that can sense a -30°C to 100°C temperature range while operating from a single 2.7-V supply. The output voltage of the LM61 is linearly proportional to temperature (10 mV/°C) and has a DC offset of 600 mV. The offset allows reading negative temperatures without the need for a negative supply. The nominal output voltage of the LM61 ranges from 300 mV to 1600 mV for a -30°C to 100°C temperature range. The LM61 is calibrated to provide accuracies of ±2°C at room temperature and ±3°C over the full -25°C to 85°C temperature range.

The linear output of the LM61, 600-mV offset, and factory calibration simplify external circuitry required in a single supply environment where reading negative temperatures is required. Because the quiescent current is less than 125 µA, self-heating is limited to a very low 0.2°C in still air. Shutdown capability for the LM61 is intrinsic because its inherent low power consumption allows it to be powered directly from the output of many logic gates.

Device Information ⁽¹⁾		
PART NUMBER	PACKAGE	BODY SIZE (NOM)
LM61	SOT-23 (3)	1.30 mm × 2.92 mm
	TO-92 (3)	4.30 mm × 4.30 mm

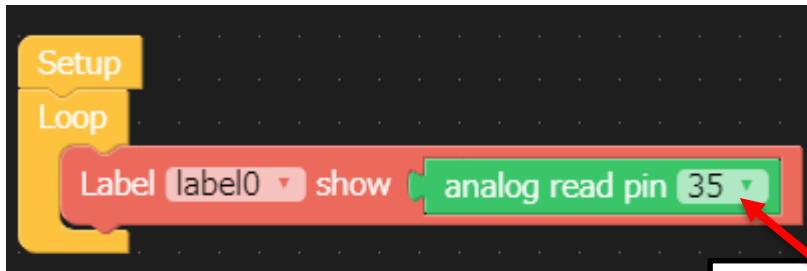
(1) For all available packages, see the orderable addendum at the end of the data sheet.

Key Specifications	
	VALUE
Accuracy at 25°C	±2°C or ±3°C
Accuracy for -25°C to 85°C	±3°C
Accuracy for -30°C to 100°C	±4°C
Temperature slope	10 mV/°C
Power supply voltage	2.7 V to 10 V
Current drain at 25°C	125 µA
Nonlinearity	±0.8°C
Output impedance	800 Ω

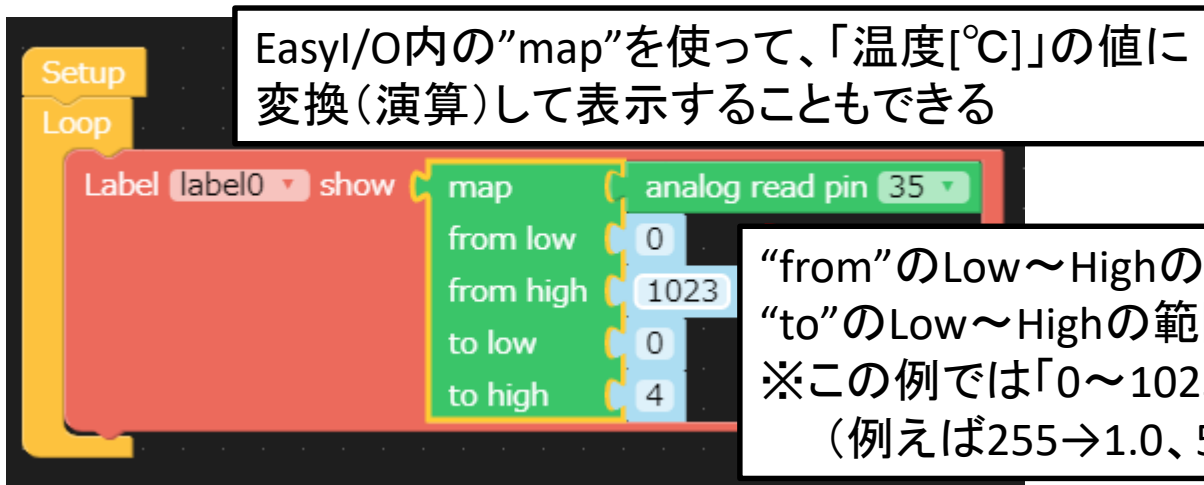
Typical Application

Copyright © 2016, Texas Instruments Incorporated
V₀ = 10 mV/°C × T(°C) + 600 mV

E8. 温度センサの読み取り



IOピン35番の電圧を読み取り、label0に設定



EasyI/O内の“map”を使って、「温度[°C]」の値に変換(演算)して表示することもできる

“from”のLow～Highの範囲の数値を、
“to”のLow～Highの範囲に(線形に)換算する
※この例では「0～1023」を「0～4」に換算
(例えば255→1.0、511→2.0、など)

このプログラムをもとに、温度を[°C]単位で表示する

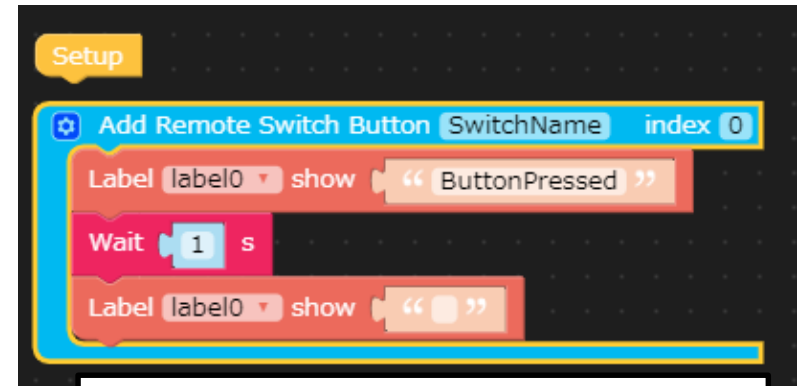
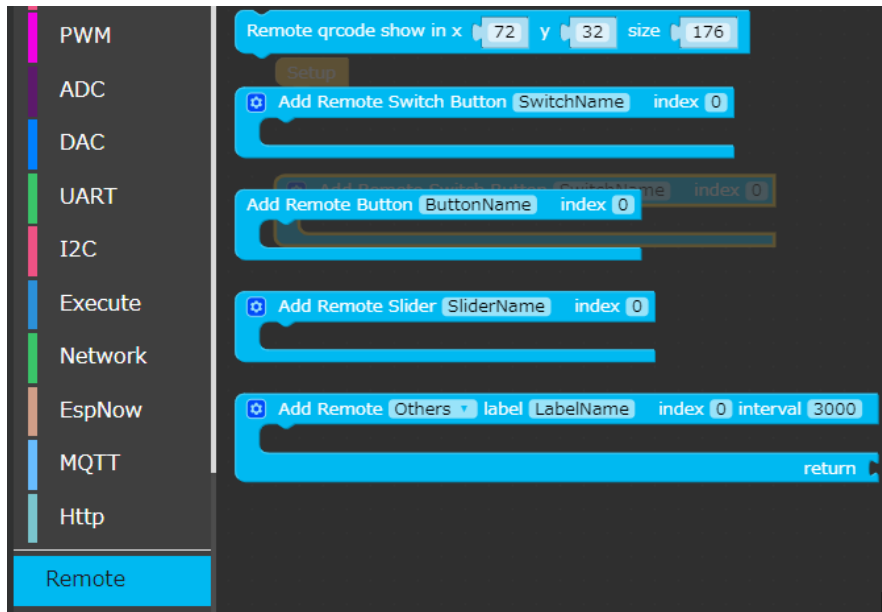
E9.インターネット連携:IoTへ

- PCやスマホだけでなく、「いろいろなもの」がインターネットにつながる世界
=IoT (Internet of Things) → AIの出番



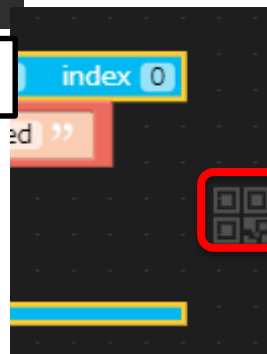
E9.インターネット連携:リモートボタン

- スマホ(ブラウザ)からM5stackを操作してみる



ブラウザに「ボタン」を表示し、それを押したらM5stackで処理を行うプログラム

“Remote”内に、スマホ連携の処理がある



(ちょっと薄くて見にくいですが)「QRコード表示アイコン」でQRコードを表示し、それをスマホ等でスキャンしてブラウザで制御ページを表示

このプログラムを実行し、スマホからM5stackを制御してみる

E9.インターネット連携:リモートラベル

- 逆にM5stackからスマホ(ブラウザ)を制御

スマホ(ブラウザ)側で表示を自動更新する間隔[ms]
(この例では3秒)

return tof0 get distance

This screenshot shows the 'Add Remote' block in Scratch. The 'interval' field is set to 3000 and is circled in red. The 'return' field is set to 'tof0 get distance' and is also circled in red. A callout box points to the interval field with the text 'スマホ(ブラウザ)側で表示を自動更新する間隔[ms] (この例では3秒)'. Another callout box points to the return field with the text 'ToFセンサで取得した距離情報'.

スマホ(ブラウザ)上にラベルを表示し、そこに情報(この例では、ToFセンサで取得した距離)を表示する例

変数の作成

変数の更新(値の取得)

This screenshot shows the Scratch interface. On the left, the 'Variables' menu is open, and the 'Create variable...' button is circled in red. A callout box points to it with the text '変数の作成'. In the center, there is a 'set distance to' block with 'tof0 get distance' selected in the dropdown, circled in red. A callout box points to it with the text '変数の更新(値の取得)'. On the right, the 'return distance' block is circled in red.

変数(Variable)を作成し(この例では"distance"という変数)、それに必要な値を代入して表示などに使用することもできる

このプログラムを参考に、M5stackで取得したセンサ情報をスマホに表示してみる

E10. インターネット連携 (Webスクレイピング)

※E10, E11の内容はEdelWorksの古川光氏の協力で作成されました

- Webページを取得し、その中の情報を抜き出して利用する (スクレイピング)
 - 例として、無料Webサービスから天気情報を取得

<https://openweathermap.org/>



“Sign Up”からアカウントを作成
※16歳以上とライセンス同意をチェック)
※メルマガを登録するか聞かれるが
必要ならチェックしておく
※何に利用するかを回答(任意)

天気情報を取得する都市を検索(例: 金沢)



APIキーを
取得する
(メモしておく)



都市名を選択



E10.インターネット連携（Webスクレイピング）

- 取得したAPI Keyと都市名を使って、以下のURLで天気情報などを取得できる（JSON形式）

`http://api.openweathermap.org/data/2.5/weather?q=[都市名]&appid=[APIキー]`

```
{
  - coord: {
    lon: 138.86,
    lat: 36.56
  },
  - weather: [
    - {
      id: 020,
      main: "Rain",
      desc: "light intensity shower rain",
      icon: "09d"
    }
  ],
  base: "stations",
  - main: {
    temp: 296.15,
    pressure: 1017,
    humidity: 83,
    temp_min: 295,
    temp_max: 297,
  },
  visibility: 10000,
  - wind: {
    speed: 2.1,
    deg: 40
  },
  - clouds: {
    all: 75
  },
  dt: 1569716359,
  - sys: {
    type: 1,
    id: 8017,
    message: 0.0074,
    country: "JP",
    sunrise: 1569703568,
    sunset: 1569746499
  },
  timezone: 32400,
  id: 1857470,
  name: "Kanazawa",
  cod: 200
}
```

ここに現在の天気
書いてある

※[都市名]のところは、kanazawa,jp などが入る
※APIキーは、先ほど記録したAPI Key

このURLへアクセスすると、
このようなデータが返送される
(JSON形式)

※JSON (JavaScript Object Notation)
Webからのデータの取得などでよく使われるデータ形式
見れば、なんとなく意味はわかるはず
詳しくは→<https://www.json.org/json-ja.html>

E10.インターネット連携（Webスクレイピング）

- 例として以下のプログラムで、指定したURLからJSONデータ取得→そこからmain部分を切り出し（パース）→それをlabel0（置いておく）で表示

The image shows a Scratch script for fetching weather data from an API and displaying it. The script is set up to run when button A is pressed. It uses an 'Http Request' block to send a GET request to the URL 'http://api.openweathermap.org/data/2.5/weather?i...'. The response is then processed using 'loads json' and 'Get Data' blocks. The data is then parsed using 'get key' blocks to extract the 'main' value from the 'weather' list. The final result is displayed on label0. The script also includes a 'Fail' block to handle errors.

“GET”を選択（HTTPのmethod）

天気情報を取得するURLを指定（都市名とAPI Keyを指定する）

取得したJSON形式のデータをパースして変数に保持

項目“weather”の0番目（最初に現れるデータ）の“main”の値を取得

このプログラムを参考に、M5stackで取得した天気や気温などの情報を表示してみる

E11. インターネット連携 (IFTTT: 準備1)

- Webサービスどうしを連携するサービス IFTTT
※IFTTT=IF This Then That: 条件で行う動作を定義
- 例として、「M5stackからWebアクセスしてGoogleスプレッドシートに情報を記録する」という動作をさせてみる

<https://ifttt.com/>

Get started with IFTTT

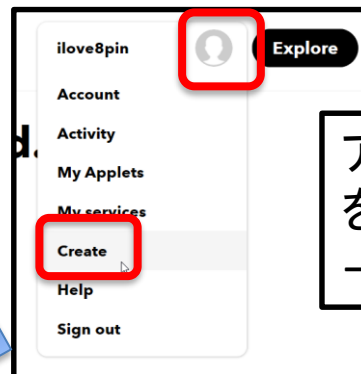
Continue with Google

Continue with Facebook

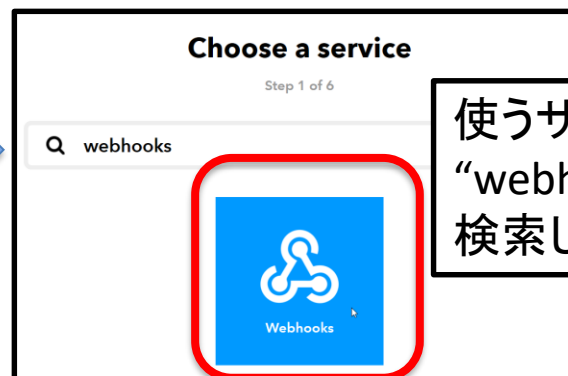
アカウントを作成

If  This Then That

動作を起こす条件を定義する(クリック)



アカウント(顔マーク)をクリック
→"Create"を選択



使うサービスとして
"webhooks"を検索して選択

E11. インターネット連携 (IFTTT: 準備2)

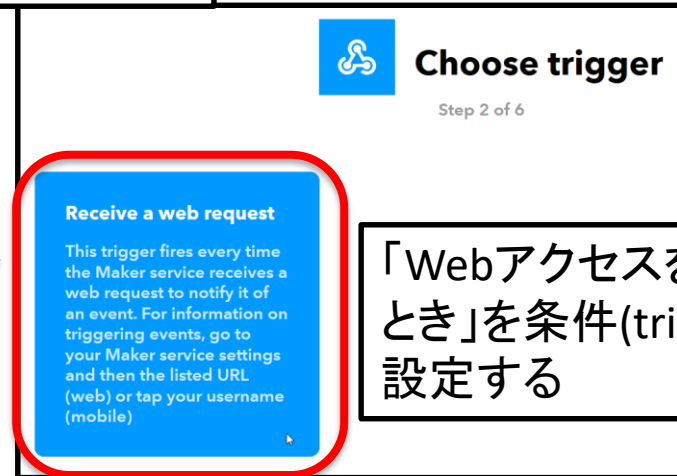
条件「M5stackからWebアクセス」のイベントを設定する



Connect Webhooks
Step 1 of 6

Integrate other services on IFTTT with your DIY projects. You can create Applets that work with any device or app that can make or receive a web request. If you'd like to build your own service and Applets, check out the IFTTT platform.

Connect

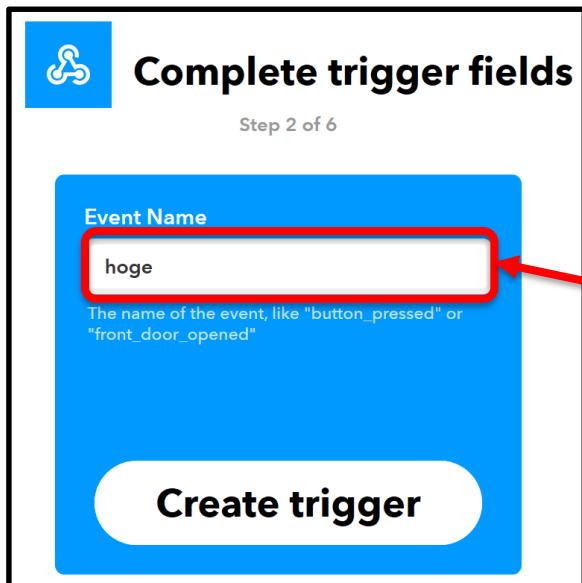


Choose trigger
Step 2 of 6

Receive a web request

This trigger fires every time the Maker service receives a web request to notify it of an event. For information on triggering events, go to your Maker service settings and then the listed URL (web) or tap your username (mobile)

「Webアクセスを受けたとき」を条件(trigger)に設定する



Complete trigger fields
Step 2 of 6

Event Name

hoge

The name of the event, like "button_pressed" or "front_door_opened"

Create trigger

このイベント(Webアクセスを受けたとき)に適切な名前をつけておく(このイベント名はあとで使う)

E11. インターネット連携 (IFTTT: 準備3)

条件 (M5stackからアクセス) が起こった時に行う動作 (GoogleSpreadsheetへ記録) を指定する

If  Then  That

条件が起こった(trigger)ときに行う動作を指定する

Choose action service

Step 3 of 6

Q google sheets



“Google sheets”を検索して指定

接続の確認やアクセス許可などを設定 (OKで進む)
※Actionは“Add row to spreadsheet”を指定

Complete action fields

Step 5 of 6

Spreadsheet name

IFTTT_Maker_Webhooks_Events

Add ingredient

Formatted row

OccurredAt ||| EventName |||
Value1 ||| Value2 ||| Value3

Add ingredient

Drive folder path

EventName

Format: some/folder/path
(defaults to "IFTTT")

Add ingredient

Create action

スプレッドシート名

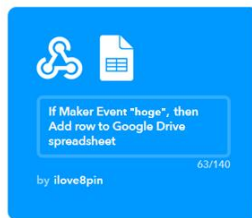
行に書き込む
フォーマット

スプレッドシート
のディレクトリパス

とりあえずすべて
defaultのままでOK

Review and finish

Step 6 of 6



Receive notifications
when this Applet runs



Finish

条件と動作の
設定を完了

E11. インターネット連携 (IFTTT: 準備4)

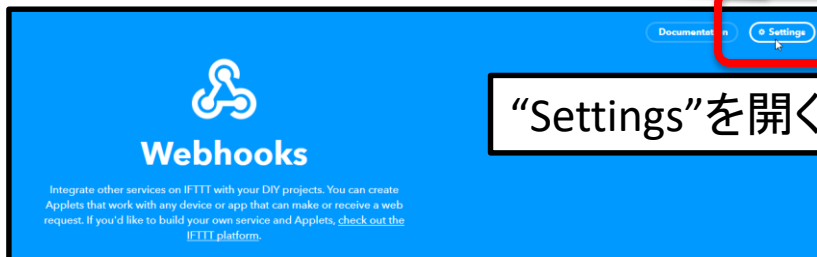
M5stackからWebアクセスするためのAPI Keyを取得する



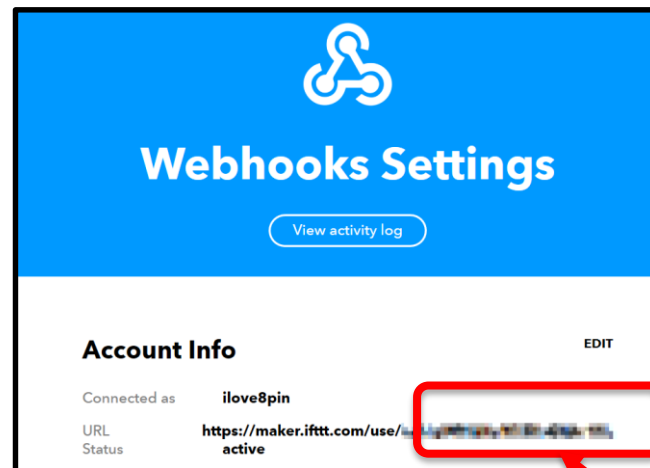
先ほど設定した
サービスを確認



“Webhooks”の
設定を確認する



“Settings”を開く



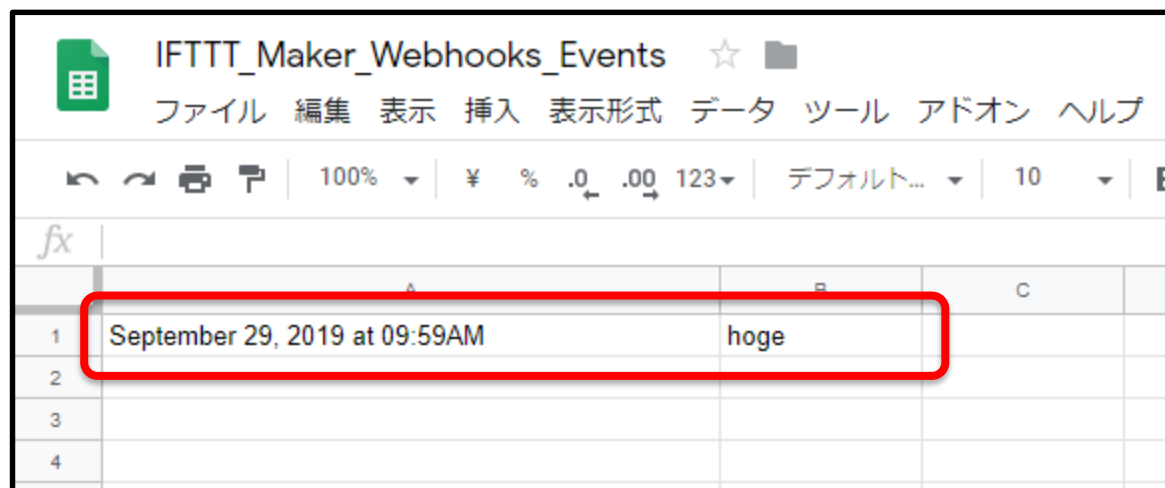
APIキー (“use/” のあとの文字列) を記録しておく

E11. インターネット連携 (IFTTTアクセス)

- 以下のURLで、Googleスプレッドシートに記録されていく(はず)のを確認する

[http://maker.ifttt.com/trigger/\[イベント名\]/with/key/\[APIキー\]](http://maker.ifttt.com/trigger/[イベント名]/with/key/[APIキー])

※[イベント名]のところは「準備2」(p.47)で指定したイベント名
※APIキーは、先ほど記録したAPI Key



The screenshot shows a Google Sheet interface. The title bar reads 'IFTTT_Maker_Webhooks_Events'. Below the title bar is a menu bar with options: 'ファイル', '編集', '表示', '挿入', '表示形式', 'データ', 'ツール', 'アドオン', 'ヘルプ'. Below the menu bar is a toolbar with various icons and settings like '100%', '¥', '%', '.0', '.00', '123', 'デフォルト...', and '10'. The main area is a spreadsheet grid. Row 1, Column A contains the text 'September 29, 2019 at 09:59AM'. Row 1, Column B contains the text 'hoge'. A red rectangle highlights the cells in row 1, columns A and B.

	A	B	C
1	September 29, 2019 at 09:59AM	hoge	
2			
3			
4			

まずはPCやスマホのWebブラウザから、このURLを直接入力し、Googleスプレッドシートに記録されるかを確認する

E11. インターネット連携 (IFTTTアクセス)

関数 "post" を定義 (引数 = "message")

ラベル label0 に "access..." を表示

Method POST

"POST" を選択 (HTTP の method)

URL "http://maker.ifttt.com/trigger/httpblock/with/ke..."

さきほどのURL

new map

key "value1" value message

引数の "message" の内容を、Key="value1" に格納

Success ラベル label0 に "Success" を表示

Fail ラベル label0 に "Fail" を表示

M5stack のボタン A を押したら、"message" = 「A」で関数 post を呼ぶ = M5stack から Web アクセスがおこる

ボタン A が wasPressed

post message "A"

ボタン B が wasPressed

post message "B"

ボタン C が wasPressed

post message "C"

このプログラムを参考に、M5stackからIFTTT経由でGoogleスプレッドシートに記録を試みる