リモッテで楽しく簡単 IoT ！
M5StickC＋エッジ AI、Python プログラム開発！

だれもが
楽しく簡単に。

このセミナーを受講していただくと。。。

- Remotte を一通り使えるようになる
- 開発者としてアプリを新規作成できるようになる
- M5StickC を使った IoT を実現できる
- Intel OpenVINO を活用したエッジ AI を実現できる

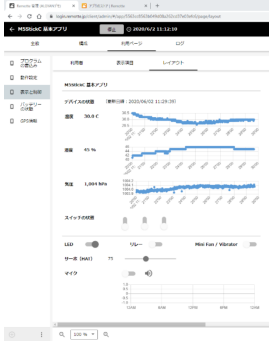
演習が沢山ありますが、実装するコードは数行程度だけで、
ほとんどはクリックと設定操作。
Remotte の操作方法を理解していただければ、色々なアプリを
非常に早く簡単に実現できるようになります！！

演習の内容

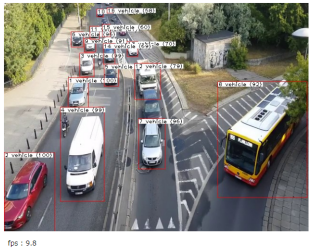
- ステーションが走っている Windows パソコンの「CPU 使用率」を
  センサー値と見立て、それを周期センス、ページ表現
- 「音声ファイルの再生」と「テキストのスピーチ」
- バーコード・ＱＲコードの読み取り
- 音声レベルメーター
- バーコードのアプリを「ＲＰＡ」っぽく使ってみる
- センサー値のキャリブレーション

演習の内容

- MQTT ブローカを準備し、M5StickC の基本アプリを実行



- Intel OpenVINO を使用した顔の検出、顔の再認識、通行カウンター（デモ）など



fps : 9.8

検出したもの

種類	信頼度（%）	座標
vehicle	100	[135,157,177,221]
vehicle	100	[0,314,106,486]
vehicle	99	[159,105,189,155]
vehicle	99	[118,219,210,417]
vehicle	98	[210,128,240,175]

前準備

以下の１から３までが済んでいることを前提に進めます。

１）リモッテ・ステーションのインストール
https://qiita.com/remotte_jp/items/39278287097a7a25432d

２）ステーションの登録
https://qiita.com/remotte_jp/items/94fb73db9fb98e9e10a2

３）簡単なアプリを動かしてみる
https://qiita.com/remotte_jp/items/324f99727c68392b80e5

４）「利用者」として使ってみる
https://qiita.com/remotte_jp/items/7ec79a400429738843d0

５）詳細画面を見てみる
https://qiita.com/remotte_jp/items/a7b5475176eb133af957

ハンズオンの進め方

1．リモッテの仕組みと特徴
2．基本的な操作方法のおさらい
3．管理・開発の機能をもう少し詳しく
★ 4．アプリ開発に関する説明と沢山の演習
5．M5StickC を使った IoT
6．Intel OpenVINO を使ったエッジ AI
7．Tips & まとめ

リモッテの仕組みと特徴

Remotte（リモッテ）が目指していること

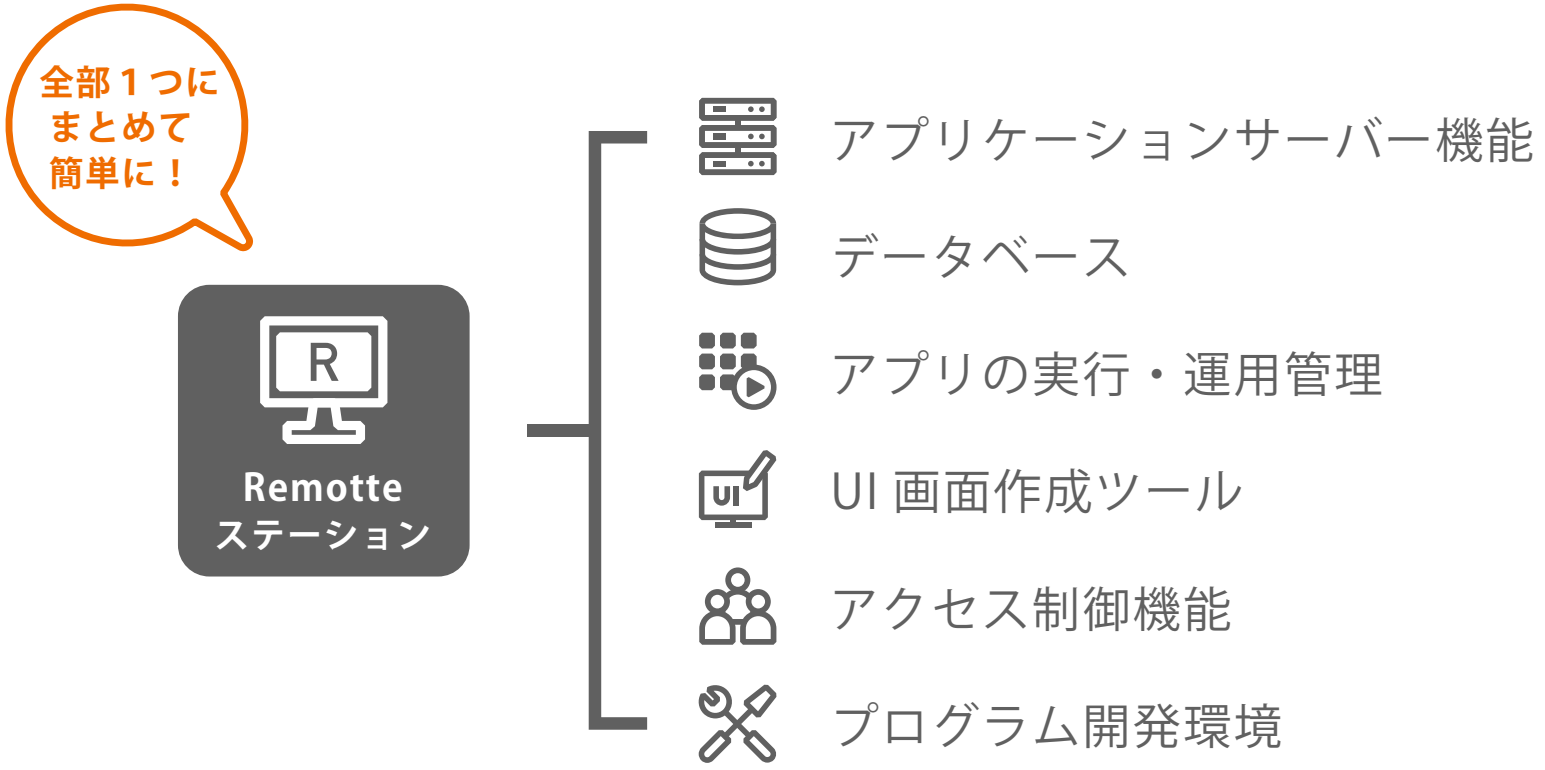
○ IoT / AI をもっと手軽に簡単に
インターネットで買い物ができる人であれば自力で IoT / AI ができるように
考慮に入れなければならない要素が少なくて済むこと
学習コストが小さいこと

○ ちょっとしたことが直ぐできる
今日の午前だけ、今回のイベントで、仕事中だけ、季節の変わり目だけ。。。
有無を知りたい、数を数えたい、時間を知りたい、どれくらいか。。。
測定・通知

何ができる？

オフィス	各種通知、入退出記録、RPA、マーケティング分析、プリンタ機器等の管理
ホーム	監視、防犯、リモコン・遠隔操作、ガーデニング、家族のケア
ショップ	データ収集、各種通知、顧客分析、セキュリティー、デジタル・サイネージ
工場	IoT、計測・ロガー、自動化、品質検査、故障予測、異常検出、人工知能、PoC
農業・畜産	環境測定、生育調査、自動化、病害虫対策、野生動物被害対策
その他	ドローン、BOT、インターバル撮影、SDR（ソフトウェア・ラジオ）

All-In-One プラットフォーム



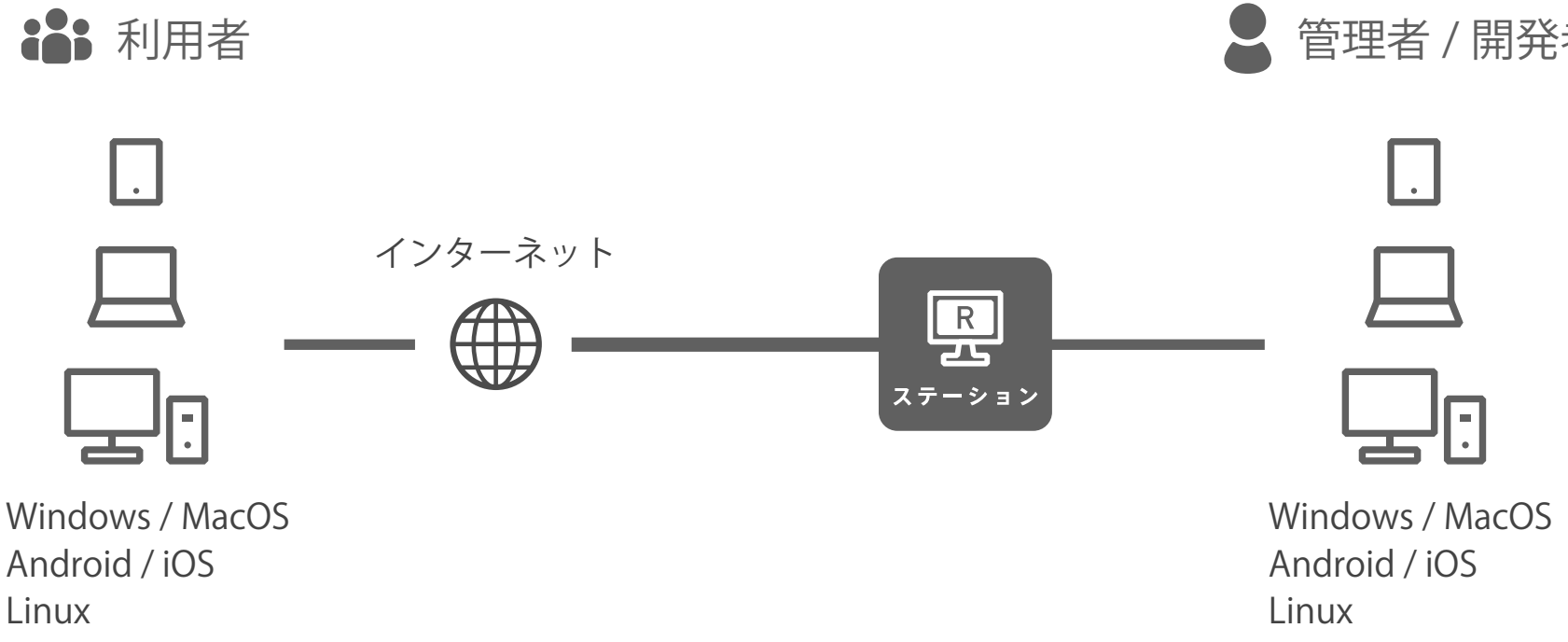
Remotte ステーション

インストール

Windows 10

- 用途に合わせてマシン環境を用意
- 就業時間中・工場稼働中だけ動作すれば良い ➡ 社内の自分の Windows PC で十分

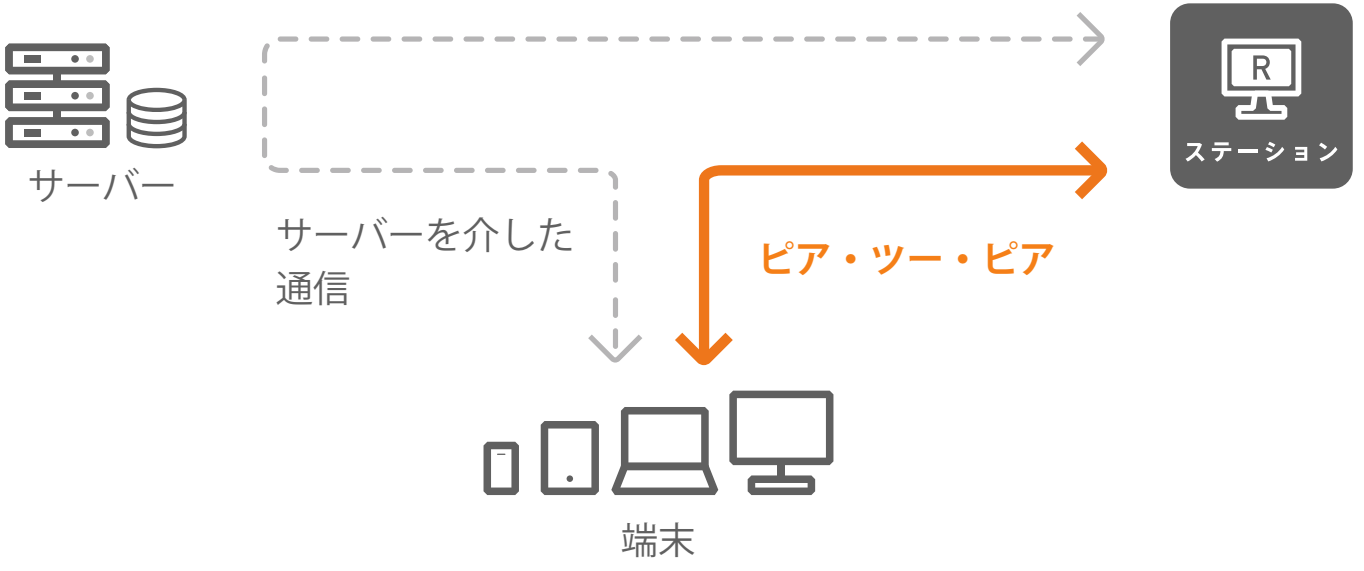
ステーションへはブラウザからアクセス



ステーションは Windows 10 のみ。

管理、開発、利用は OS を問わない。

WebRTC ピア・ツー・ピア接続による直接通信



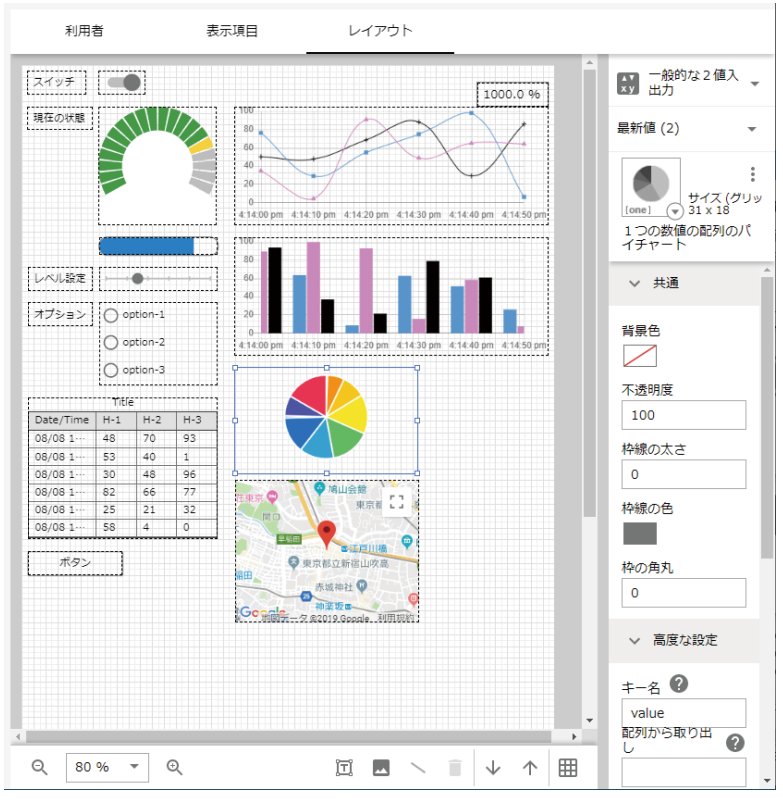
利用可能なブラウザ

Windows：Chrome（推奨）、新しい Edge、Firefox
MacOS　：Safari
Linux　：Chrome（推奨）、Firefox

Android：Chrome（推奨）、Firefox
iOS　：Safari

画面作成はノン・プログラミングで

レイアウト編集



１００種類以上の部品

「Remotte ストア」からすぐ使えるソフトを

アプリ

構成要素

Caffè

基本的な操作方法のおさらい

1）起動方法

2）何かおかしくなった時の対処方法

3）簡単なアプリを動かしてみる

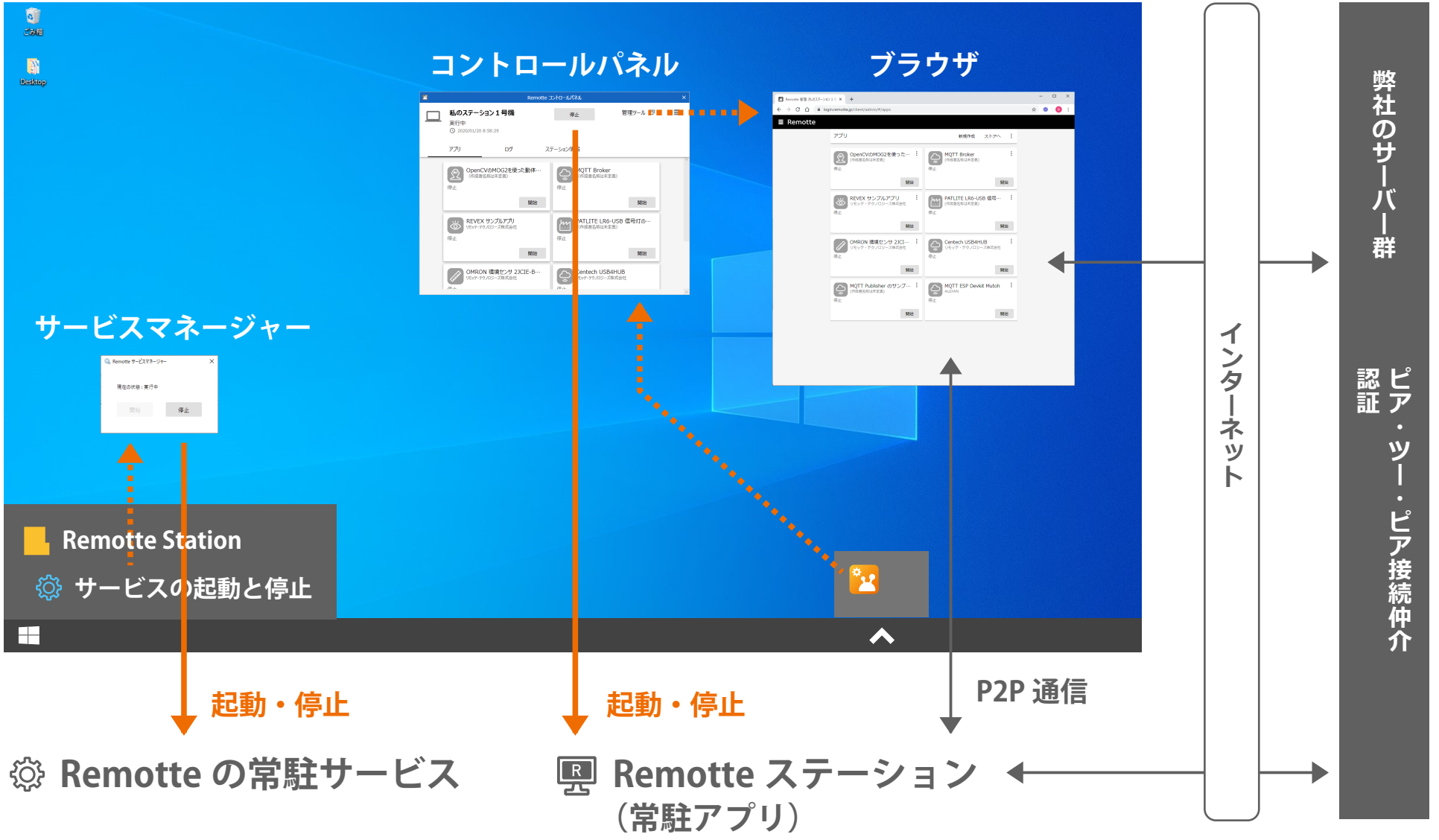
https://qiita.com/remotte_jp/items/324f99727c68392b80e5

4）「利用者」として使ってみる

https://qiita.com/remotte_jp/items/7ec79a400429738843d0

5）詳細画面を見てみる

https://qiita.com/remotte_jp/items/a7b5475176eb133af957


コントロールパネル
ブラウザ
サービスマネージャー
Remotte Station
サービスの起動と停止
起動・停止
起動・停止
P2P 通信
Remotte の常駐サービス
Remotte ステーション
（常駐アプリ）
インターネット
弊社のサーバー群
ピア・ツー・ピア接続仲介
認証

何かおかしくなったら。。。

1）ステーションの再起動

コントロールパネルで
「停止」を押してから
再度「起動」を押す

2）サービスの再起動　＋　ステーションの再起動

サービスマネージャーで
「停止」を押してから
再度「開始」を押す

再度
ステーションを
起動

Caffè

管理・開発の機能をもう少し詳しく

1）ユーザーガイドの見方

https://www.remotte.jp/ja/user_guide

2）コントロールパネルのオプション設定

3）Android や iOS など他のデバイスを使って管理

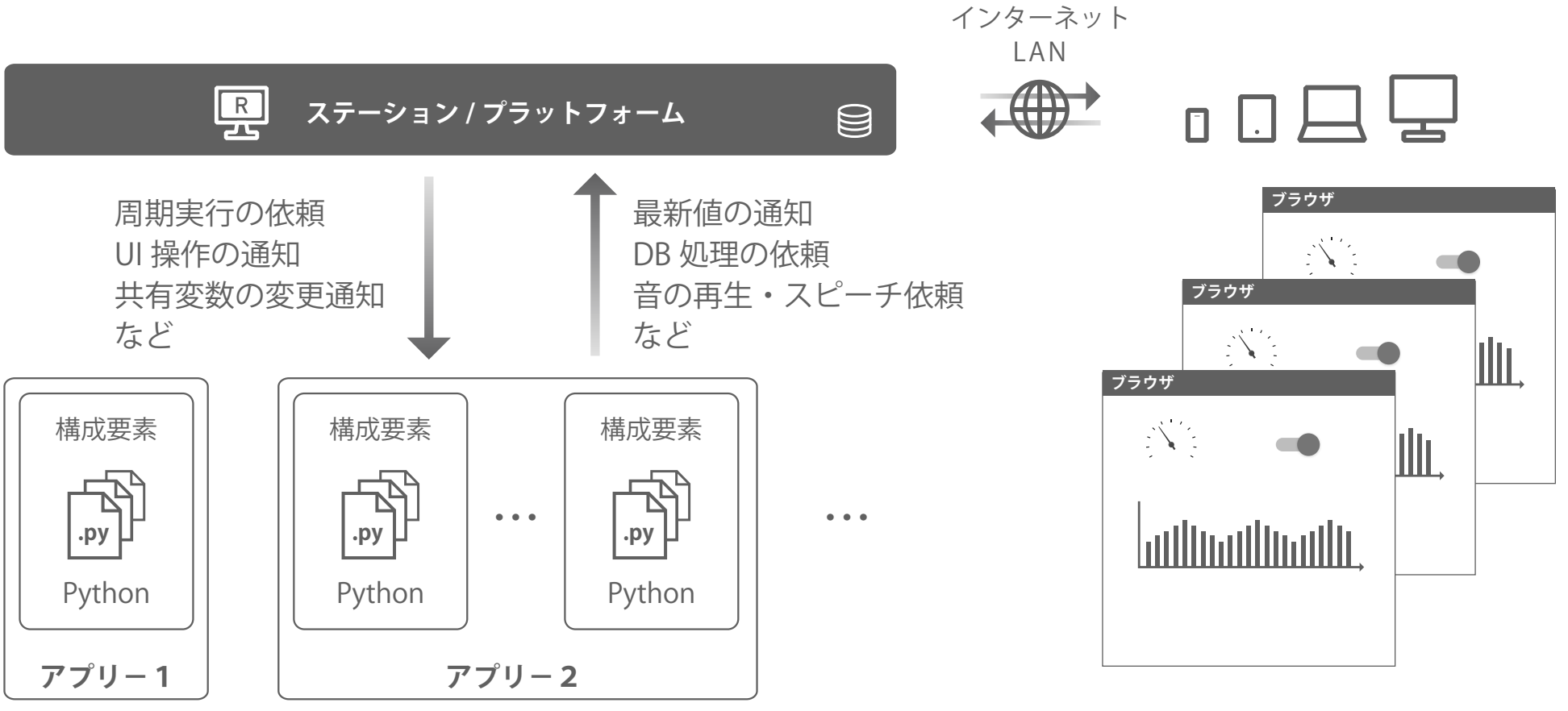
4）管理ツールの各種機能

5）開発者になると何が出来る？

https://www.remotte.jp/ja/user_guide/devel/register

Caffè

アプリ実行の仕組み



複数のアプリを同時に実行できる

アプリを実行しておくだけならインターネット接続は不要！

構成要素のプログラミング

Remotte における構成要素は Python でいうところの「パッケージ」

__init__(self, sys, opt, log) ―― **プラットフォームが最初に呼び出す関数**

sys Python プログラムが呼び出すプラットフォーム側の関数が含まれるクラス

opt Python プログラムに対するオプション設定（プログラム変数）の値を含むクラス

log Python プログラムがログを出力するためのロガークラス

ステーションが呼び出す関数例

__init__()
sense()
control()
new_video_frame()
new_audio_frame()

ステーションの機能を呼び出す / 参照する例

sys . set_value()
. get_last_value()
. kill_me()
. set_video_frame()
. set_audio_frame()

opt . オプション設定値
. __video__
. __audio__

log . debug()
. info()
. warning()
. error()
. critical()

構成要素の種類

一般データ系

オン・オフ
１つの数値
２つの数値
３つの数値
テキスト
画像
（および上記の配列）

メディア系

標準（UVC, USB)
ネットワーク
ブラウザ
デスクトップ
カスタム

解析系

ビデオ解析
音声解析

カスタム系

カスタム

「構成要素」のことを「 **IOC**（**I**nput **O**utput **C**omponent）」とも呼びます。

役割り

～ センス

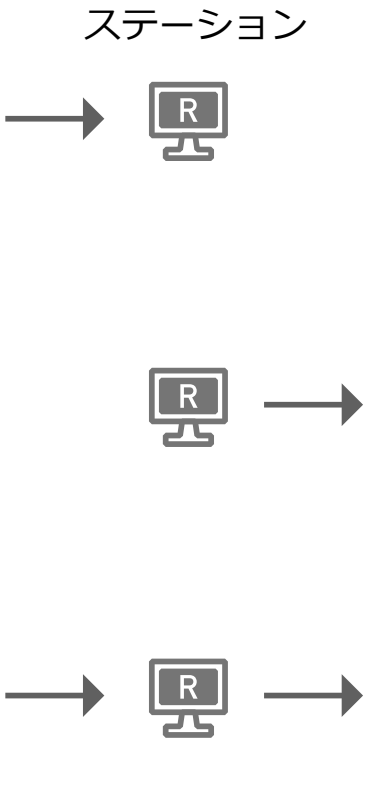
ステーションが **外部から** 何らかのデータを取得

制御

ステーションが **外部に** 何らかのデータを送信

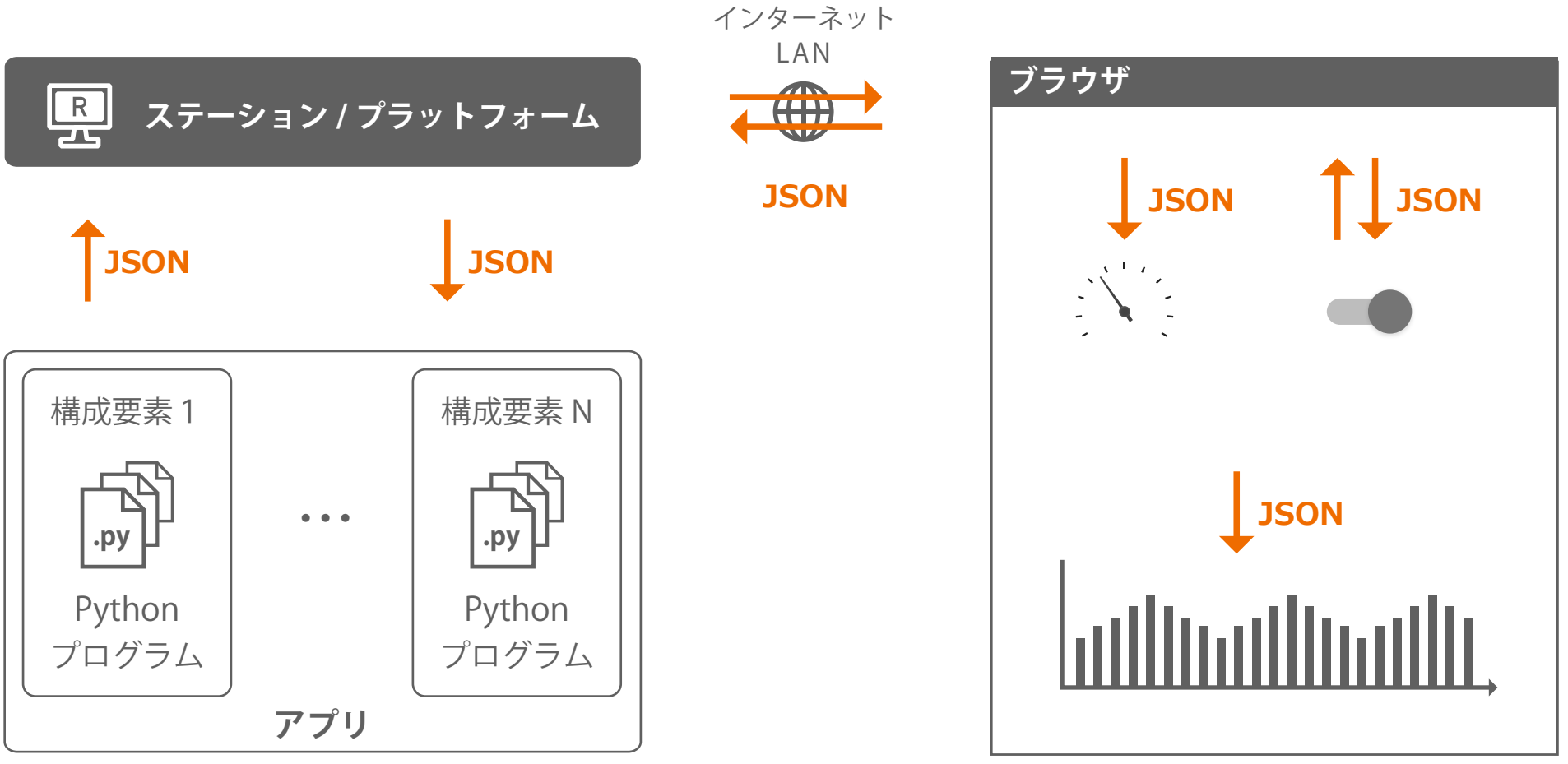
▲▼ 入出力

センスと制御の **両方** の性質

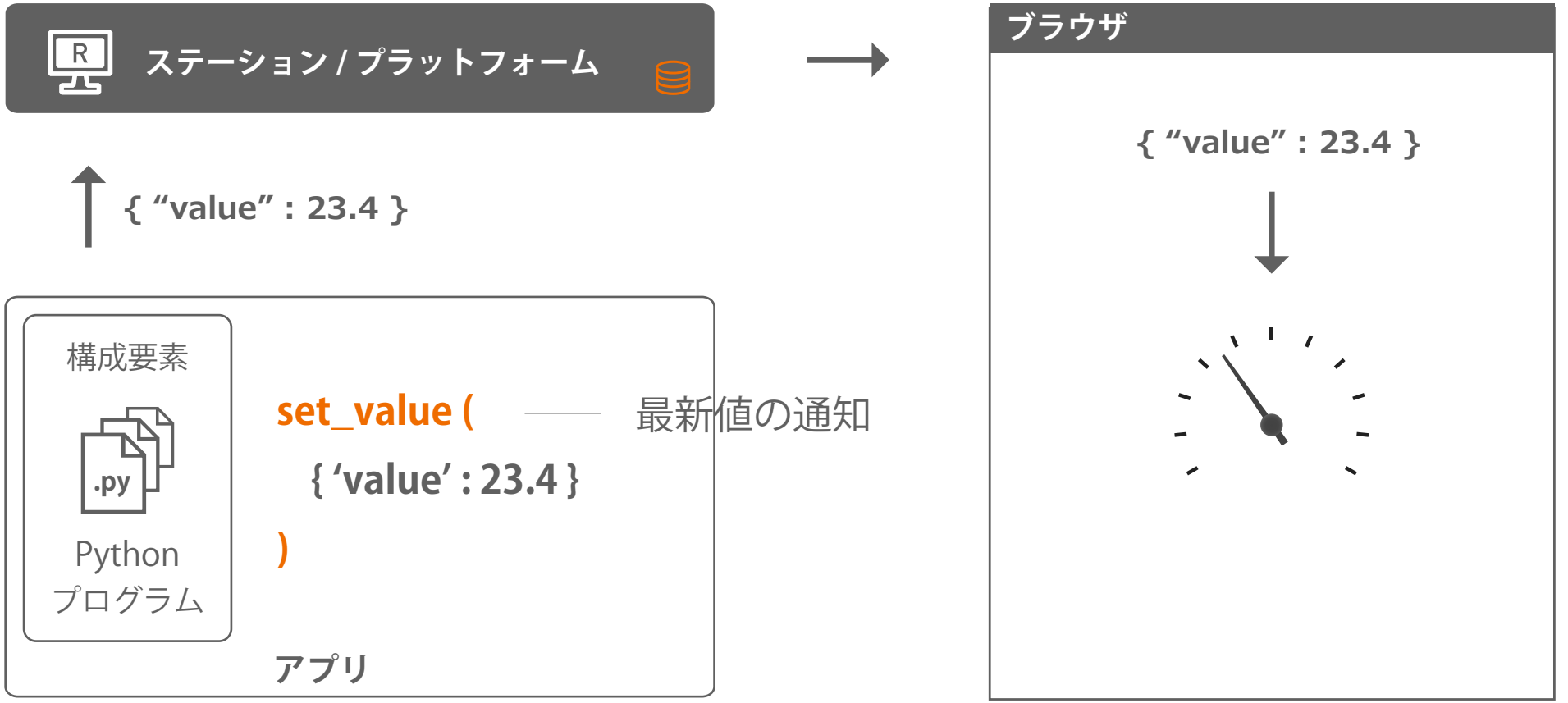


	カテゴリー	役割り	構成要素タイプ	例
ox	オン・オフ	センス	オン・オフセンス	ドアセンサーの開閉
		制御	オン・オフ制御	リセットボタン
		入出力	オン・オフ入出力	スピーカーのミュート
one	１つの数値		１つの数値センス	温度、湿度センサー
			１つの数値制御	各種の調整つまみ
			１つの数値入出力	音量調節
xy	２つの数値		２つの数値センス	GPS データ
			２つの数値制御	範囲設定
			２つの数値入出力	左右バランス
xyz	３つの数値		３つの数値センス	３次元加速度
			３つの数値制御	色の設定
			３つの数値入出力	年月日や時分秒
text	テキスト		テキストセンス	バーコード読み取り
			テキスト制御	音声合成
			テキスト入出力	メッセージング
	画像		画像センス	物体認識

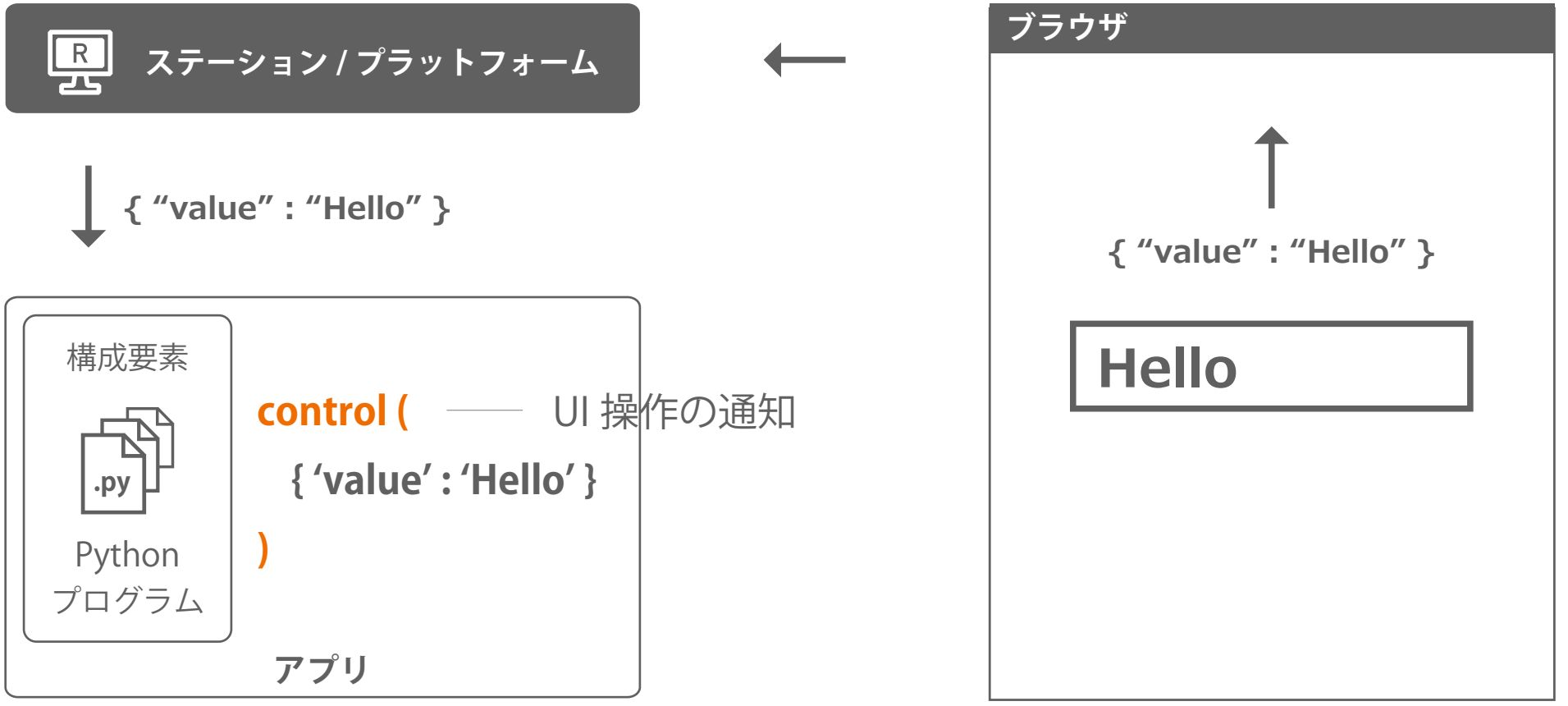
データは JSON 形式で



センス系の場合


R
ステーション / プラットフォーム
{ “value” : 23.4 }
構成要素
.py
Python
プログラム
set_value (
{ ‘value’ : 23.4 }
)
最新値の通知
アプリ
ブラウザ
{ “value” : 23.4 }

制御系の場合


ステーション / プラットフォーム
{ “value” : “Hello” }
構成要素
.py
Python
プログラム
control (
{ ‘value’ : ‘Hello’ }
)
UI 操作の通知
アプリ
ブラウザ
{ “value” : “Hello” }
Hello

データの例

オン・オフ　　{ “value” : True }
１つの数値　　{ “value” : 12.3 }
２つの数値　　{ “value” : [26.3, 68] }
３つの数値　　{ “value” : [-3.5712, 8.3512, 29.3451] }
テキスト　　{ “value” : “Hello” }
画像　　{ “value” : “base64 化された JPEG や PNG” }

配列化したデータも扱える

分布やスペクトラムデータを表す　　{ “value” : [48.1, 27.3, 70.9, 15.2, 46.6, ・・・ , 82.8] }

キー名について、デフォルトが “value” なだけであって、好きに変更して良い

ステータスを表す　　{ “status” : “OK” }

１つのオブジェクトに複数のキーを盛り込んでも良い

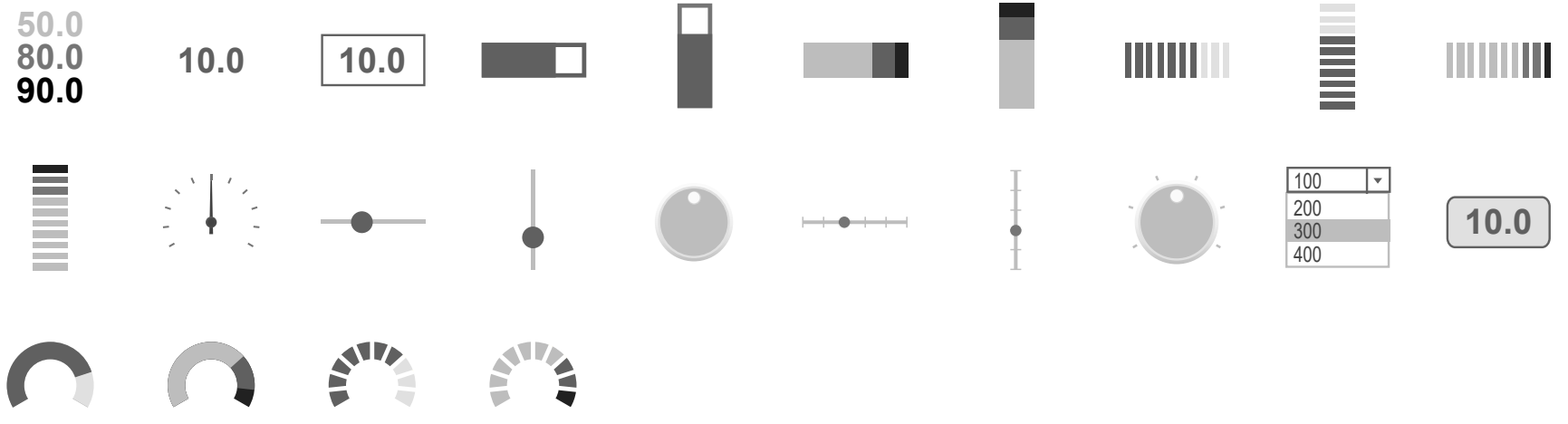
環境センサー付きの開閉センサー　　{ “door” : True, “Temperature” : 26.3, “Humidity” : 68 }

画面で使う部品（HMI：Human Machine Interface）

１００種類以上の部品が用意されていて、カスタマイズが可能

オン・オフの最新値用

１つの数値の最新値用



２つの数値の最新値用

10, 20

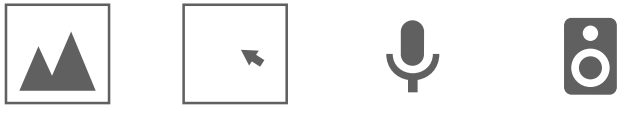
３つの数値の最新値用

10,20,30

テキストの最新値用

Hello

メディアの最新値用



各種配列の最新値用

○×
○
×
○

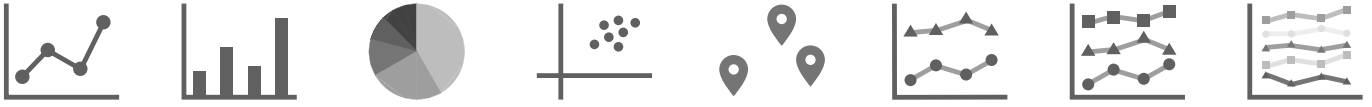
one
100
101
102

x	y
1	10
2	25
3	80

x	y	z
1	10	32
2	25	55
3	80	96

text
Line 1
Line 2
Line 3

custom
{ data 1 }
{ data 2 }
{ data 3 }



各種の履歴用

T	○×
t	○
t	×
t	○

T	one
t	100
t	101
t	102

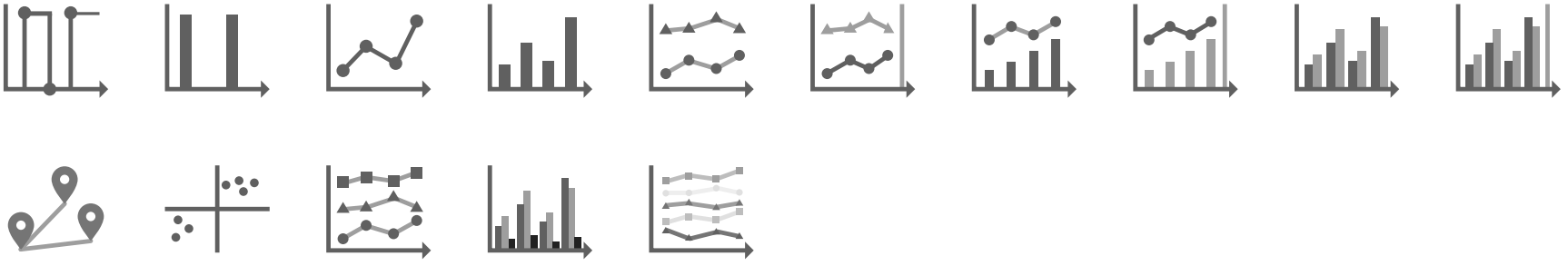
T	x	y
t	1	10
t	2	25
t	3	80

T	x	y	z
t	1	10	32
t	2	25	55
t	3	80	96

T	text
t	Line 1
t	Line 2
t	Line 3

T	custom
t	{ data 1 }
t	{ data 2 }
t	{ data 3 }

T	?	?	?
t	1	aa	T
t	2	bb	F
t	3	cc	T



まずは１つ作ってみよう！！（センス系）

ステーションが走っている Windows パソコンの「CPU 使用率」を
センサー値と見立て、それを周期センス、ページ表現する

Qiita の投稿

[Remotte 開発] まずは作ってみよう！！（演習 1 ）

https://qiita.com/remotte_jp/items/2c7f593c708d4ae04bcc

演習のまとめ

アプリ作成の基本的な流れ

アプリの新規作成

構成要素の追加

- インストールスクリプトの記述と実行
- Python コードの記述（ set_value 関数によって最新値を通知する）
- 構成要素の動作設定

画面の作成

再利用のために保存

センスする方法は主に３つ

１．「プログラム」タブにてプラットフォームに定期センスを依頼する

```
class InputSense(object):
    def sense(self):
        ・・・
```

２．MQTT やシリアル通信などで非同期に入力を得る

３．自分でスレッドを起動して好きにする

```
class InputSense(object):
    def run(self):
        ・・・

    def __init__(self, sys, opt, log):
        thread = threading.Thread(target=self.run)
        thread.start()
```

次に制御系のアプリを作ってみよう！！

「音声ファイルの再生」と「テキストのスピーチ」を題材にする

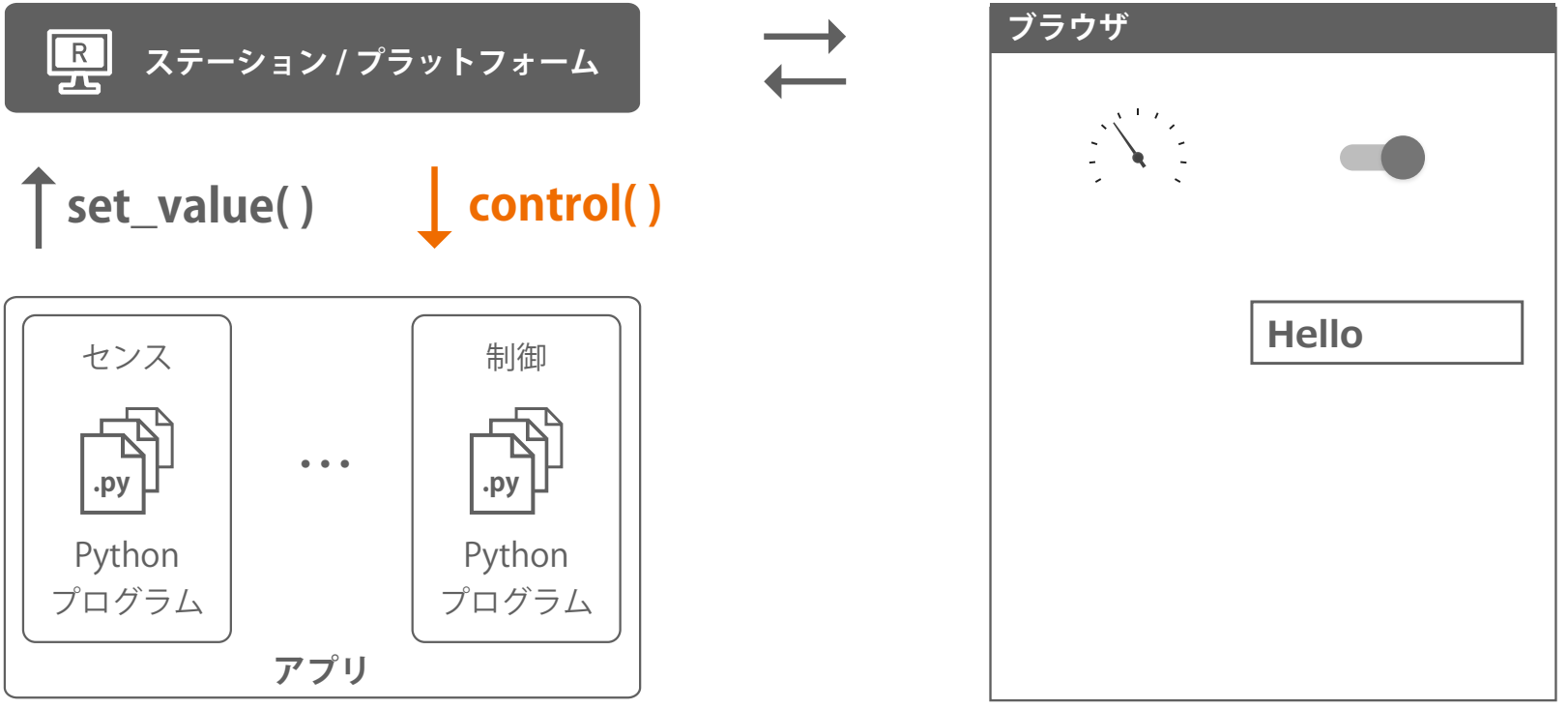
Qiita の投稿

[Remotte 開発] 制御系のアプリ（演習２）

https://qiita.com/remotte_jp/items/ce77c0f5b796ccd818a8

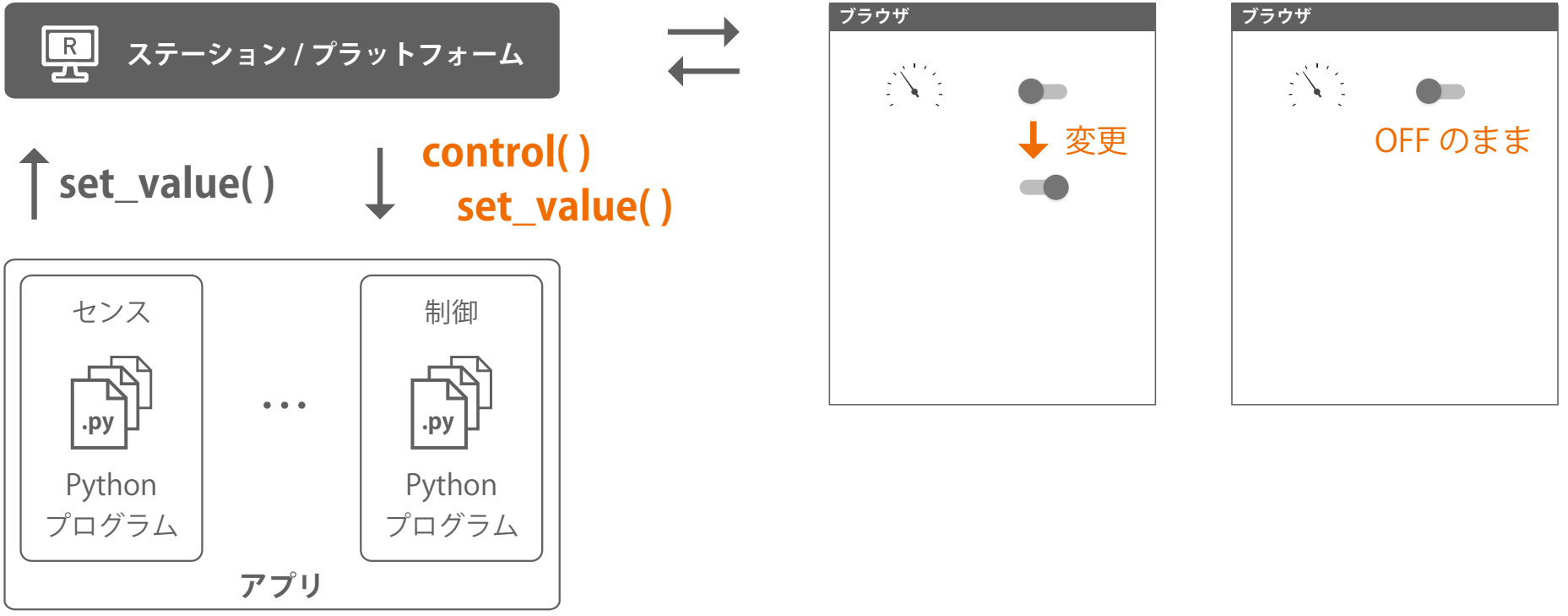
演習のまとめ

利用ページで操作・入力した内容は、Python コードの control 関数に伝えられる



複数のブラウザでアクセスしている場合は？

control 関数の中で set_value しないと、他の人に伝わらない



Caffè

メディア系の構成要素の種類と設定

ビデオ解析、音声解析の演習に進む前に、メディアのソースの構成方法の確認

カテゴリー
標準（UVC, USB）
URL（ネットワーク・ファイル）
ブラウザ
デスクトップ
カスタム

X

役割り
ビデオ
音声
ビデオと音声

種類のメディア

解析系の構成要素の種類と設定

カテゴリー
ビデオ解析
音声解析

X

役割り
値出力のみ
メディアのみ出力
値とメディアの両方を出力

互換タイプ

出力する値（set_value する内容）が下記のいずれか 1 種類であれば、設定しておくことによって HMI 選択が楽になる（通常はデフォルト「無し」のまま）。

オン・オフセンス
一般的な 1 値センス
一般的な 2 値センス
一般的な 3 値センス
テキストセンス

ビデオ解析の演習

「バーコード・ＱＲコードの読み取り」を題材にする

Qiita の投稿

[Remotte 開発] ビデオ解析系のアプリ（演習３）

https://qiita.com/remotte_jp/items/6f379b1f32f5c78f61ee

演習のまとめ

- ビデオ解析に使うソースを、他の構成要素から選択する
- ソースの解像度やフレームレートを設定する
- ソースの属性は、__init__( ) 関数の opt 引数にて与えられる
- new_video_frame( ) 関数に解析すべき画像が周期的に伝えられる
- 解析後の画像は set_video_frame( ) にてプラットフォームに渡す
- 解析系の構成要素は __init__( ) 関数の中で print( ) してもコンソールに出力されない（デバッグ時に注意）

音声解析の演習

「レベルメーター」を題材にする

Qiita の投稿

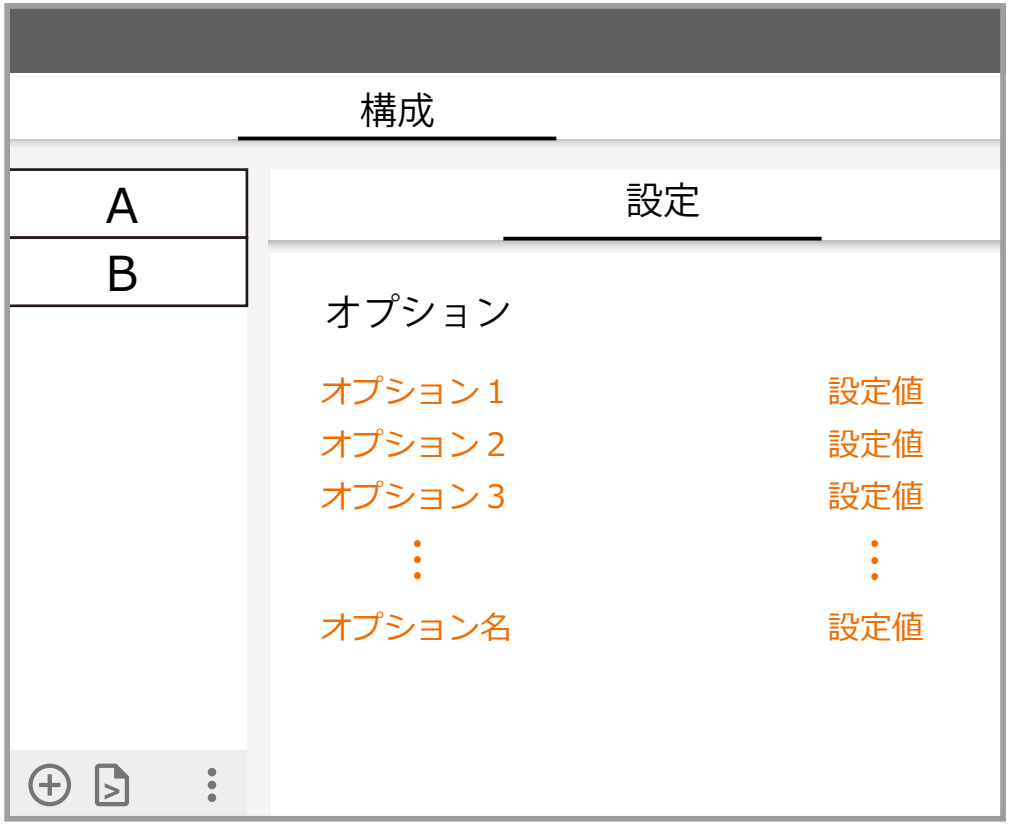
[Remotte 開発] 音声解析系のアプリ（演習４）

https://qiita.com/remotte_jp/items/f759252cafa6dccc80c1

演習のまとめ

- 音声解析に使うソースを、他の構成要素から選択する
- 音声データは１６ビット、２チャンネル、サンプリングレートは４８kHz
- new_audio_frame( ) 関数に解析すべき音声が周期的に伝えられる
- ２０ミリ秒毎に呼び出され、振幅値がチャンネル毎に９６０個格納される
- print(type(audio_frame), len(audio_frame), type(audio_frame[0]))
  <class 'numpy.ndarray'> 1920 <class 'numpy.int16'>
- 解析後の音声は set_audio_frame( ) にてプラットフォームに渡す
- 解析系の構成要素は __init__( ) 関数の中で print( ) してもコンソールに出力されない（デバッグ時に注意）

構成要素のオプション設定

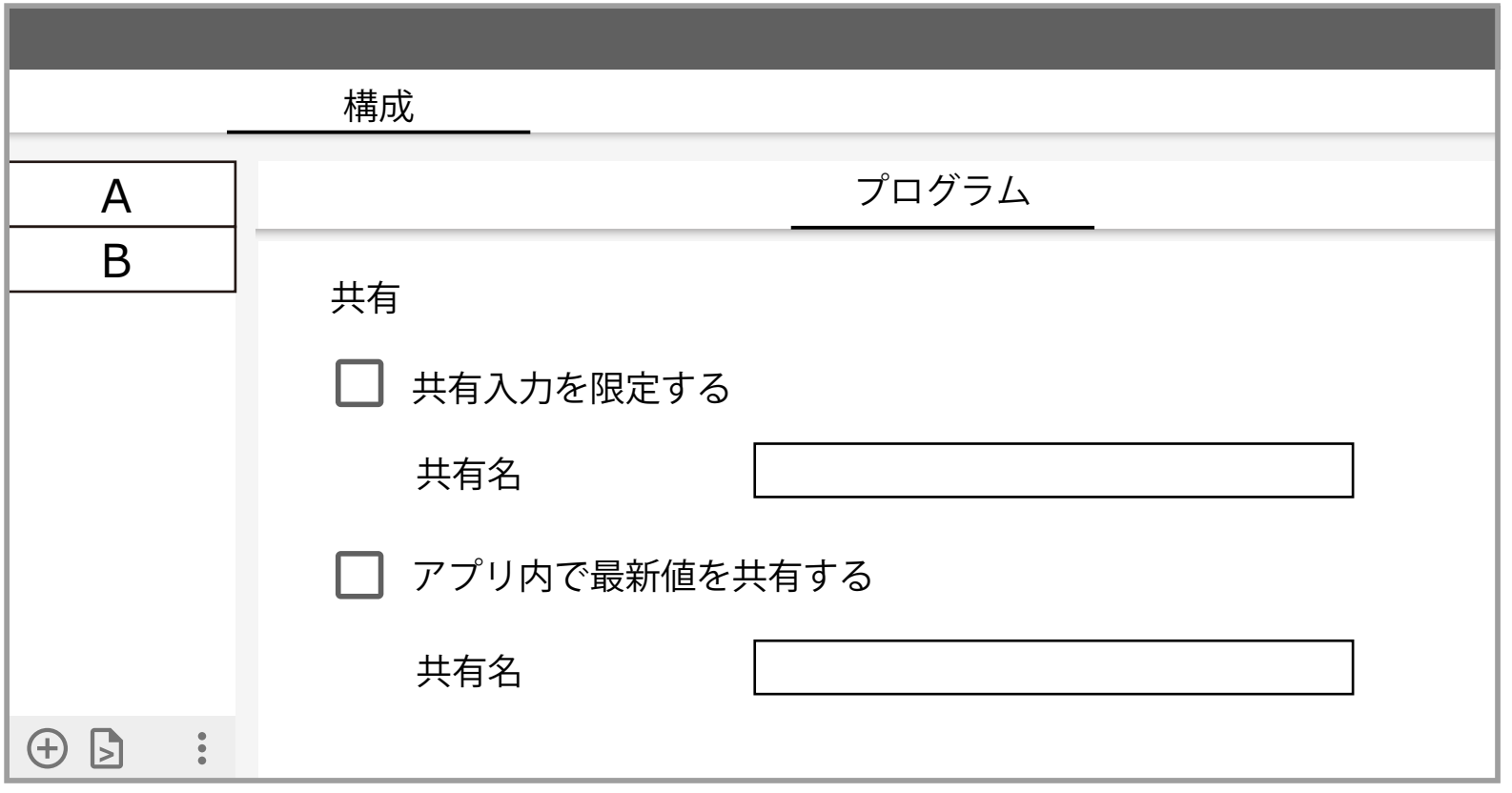


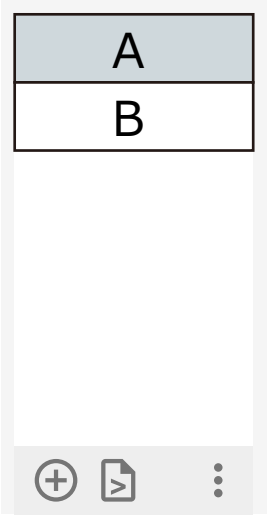
ブール
数値
数値 2 つの配列
数値 3 つの配列
数値 4 つの配列
パーセント値
リスト選択
文字列
テキスト
RGB 色
ラベル

Python
プログラム

__init__(self, sys, opt, log)

構成要素間のデータのやり取り





共有

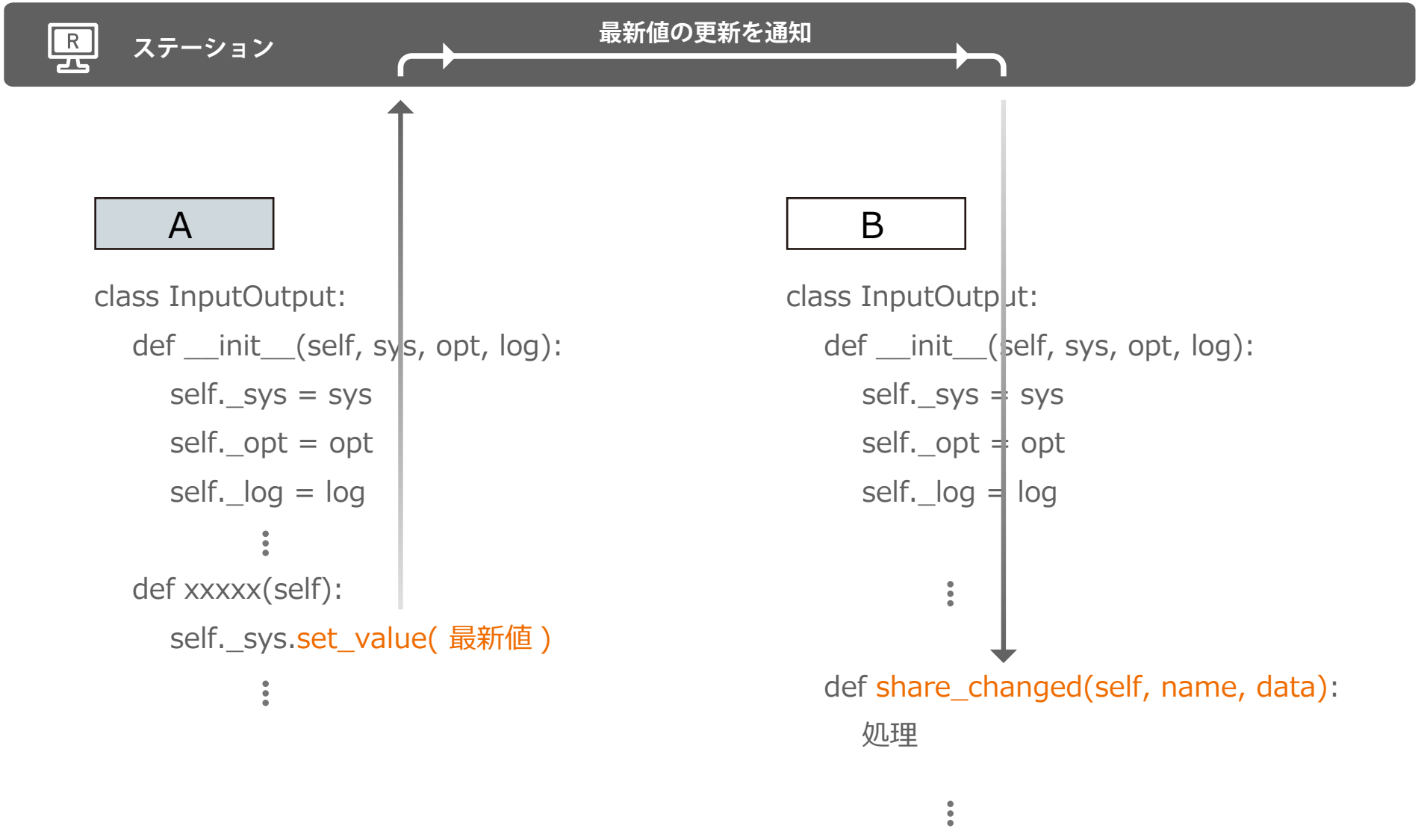
☐ 共有入力を限定する

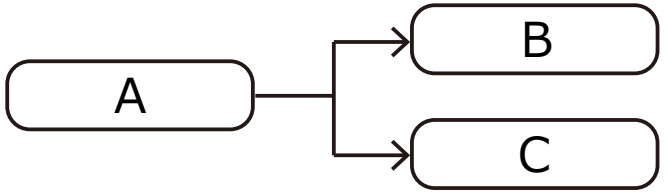
☑ アプリ内で最新値を共有する

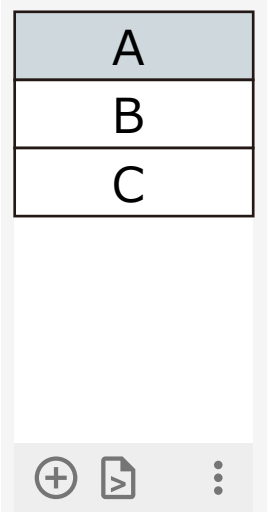
共有名 [ ]　※ 共有名はオプションで、空でも良い

A で set_value (data) されると、

B の share_changed (self, name, data) が呼ばれる。


ステーション
最新値の更新を通知
A
class InputOutput:
def __init__(self, sys, opt, log):
self._sys = sys
self._opt = opt
self._log = log
def xxxxx(self):
self._sys.set_value(最新値)
B
class InputOutput:
def __init__(self, sys, opt, log):
self._sys = sys
self._opt = opt
self._log = log
def share_changed(self, name, data):
処理





共有

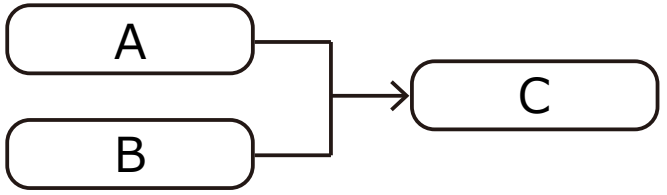
☐ 共有入力を限定する

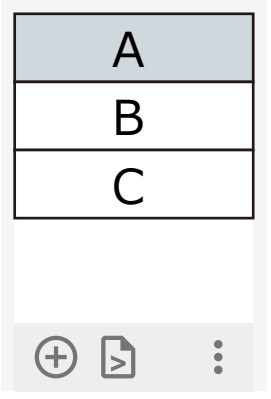
☑ アプリ内で最新値を共有する

共有名

A で set_value (data) されると、

B および C の share_changed (self, name, data) が呼ばれる。



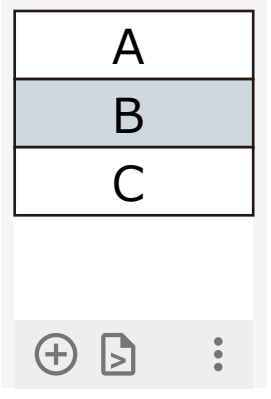


共有

共有入力を限定する

アプリ内で最新値を共有する

共有名

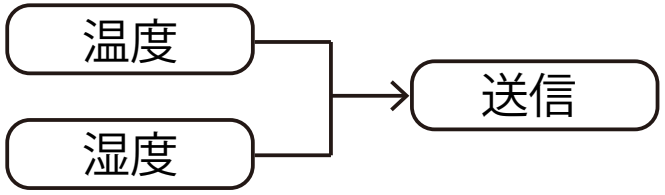


共有

共有入力を限定する

アプリ内で最新値を共有する

共有名



共有名で区別する方法

温度

アプリ内で最新値を共有する

共有名 temperature

湿度

アプリ内で最新値を共有する

共有名 humidity

送信

```
def  share_changed ( self, name, data) :
    if  name == 'temperature'  :
```

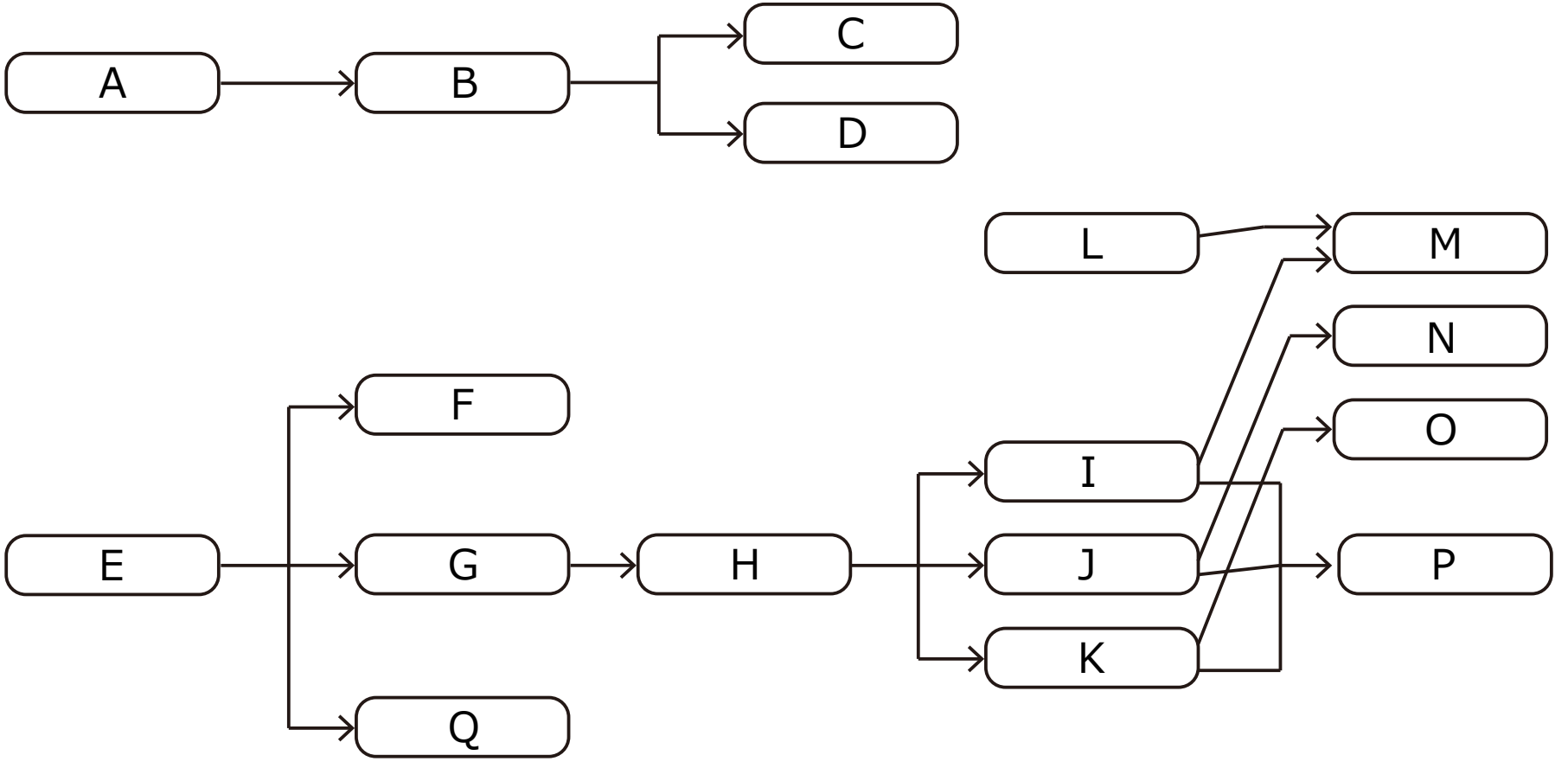
キー名で区別する方法

```
set_value(
    { 'temperature' : 23.4 }
)
```

```
set_value(
    { 'humidity' : 23.4 }
)
```

```
def  share_changed ( self, name, data) :
    if  'temperature' in data:
```

これでどんな形でも作れるわけですが。。。



データ共有の演習をしよう！！

1．「バーコード・ＱＲコードの読み取り」のアプリを「ＲＰＡ」っぽく使ってみる
2．アプリ「キャリブレーション機能のサンプル」
3．構成要素「1 つの数値のキャリブレーション」

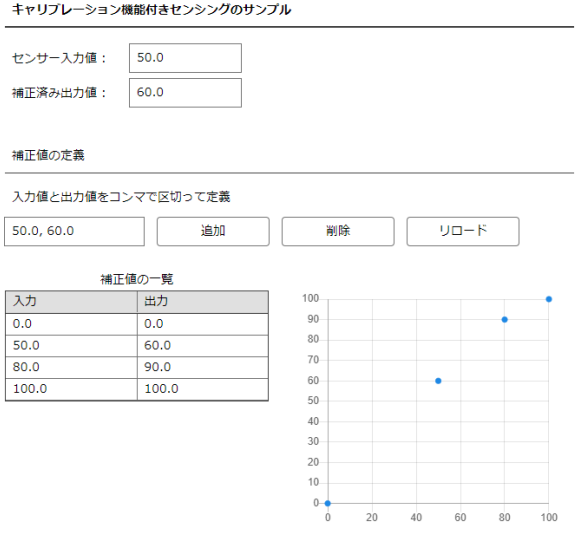
テキストをキーボード出力

```
python -m pip install pyautogui

import pyautogui as pg

pg.FAILSAFE = False
pg.typewrite(data['value'][0])
```

- [x] サインイン状態でのみ実行する



Caffè

M5StickC を使った IoT

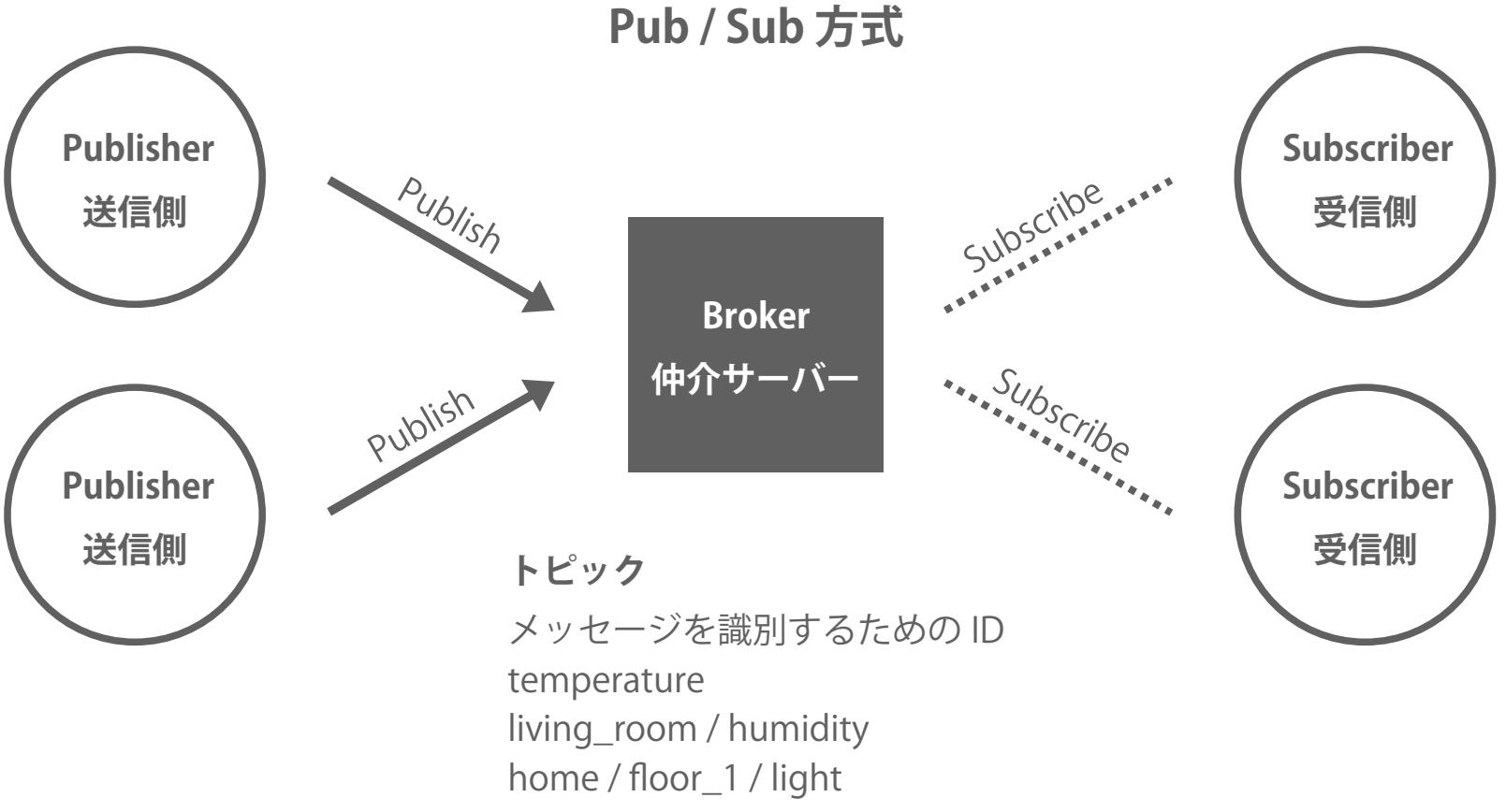
1）MQTT ブローカーの準備

2）MQTT 用のポートを開く

3）M5StickC 基本アプリ

4）キャリブレーション機能の追加

MQTT と複数デバイスの対応方法



Intel OpenVINO を使ったエッジ AI

1）位置情報オプション設定
2）推論に使用するデバイスの選択（CPU、GPU、MYRIAD）
3）画像の出力とフレームレート
4）顔の検出と年齢、性別の推定
5）顔の再認識
6）通行カウンター

OpenVINO ツールキット

- ディープラーニングの推論処理を、実行するハードウェア構成に応じて
  モデルの最適化を行ない、高速に処理してくれるツール群。
- CPU、GPU、VPU、FPGA のどれで行うか指定するだけで勝手にやってくれる。
- 無料で使える。
- サポートしている学習済みモデル

  Caffe 形式
  TensorFlow 形式
  MXNet 形式
  ONNX 形式
  など

  PyTorch, Chainer, NNC などのフレームワークから
  ONNX 形式で出力すればオッケー！

OpenVINO Open Model Zoo

学習済みモデルを公開・提供している

- 顔や人、車、物体の検出
- 年齢や性別、感情などの属性推定
- 頭の向き、視線の認識
- 姿勢推定
- 車のナンバープレートの認識
- 文字・テキスト認識

など

Tips & まとめ

コミュニティー・質問


Facebook（Remotte User Group）


Twitter（Remotte Technologies, Inc.）

Qiita（Remotte JP）

teratail（remotte.jp）

取り扱って欲しいデバイスやアプリについて投稿してください。
開発の優先順位を上げて、Remotte ストアにて公開します。