

第11期 成果発表会

『BLE タグで置き傘を管理したい』

2021年10月08日

先端IT活用推進コンソーシアム
クラウド・テクノロジー活用部会
キヤノンITソリューションズ株式会社 上村準也

はじまり

- 朝、出勤前に気になること...
 - 帰りに雨降るかな？傘は会社にあったかな？
 - 自宅に無ければ会社にあるのでは、という話はおいて...
 - 機械可読な仕組みを用意すべき(空気を読む家への接続)
- 空気を読む家のキッチン 冷蔵庫編
https://www.slideshare.net/aitc_jp/31-133093969
 - 過去に荒本さんが行った発表
 - BLEタグを利用した冷蔵庫の在庫管理(他の方法も検討)
 - モルモットのひとりとして参加

過去の実験

- 空気を読む家のキッチン 冷蔵庫編
 - 活動費でBLEタグを購入
 - MYNTはTileなど並び初期からあるスマートタグ
 - 機能が少なくて安いモデルを4個まとめて

注文日 2019年1月23日	合計 ¥ 5,899	お届け先 ローソン大田西馬込一丁目	注文番号 503-3933622-9995045 注文内容を表示 領収書等
-------------------	---------------	----------------------	--



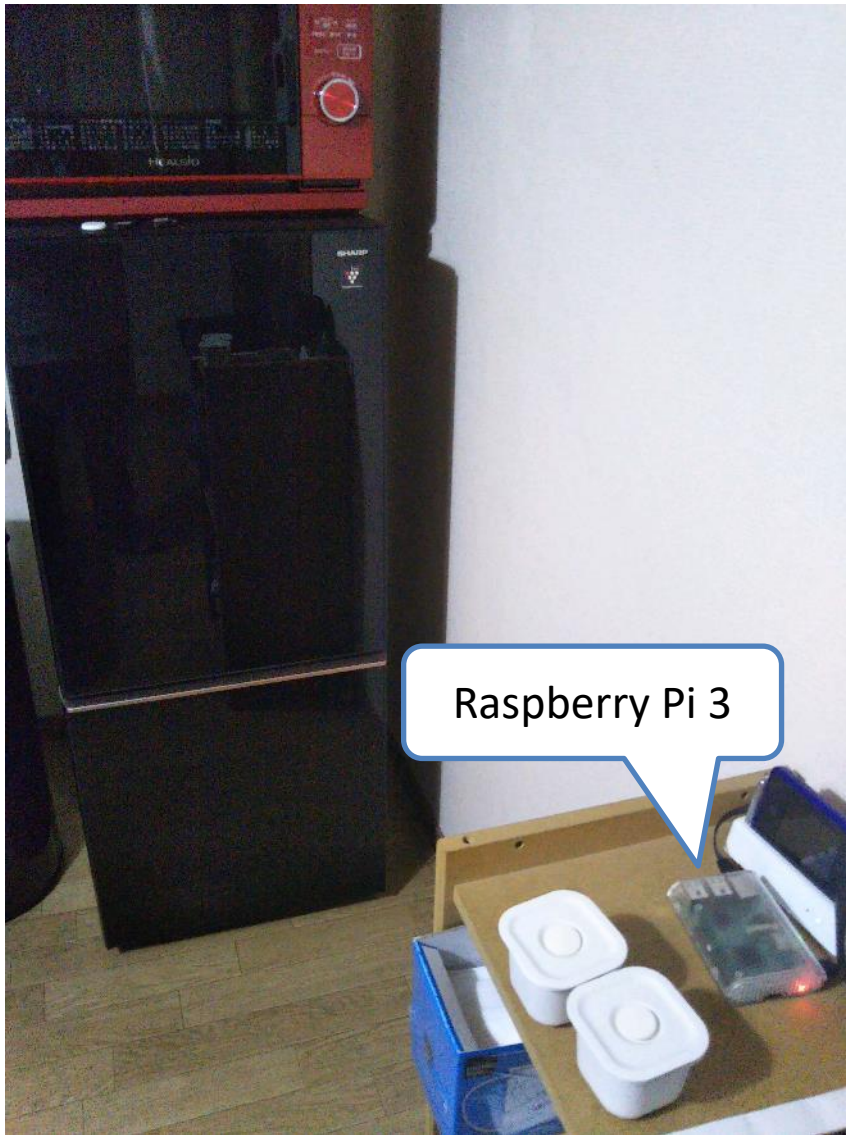
MYNT ES - 貴重品探索コンパス 電話探知機 キーファインダー 財布追跡機 鍵 ウォレット スマホ 携帯 数秒で紛失物を見つける(4個パック、黒2+白2)
返品期間：2019/02/23まで

 再度購入

商品レビューを書く

注文を非表示にする

過去の実験



今回の実験

- BLEタグで置き傘を管理したい
 - 馬鹿正直にやっていると時間がかかる(出社日も少ない...)
 - 部会の雑談で出たアイデア、電波遮断ポーチを追加購入
 - BLEタグをポーチから出し入れして開発
 - 蓋をバリッと開くだけで、直ぐ検知される
 - 複数タグにそれぞれ意味がある場合はポーチ1つでは足りないかも？

注文日
2021年7月27日

合計
¥ 447

お届け先
上村準也 ▾

注文番号 250-4489692-5411023
注文内容を表示 | 領収書等 ▾

2021/07/29に配達しました

ご注文商品を郵便受けに配達しました。



E買い物ネットワーク スマホの電波を完全遮断 電波遮断携帯圏外ポーチ 公共施設
病院などで Wi-Fi Bluetooth GPS
返品期間：2021/08/28まで

 再度購入

出品者を評価

商品レビューを書く

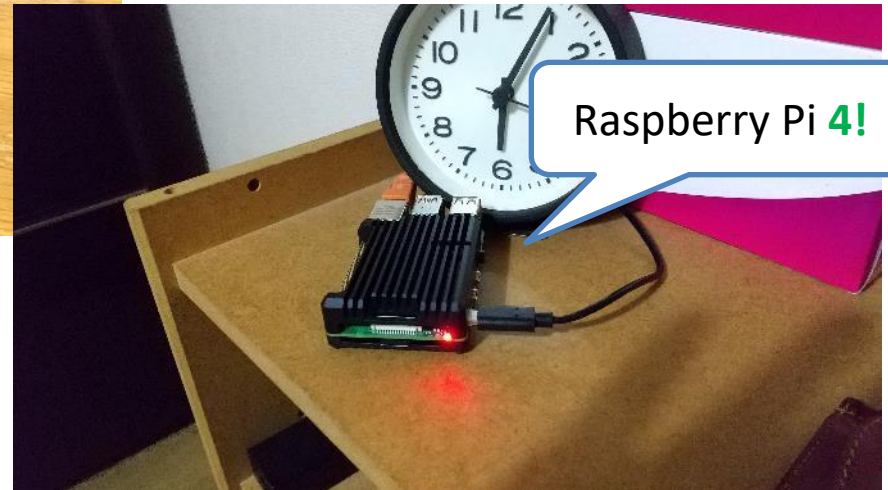
注文を非表示にする

今回の実験

電波遮断ポーチ



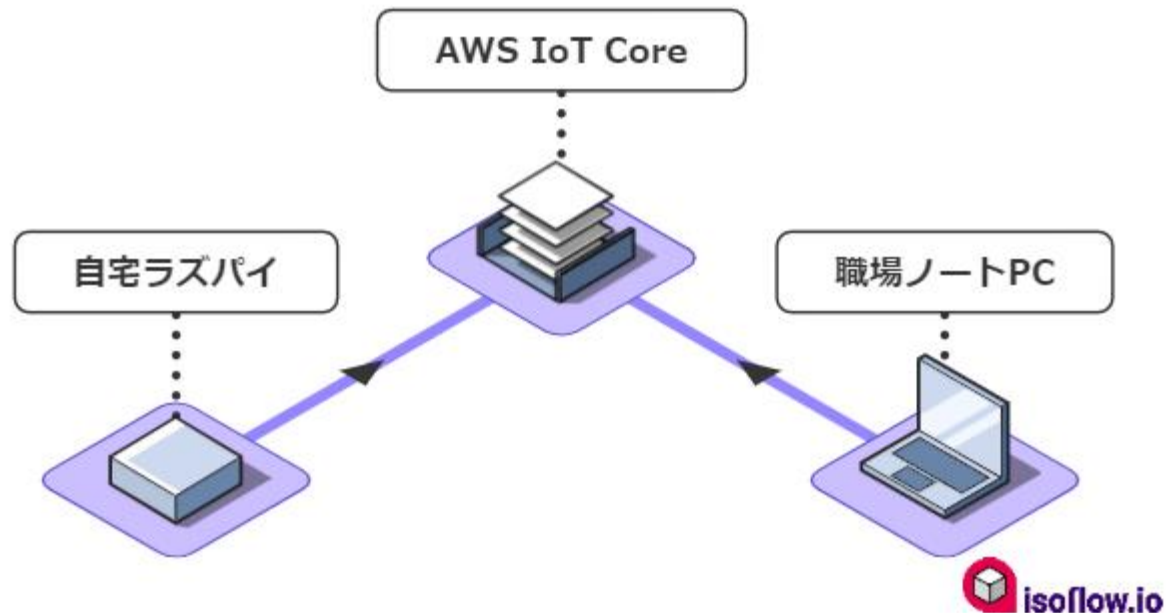
MYNT ES



Raspberry Pi 4!

構成

- 自宅と職場の両方から情報をアップロード
 - アップロードした情報の見方はあとで考える
 - 以前から触って見たかったAWS IoT Coreを試す



AWS IoT Core (Java)

AWS IoT にどのように接続していますか？

AWS IoT への接続方法に最適なプラットフォームと SDK を選択します。

プラットフォームの選択

Linux/OSX



Windows



AWS IoT デバイス SDK の選択

Node.js



Python



Java



留意が必要な前提条件:

デバイスは、Java JDK、Maven、Git がインストールされていて、ポート 8883 でパブリックインターネットに TCP 接続されている必要があります。

AWS IoT Device SDK とドキュメントをお探しの場合
[AWS IoT デバイス SDK の表示](#)

次へ

AWS IoT Core (Java)

AWS IoT に接続する

モノの登録

ステップ 1/3

「モノ」とはクラウド内部の物理デバイスの表現とレコードを意味します。物理デバイスが AWS IoT と連携するには、モノが必要です。モノを作成すると、Thing Shadow も作成されます。

代わりに既存のモノを選択しますか?

名前

オプション設定の表示 (後でも実行できます) ▲

戻る

次のステップ

AWS IoT Core (Java)

AWS IoT に接続する

接続キットのダウンロード

ステップ 2/3

次の AWS IoT が作成されます。

AWS IoT レジストリ内のモノ	Pi4-01
メッセージを送受信するポリシー	Pi4-01-Policy ポリシーのプレビュー

接続キットには以下が含まれます。

証明書とプライベートキー	Pi4-01.cert.pem, Pi4-01.private.key
AWS IoT デバイス SDK	Java SDK
メッセージを送受信するスクリプト	start.sh
デバイスがメッセージの接続と公開を行うには、接続キットをダウンロードする必要があります。 接続キットのダウンロード Linux/OSX	

[戻る](#)

[次のステップ](#)

AWS IoT に接続する

デバイスの設定とテスト

ステップ 3/3

デバイスの設定とテストを行うには、次のステップを行います。

ステップ 1: デバイスで接続キットを解凍する

```
unzip connect_device_package.zip
```

ステップ 2: 実行権限を追加する

```
chmod +x start.sh
```

ステップ 3: 開始スクリプトを実行する下記のようなメッセージがモノに表示されます

```
./start.sh
```

デバイスからのメッセージを待機中

戻る

完了

AWS IOT に接続する

デバイスの設定とテスト

ステップ 3/3

デバイスの設定とテストを行うには、次のステップを行います。

ステップ 1: デバイスで接続キットを解凍する

```
unzip connect_device_package.zip
```

ステップ 2: 実行権限を追加する

```
Set-ExecutionPolicy -ExecutionPolicy Bypass -Scope Process
```

ステップ 3: 開始スクリプトを実行する下記のようなメッセージがモノに表示されます

```
.\start.ps1
```

デバイスからのメッセージを待機中

戻る

完了

AWS IoT > MQTT test client

MQTT テストクライアント 情報

MQTT テストクライアントを使用して、AWS アカウントで渡される MQTT メッセージをモニタリングできます。デバイスは、トピック MQTT メッセージトピックにサブスクライブし、トピックに MQTT メッセージを発行できます。

トピックをサブスクライブする

トピックに公開する

トピックのフィルター 情報

トピックフィルタは、サブスクライブするトピックを記述します。トピックフィルタには、MQTT ワイルドカード文字を含めることができます。

▶ 追加設定

サブスクライブ

サブスクリプション

sdk/test/java

♡ ×

sdk/test/java

▼ sdk/test/java

hello from blocking publisher - 7

▼ sdk/test/java

hello from non-blocking publisher - 7

▼ sdk/test/java

hello from blocking publisher - 6

```
File Edit Setup Control Window KanjiCode Help
pi@raspberrypi:~/workspace $ ./start.sh

Running pub/sub sample application...
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by com.google.inject.internal.cglib.core.$Ref
va.lang.ClassLoader.defineClass(java.lang.String,byte[],int,int,java.security.Pr
WARNING: Please consider reporting this to the maintainers of com.google.inject.
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflect
WARNING: All illegal access operations will be denied in a future release
[INFO] Scanning for projects...
[INFO] Inspecting build with total of 1 modules...
[INFO] Installing Nexus Staging features:
[INFO] ... total of 1 executions of maven-deploy-plugin replaced with nexus-st
[INFO]
[INFO] -----< com.amazonaws:aws-iot-device-sdk-java-samples >-----
[INFO] Building aws-iot-device-sdk-java-samples 1.3.9
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- exec-maven-plugin:3.0.0:java (default-cli) @ aws-iot-device-sdk-java-
Cert file: ../Pi4-01.cert.pem Private key: ../Pi4-01.private.key
Oct 05, 2021 11:31:57 AM com.amazonaws.services.iot.client.core.AwsIotConnection
INFO: Connection successfully established
Oct 05, 2021 11:31:57 AM com.amazonaws.services.iot.client.core.AbstractAwsIotCl
INFO: Client connection active: sdk-java
1633429917610: >>> hello from blocking publisher - 1
1633429917610: >>> hello from non-blocking publisher - 1
1633429917782: <<< hello from blocking publisher - 1
1633429917945: <<< hello from non-blocking publisher - 1
1633429918608: >>> hello from non-blocking publisher - 2
1633429918625: >>> hello from blocking publisher - 2
1633429918779: <<< hello from non-blocking publisher - 2
1633429918951: <<< hello from blocking publisher - 2
1633429919608: >>> hello from non-blocking publisher - 3
1633429919643: >>> hello from blocking publisher - 3
1633429919780: <<< hello from non-blocking publisher - 3
1633429919954: <<< hello from blocking publisher - 3
1633429920609: >>> hello from non-blocking publisher - 4
1633429920652: >>> hello from blocking publisher - 4
1633429920781: <<< hello from non-blocking publisher - 4
1633429920951: <<< hello from blocking publisher - 4
1633429921609: >>> hello from non-blocking publisher - 5
1633429921658: >>> hello from blocking publisher - 5
1633429921781: <<< hello from non-blocking publisher - 5
1633429921952: <<< hello from blocking publisher - 5
1633429922610: >>> hello from non-blocking publisher - 6
1633429922664: >>> hello from blocking publisher - 6
1633429922781: <<< hello from non-blocking publisher - 6
```


- git と maven をインストールしておく

```
pi@raspberrypi:~ $ sudo apt install git
```

```
pi@raspberrypi:~ $ sudo apt install openjdk-11-jdk-headless
```

```
pi@raspberrypi:~ $ sudo apt install maven
```

- JREでなくJDK相当のJava環境が必要です
- mavenにはJAVA_HOMEを明示的に示す必要あり

AWS IoT Core (Java) のコツ

AWS IoT > ポリシー > Win10-01-Policy

ポリシー

Win10-01-Policy

アクション ▾

概要

ポリシー ARN

証明書

ポリシー ARN は、このポリシーを一意に識別します。詳細はこちら

バージョン

グループ

arn:aws:iot:us-east-1:106185572329:policy/Win10-01-Policy

コンプライアンス違反

作成日

10月 06, 2021, 20:43:58 (UTC+0900)

ポリシードキュメント

ポリシードキュメントでは、リクエストの権限を定義しています。詳細はこちら

☒ デフォルトとして設定

キャンセル 新しいバージョンとして保存

```

5      "Effect": "Allow",
6      "Action": [
7          "iot:Publish",
8          "iot:Receive",
9          "iot:RetainPublish"
10     ],
11     "Resource": [
12         "arn:aws:iot:us-east-1:106185572329:topic/sdk/test/java",
13         "arn:aws:iot:us-east-1:106185572329:topic/sdk/test/Python",
14         "arn:aws:iot:us-east-1:106185572329:topic/topic_1",
15         "arn:aws:iot:us-east-1:106185572329:topic/topic_2",
16         "arn:aws:iot:us-east-1:106185572329:topic/ble/office"
17     ]
18 },
19 {
20     "Effect": "Allow",
21     "Action": [
22         "iot:Subscribe"
23     ],
24     "Resource": [
25         "arn:aws:iot:us-east-1:106185572329:topicfilter/sdk/test/java",
26         "arn:aws:iot:us-east-1:106185572329:topicfilter/sdk/test/Python",
27         "arn:aws:iot:us-east-1:106185572329:topicfilter/topic_1",
28         "arn:aws:iot:us-east-1:106185572329:topicfilter/topic_2"
29     ]
30 },
31 {
32     "Effect": "Allow",
33     "Action": [
34         "iot:Connect"

```

- 独自のトピックを使い始めるためにはポリシーを編集
- モノの証明書に紐付く

AWS IoT Core (Java)

MQTT テストクライアント 情報

MQTT テストクライアントを使用して、AWS アカウントで渡される MQTT メッセージをモニタリングできます。デバイスは、トピックによって識別された MQTT メッセージを発行し、その状態を AWS IoT に伝えます。MQTT メッセージトピックにサブスクライブし、トピックに MQTT メッセージを発行できます。

トピックをサブスクライブする

トピックに公開する

トピックのフィルター 情報

トピックフィルタは、サブスクライブするトピックを記述します。トピックフィルタには、MQTT ワイルドカード文字を含めることができます。

ble/home

▶ 追加設定

サブスクライブ

サブスクリプション

ble/home

♡ ✕

ble/home

▼ ble/home

80:EA:CA:71:0A:8C MYNT-ES

- サンプルコードを改造して
-payload オプションで任意の文字列を送信できるようにした

```

raspberrypi.local - uemuraj@ik1-421-42813:~ VT
File Edit Setup Control Window KanjiCode Help
pi@raspberrypi:~/workspace/aws-iot-device-sdk-java/aws-iot-device-sdk-java-samples$ java @PublishToIoTCore -payload "80:EA:CA:71:0A:8C MYNT-ES"
Cert file:/home/pi/workspace/Pi4-01.cert.pem Private key: /home/pi/workspace/Pi4-01.private.key
Oct 06, 2021 1:08:13 PM com.amazonaws.services.iot.client.core.AwsIotConnection onConnectionSuccess
INFO: Connection successfully established
Oct 06, 2021 1:08:14 PM com.amazonaws.services.iot.client.core.AbstractAwsIotClient onConnectionSuccess
INFO: Client connection active: sdk-java
1633522094123: >>> 80:EA:CA:71:0A:8C MYNT-ES
Oct 06, 2021 1:08:14 PM com.amazonaws.services.iot.client.core.AwsIotConnection onConnectionClosed
INFO: Connection permanently closed
Oct 06, 2021 1:08:14 PM com.amazonaws.services.iot.client.core.AbstractAwsIotClient onConnectionClosed
INFO: Client connection closed: sdk-java
pi@raspberrypi:~/workspace/aws-iot-device-sdk-java/aws-iot-device-sdk-java-samples$

```

- 自宅はラズパイ
 - 過去の実験よりハードもOSも新しい
 - ただ、同じ手順でちゃんとMYNTを検出できる
- 職場は...
 - 私物のハードウェアなんて設置できない、通信もダメ
 - LinuxのサーバはあるがBluetoothデバイス無し
 - WindowsのノートPCが支給されていてBluetoothデバイス有り

前から興味があったのでWindowsでBluetoothデバイスを扱うプログラミングをすることにしたが...

- 環境は新しいが前回と同じ方法でタグ検出

```
pi@raspberrypi:~ $ cat /proc/cpuinfo | grep Model
Model          : Raspberry Pi 4 Model B Rev 1.2
pi@raspberrypi:~ $ cat /etc/os-release | grep PRETTY
PRETTY_NAME="Raspbian GNU/Linux 10 (buster)"
pi@raspberrypi:~ $ sudo hcitool lescan
LE Scan ...
60:C7:0B:96:A3:26 (unknown)
60:A5:EF:3C:5F:60 (unknown)
80:EA:CA:71:0A:8C MYNT-ES
80:EA:CA:71:0A:8C (unknown)
79:A3:2A:6E:F3:A4 (unknown)
79:A3:2A:6E:F3:A4 (unknown)
D8:E8:8D:17:83:75 (unknown)
80:EA:CA:71:13:4D MYNT-ES
80:EA:CA:71:13:4D (unknown)
49:20:EB:F4:44:04 (unknown)
```

おわりに

ありがとうございました



<http://aitc.jp>



<https://www.facebook.com/aitc.jp>



ハルミン

AITC非公式イメージキャラクター