

IoTデバイス活用マニュアル



センサ種類：ブザー

型番：Grove - Buzzer


ストーリー：クラウド側から、デバイスのブザーを鳴らす。ブザーをデバイス側の操作で停止できる。ブザーの状況はクラウドで確認できる。

。



改訂記録：

2020/12/07 初版 作成 陣内



注意

本書の内容は、IoT Hubを使ったバージョン(a)と、IoT Centralを使用したバージョン(b)の2つを扱う。
それぞれ、以下の通り、現状では一長一短がある。

(a) IoT Hubを使ったバージョンでは、ブザーを鳴らすのに、Azureポータルからの手動操作のみをサポートしている。PowerAutomateから操作できるようにするには、プログラムの開発が必要になり、今後の検討課題である。

ArduinoのボードマネージャのESP8266のバージョンは 2.3.0を使用する。

(b) IoT Centralを使ったバージョンでは、ブザーをPower Automateから制御することができる。
しかし、PowerBI、CosmosDBとは連携していない。連携には、間にもう一段階サービスが必要と思われる。

ArduinoのボードマネージャのESP8266のバージョンは最新版2.7.4を使用する。

以上 2020/12/07時点

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by circles of varying sizes, some with concentric rings, and the lines are thin and grey. The diagram is partially cut off by the top and left edges of the frame.

(a)IoT Hub版

A decorative network diagram in the bottom-right corner, similar to the one in the top-left. It shows a cluster of nodes connected by lines, with some nodes having concentric circles. The diagram is partially cut off by the bottom and right edges of the frame.



1.Arduinoのプログラムの変更

ライブラリの読み込みとコードの変更箇所



プログラム

サンプルプログラム： [Buzzer_WIOnodeInput_IoTHUB](#)

修正箇所： 以下の該当箇所を利用環境に応じて修正する。

- WiFi の SSID、パスワード
- Azure IoT Hub で発行されたデバイスの接続文字列



2. Stream Analyticsのクエリ変更

修正不要

講座の標準サンプルの温度計の設定のまま
利用できます。





3. PowerBIでのレポートの作成

修正不要

講座の標準サンプルの温度計の設定
のまま利用できます。

データセットを開く

Power BI FactoryScientist受講生サイト

新しい外観をオンにする

検索

ホーム

お気に入り

最近

アプリ

自分と共有

詳細

ワークスペース

FactoryScientist受...

ダッシュボード

確認ダッシュボード

レポート

稼働状況マップ

ブック

データセット

データフロー

データフローがありません

データを取得

ページ 1

フィルター

検索

このページでのフィルター

ここにデータ フィールド...

すべてのページでのフィル...

ここにデータ フィールド...

視覚化

フィールド

検索

fstable

device

duration

time

その他のビジュアルの取得

ビジュアルをファイルからインポート

視覚エフェクトの削除

既定の視覚化の復元

ドリルダウン

クロス レポート

オフ

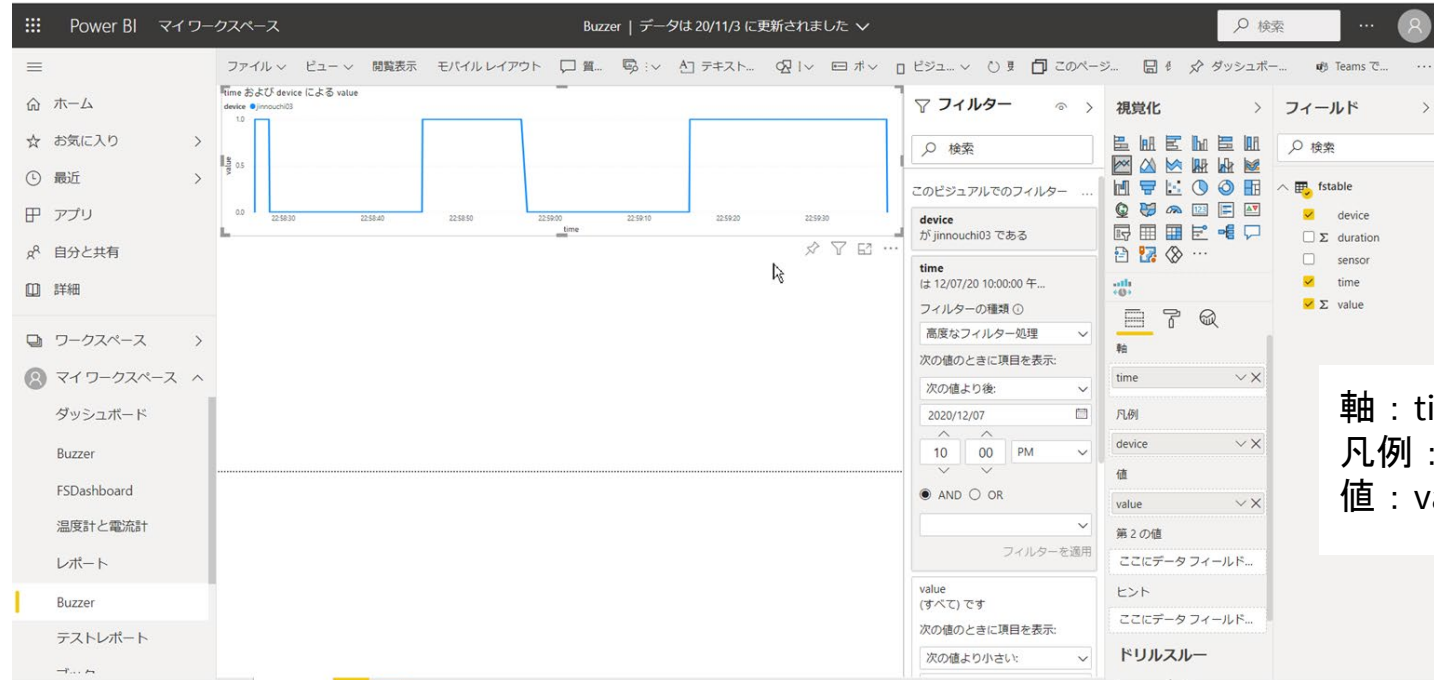
すべてのフィルターを保持...

オン

ドリルスルー フィールド...

fsdatasetを選択する

折れ線グラフの設定



A decorative network diagram in the top-left corner, consisting of various sized circles (nodes) connected by thin lines (edges). Some nodes are solid grey, while others are hollow with a grey outline. The connections form a complex, branching structure.

4. Azureからブザーを鳴らす

A decorative network diagram in the bottom-right corner, similar to the one in the top-left, featuring a cluster of interconnected nodes and edges.

Azureからの操作でブザーを鳴らす

「全てのリソース」→「(自分の)IoT Hub」→「IoT Device」→「(ブザーを鳴らすデバイス)」を選択

The image consists of two screenshots from the Microsoft Azure portal, illustrating the process of sending a message to an IoT device.

Left Screenshot: Shows the 'jinnouchi03' IoT device page. A red circle highlights the 'デバイスへのメッセージ' (Message to device) link in the left-hand navigation pane. A red arrow points from this link to the right screenshot.

Right Screenshot: Shows the 'デバイスへのメッセージ' (Message to device) page. A red circle highlights the 'メッセージの送信' (Send message) button. Below this, a large text input area is shown with a red arrow pointing to it, indicating where to enter the message.

Annotations:

- ① デバイスへのメッセージ (Message to device)
- ② ここにメッセージを何か入力（アラート理由など） (Enter the message here (e.g., alert reason))
- ③ メッセージの送信 (Send message)
- ④ ブザーはWioNodeのfuncボタン長押しで停止 (The buzzer is stopped by long pressing the func button of WioNode)

IoT Hub 版の検討課題

- ・ ブザーを鳴らすためにAzureポータルからの手動操作が必要。
- ・ PowerAutomateからAzureポータルのこの操作を直接呼び出すことができないため、自動化のためには、この操作に該当するプログラム開発が必要。
- ・ プログラミングには以下のリンク先にあるような .Net、Java、Node.js、Python、iOSなどのプログラミング方法がある。

[デバイスに IoT Hub でクラウドからメッセージを送信する \(.NET\)](#)

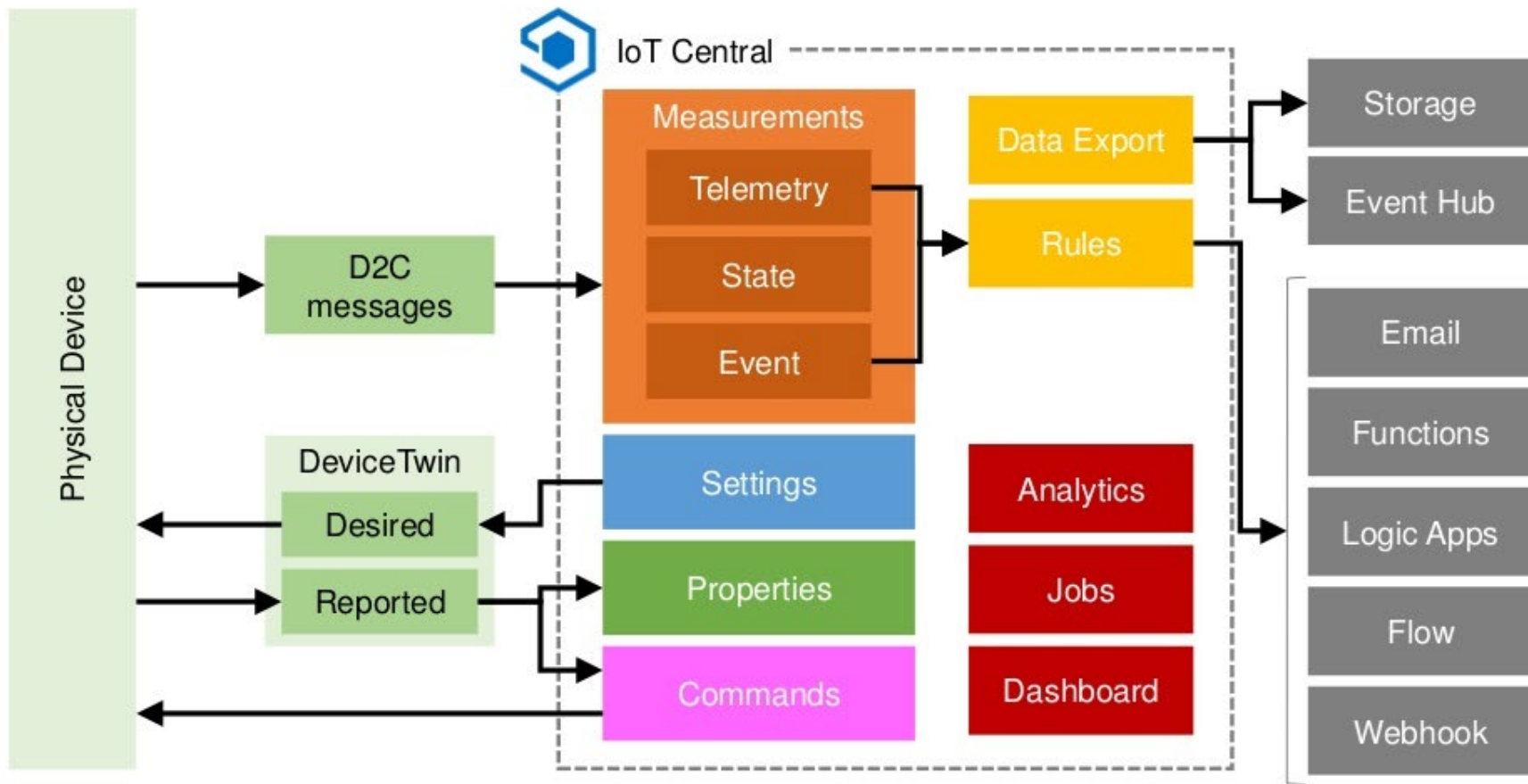
A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by circles of varying sizes, some with concentric rings, and the lines are thin and grey. The diagram is partially cut off by the top and left edges of the frame.

(b)IoT Central版

A decorative network diagram in the bottom-right corner, similar to the one in the top-left. It shows a cluster of interconnected nodes and lines, with some nodes having concentric circles. The diagram is partially cut off by the bottom and right edges of the frame.

IoT Central とは

- IoTソリューションを手軽に構築できるプラットフォームです。
- これまでの講座で使用したIoT Hub、Stream Analytics、PowerBIのような機能をまとめて簡略版で提供しています。
- IoT Centralの中にもIoT Hubが含まれています。



IoT Central 版 作成の流れ

1. Azure IoT Centralアプリケーションの作成
 - a. IoT Centralアプリケーションの作成
 - b. デバイステンプレートの作成
 - c. ビューの作成
 - d. デバイステンプレートの公開
 - e. デバイスの作成
2. Arduinoプログラムの修正
3. IoT Centralからブザーを鳴らす
4. PowerAutomateからブザーを鳴らす



1.a IoT Centralアプリケーションの作成



IoT Central アプリケーションの作成

「全てのリソース」→「新規追加」→「IoT Central」を検索

The screenshot shows the Microsoft Azure Marketplace interface. The search bar at the top contains the text 'IoT Central'. The results are displayed in a grid of cards. The first card is for 'IoT Central application' by Microsoft. This card is highlighted with a red box, and its '作成' (Create) button is circled in red. A red arrow points from the text '作成→「IoT Central Application」を選択' to this button. Other cards visible include 'Air Compressor Predictive Maintenance (IoT Central)' by KAISPE LLC, 'Omnitech Unified Commerce Cloud - IoT' by Tofugear Limited, 'KeyScaler - IoT Security for Microsoft Azure' by Device Authority, and 'AXS GUARD Firewall - VPN - WAF - IAM for Azure' by AXS GUARD / Able NV.

Microsoft Azure

リソース、サービス、ドキュメントの検索 (G+)

ホーム > すべてのリソース > 新規 >

Marketplace

プライベート マーケットプレイス (レビュー)

保存リスト

最近作成

サービス プロバイダー

カテゴリ

開始

AI + Machine Learning

分析

ブロックチェーン

Compute

コンテナー

データベース

開発者ツール

DevOps

ID

統合

Marketplace を検索

料金 : すべて

オペレーティング システム : すべて

Publisher Type : すべて

Offer Type : すべて

結果をすべて表示


Tile view

IoT Central application
Microsoft
IoT (モノのインターネット) の SaaS の簡単さをご体験ください。クラウドの専門知識は不要です。

作成

作成 → 「IoT Central Application」を選択

IoT Central アプリケーションの作成



Microsoft Azure

リソース、サービス、ドキュメントの検索 (G+)

jinnouchi@factoryscient...
FACTORYSCIENTIST.COM

ホーム > 新規 > Marketplace > IoT Central application >

IoT Central アプリケーション

IoT Central アプリケーション

リソース名 *

jinnouchi-iotc-app1

アプリケーション URL *

jinnouchi-iotc-app1.azureiotcentral.com

サブスクリプション *

TA

リソースグループ *

jinnouchi

新規作成

料金プラン *

Standard 0

価格の詳細をご確認ください

テンプレート *

カスタム アプリケーション

アプリケーション テンプレートの詳細情報

場所 *

日本

作成 Automation オプション

リソース名：IoT Centralとわかる名前をつける

アプリケーション名：IoT CentralアプリケーションのURLの一部になる。

サブスクリプション：使用するサブスクリプション

リソースグループ：講座では自分の番号+名前をつける

料金プラン：利用する通信量によりプランを決める。

最初は **Standard0** で良い

プランの詳細な説明はこちら：

<https://azure.microsoft.com/en-us/pricing/details/iot-central/?rtc=1>

テンプレート：1から作成するのでカスタムアプリケーションにする
作りたい類似のアプリがあればそれを選んでよい。

場所：日本

IoT Central アプリケーションの作成結果

The screenshot shows the Microsoft Azure portal interface for an IoT Central application named 'jinnouchi-iotc-app'. The left sidebar contains navigation links: Home, Summary, Tags, Settings, Properties, Monitoring, Metrics, Automation, and Tasks. The main content area displays the application's details under the 'Basic' tab. The details include the resource group 'Jinnouchi', location 'Japan', subscription 'TA', and subscription ID '76dc27d2-d03c-49d7-90f7-125f6e381b32'. A link to the application's URL is provided: 'https://jinnouchi-iotc-app.azureiotcentral.com'. An arrow points to this URL with the text 'このURLがIoT Centralアプリケーション' (This URL is the IoT Central application).

Microsoft Azure

リソース、サービス、ドキュメントの検索 (G+/I)

ホーム >

jinnouchi-iotc-app
IoT Central アプリケーション

検索 (Ctrl+/)

削除

概要

タグ

設定

プロパティ

監視

メトリック

オートメーション

タスク

基本

リソース グループ (変更) : Jinnouchi

場所 : 日本

サブスクリプション (変更) : TA

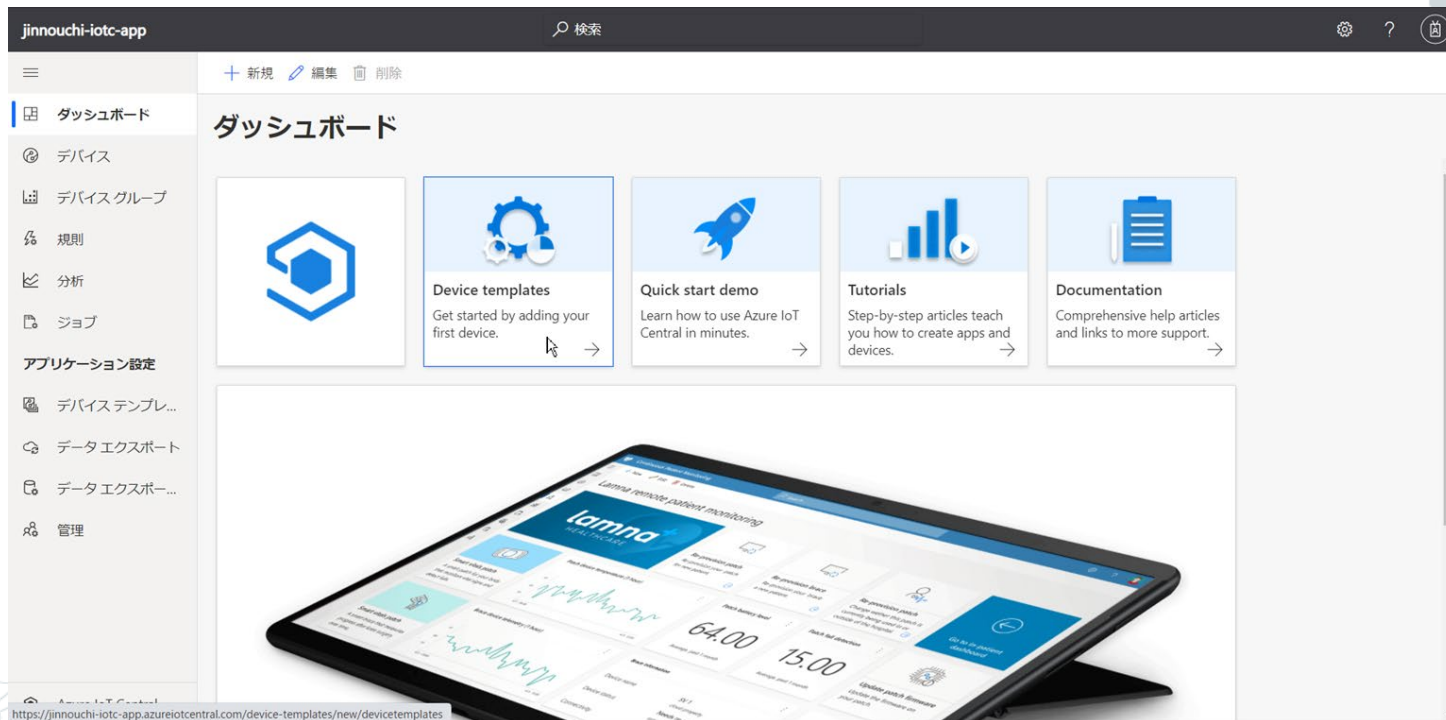
サブスクリプション ID : 76dc27d2-d03c-49d7-90f7-125f6e381b32

タグ (変更) : タグを追加するにはここをクリック

IoT Central アプリケーション... : <https://jinnouchi-iotc-app.azureiotcentral.com>

このURLがIoT Centralアプリケーション

作成した IoT Central アプリケーションのトップページ





1.b デバイステンプレートの作成



デバイステンプレートとは

・ デバイステンプレートとは、デバイスとIoT Centralでやり取りするインターフェースの定義。
やり取りの内容には次のものが含まれる

- ・ プロパティ
- ・ テレメトリ
- ・ コマンド
- ・ クラウドプロパティ
- ・ カスタマイズ
- ・ ビュー

この事例では、テレメトリとコマンド、ビューの設定を行う。

注：プロパティは、設定してもうまく動作しなかった。

デバイステンプレートの作成

jinnouchi-iotc-app

検索

デバイス テンプレート

+ 新規

① デバイス テンプレ...

デバイス テンプレートの作成

デバイス テンプレートは、デバイスの特性と動作を定義するブループリントです。 [詳細情報](#)

② デバイス テンプレートの作成

Azure IoT Central

デバイステンプレートの作成

jinnouchi-iotc-app

検索



ダッシュボード

デバイス

デバイス グループ

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...

データ エクスポート

データ エクスポート...

管理

Azure IoT Central

デバイス テンプレート > 新規作成



種類の選択



確認

種類の選択

デバイス テンプレートはブループリントのようなものです。アプリケーションに接続するデバイスの特性と動作を定義します。

カスタム デバイス テンプレートの作成

IoT デバイス
機能モデルをインポートするか、初めから機能を作成します。

Azure IoT Edge
Azure IoT Edge とゲートウェイのシナリオを備えたテンプレートを作成します。

おすすめのデバイス テンプレート

次へ: カスタマイズ

キャンセル

デバイステンプレートの作成

jinnouchi-iotc-app

検索

?

≡

ダッシュボード

デバイス

デバイスグループ

規則

分析

ジョブ

アプリケーション設定

デバイステンプレ...

データエクスポート

データエクスポー...

管理

デバイステンプレート > 新規作成

● 種類の選択

● カスタマイズ

○ 確認

カスタマイズ

デバイステンプレート名*

BuzzerTemplate

① デバイステンプレート名を入力する

デバイスと接続するゲートウェイ デバイスを作成するには、下のチェック ボックスをオンにする必要があります。

☐ ゲートウェイ デバイス

* 必須

次: レビュー

[消へ] をクリックします。

キャンセル

デバイステンプレートの作成

jinnouchi-iotc-app

検索



ダッシュボード

デバイス

デバイス グループ

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...

データ エクスポート

データ エクスポー...

管理

デバイス テンプレート > 新規作成

- 種類の選択
- カスタマイズ
- 確認

確認

空のテンプレートを作成して、機能とインターフェイスを追加できるようにします。インターフェイスは、インポートすることも最初から作成することもできます。完了すると、テンプレートを公開してデバイスに接続できるようになります。

基本情報

デバイステンプレートの種類	IoT デバイス
デバイステンプレート名	BuzzerTemplate

作成

[前へ] をクリックします。

キャンセル

デバイステンプレートの作成 — モデルの作成

jinnouchi-iotc-app

検索

バージョン テストデバイスの管理 公開 名前の変更 削除

デバイステンプレート > BuzzerTemplate

BuzzerTemplate

モデルの作成

カスタムモデルを最初から作成するか、既存のモデルをインポートします。

カスタムモデル

空のモデルで開始し、最初からデバイスを作り上げます。

モデルのインポート

まずモデル ファイルをインポートします。

デバイステンプレートの作成 — インターフェースの追加

jinnouchi-iotc-app

検索

?

三

ダッシュボード

デバイス

デバイス グループ

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...


データエクスポート

データエクスポ...

管理

バージョン テスト デバイスの管理 公開 名前の変更 削除

デバイス テンプレート > BuzzerTemplate > モデル



BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル


コンポーネントの追加 エクスポート 削除

セアラル

モデルは、デバイスが IoT Central アプリケーションと対話する方法を定義します。追加の機能を使用してモデルをカスタマイズするか、機能を継承するインターフェイスを追加するか、他のインターフェイスに基づく新しいコンポーネントを追加します。[詳細情報](#)


機能

モデルに機能がありません。機能は、デバイスが実行できる操作と、それが IoT Central に送信するデータのサブタイプを示します。




既定のコンポーネントへの機能の追加

既定のコンポーネントはこのデバイス モデルに関連付けられているため、このイ...



継承されたインターフェイスの追加

他の一般的なインターフェイスから特殊なインターフェイスを作成します (例: "ルーム" インターフェイスを拡張する "会議室" インターフェイス)。



コンポーネントの追加

コンポーネントを使用すると、他の類似のインターフェイスで構成されるモジュール インターフェイスを作成できます (例: "デバイス情報", "カメラ", "加減速度")

デバイステンプレートの作成 — インターフェースの追加

jinnouchi-iotc-app

検索

バージョン

テストデバイスの管理

公開

名前の変更

削除

ダッシュボード

デバイス

デバイスグループ

規則

分析

ジョブ

アプリケーション設定

デバイステンプレ...

データエクスポート

データエクスポ...

管理

デバイステンプレート > BuzzerTemplate > モデル > BuzzerTemplate > インターフェースの追加

BuzzerTemplate

アプリケーションが更新されました: Never インターフェースが公開されました: Never

モデル

BuzzerTemplate

クラウドのプロパティ


カスタマイズ

ビュー


戻る

インターフェースの追加

独自のカスタム インターフェースを作成するか、インターフェースをインポートできます。



カスタム
空のインターフェイスで開始し、適宜作成します。



インターフェースのインポート
ファイルからインターフェースをインポートします。

51

デバイステンプレートの作成 — インターフェースの追加 — 機能の追加

jinnouchi-iotc-app

検索

バージョン

テスト

デバイスの管理

公開

名前の変更

削除

ダッシュボード

デバイス

デバイスグループ

規則

分析

ジョブ

アプリケーション設定

デバイステンプレ...

データエクスポート

データエクスポー...

管理

バージョン

テスト


デバイスの管理

公開

名前の変更

削除

デバイス テンプレート > BuzzerTemplate > モデル > BuzzerTemplate > インターフェイス



BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

インターフェイス

クラウドのプロパティ

カスタマイズ

ビュー

保存

機能の追加

ID の編集

バージョン

エクスポート

削除

インターフェイス

継承インターフェイス

ドラフト

より一般的な他のインターフェイスから機能を継承する専用インターフェイスを作成します (例: "ルーム" インターフェイスを拡張する "会議室" インターフェイス)。 [詳細情報](#)

機能

Give your interface a display name and a friendly name, choose a capability type and a semantic type, and then choose how your data will be measured and displayed. Keep adding capabilities until you've fully described your interface.

表示名

名前 *

機能の種類 * ①

セマンティックの種類 ①

Telemetry

なし

+

機能の追加

デバイステンプレートの作成 – インターフェースの追加 – 機能の追加

jinnouchi-iotc-app 検索

バージョン テストデバイスの管理 公開 名前の変更 削除

ダッシュボード
デバイス
デバイスグループ
規則
分析
ジョブ
アプリケーション設定
デバイステンプレート

デバイステンプレート > BuzzerTemplate > モデル > BuzzerTemplate > インターフェイス

BuzzerTemplate
アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル < BuzzerTemplate
インターフェイス

クラウドのプロパティ
カスタマイズ
ビュー

機能
Give your interface a display name and a friendly name, choose a capability type and a semantic type, and then choose how your data will be measured and displayed. Keep adding capabilities until you've fully described your interface.

①

表示名 *	名前 *	機能の種類 *	セマンティックの種類 ①
Alert	Alert	Telemetry	State

状態の値
値スキーマ *

Integer

② + 追加

複合型を定義する必要があります
各状態の値と表示名を定義してください

テレメトリの追加を行う
表示名: Alert
名前: Alert
機能の種類: Telemetry
セマンティックの種類: State
値スキーマ: Integer

デバイステンプレートの作成 — インターフェースの追加 — 機能の追加

jinnouchi-iotc-app

検索

バージョン テスト デバイスの管理 公開 名前の変更 削除

デバイス テンプレート > BuzzerTemplate > モデル > BuzzerTemplate > インターフェイス

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

インターフェイス

クラウドのプロパティ

カスタマイズ

ビュー

①

保存 機能の追加 ID の編集 バージョン エクスポート 削除

Alert

Alert

状態の値

値スキーマ *

Integer

表示名 名前 値

Normal	Normal	0	×
Alert	Alert	1	×

+ 追加

単位 表示単位 コメント 説明

なし

Alert有無

0:Normal、1:Alert

機能の追加

テレメトリの値の定義を行う

表示名: Normal

名前: Normal

値: 0

表示名: Alert

名前: Alert

値: 1

デバイステンプレートの作成 — インターフェースの追加 — 機能の追加

jinnouchi-iotc-app

検索

バージョン テスト デバイスの管理 公開 名前の変更 削除

ダッシュボード

デバイス

デバイス グループ

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...

データエクスポート

データエクスパー...

管理

デバイス テンプレート > BuzzerTemplate > モデル > BuzzerTemplate > インターフェイス

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

インターフェイス

クラウドのプロパティ

カスタマイズ

ビュー

①

AlarmOn AlarmOn Command

オフラインの場合にキュー... コメント 説明

オン

要求

応答

②

AlarmOff AlarmOff Command

オフラインの場合にキュー... コメント 説明

オン

要

2つのコマンドを追加する

表示名: AlertOn

名前: AlertOn

機能の種類: Command

オフラインの場合にキュー:

オン

表示名: AlertOff

名前: AlertOff

機能の種類: Command

オフラインの場合にキュー:

デバイステンプレートの作成ーインターフェースIDの編集

jinnouchi-iotc-app

検索

バージョン テスト デバイスの管理 公開 名前の変更 削除

デバイス テンプレート > BuzzerTemplate > モデル > BuzzerTemplate > インターフェイス

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

インターフェイス

クラウドのプロパティ

カスタマイズ

ビュー

アプリケーション設定

デバイス テンプレ...

データ エクスポート

データ エクスポー...

管理

保存 + 機能の追加 ID の編集 バージョン エクスポート 削除

インターフェイス 継承インターフェイス

より一般的な他のインターフェイスから機能を継承する専用インターフェイスを作成します (例: "ルーム" エイス)。詳細情報

機能

Give your interface a display name and a friendly name, choose a capability type and a semantic type, and the displayed. Keep adding capabilities until you've fully described your interface.

表示名 *	名前 *	機能の種類 * ①
Alert	Alert	Telemetry

状態の値

値スキーマ *

Integer

①

②

③

インターフェイスの名前を付ける

表示名: BuzzerIF

名前: BuzzerIF

表示名 ①

BuzzerIF

名前空間 * ①

jinnouchilotcApp

名前 * ①

BuzzerIF

バージョン ①

1

インターフェイス @ID

dtmijinnouchilotcApp:Buz...

保存 キャンセル

バージョン

テスト

デバイスの管理

公開

名前の変更

削除

バージョン

機能の追加

IDの編集

バージョン

エクスポート

削除

バージョン

機能の追加

IDの編集

バージョン

エクスポート

削除

より一般的な他のインターフェイスから機能を継承する専用インターフェイスを作成します (例: "ルーム" インターフェイスを拡張する "会議室" インターフェイス)。 [詳細情報](#)

機能

Give your interface a display name and a friendly name, choose a capability type and a semantic type, and then choose how your data will be measured and displayed. Keep adding capabilities until you've fully described your interface.

表示名 *	名前 *	機能の種類 * ①	セマンティックの種類 ①		
<input type="text" value="Alert"/>	<input type="text" value="Alert"/>	<div>Telemetry</div>	<div>State</div>	<div></div>	<div></div>
<input type="text" value="AlarmOn"/>	<input type="text" value="AlarmOn"/>	<div>Command</div>		<div></div>	<div></div>
<input type="text" value="AlarmOff"/>	<input type="text" value="AlarmOff"/>	<div>Command</div>		<div></div>	<div></div>

モデル

BuzzerTemplate

BuzzerIF

クラウドのプロパティ

カスタマイズ

ビュー

作成したインターフェースの確認



1.c ビューの作成



ビューの作成 - テレメトリ タイルの追加

jinnouchi-iotc-app

検索

バージョン テスト デバイスの管理 公開 名前の変更 削除

デバイス テンプレート > BuzzerTemplate > ビュー

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

BuzzerIF

クラウドのプロパティ

カスタマイズ

ビュー

保存 削除 プレビュー デバイスの構成

す。

ヘ テレメトリ

グラフ、状態、イベント、最後の既知の値 (LKV)、主要業績評価指標 (KPI) のタイルを作成します。

Alert

+

テレメトリ

ヘ プロパティ

デバイスに関する基本情報を表示する一覧 (グリッド) またはマップタイルを作成します。

プロパティを選択する

①

②

③

タイルの追加

ビューの作成 - テレメトリ タイルのグラフ種類変更

jinnouchi-iotc-app

検索

?

三

ダッシュボード

デバイス

デバイス グループ

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...

データ エクスポート

データ エクスポー...

管理

バージョン

テスト デバイスの管理

公開

名前の変更

削除

デバイス テンプレート > BuzzerTemplate > ビュー

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

BuzzerIF

クラウドのプロパティ

カスタマイズ

ビュー

保存

削除

プレビュー

デバイスの構成

す。

ヘ テレメトリ

グラフ、状態、イベント、最後の既知の値 (LKV)、主要業績評価指標 (KPI) のタイルを作成します。

テレメトリを選択する

+ テレメトリ

ヘ プロパティ

デバイスに関する基本情報を表示する一覧 (グリッド) またはマップ タイルを作成します。

プロパティを選択する

+ プロパティ

Alert

タイムスタンプ

状態名

値

	Alert	Normal
	Alert	Alert

ビューの作成 - テレメトリ タイルのグラフ種類変更

jinnouchi-iotc-app

検索

バージョン テスト デバイスの管理 公開 名前の変更 削除

デバイス テンプレート > BuzzerTemplate > ビュー

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

BuzzerIF

クラウドのプロパティ

カスタマイズ

ビュー

保存 削除 プレビュー デバイスの構成

す。

ヘ テレメトリ

グラフ、状態、イベント、最後の既知の値 (LKV)、主要業績評価指標 (KPI) のタイルを作成します。

テレメトリを選択する

+ テレメトリ

ヘ プロパティ

デバイスに関する基本情報を表示する一覧 (グリッド) またはマップタイルを作成します。

プロパティを選択する

+ プロパティ

Alert

KPI

状態の履歴

棒グラフ

ヒートマップ

折れ線グラフ

円グラフ

最後の既知の値

状態グラフ

どれでも良いが、ここでは「状態グラフ」を選択してみる

ビューの作成 - コマンド タイルの追加

jinnouchi-iotc-app

検索

バージョン テスト デバイスの管理 公開 名前の変更 削除

デバイス テンプレート > BuzzerTemplate > ビュー

BuzzerTemplate

アプリケーションが更新されました: Never インターフェイスが公開されました: Never

モデル

BuzzerTemplate

BuzzerIF

クラウドのプロパティ

カスタマイズ

ビュー

コマンド

デバイスにコマンドを送信するタイルを作成します。コマンド タイルを作成するために選択できるのは 1 台のデバイスのみです。

① AlarmOn

カスタム タイル

カスタム画像、テキスト、マークダウンを含むタイルを作成します。

☐ 画像

☐ ラベル

② タイルの追加

Alert



ビューの作成 - 保存

jinnouchi-iotc-app

検索



ダッシュボード



デバイス



デバイス グループ



規則



分析



ジョブ



アプリケーション設定



デバイス テンプレ...



データ エクスポート



データ エクスポー...



管理



Azure IoT Central



バージョン



テスト デバイスの管理



公開



名前の変更



削除



デバイス テンプレート > BuzzerTemplate > ビュー

BuzzerTemplate

アプリケーションが更新されました: Never

インターフェイスが公開されました: Never

モデル



保存



削除



プレビュー デバイスの構成

BuzzerTemplate

BuzzerIF

クラウドのプロパティ

カスタマイズ

ビュー

コマンド

デバイスにコマンドを送信するタイルを作成します。コマンド タイルを作成するために選択できるのは 1 台のデバイスのみです。

コマンドを選択する



カスタム タイル

カスタム画像、テキスト、マークダウンを含むタイルを作成します。

☐ 画像

☐ ラベル

タイルの追加

クリア

Alert



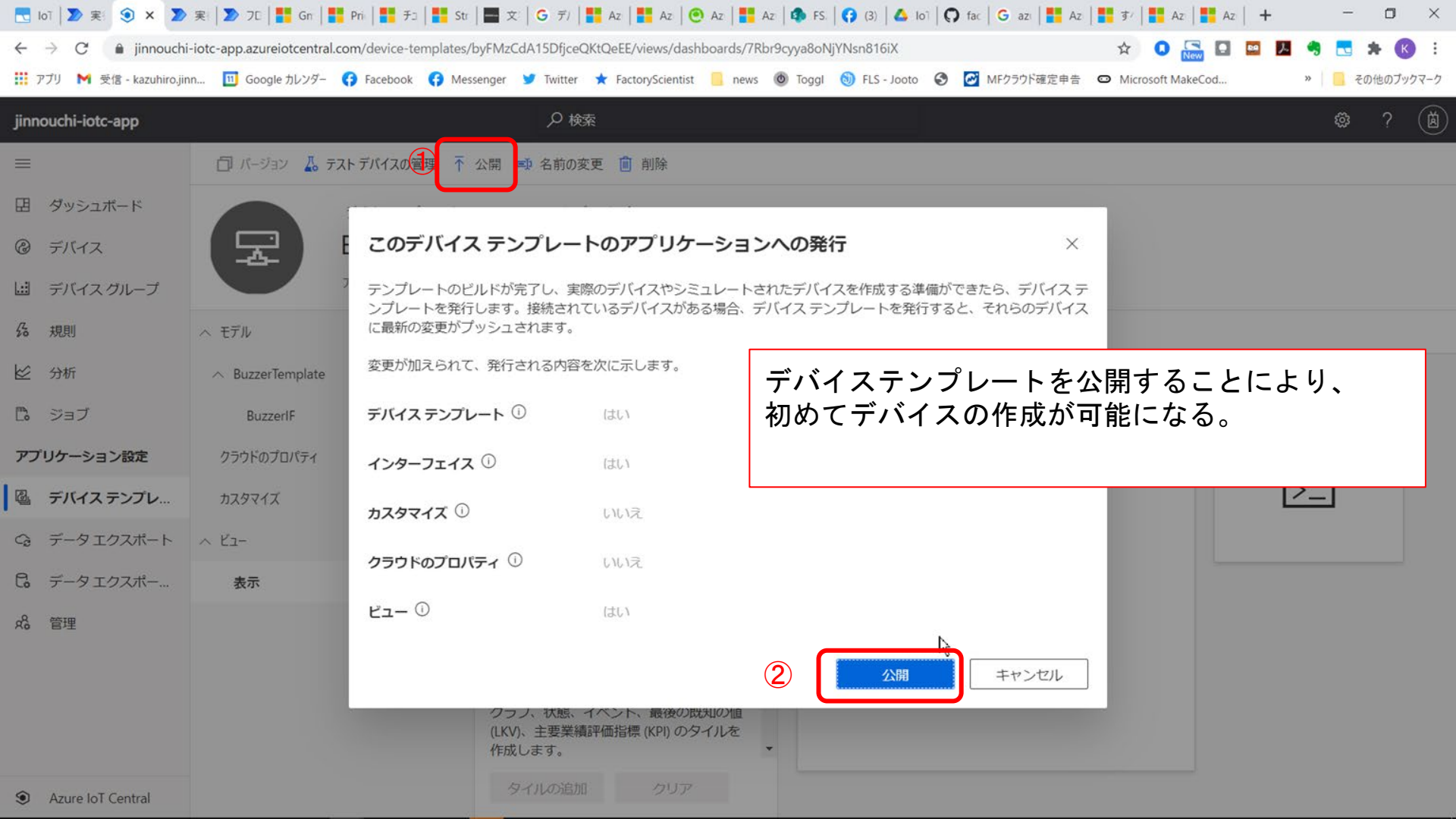
AlarmOn





1.d デバイステンプレートの公開





このデバイス テンプレートのアプリケーションへの発行

テンプレートのビルドが完了し、実際のデバイスやシミュレートされたデバイスを作成する準備ができたなら、デバイス テンプレートを発行します。接続されているデバイスがある場合、デバイス テンプレートを発行すると、それらのデバイスに最新の変更がプッシュされます。

変更が加えられて、発行される内容を次に示します。

デバイス テンプレート ①	はい
インターフェイス ①	はい
カスタマイズ ①	いいえ
クラウドのプロパティ ①	いいえ
ビュー ①	はい

デバイス テンプレートを公開することにより、初めてデバイスの作成が可能になる。

②

公開

キャンセル

A decorative network diagram in the top-left corner, featuring a complex web of interconnected nodes and lines. The nodes are represented by circles of varying sizes, some with concentric rings, and the lines are thin and grey. The overall structure is organic and branching.

1.e デバイスの作成

デバイスの作成

jinnouchi-iotc-app

検索

メニュー

ダッシュボード

① デバイス

デバイス グループ ②

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...

データ エクスポート

データ エクスパー...

管理

Azure IoT Central

デバイス

テンプレートをフィルタリングして...

すべてのデバイス

新規 インポート エクスポート 承認 ブロック ブロック解除 ゲートウェイに接続 移行 削除

① メニューの「デバイス」を選択し
② デバイステンプレートを選択し
③ 「デバイスの作成」へ

デバイスの作成

監視、格納、分析のために、デバイスから IoT Central にデータが送信されます。詳細情報

③ デバイスの作成

デバイスの作成

新しいデバイスを作成します

新しいデバイスを作成するには、デバイステンプレート、名前、一意のIDを選択してください。 [詳細情報](#)

デバイステンプレート*

BuzzerTemplate

デバイス名* ①

Buzzer01

デバイスID* ①

Buzzer01

このデバイスをシミュレートしますか？
シミュレートされたデバイスでは、実際のデバイスに接続する前にアプリケーションの動作をテストできるテレメトリが生成されます。

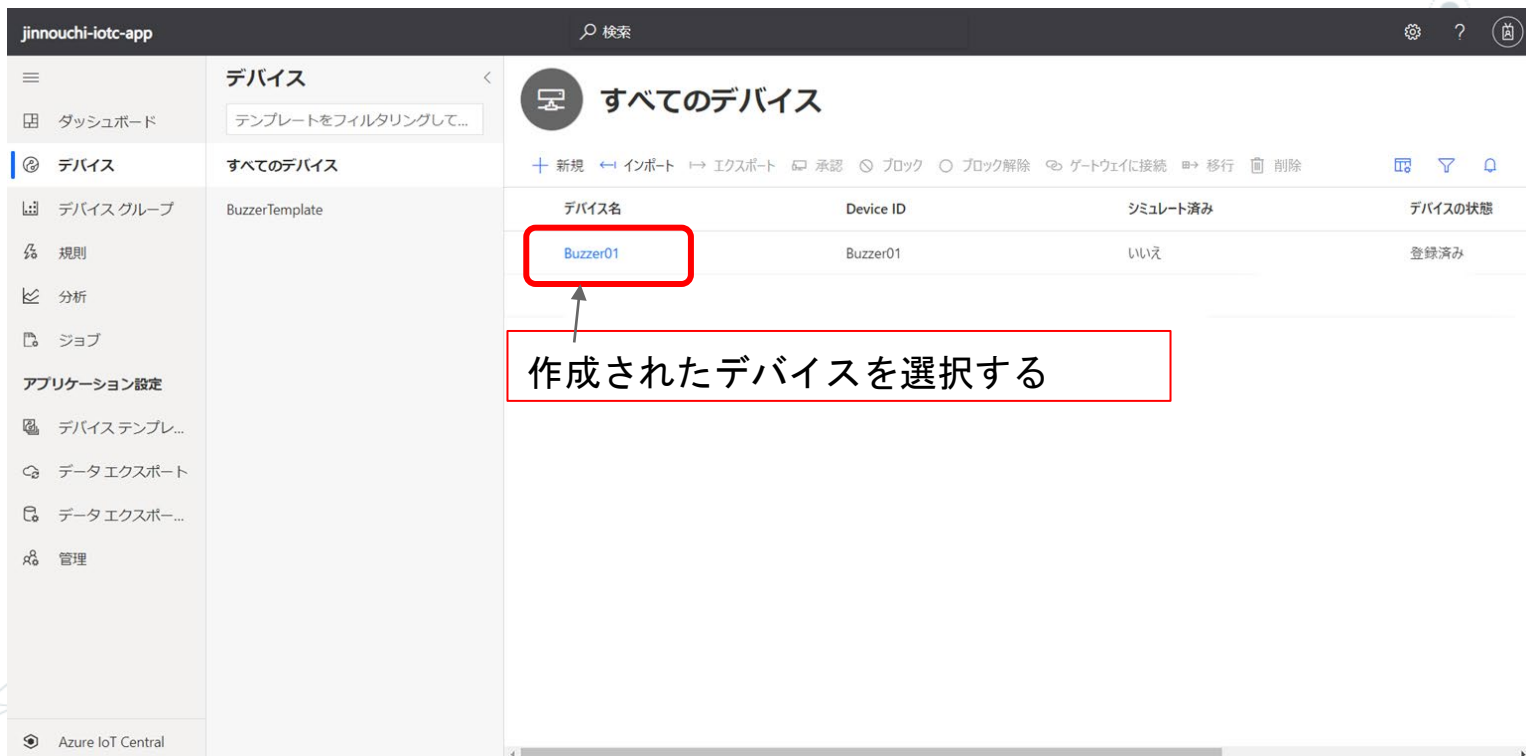
☐ いいえ

* 必須

作成 キャンセル

いいえ

デバイスの作成



The screenshot shows the 'jinnouchi-iotc-app' interface. The left sidebar contains a menu with items like 'ダッシュボード', 'デバイス', 'デバイスグループ', '規則', '分析', 'ジョブ', 'アプリケーション設定', 'デバイステンプレ...', 'データエクスポート', 'データエクスポート...', and '管理'. The main area is titled 'すべてのデバイス' (All Devices) and displays a table of devices. The table has columns for 'デバイス名' (Device Name), 'Device ID', 'シミュレート済み' (Simulated), and 'デバイスの状態' (Device Status). The first row shows 'Buzzer01' with 'Buzzer01' as the Device ID, 'いいえ' (No) for simulated, and '登録済み' (Registered) status. A red box highlights 'Buzzer01' in the 'デバイス名' column, and a red arrow points to it from a text box that says '作成されたデバイスを選択する' (Select the created device).

デバイス名	Device ID	シミュレート済み	デバイスの状態
Buzzer01	Buzzer01	いいえ	登録済み

作成されたデバイスの画面

The screenshot shows the 'jinnouchi-iotc-app' interface. The top navigation bar includes a search icon and the text '検索'. The left sidebar contains a menu with items: 'ダッシュボード', 'デバイス' (selected), 'デバイス グループ', '規則', '分析', 'ジョブ', 'アプリケーション設定', 'デバイス テンプレ...', 'データ エクスポート', 'データ エクスポート...', and '管理'. The main content area displays the 'Buzzer01' device page. The breadcrumb path is 'デバイス > BuzzerTemplate > Buzzer01'. The device name 'Buzzer01' is prominently displayed. Below the name are tabs for '表示' (selected), 'コマンド', and '生データ'. A red box highlights the '接続' (Connect) button in the top right corner of the main content area. An arrow points from this button to another red box containing the text 'デバイスの接続情報を開く'. The main content area also shows a large 'Alert' panel with the text 'データの待機中' (Waiting for data) and a smaller 'AlarmOn' panel with a '>-' icon. The bottom status bar shows 'Azure IoT Central'.

接続

デバイスの接続情報を開く

デバイスの接続情報

The screenshot shows the 'jinnouchi-iotc-app' interface with a sidebar menu on the left containing options like 'ダッシュボード', 'デバイス', 'デバイス グループ', '規則', '分析', 'ジョブ', 'アプリケーション設定', 'デバイス テンプレ...', 'データ エクスポート', 'データ エクスポート...', and '管理'. The main area displays 'デバイス > Buzzer' with a '表示' button. A modal dialog titled 'デバイス接続' is open, showing the following fields:

- ID スコープ ①: 0ne001D7C3C
- デバイス ID ①: Buzzer01
- Authentication type: Shared Access Signature (SAS)
- 主キー ①: /ArFJ+KlJvaGnlcVvF4vQ1GBuO/GScYIVhus+Szlgw=
- セカンダリキー ①: VnbFbT7fauWnQdO4FTG8WAViJOryIM9mvH8Zi0eFI+k=

A red circle highlights the 'ID スコープ', 'デバイス ID', and '主キー' fields. A red box on the right contains the following text:

デバイスの接続情報は、
以下の3つ

- ・ IDスコープ
- ・ デバイスID
- ・ 主キー
またはセカンダリキー

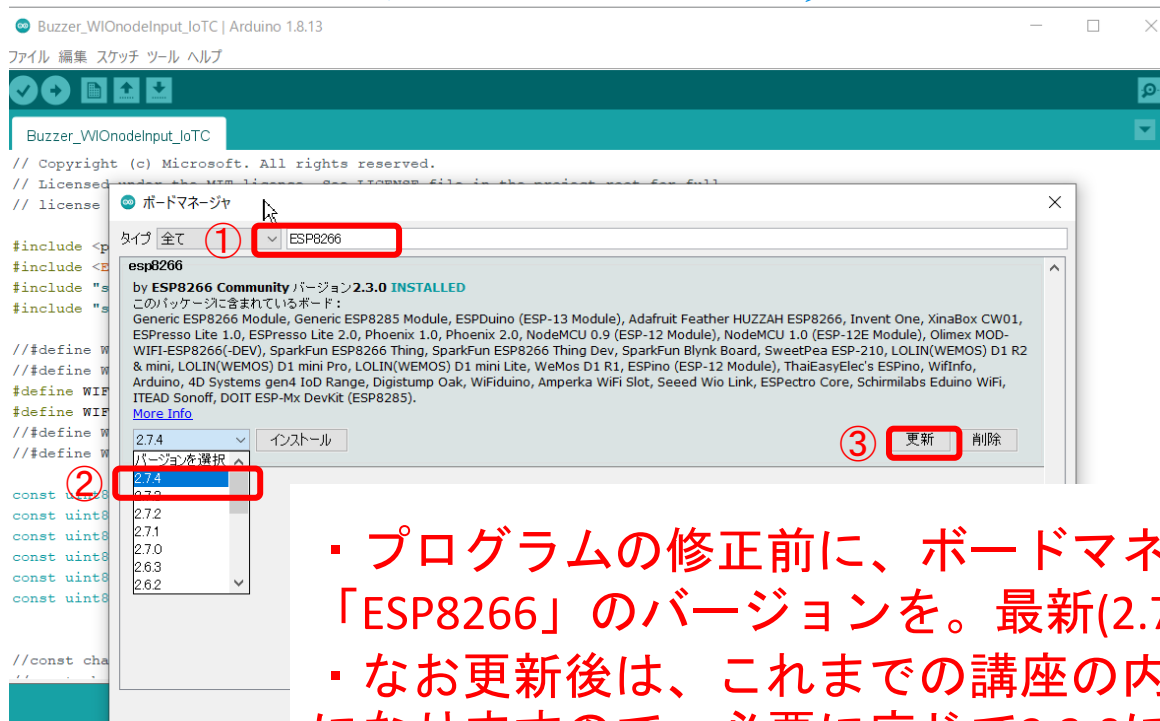


2.Arduinoのプログラムの変更

コードの変更箇所



ESP8266ボードのバージョンアップ



- ・ プログラムの修正前に、ボードマネージャーで、「ESP8266」のバージョンを。最新(2.7.4)に更新します。
- ・ なお更新後は、これまでの講座の内容のコンパイルはエラーになりますので、必要に応じて2.3.0に戻してください。

ボードの設定

WIFI101 / WIFININA Firmware Updater

ボード: "Generic ESP8266 Module"
Builtin Led: "2"
Upload Speed: "115200"
CPU Frequency: "80 MHz"
Crystal Frequency: "26 MHz"
Flash Size: "512KB (FS:32KB OTA:~230KB)"
Flash Mode: "DIO"
Flash Frequency: "40MHz"
Reset Method: "dtr (aka nodemcu)"
Debug port: "Disabled"
Debug Level: "なし"
lwIP Variant: "v2 Lower Memory"
VTables: "Flash"
Exceptions: "Legacy (new can return nullptr)"
Erase Flash: "Only Sketch"
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"
SSL Support: "All SSL ciphers (most compatible)"
シリアルポート
ボード情報を取得

書き込み装置

ブートローダを書き込む

設定内容

Builtin LED: "2"

Upload Speed:"115200"

CPU Frequency:80 MHz"

Crystal Frequency:"26MHz"

Flash Size:"512KB(FS:32KB OTA-230KB)"

Flash Mode:"DIO"

Flash Frequency:"40MHz"

Reset Method:"dtr(aka nodemcu)"

Debug port:"Disabled"

Debug Level:"なし"

lwIP Variant:"v2 Lower Memory"

VTable:"Flash"

Exceptions:"Legacy(new can return nullptr)"

Erase Flash:"Only Sketch"

Espressif FW:"All SSL chipers(most compatible)"

プログラムのダウンロードと修正

- ・ 以下のリンク先の「Buzzer_WIOnodeInput_loTC」フォルダー式をダウンロードします。

[Buzzer](#)

- ・ Buzzer_WIOnodeInput_loTC.ino の以下の個所を、ご利用のWiFi環境、及び作成したデバイスの接続情報に併せて修正します。

```
#define WIFI_SSID "<ENTER WIFI SSID HERE>"
#define WIFI_PASSWORD "<ENTER WIFI PASSWORD HERE>"

const char* SCOPE_ID = "<ENTER SCOPE ID HERE>";
const char* DEVICE_ID = "<ENTER DEVICE ID HERE>";
const char* DEVICE_KEY = "<ENTER DEVICE primary/secondary KEY HERE>";
```



3.IoT Centralからブザーを鳴らす



Arduino のシリアルモニタでの動作確認



```
*** Wake up WioNode...
- Connecting to WiFi...
- - iotc.dps : getting auth...
- - iotc.dps : getting operation id...
- - iotc.dps : getting host name...
-
hostname: iotc-d14e00d8-219f-4f89-9a6f-1b00e809c295.azure-devices.net
deviceId: Buzzer01
username: iotc-d14e00d8-219f-4f89-9a6f-1b00e809c295.azure-devices.net/Buzzer01/api-version=2016-11-14
password: SharedAccessSignature sr=iotc-d14e00d8-219f-4f89-9a6f-1b00e809c295.azure-devices.net%2FBuzzer01&sig=tdC

- Event recieved 1
- Is connected ? YES (64)
- Event recieved 2
- - [MessageSent] event was received. Payload => [{"Alert":0}]

- deviceTwinGetStateCallback
- topicName $iothub/twin/res/200/?$rid=0
- payload {"desired":{"$version":1},"reported":{"$version":1}}
- Event recieved 3
- - [SettingsUpdated] event was received. Payload => [{"desired":{"$version":1},"reported":{"$version":1}}]
```

正常に起動すると、次のような表示が出ます。

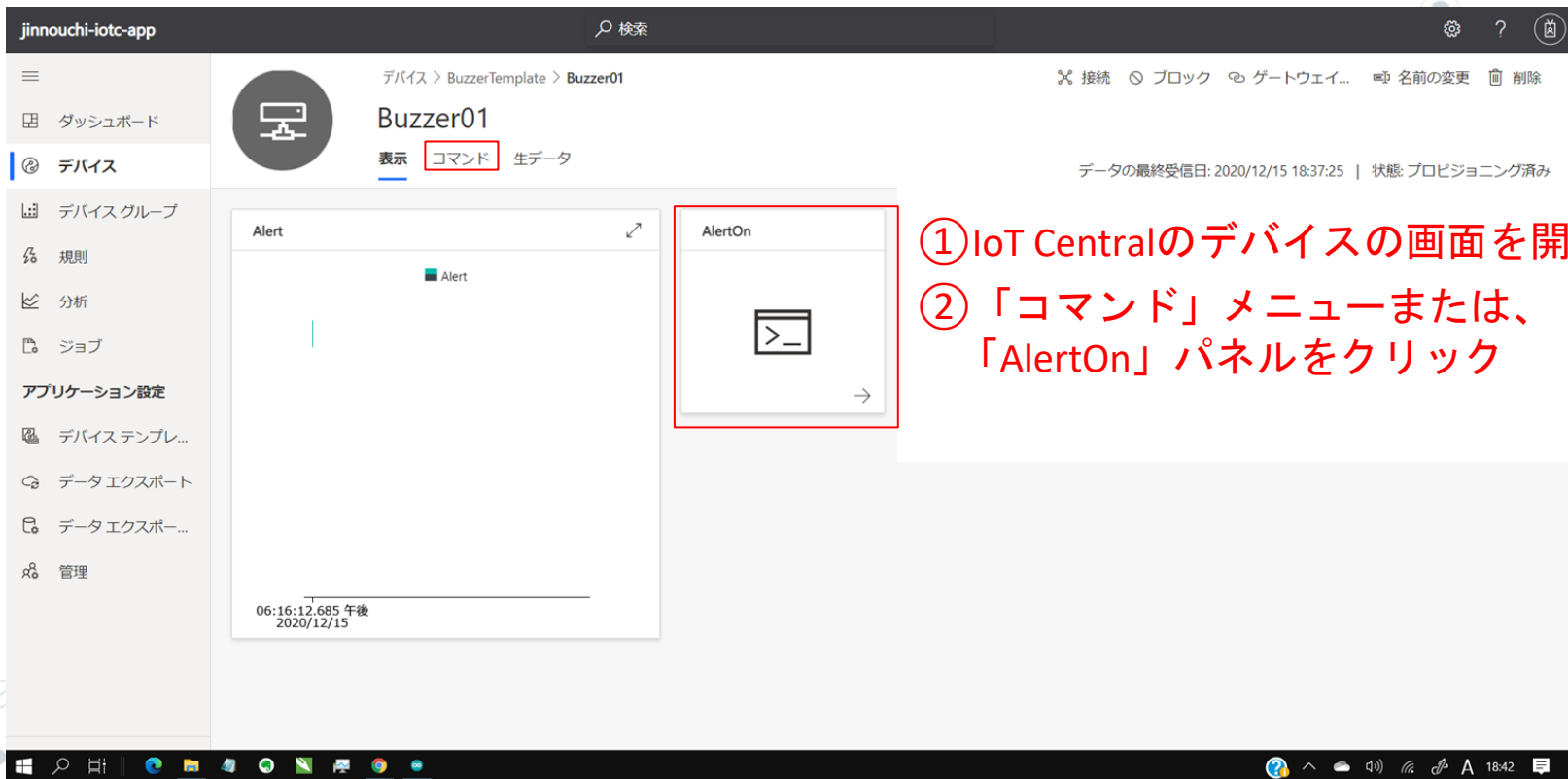
IoT Centralへの接続成功

Alertの初期化成功

☒ 自動スクロール ☐ タイムスタンプを表示

CRおよびLF 115200 bps 出力をクリア

IoT Central からブザーを鳴らす



jinnouchi-iotc-app

検索

三

ダッシュボード

デバイス

デバイス グループ

規則

分析

ジョブ

アプリケーション設定

デバイス テンプレ...

データ エクスポート

データ エクスポート...

管理

デバイス > BuzzerTemplate > Buzzer01

Buzzer01

表示 コマンド 生データ

接続 ブロック ゲートウェイ... 名前の変更 削除

データの最終受信日: 2020/12/15 18:37:25 | 状態: プロビジョニング済み

Alert

Alert

AlertOn

→

06:16:12.685 午後
2020/12/15

①IoT Centralのデバイスの画面を開く

②「コマンド」メニューまたは、「AlertOn」パネルをクリック

IoT Central からブザーを鳴らす

jinnouchi-iotc-app

検索

デバイス > BuzzerTemplate > Buzzer01

接続 ブロック ゲートウェイ... 名前の変更 削除

Buzzer01

表示 コマンド 生データ

データの最終受信日: 2020/12/15 18:37:25 | 状態: プロビジョニング済み

実行

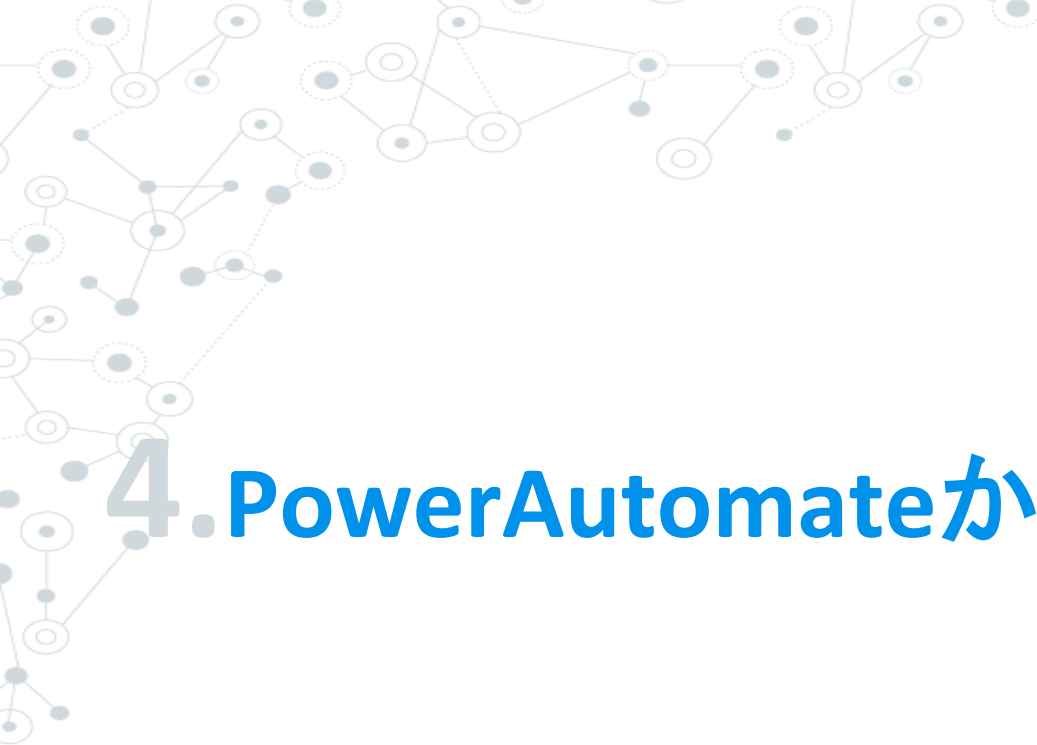
このボタンを押すとブザーが鳴ります。

実行

このボタンでブザーが停止します。

ブザーの停止は、WioNodeのFuncボタンを長押しによっても、できます。

Azure IoT Central



4. PowerAutomateからブザーを鳴らす



Power Automate の設定

役に応じたリソースを検索してください

環境 factoryscientist.com (既...)

保存 フロー チェッカー テスト

AlertOn

手動でフローをトリガーします

アクションを選択してください

IoT

すべて 組み込み 標準 プレミアム カスタム 自分のクリップボード

Azure DevOps Azure IoT Central V2 Azure IoT Central V3 FORCAM FORCE...

トリガー アクション もっと見る

作業項目を更新する PREMIUM Azure DevOps

作業項目を作成する PREMIUM Azure DevOps

新しいビルドをキューに入れる PREMIUM Azure DevOps

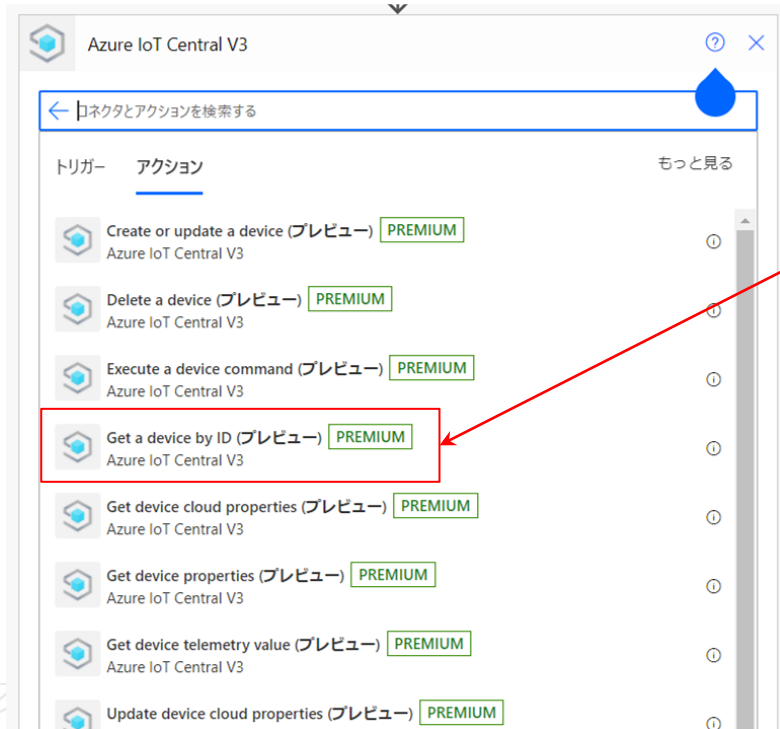
新しいリリースを作成する PREMIUM Azure DevOps

①「新しいステップ」でアクションの追加で、「IoT」を検索します。

②「Azure IoT Central V3」を選択します。

③IoT Centralへの認証情報を聞かれるので入力します。

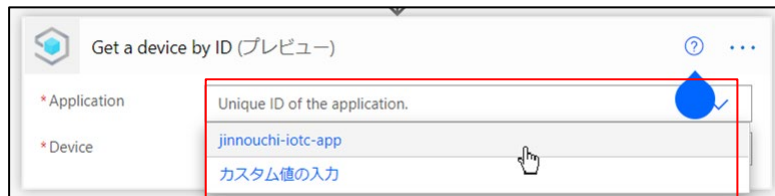
Power Automate の設定



まず、IoT Centralへの接続を確認するために、一番簡単なアクションで接続確認します。

「Get a device by ID」を選択します。

Power Automate の設定

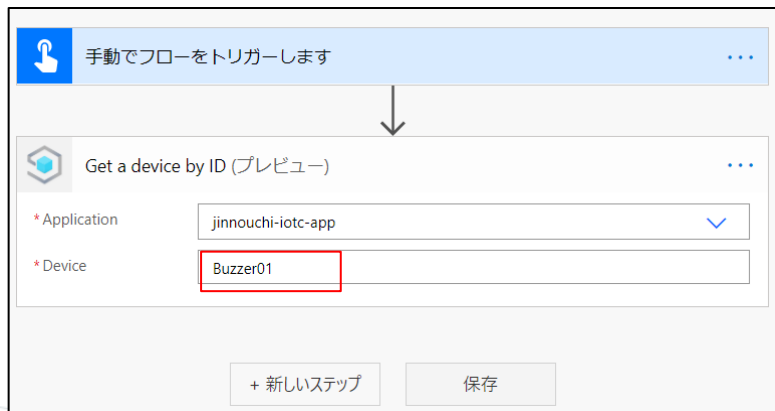


Get a device by ID (プレビュー)

* Application Unique ID of the application.
jinnouchi-iotc-app
カスタム値の入力

* Device

Application : プルダウンから「IoT Central アプリケーション名」を選択します。



手でフローをトリガーします

Get a device by ID (プレビュー)

* Application jinnouchi-iotc-app

* Device Buzzer01

+ 新しいステップ 保存

Device : デバイス名を入力します。

テスト実行

環境
factoryscientist.com (既...

トリガーします

ID (プレビュー)

jinnouchi-iotc-app

Buzzer01

+ 新しいステップ

保存

フローの実行

Buzzer
所有者: 陣内 和宏

サインイン
このフローは次のアプリを使用します。緑のチェックは準備完了を意味します。

Azure IoT Central V3

続行 キャンセル

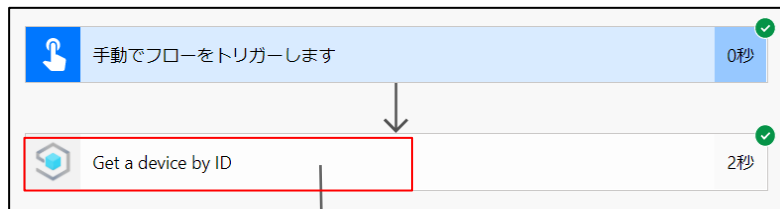
環境
factoryscientist.com (既...

フローの実行

Buzzer
所有者: 陣内 和宏

このフローでは Azure IoT Central V3 を使用しています。
[接続とアクションを確認する](#)

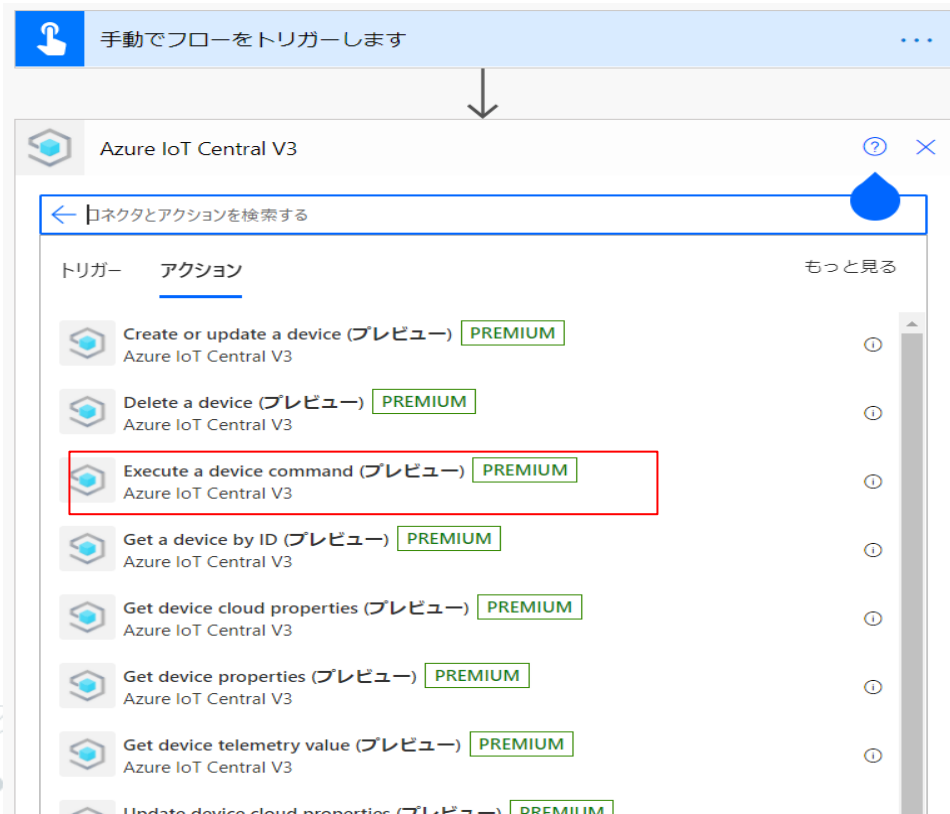
フローの実行 キャンセル



A screenshot of the "Get a device by ID" step results. The step name and duration "2秒" are at the top. Below is a section titled "入力" (Input) containing two fields: "Application" with the value "jinnouchi-iotc-app" and "Device" with the value "Buzzer01". Below this is a section titled "出力" (Output) containing two fields: "Device Name" with the value "Buzzer01" and "Device Template" which is currently empty.

正常に実行できたら、このような結果が表示されます。

Power Automate からブザーを鳴らす



IoT Centralへの接続ができることが確認出来たら、フローを作りなおします。

「Execute a device command」を選択します。

Power Automate からブザーを鳴らす

Execute a device command (プレビュー)			
* Application	Unique ID of the application.	①	▼
* Device	Unique ID of the device.	②	
* Device Component	Name of the device component.	④	▼
* Device Command	Name of this device command.	⑤	▼
Device Template	The device template definition for the device.	③	▼

設定の順番が重要です。
③を設定しないと、④、⑤の選択肢が
出てきません。

Execute a device command (プレビュー)

* Application: jinnouchi-iotc-app

* Device: Buzzer01

* Device Component: Name of the device component.
 'Device Component' が必要です。

* Device Command: Name of this device command.
 'Device Command' が必要です。

Device Template: The device template definition for the device.

Connection timeout: dtmi:modelDefinition:icdlcwh:tnpnjo6yya
 カスタム値の入力

Command response timeout: response timeout in seconds to wait for a command completion on a device.

Device Command Request Payload: The payload for the device command.

Device Templateは、設定したIDではなく、このような内部文字列が出てきますが、選択肢から設定します。

Execute a device command (プレビュー)

* Application: jinnouchi-iotc-app

* Device: Buzzer01

* Device Component: Name of the device component.
 BuzzerIF

* Device Command: カスタム値の入力
 'Device Command' が必要です。

Device Template: dtmi:modelDefinition:icdlcwh:tnpnjo6yya

Device Componentには、インターフェース名「BuzzerIF」を設定します。



Execute a device command (プレビュー)

* Application: jinnouchi-iotc-app

* Device: Buzzer01

* Device Component: BuzzerIF

* Device Command: Name of this device command.

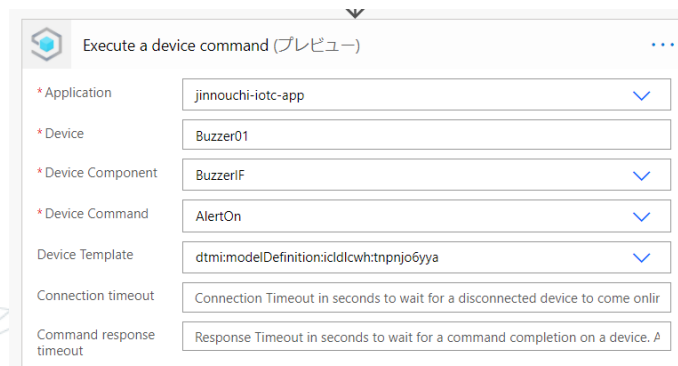
Device Template: AlertOn

Connection timeout: カスタム値の入力

Command response timeout: Response Timeout in seconds to wait for a command completion on a device. /

Device Command Request Payload: The payload for the device command.

Device Commandには、「AlertOn」を設定します。



Execute a device command (プレビュー)

* Application: jinnouchi-iotc-app

* Device: Buzzer01

* Device Component: BuzzerIF

* Device Command: AlertOn

Device Template: dtmi:modelDefinition:icldlcwh:tnpnjo6yya

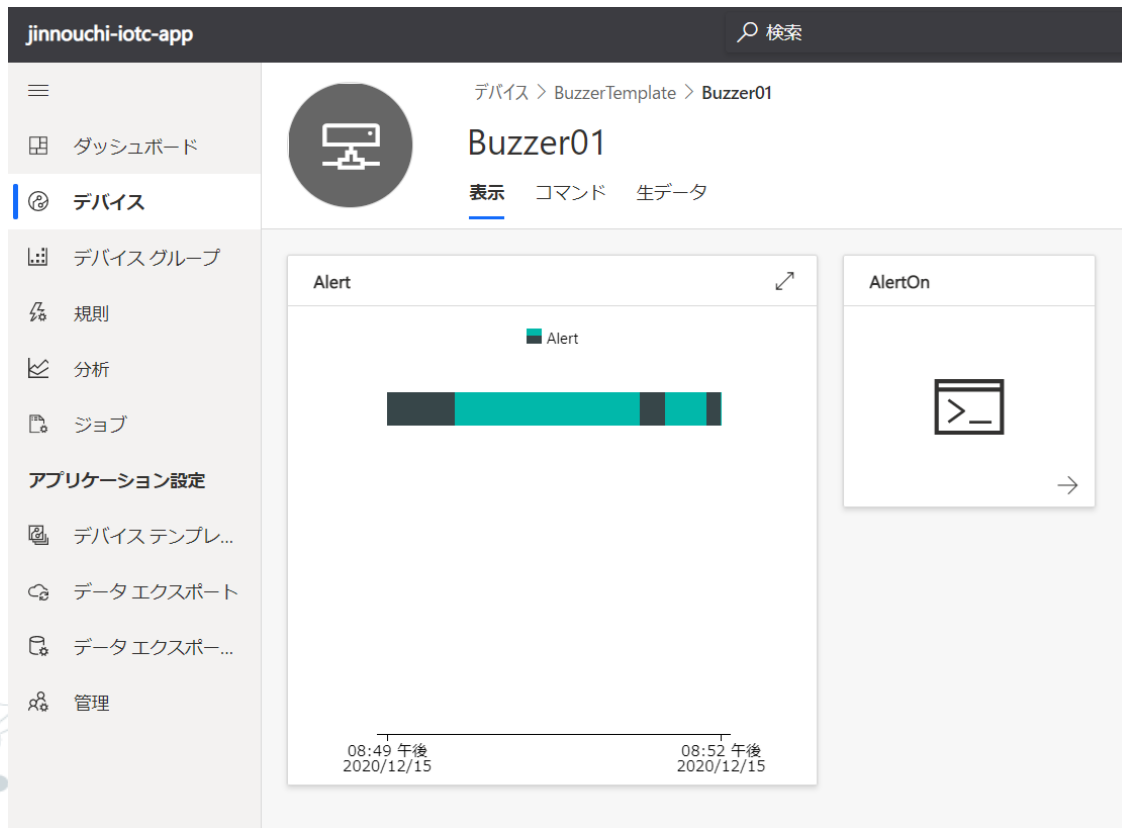
Connection timeout: Connection Timeout in seconds to wait for a disconnected device to come onlinr

Command response timeout: Response Timeout in seconds to wait for a command completion on a device. /

最終的に、このように設定できればOK

テスト実行してみてください。

IoT Central でブザーの状況を確認



何回か、ブザーを鳴らしたり、止めたりした後、IoT Central でデバイスを開くと、このように、ブザーのオン、オフの状況が表示されます。

参考資料

[Azure IoT Central のドキュメント](#)
[デバイス テンプレートとは](#)

[Connection a cheap ESP8266 to Azure IoT Central](#)

[Azure/iot-central-firmware](#) ←本書はこのサンプルプログラムをもとにしている

。