

# Cloud Robotics FX V2

セットアップガイド ver.0.9

# 利用条件

本ハンズオンセッションで利用する Cloud Robotics FX V2 は、サンプルコードであり、OSS として提供されます。

このサンプルコードは、MIT License の下で提供され、利用者の責任においてのみ、利用されるものとなります。

# 利用ガイド

Cloud Robotics Azure Platform V1 SDK のハンズオン資料に則って、既に環境を作成されている方は、ディプロイされた Cloud Robotics FX (V1) = Cloud Service を削除（或いは、停止）して頂き、Cloud Robotics FX V2 = Service Fabric の環境だけを追加で作成して頂ければ、他の環境 (IoT Hub, SQL Database, ... ) をそのまま流用することができます。

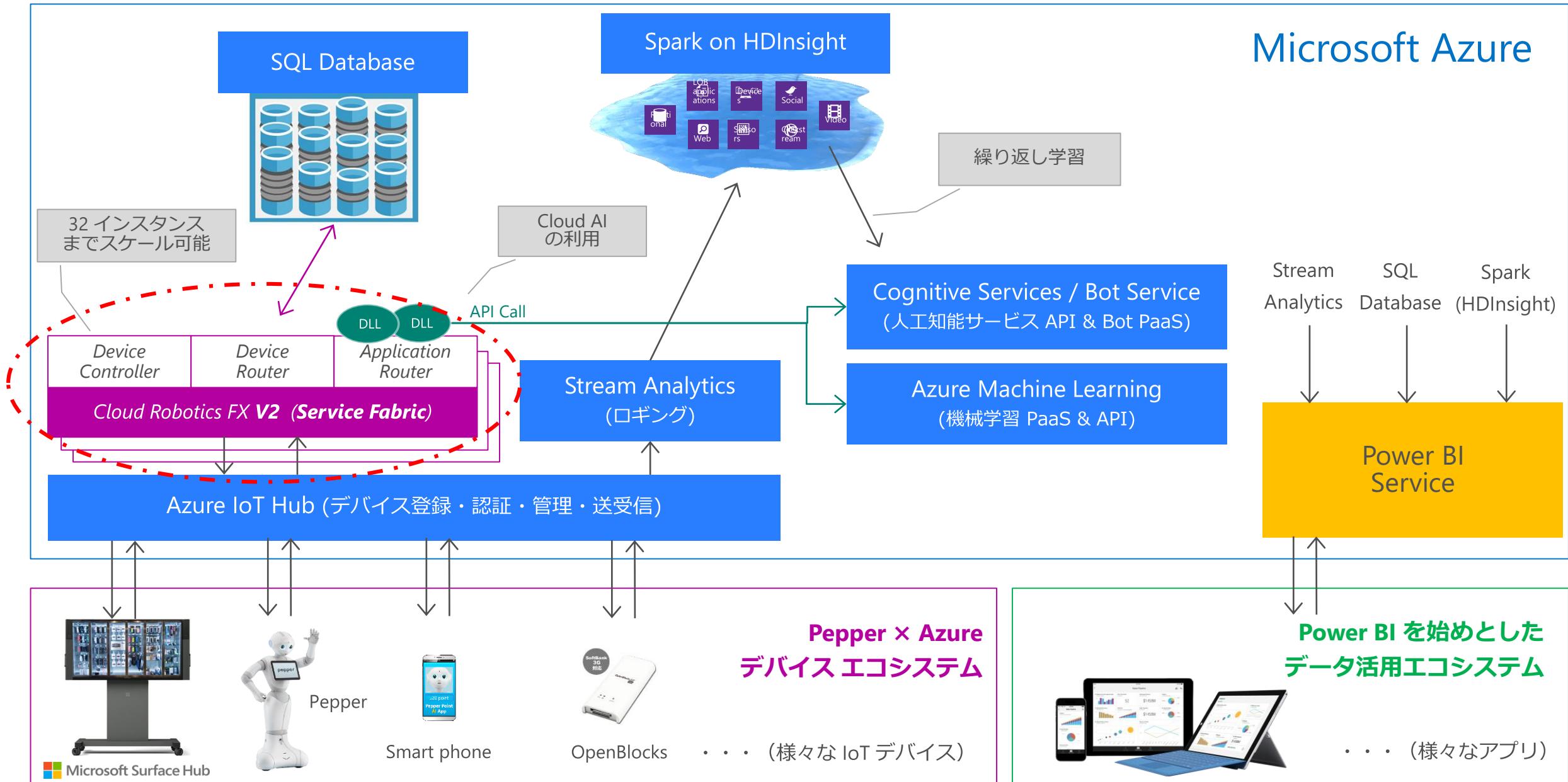
Tutorial についても、Cloud Robotics FX 部分のみの作業で完了となります。

# 事前準備

- ① Azure サブスクリプションをご用意ください
- ② Azure Portal ([portal.azure.com](https://portal.azure.com)) へのログインの確認と権限の確認（仮想マシンなどを作成できるか）をお願い致します
- ③ Windows PC (Windows 7 以降、推奨は、Windows 10) をご用意ください
- ④ Visual Studio 2015 もしくは、Visual Studio 2017 をご用意ください。Community 版は、非商用目的でご利用頂けます。  
<https://www.microsoft.com/ja-jp/dev/products/community.aspx>
- ⑤ Azure SDK for Visual Studio 2015 のインストールをお願い致します  
<https://go.microsoft.com/fwlink/?LinkId=518003&clcid=0x411>
- ⑥ Visual Studio のバージョンに合わせた Azure Service Fabric SDK とツールのインストールをお願い致します  
<https://docs.microsoft.com/ja-jp/azure/service-fabric/service-fabric-get-started>
- ⑦ Microsoft Azure Storage Explorer のインストールをお願い致します  
<http://storageexplorer.com/>
- ⑧ SQL Server Management Studio のインストールをお願い致します  
<https://msdn.microsoft.com/en-us/library/mt238290.aspx>
- ⑨ Power BI サービスへのサインアップ & サインをお願い致します (option)  
<http://powerbi.com/>  
※会社のメールアドレスで登録をお願い致します（組織アカウントのみ有効です = マイクロソフトアカウントやフリームのアドレスは利用できません）

# Cloud Robotics Azure Platform

■ Azure PaaS  
■ Robotics SDK



# Cloud Robotics FX V2 (App Framework Sample Code)

- Service Fabric - Reliable Service (Stateful Service) ベース
  - Azure SQL Database や DocumentDB なども利用している基盤
    - <https://docs.microsoft.com/ja-jp/azure/service-fabric/>
- Device Controller
  - Device への制御情報伝達
    - Device 起動直後の初期処理
    - Device 上のアプリ更新の制御
- Device Router
  - Device 間のルーティング
    - 1 : 1 の通信
    - 1 : N (グループ) の通信
  - ルーティング定義がなければ、何もしない
- Application Router
  - Device からの受け取ったデータを処理するクラス (DLL) を BLOB から動的に読み込む
  - 出力結果は、Device Router に出力先を任せるか、データ処理用クラス (DLL) から指定することも可能
  - よく利用されるシナリオに対するデータ処理用クラス (DLL) は、パッケージとして用意

# Cloud Robotics SDK

- Cloud Robotics FX ソースコード
- Cloud Robotics FX (IoT Hub) との通信手順サンプルコード
  - C# (.NET), Python
- Cloud Robotics Definition & Management Tool
  - Device の登録・変更・削除/ステータス確認と変更/メッセージ内容確認
  - Device Routing の登録・変更・削除
  - Device Group の登録・変更・削除
  - Application の登録・変更・削除/ステータス確認と変更/DLL のディプロイ
  - Application Routing の登録・変更・削除
  - Cloud Robotics FX のトレースログ参照
  - Cloud Robotics FX に対応したデバイス (Pepper 含む) の送受信用シミュレータ
- Cloud Robotics Test Driver
  - Cloud Robotics FX 上で動作するアプリ (DLL) の単体テスト用テストドライバー
- Cloud Robotics Load Test New
  - IoT Hub + Cloud Robotics FX に対する負荷発生ツール
- Cloud Robotics API
  - 標準的なシナリオベースの Cognitive Services ベースの App DLL

# Cloud Robotics Definition Tool

## GUI 管理ツール (IoT Hub Device 管理機能含む)

The screenshot shows the main interface of the Cloud Robotics Definition & Management Tool. The top navigation bar includes tabs for Connection, Device Master, Device Group, Device Routing, App Master, App Routing, and FX Trace Log. The 'Device Master' tab is currently selected. Below the tabs is a toolbar with buttons for Create, Refresh, Update, Delete, and SAS Token... A dark blue callout box labeled 'デバイス管理機能' (Device Management Function) points to the 'Device Master' tab.

**IoT Hub Devices | Device Master Table**  
Total: 12

ID	PrimaryKey	SecondaryKey	ConnectionString	Status	Created Date	Updated Date
hub01	InJirmWhLF8...	3+QHntax35...	HostName=s...	Disconnected	2016/11/05 5...	2016/11/05 5...
hub03	bekOKoYG+3...	spPvoUtXpfEt...	HostName=s...	Disconnected	2016/05/22 4...	2016/06/28 2...
hub04	oE7Y3Zgxeu...	zxr1bmWHd3...	HostName=s...	Disconnected	2016/05/20 1...	2016/05/20 1...
hub_test01	5W0Tsoj84Jn...	T0al447wTFV...	HostName=s...	Disconnected	2016/11/05 1...	2016/11/05 1...
NetAppCtl	qrahqBZHKO...	Xspo5vCJIRK...	HostName=s...	Disconnected	2016/05/24 5...	2016/06/29 4...
pepper009	Q08zugVDx9r...	aom+jCdnBC...	HostName=s...	Disconnected		
pepper01	1b0lawOLDxl...	67-PA7L+L5...	HostName=s...	Disconnected	2016/11/05 5...	2016/11/05 5...
pepper02	icN8DPkxGC2...					
pepper03	ToHDijpcj5tw...					
pepper04	ULgT8Ct2s7G...					
pepper_test01	C9HQtpB4aY...					
UWPApp01	0ZMoM0a1L5...					

**Device Update Form**

IoT Hub Device

Device ID: pepper01

Primary Key:

Secondary Key:

RBFX Device Master Table

Device Type: Pepper

Status: Active

Resource Group ID:

Description: Pepper 沙留1号機

Buttons: Update, Restore, Cancel

The screenshot shows the 'Device Group' tab selected in the navigation bar. A dark blue callout box labeled 'デバイスグループ管理機能' (Device Group Management Function) points to the 'Device Group' tab.

**Device Group Table**  
Device Group ID (Filter): \* List Total: 8

DeviceGroupId	DeviceId	Registered_Date
DG01	DEVTEST02	2016/04/13
DG01	DEVTEST03	2016/04/13
DG01	hub01	2016/09/10 0...
hubgroup01	hub01	2016/09/16 1...
hubgroup01	hub03	2016/09/16 1...
hubgroup01	hub04	2016/09/16 1...
pepperG01	pepper01	2016/05/17
pepperG01	pepper02	2016/05/17

**Edit Device Group**

Device Group ID: DG01

Device List

- hub01
- hub03
- hub04
- NetAppCtl
- pepper01
- pepper02
- pepper03
- pepper04
- UWPApp01

Selected Device List

- DEVTEST02
- DEVTEST03

Buttons: Create, Update, Cancel

# Cloud Robotics Device Simulator

## デバイスシミュレーター

Cloud Robotics Definition & Management Tool

Connection Device Master Device Group Device Routing App Master App Routing FX Trace Log

Actions  
Create Refresh Update Delete SAS Token...

IoT Hub Devices | Device Master Table  
Total: 12

Id	PrimaryKey	SecondaryKey	ConnectionString	ConnectionState	LastActivityTime	LastConnection:
hub01	InJjrmWhLF8...	3+QHNtax35...	HostName=s...	Disconnected	2016/11/05 5...	2016/11/05 5...
hub03	bekOKoYG+3...	spPvoUtxpfe...	HostName=s...	Disconnected	2016/05/22 4...	2016/06/28 2...
hub04	oE7Y3Zgxceu...	zxr1bmWHD3...	HostName=s...	Disconnected	2016/05/20 1...	2016/05/20 1...
hub_test01	5W0Tsoj84Jn...	T0al447wTFV...	HostName=s...	Disconnected	2016/11/05 1...	2016/11/05 1...
NetAppCtl	qrahqBZHKOj...	Xspo5vCJIRK...	HostName=s...	Disconnected	2016/05/24 5...	2016/06/29 4...
pepper009	Q08zugVDx9r...	aom+jCdnBC...	HostName=s...	Disconnected	2016/05/24 5...	2016/06/29 4...
pepper01	1tC1GmD...	3+QHNtax35...	HostName=s...	Disconnected	2016/11/05 5...	2016/11/05 5...
pepper02				Disconnected	2016/05/17 1...	2016/05/24 1...
pepper03				Disconnected	2016/05/24 5...	2016/05/24 5...
pepper04				Disconnected	2016/05/24 5...	2016/05/24 5...
pepper_test				Disconnected	2016/11/05 1...	2016/11/05 1...
UWPApp01	UZM0M0d1Cj...	VX485gGD0A...	HostName=s...	Disconnected	2016/05/24 5...	2016/05/23 4...

Launch Device Simulator

Copy data for all device

Copy data for selected device

Copy connection string for selected device

デバイスシミュレーター

Device Simulator

```
{"RbHeader": { "RoutingType": "D2D", "RoutingKeyword": "Default", "AppId": "PepperShopApp", "AppProcessingId": "", "MessageId": "detect", "MessageSeqno": "1", "SendDateTime": "2016-11-08 15:29:42.593" }, "RbBody": { "visitor": "u001", "gender": "f", "age": "29" }}
```

pepper Detect

Send to IoT Hub Receive from IoT Hub (Waiting) Launch Message Edit Form

IoT Hub Host: [REDACTED].azure-devices.net

DeviceId: pepper01 DeviceKey: [REDACTED]

Device Simulator

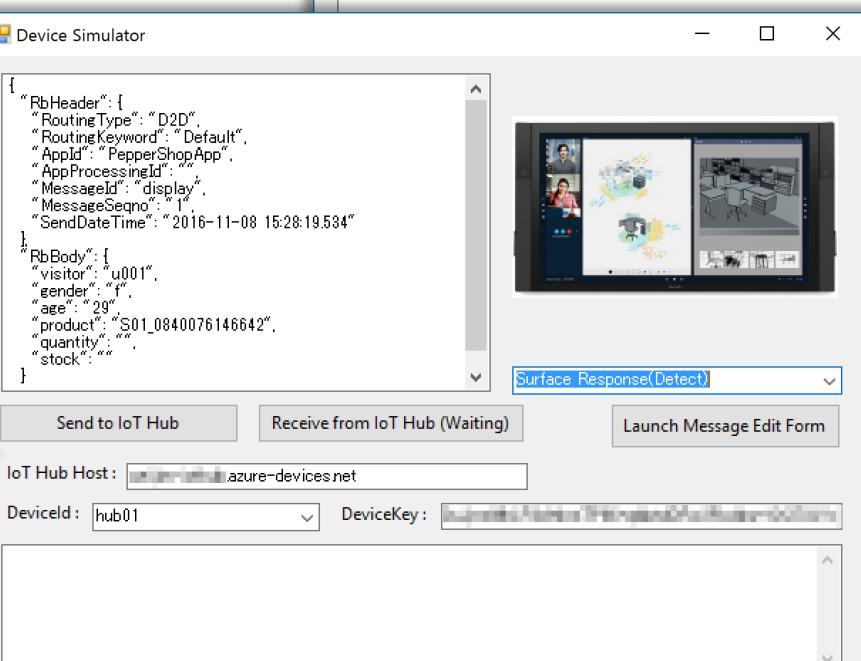
```
{"RbHeader": { "RoutingType": "D2D", "RoutingKeyword": "Default", "AppId": "PepperShopApp", "AppProcessingId": "", "MessageId": "display", "MessageSeqno": "1", "SendDateTime": "2016-11-08 15:28:19.534" }, "RbBody": { "visitor": "u001", "gender": "f", "age": "29", "product": "S01_0840076146642", "quantity": "", "stock": "" }}
```

Surface Response(Detect)

Send to IoT Hub Receive from IoT Hub (Waiting) Launch Message Edit Form

IoT Hub Host: [REDACTED].azure-devices.net

DeviceId: hub01 DeviceKey: [REDACTED]



# Cloud Robotics Power BI Template

The screenshot shows a Power BI dashboard titled "CRobo\_Dashboard". The dashboard includes the following components:

- イベント数**: A large number 1328 displayed prominently.
- 性別比率**: A donut chart showing gender distribution with labels m, k, and f.
- カテゴリ別売上比率**: A pie chart showing sales distribution across categories: ヘルスケア (large teal), イヤホ... (small grey), and iPhoneケース (small red).
- 年代別来店者数**: A bar chart showing the number of visitors by age group: 10代 (3), 20代 (182), 30代 (26), 40代 (12), and 50代以上 (14). The total percentage is 466.7%.
- 時間帯別売上額**: A bar chart showing sales volume by time period (10 to 23) in thousands of yen.
- IoT Hub Log**: A table listing log entries from an IoT hub, including columns like localdate, localhour, localminute, localsecond, iothub\_conn\_id, rbf\_message\_type, visitor, gender, age, emotion, product, quantity, and stock.
- 興味のある色**: A color palette section featuring "Candy Stripe Multi/Blue" as the primary color, with other colors like Pink/White, グリーン, Fuchsia Pink, etc., shown in smaller squares.



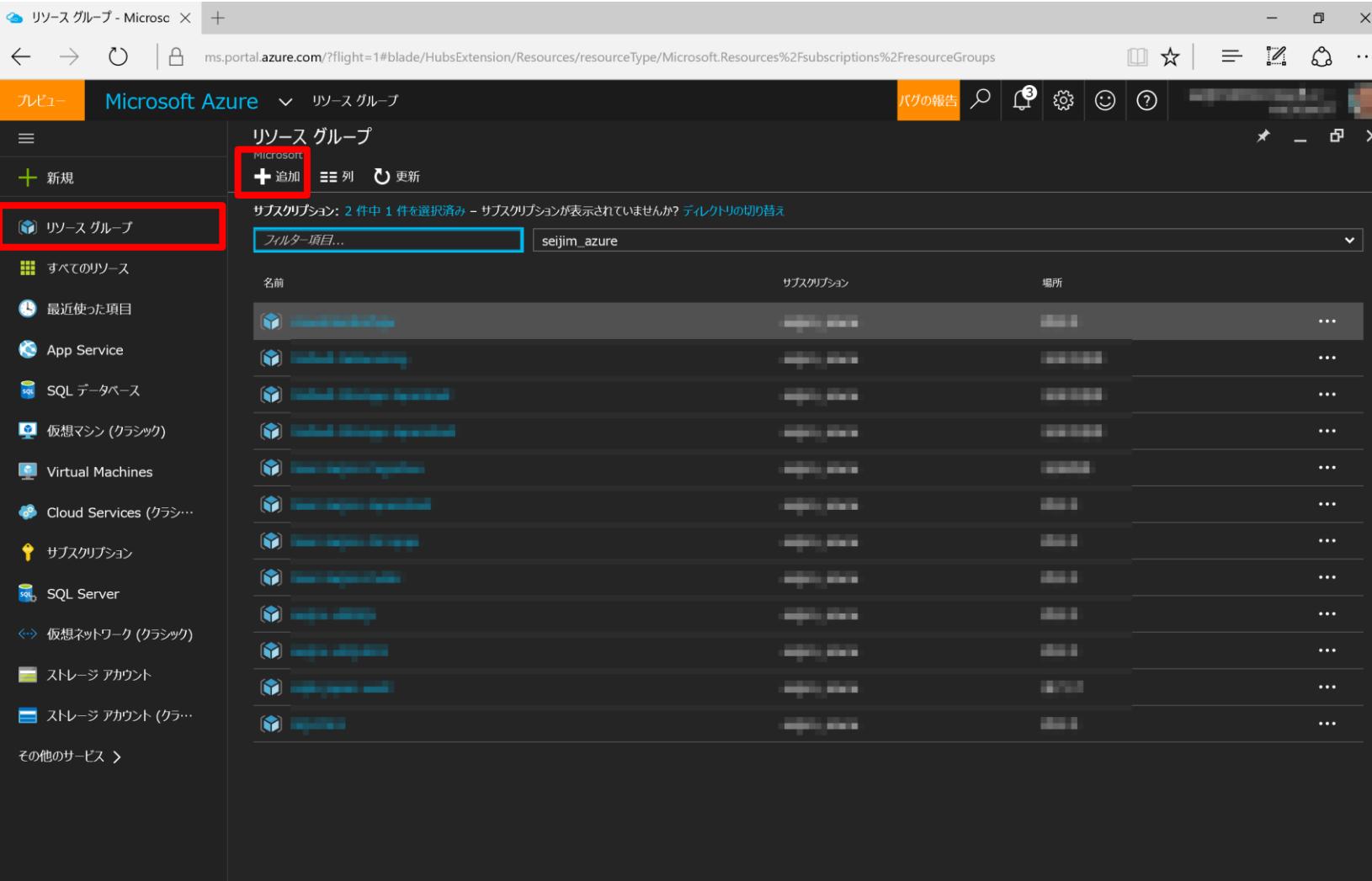
# Tutorial 01

## リソースグループ & ストレージの作成

# T01\_01：リソースグループの新規作成

Azure ポータル (portal.azure.com) にログインし、該当のサブスクリプションかどうか確認

- [リソースグループ]→[+追加]

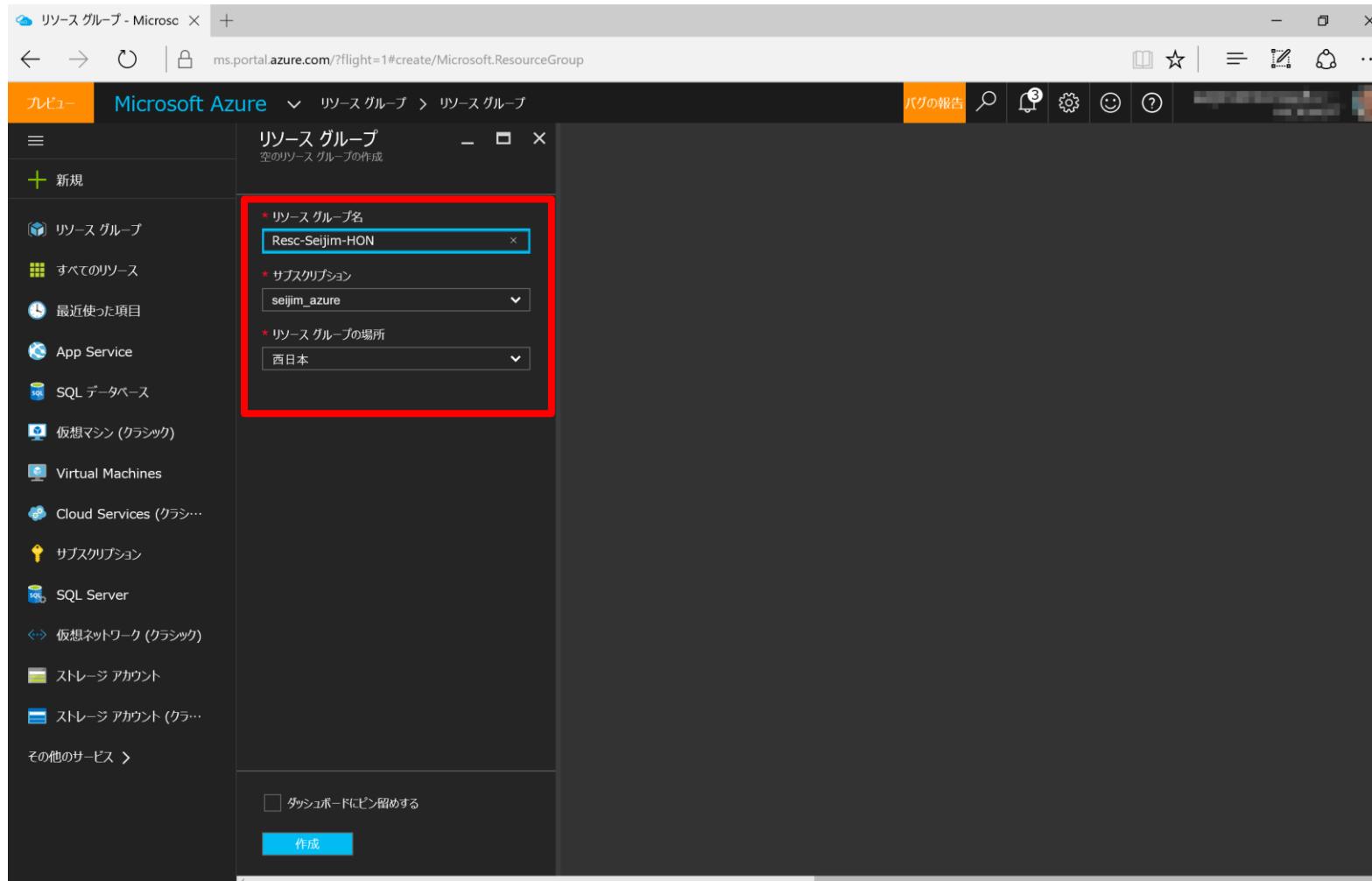


The screenshot shows the Microsoft Azure portal interface. The title bar reads "リソース グループ - Microsoft Azure". The address bar shows the URL "ms.portal.azure.com/?flight=1#blade/HubsExtension/Resources/resourceType/Microsoft.Resources%2Fsubscriptions%2FresourceGroups". The left sidebar has a red box around the "Resource Groups" link under the "New" section. The main content area is titled "Resource Groups" and shows a list of resource groups. At the top right of this area, there is another red box around the "+ Add" button.

# T01\_02：リソースグループ作成パラメータ入力

以下のように設定し、[作成] ボタンを押します

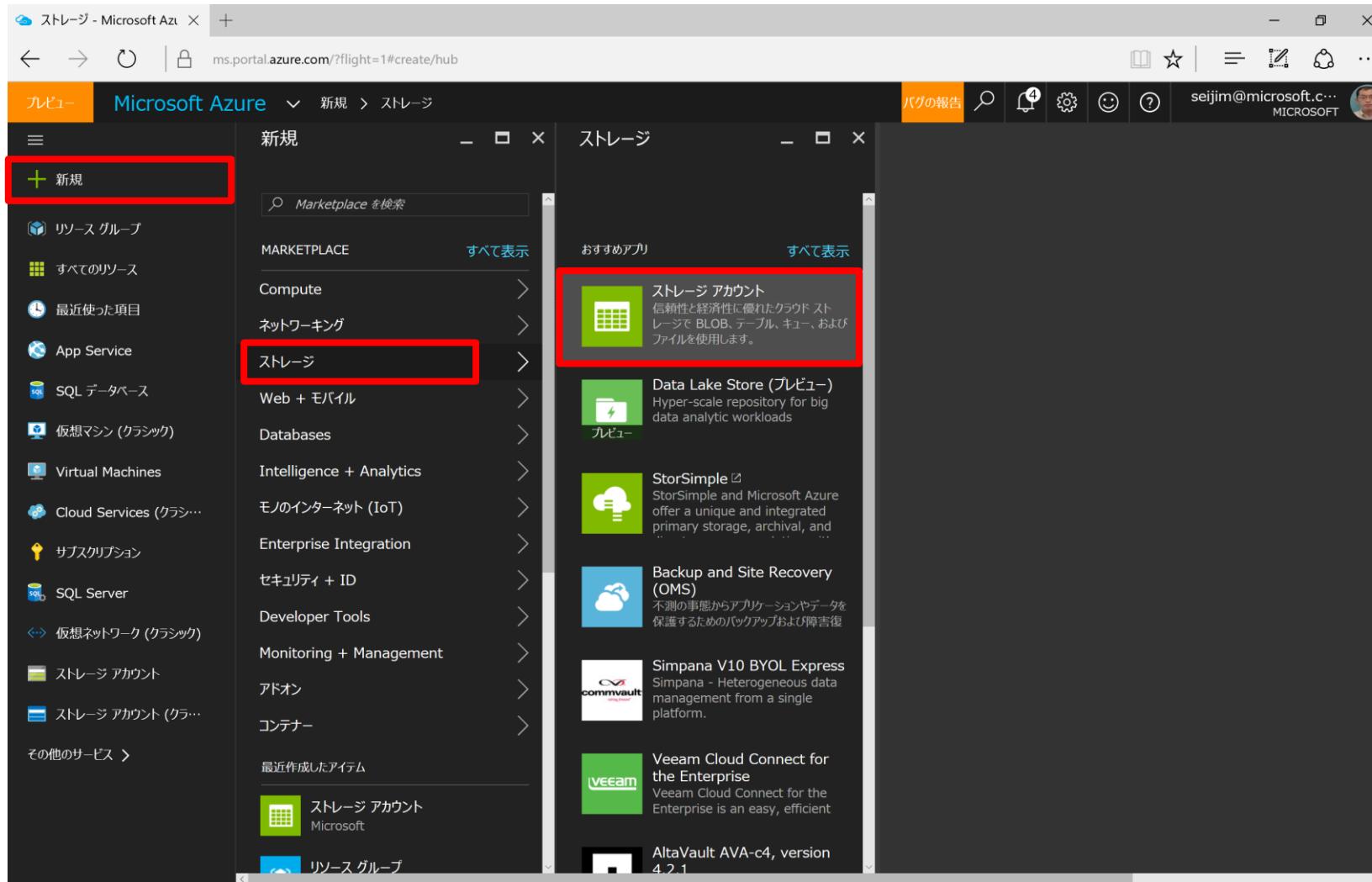
- [名前]={ 管理し易い名前で }
- [場所]={ 東日本 or 西日本 }



# T01\_03：ストレージアカウントの新規作成

Azure ポータル (portal.azure.com) にログインし、該当のサブスクリプションかどうか確認

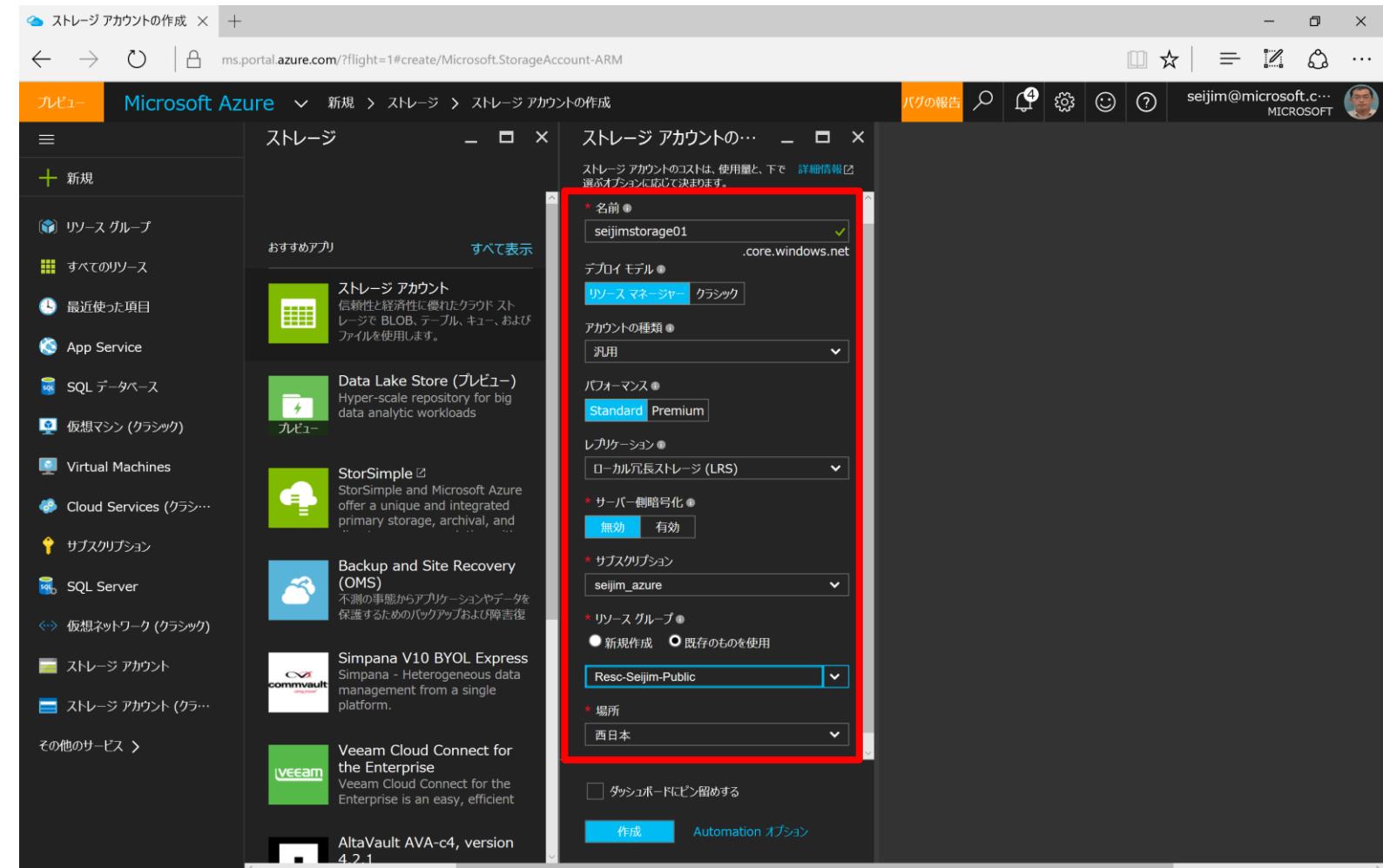
- [+新規]→[ストレージ]→[ストレージアカウント]



# T01\_04：ストレージアカウント作成パラメータ入力

以下のように設定し、[作成] ボタンを押します

- [名前]={ 管理し易い名前で }
- [レプリケーション]={ ローカル冗長 (LRS) }
- [リソースグループ]={ T01\_02 の設定値 }
- [場所]={ 東日本 or 西日本 }



# T01\_05：ストレージアカウントの確認

Azure ポータルから以下のように選択をします

- [ストレージアカウント]→[該当のストレージアカウント]→[アクセスキー]
- [ストレージアカウント名], [key1] をメモ帳などに保存しておきます

The screenshot shows the Microsoft Azure portal interface. The left sidebar is collapsed. The main navigation bar shows the current page as 'アクセスキー - Microsoft' and the URL as 'ms.portal.azure.com/?flight=1#resource/subscriptions.../resourceGroups/Resc-Seijim-JapanWest/providers/Microsoft.Storage/storageAccounts/sejim.../AccessKeys'. The top right corner has icons for reporting a bug, search, notifications (4), settings, and help.

The main content area is titled 'sejim... - アクセスキー' (Storage Account Access Keys). It displays a list of access keys for the storage account 'sejim...'. The first key, 'key1', is highlighted with a red box. The 'Access Key' button in the 'Actions' section of the key list is also highlighted with a red box.

The left sidebar has several collapsed sections, but the 'Storage Account' section is expanded, showing the storage account 'sejim...' selected. Other collapsed sections include 'Resource Groups', 'All Resources', 'Recent Items', 'App Service', 'SQL Database', 'Virtual Machines', 'Cloud Services', 'Subscription', 'Storage Account (Classic)', and 'Other Services'.

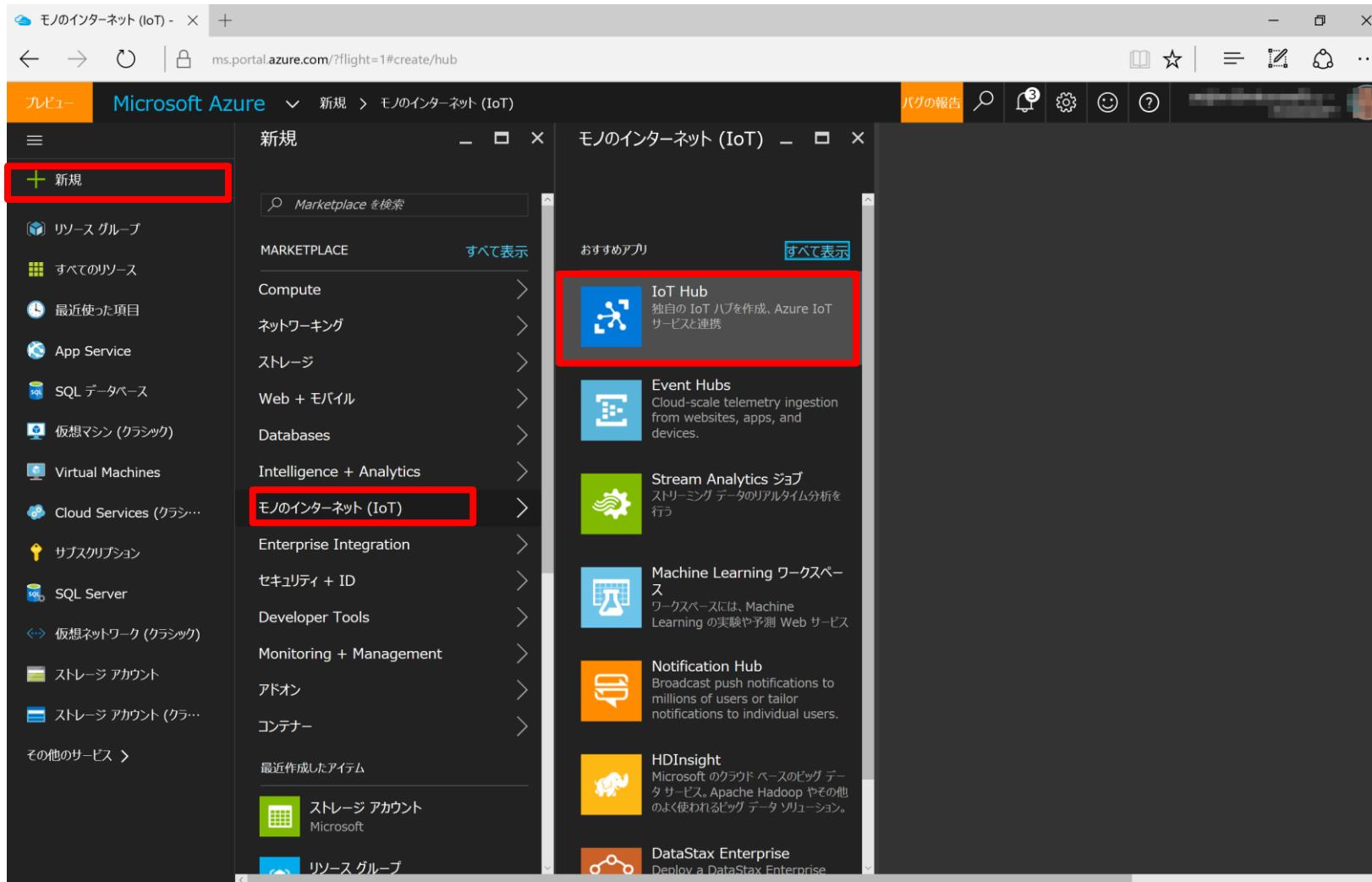
# Tutorial 02

## Azure IoT Hub の作成

# T02\_01 : IoT Hub の新規作成

Azure ポータルから以下のように選択をします

- [新規]→[モノのインターネット]→[IoT Hub]



# T02\_02 : IoT Hub 作成パラメータ入力

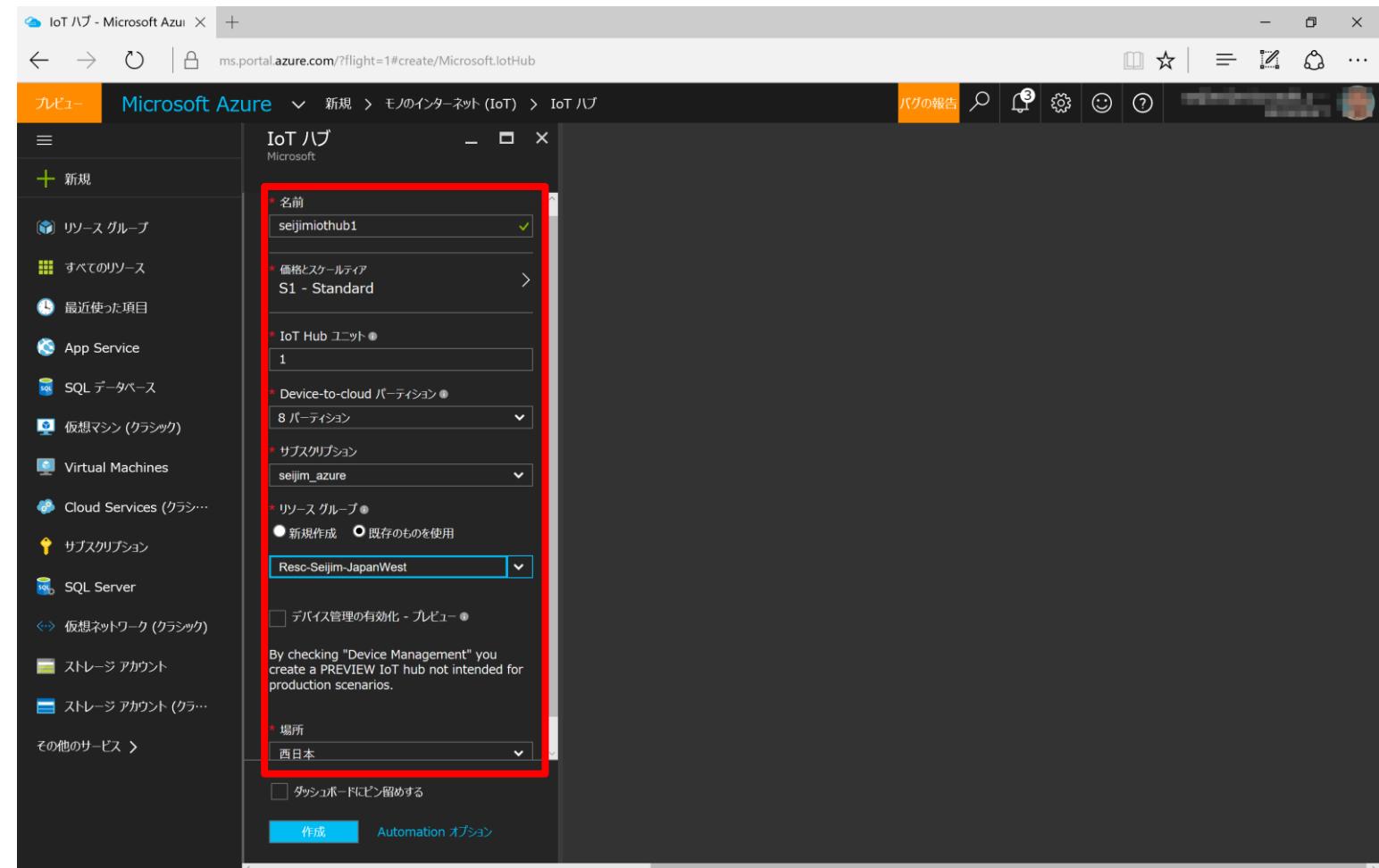
以下のように設定し、[作成] ボタンを押します

- [名前]={ 小文字で重複なく設定 }
- [Device-to-cloud パーティション]={ 4 }
- [リソースグループ]={ T01 の設定値 }
- [デバイス管理の有効化-プレビュー]={ 無し }
- [場所]={ 東日本 or 西日本 }

## [Point]

価格とスケールティアは [F1 - Free] でも構いません。  
但し、パーティション数が 2 に制約される為、Cloud  
Robotics FX もインスタンス数は最大 2 に制約されます。

価格とスケールティアやスケールユニットは、後で  
変更できますが、パーティション数(キューの数)は、  
後で変更はできません。



# T02\_03 : IoT Hub 作成後の確認

Azure ポータルから以下のように選択をします

- [IoT Hub]→[該当の IoT Hub]→[共有アクセスポリシー]→[iothubowner]
- [接続文字列-プライマリキー] をメモ帳などに保存しておきます

The screenshot shows the Azure portal interface. On the left, the navigation menu is visible with various service icons. In the center, under the 'IoT Hub' section, the 'Shared access policy' page is displayed. A specific policy named 'iothubowner' is highlighted with a red box. This policy grants 'RegistryWrite' and 'DeviceConnect' permissions. Below the policies, there are sections for 'Service' and 'Device' access. At the bottom of the page, there are links for 'Message', 'File upload', 'Scale', 'Operational monitoring', and 'Diagnostics'.

This screenshot shows the detailed view of the 'iothubowner' shared access policy. It includes fields for the policy name ('iothubowner'), access rights ('RegistryWrite, DeviceConnect'), and two connection strings: 'Primary key' and 'Secondary key'. The 'Primary key' field is highlighted with a red box. Both keys begin with 'HostName=' followed by a unique identifier.

# T02\_04 : IoT Hub 作成後の確認と設定

Azure ポータルから以下のように選択をします

- [消費者グループ]={ cloudroboticsfxcg } を設定し、[保存] ボタンを押します

The screenshot shows two windows from the Microsoft Azure portal. The left window is titled 'Properties - Microsoft A' and displays the 'Endpoints' section for an IoT Hub named 'seijim-iothub'. The right window is a modal titled 'Properties' for the same IoT Hub, showing the 'Device-to-cloud 設定' tab. In the left window, the 'Endpoints' section lists an 'Events' endpoint with the value 'messages/events'. In the right window, the 'Consumer Group' dropdown is set to '\$Default' and contains the value 'cloudroboticsfxcg'. Both windows have red boxes highlighting the 'Endpoints' section in the left window and the 'Consumer Group' dropdown in the right window.

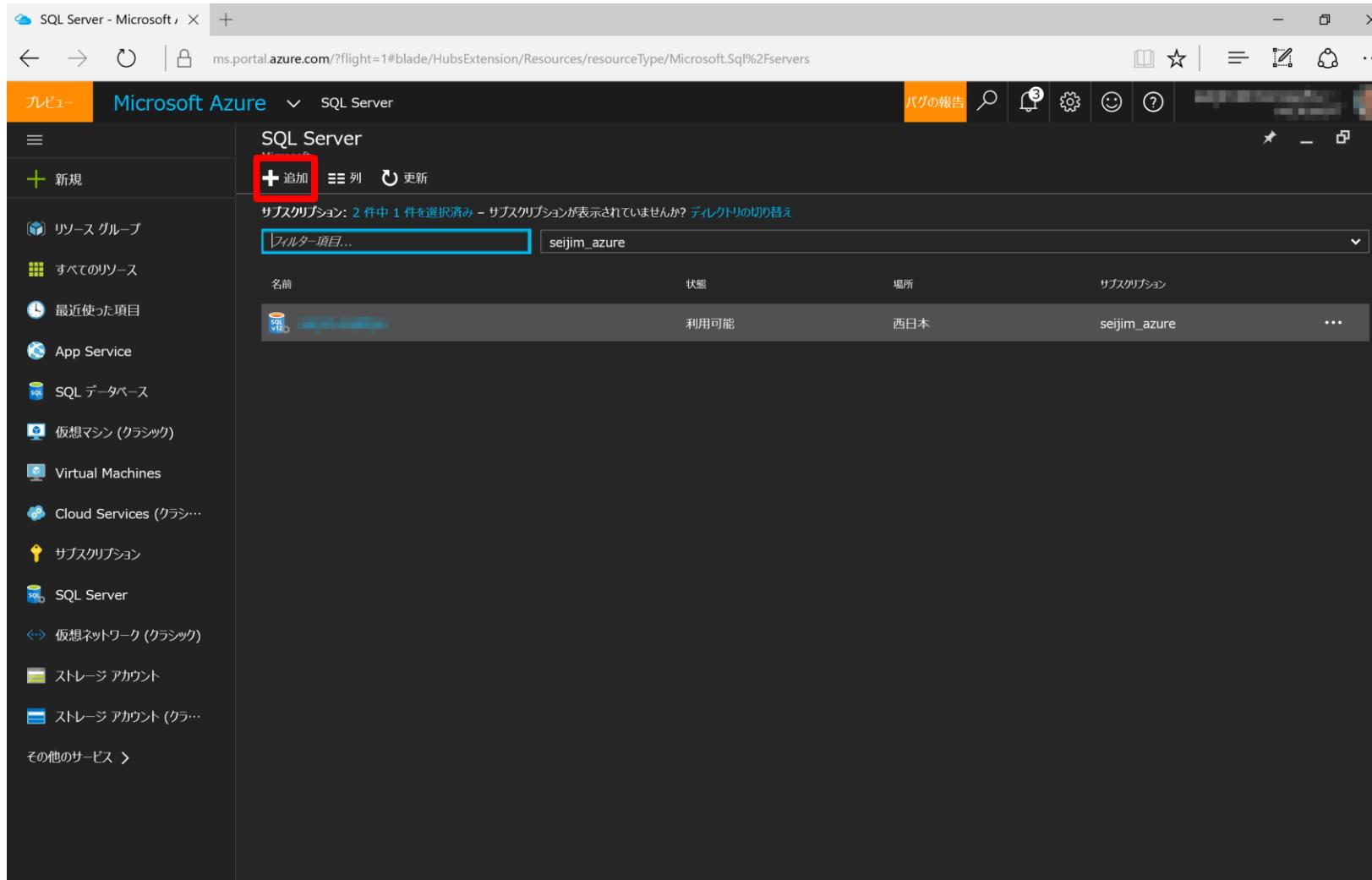
# Tutorial 03

## SQL Database の作成

# T03\_01 : SQL Server の新規作成

Azure ポータルから以下のように選択をします

- [SQL Server]→[+追加]



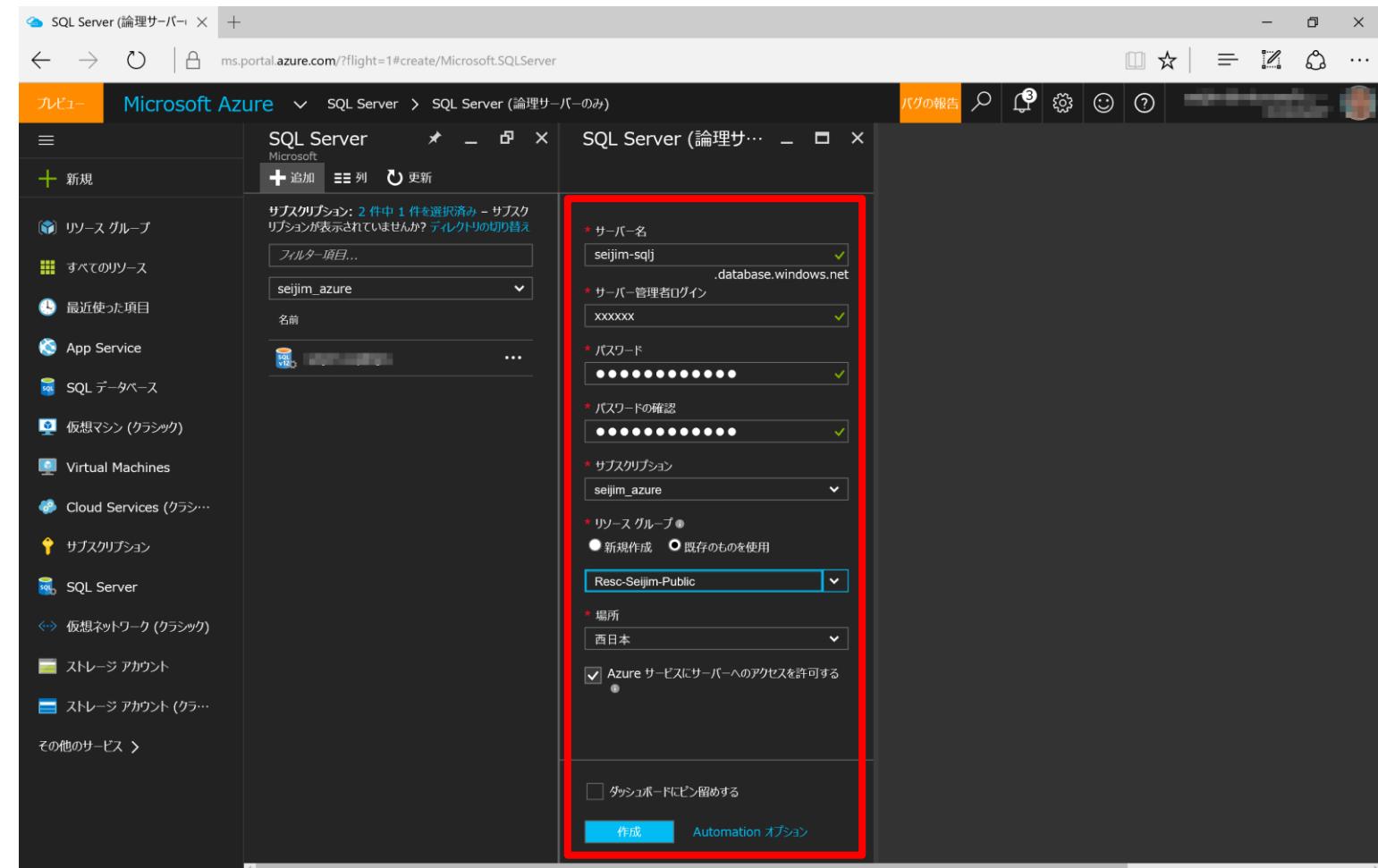
The screenshot shows the Microsoft Azure portal interface for managing SQL Server instances. On the left, there's a sidebar with various service icons like Resource Groups, App Service, and Virtual Machines. The main content area is titled 'SQL Server' and shows a list of existing servers. At the top right of this area, there's a button labeled '+ 追加' (Add), which is highlighted with a red box. Below the button, there's a message about subscriptions and a search bar with the text 'seijim\_azure'. The table below lists one server entry:

名前	状態	場所	サブスクリプション
[Redacted]	利用可能	西日本	seijim_azure

# T03\_02 : SQL Server 作成パラメータ入力

以下のように設定し、[作成] ボタンを押します

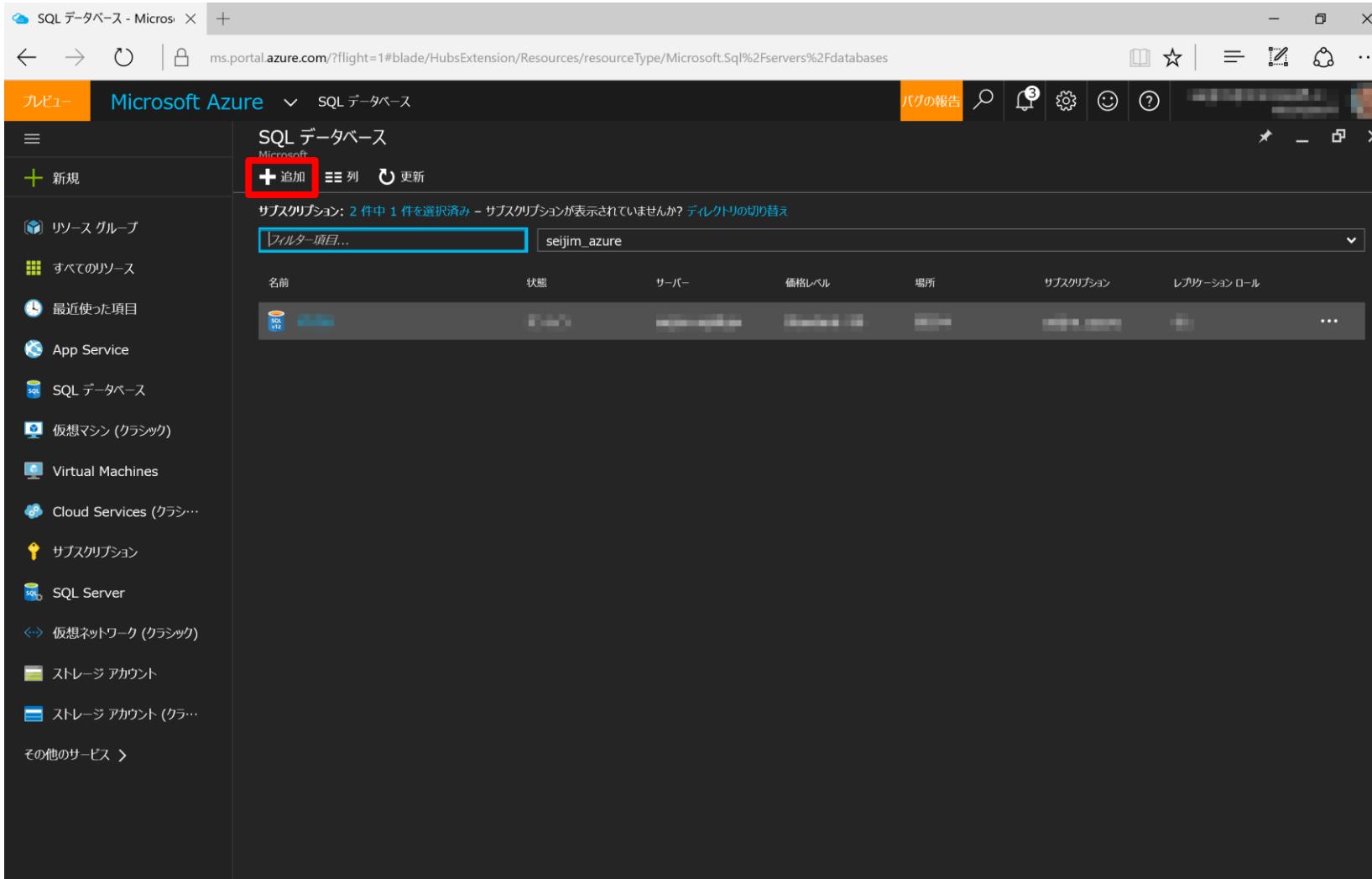
- [サーバー名]={ 小文字で重複なく設定 }
- [サーバー管理者ログイン]={ 管理UIDを設定 }
- [パスワード]={ 複雑な文字列を設定 }
- [リソースグループ]={ T01 の設定値 }
- [場所]={ 東日本 or 西日本 }



# T03\_03 : SQL Database の新規作成

Azure ポータルから以下のように選択をします

- [SQL データベース]→[+追加]



The screenshot shows the Microsoft Azure portal interface for managing SQL databases. The left sidebar contains a navigation menu with items like 'リソース グループ', 'すべてのリソース', '最近使った項目', 'App Service', 'SQL データベース', '仮想マシン (クラシック)', 'Virtual Machines', 'Cloud Services (クラシック)', 'サブスクリプション', 'SQL Server', '仮想ネットワーク (クラシック)', 'ストレージ アカウント', 'ストレージ アカウント (クラシック)', and 'その他のサービス >'. The main content area is titled 'SQL データベース' and shows a table of existing databases. At the top of this area, there is a toolbar with icons for 'バグの報告', '検索', '通知', '設定', 'ヘルプ', and 'アバター'. Below the toolbar, a red box highlights the '＋追加' (Add) button. The table has columns for '名前' (Name), '状態' (Status), 'サーバー' (Server), '価格レベル' (Price Level), '場所' (Location), 'サブスクリプション' (Subscription), and 'レプリケーション ロール' (Replication Role). A search bar at the top of the table is set to 'seijim\_azure'.

# T03\_04 : SQL Database 作成パラメータ入力

以下のように設定し、[作成] ボタンを押します

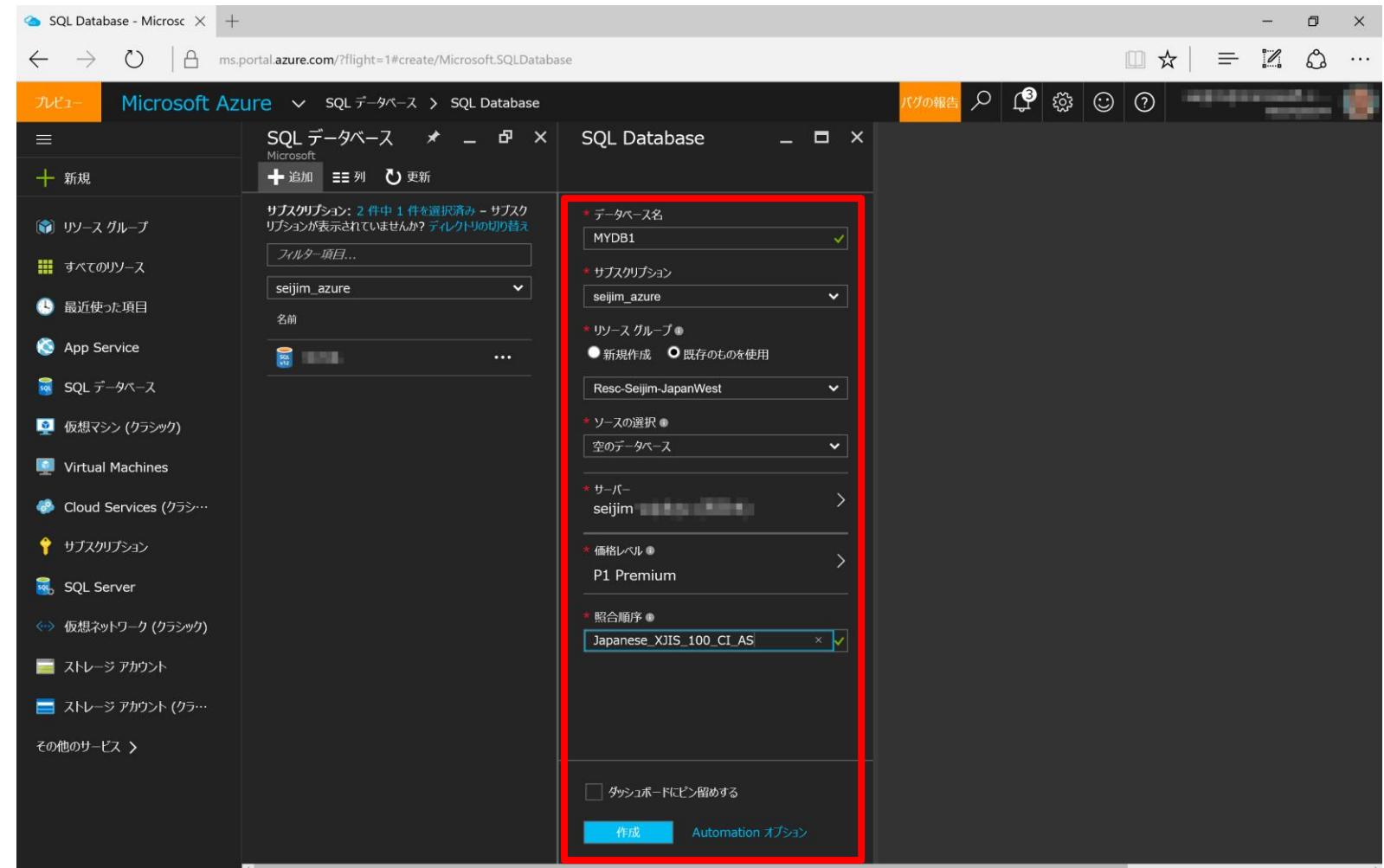
- [データベース名]={ 管理し易い名前で設定 }
- [リソースグループ]={ T01 の設定値 }
- [サーバー]={ 作成した SQL Server を選択 }
- [価格レベル]={ S0 以上 }
- [照合順序]={ Japanese\_XJIS\_100\_CI\_AS }

## [Point]

性能面を重視する場合、価格レベルは Premium をお勧めします。

RBApp.IoTHubLog などのログテーブルに、クラスター化リストアインデックスを利用すると効果的です。一般的には 10~15倍のデータ圧縮が可能となり、DWHクエリ性能が最大で数百倍向上します。

RBFX をスキーマとする OLTP 系のテーブル群に、インメモリ OLTP 機能を利用すると、IO 性能が格段に向上します。一般的には 数倍~30倍の範囲となります。



# 価格レベル毎の詳細

機能	Basic	Standard				Premium					
		S0	S1	S2	S3*	P1	P2	P4	P6	P11	P15
使用可能時間 SLA	99.99%	99.99%				99.99%					
最大 DB サイズ	2 GB	250 GB				500 GB				1 TB or 4 TB	
DTU (Database Throughput Units)	5	10	20	50	100	125	250	500	1,000	1,750	4,000
In-Memory OLTP ストレージサイズ	N/A	N/A				1 GB	2 GB	4 GB	8 GB	14 GB	32 GB
Point In Time Restore	過去7日間の任意の時点	過去 14 日間の任意の時点				過去 35 日間の任意の時点					
Disaster Recovery	アクティブ Geo レプリケーション (最大 4 つまでの [読み取り可能] オンライン セカンダリー バックアップ)										
パフォーマンス目標	1 時間当たりトランザクション数	1 分当たりのトランザクション数				1 秒当たりのトランザクション数					
	16,600/h	521/m	934/m	2,570/m	5,100/m	105/s	213/s	425/s	850/s	1,488/s	3,400/s
予測可能性	良い(時間単位)	高い(分単位)				最高(秒単位)					
TPM 換算値	276	521	934	2,570	5,100	6,300	13,680	26,820	44,100	77,175	176,366
最大同時ログイン数	30	60	90	120	200	200	400	800	1,600	2,400	6,400
最大セッション数	300	600	900	1,200	2,400	30,000	30,000	30,000	30,000	30,000	30,000
1 時間当たりの料金	¥0.78/時	¥2.33/時	¥4.66/時	¥11.65/時	¥23.28/時	¥72.05/時	¥144.08/時	¥288.66/時	¥576.30/時	¥1,086.30/時	¥2,482.68/時
1 ヶ月当たりの料金	¥612/月	¥1,734/月	¥3,468/月	¥8,670/月	¥17,340/月	¥53,550/月	¥107,202/月	¥214,812/月	¥428,808/月	¥808,248/月	¥1,847,113.92/月

SQL Database のオプションとパフォーマンス: 各サービス レベルで使用できる内容について理解する

<https://docs.microsoft.com/ja-jp/azure/sql-database/sql-database-service-tiers>

# T03\_05 : SQL Database 作成後の確認

Azure ポータルから以下のように選択をします

- [SQL データベース]→[該当の DB]→[データベース接続文字列の表示]
- [ADO.NET (SQL 認証)] の接続文字列をメモ帳などにコピペし、ユーザーID/パスワードを編集し、保存しておきます

The screenshot shows the Azure portal interface for a 'MYDB' SQL database. On the left sidebar, under 'SQL データベース', the 'MYDB' database is selected. The main pane displays the database's properties, including its resource group ('Resc-Seijim-JapanWest'), server name ('seijim'), and status ('Online'). A red box highlights the 'データベース接続文字列の表示' (Show connection string) link in the '概要' (Overview) section. Below this, a chart shows DTU percentage over time.

The screenshot shows the 'Database connection string' configuration page. It lists two connection options: 'ADO.NET (SQL 認証)' and 'ADO.NET (Active Directory パスワード認証)'. The 'ADO.NET (SQL 認証)' section contains a connection string: 'Server=tcp:seijim...database.windows.net,1433;Initial Catalog=...;Persist Security Info=False;User ID=...;Password=...;MultipleActiveResultSets=True;Encrypt=True;TrustServerCertificate=False;Connection Timeout=30;'. A red box highlights this connection string. Below it, there are several other connection strings listed in a scrollable list.

# Tutorial 04

SQL Database 上にテーブルを作成

# DB テーブル定義

# Cloud Robotics FX - SQLDB : デバイス系定義①

```
CREATE TABLE [RBFX].[DeviceMaster]
```

```
(  
    [SeqId] int IDENTITY NOT NULL,  
    [DeviceId] NVARCHAR(100) NOT NULL,  
    [DeviceType] NVARCHAR(100) NULL,  
    [Status] VARCHAR(20) NULL,  
    [ResourceGroupId] VARCHAR(40) NULL,  
    [Description] NVARCHAR(1000) NULL,  
    [Registered_DateTime] DATETIME NULL,  
CONSTRAINT [PK_DeviceMaster] PRIMARY KEY CLUSTERED  
(  
    [DeviceId] ASC  
)
```

```
CREATE TABLE [RBFX].[DeviceRouting]
```

```
(  
    [SeqId] int IDENTITY NOT NULL,  
    [DeviceId] NVARCHAR(100) NOT NULL,  
    [RoutingKeyword] NVARCHAR(100) NOT NULL,  
    [TargetType] VARCHAR(20) NULL,  
    [TargetDeviceGroupId] NVARCHAR(100) NULL,  
    [TargetDeviceId] NVARCHAR(100) NULL,  
    [Status] VARCHAR(20) NULL,  
    [Description] NVARCHAR(1000) NULL,  
    [Registered_DateTime] DATETIME NULL,  
CONSTRAINT [PK_DeviceRouting] PRIMARY KEY CLUSTERED  
(  
    [DeviceId] ASC,  
    [RoutingKeyword] ASC  
)
```

## [RBFX].[DeviceMaster] : デバイスマスター情報テーブル

- [DeviceId] 列は、IoT Hub に登録する DeviceId と同一の値で登録
- デバイスのタイプや情報を格納
- [Status] 列の値は、"Active" or "NA" で、"Active" な行のみ対象とする
- [ResourceGroupId] 列の値は、SaaS の仕組みの中では、顧客の契約部門を意味する
- このマスターに登録されていないデバイスについては、ルーティングを実施しない

## [RBFX].[DeviceRouting] : デバイスルーティング定義テーブル

- 送信元の [DeviceId] からルーティング対象のデバイス、または、デバイスグループを特定
- [DeviceId] + [RoutingKeyword] でルーティングが決定される
- [TargetType] 列の値は、"Device" or "DeviceGroup" で、その値によって送信先の列を特定
- [Status] 列の値は、"Active" or "NA" で、"Active" な行のみ対象とする

# Cloud Robotics FX - SQLDB : デバイス系定義②

```
CREATE TABLE [RBFX].[DeviceGroup]
(
    [SeqId] int IDENTITY NOT NULL,
    [DeviceGroupId] NVARCHAR(100) NOT NULL,
    [DeviceId] NVARCHAR(100) NOT NULL,
    [Registered_DateTime] DATETIME NULL,
    CONSTRAINT [PK_DeviceGroup] PRIMARY KEY CLUSTERED
    (
        [DeviceGroupId] ASC,
        [DeviceId] ASC
    )
)
```

```
CREATE INDEX [IDX1] ON [RBFX].[DeviceGroup]
(
    [DeviceId] ASC
)
```

[RBFX].[DeviceGroup] : デバイス グループ情報テーブル

- [DeviceGroupId] 列は、[RBFX].[DeviceRouting] テーブルの [TargetDeviceGroupId] と同一の値
- [DeviceId] 列は、IoT Hub に登録する DeviceId と同一の値

# Cloud Robotics FX - SQLDB : アプリ系定義

```
CREATE TABLE [RBFX].[AppMaster]
(
    [SeqId] int IDENTITY NOT NULL,
    [AppId] NVARCHAR(100) NOT NULL,
    [StorageAccount] NVARCHAR(256) NULL,
    [StorageKeyEnc] VARBINARY(2000) NULL,
    [AppInfoEnc] VARBINARY(8000) NULL,
    [AppInfoDeviceEnc] VARBINARY(8000) NULL,
    [Status] VARCHAR(20) NULL,
    [Description] NVARCHAR(1000) NULL,
    [Registered_DateTime] DATETIME NULL,
    CONSTRAINT [PK_AppMaster] PRIMARY KEY CLUSTERED
    (
        [AppId] ASC
    )
)

CREATE TABLE [RBFX].[AppRouting]
(
    [SeqId] int IDENTITY NOT NULL,
    [AppId] NVARCHAR(100) NOT NULL,
    [AppProcessingId] NVARCHAR(100) NOT NULL,
    [BlobContainer] NVARCHAR(100) NULL,
    [FileName] NVARCHAR(100) NULL,
    [ClassName] NVARCHAR(100) NULL,
    [Status] VARCHAR(20) NULL,
    [DevMode] VARCHAR(5) NULL,
    [DevLocalDir] NVARCHAR(1000) NULL,
    [Description] NVARCHAR(1000) NULL,
    [Registered_DateTime] DATETIME NULL,
    CONSTRAINT [PK_AppRouting] PRIMARY KEY CLUSTERED
    (
        [AppId] ASC,
        [AppProcessingId] ASC
    )
)
```

## [RBFX].[AppMaster] : アプリケーション (DLL) 格納先情報テーブル

- アプリケーションシステムの単位で設定
- アプリ (DLL) 格納先の Storage Account と Key、アプリに渡すべき接続情報などを格納
- [StorageKeyEnc] 列 : Storage Key は、SQL 暗号化関数で暗号化する
- [AppInfoEnc] 列 : クラウドサイドのアプリ固有の DB 接続情報などを、SQL 暗号化関数で暗号化する
- [AppInfoDeviceEnc] 列 : デバイスサイドのアプリ固有の DB 接続情報などを、SQL 暗号化関数で暗号化する
- [Status] 列の値は、“Active” or “NA” で、“Active” な行のみ対象とする

## [RBFX].[AppRouting] : アプリケーション (DLL) ルーティング定義テーブル

- デバイスから送信される電文 (JSON) の [AppId] と [AppProcessingId] からアプリ (DLL) をマッピング
- アプリ (DLL) 格納先情報は、[RBFX].[AppRoutingMaster] テーブルと [BlobContainer] 列から取得
- [ClassName] 列の値は、<名前空間>.<クラス名>を設定
- [Status] 列の値は、“Active” or “NA” で、“Active” な行のみ対象とする
- [DevMode] 列の値は、“True” の時は、BLOB の格納先は見ずに、[DevLocalDir] からロード
  - ローカル PC などで動作させて、DLL を単体テストする場合などに利用

# Cloud Robotics FX - SQLDB : 顧客系定義 (SaaS 用途)

```
CREATE TABLE [RBFX].[CustomerInfo]
(
    [SeqId] int IDENTITY NOT NULL,
    [CustomerId] NVARCHAR(100) NOT NULL,
    [CustomerName] NVARCHAR(100) NULL,
    [Description] NVARCHAR(1000) NULL,
    [Registered_DateTime] DATETIME NULL,
    CONSTRAINT [PK_CustomerInfo] PRIMARY KEY CLUSTERED
    (
        [CustomerId] ASC
    )
)
```

```
CREATE TABLE [RBFX].[CustomerResource]
(
    [SeqId] int IDENTITY NOT NULL,
    [CustomerId] NVARCHAR(100) NOT NULL,
    [ResourceGroupId] NVARCHAR(40) NOT NULL,
    [ResourceGroupName] NVARCHAR(100) NULL,
    [SqlConnectionStringEnc] VARBINARY(2000) NULL,
    [Description] NVARCHAR(1000) NULL,
    [Registered_DateTime] DATETIME NULL,
    CONSTRAINT [PK_CustomerResource] PRIMARY KEY CLUSTERED
    (
        [CustomerId] ASC,
        [ResourceGroupId] ASC
    )
)
```

```
CREATE INDEX [IDX1] ON [RBFX].[CustomerResource]
(
    [ResourceGroupId] ASC
)
```

[RBFX].[CustomerInfo] : 顧客情報テーブル

[RBFX].[CustomerResource] : 顧客リソース管理テーブル

- リソースグループ = 部門などの契約単位で、リソースグループで Pepper やデバイスを管理する
- アプリケーション (DLL) のアクティベーションもこのリソースグループ単位で管理する
- [ResourceGroupId] 列は、ハイフン付きの GUID 値
- [SqlConnectionStringEnc] 列は、顧客毎の DB の接続情報で、SQL 暗号化関数で暗号化する

# T04\_01 : SQL Server ファイアウォール設定

Azure ポータルから以下のように選択をします

- [SQL Server]→[該当の Server]→[ファイアウォール設定の表示]
- [+クライアント IP の追加] を押し、名前を付けて、[保存] を押します

The screenshot shows the Azure portal interface for managing a SQL Server instance named 'seijim'. The left sidebar lists various Azure services like Resource Groups, App Services, and Virtual Machines. The main panel displays the 'SQL Server' blade for 'seijim'. In the center, there's a summary card for the server, and below it, a table showing database details. At the bottom of the main content area, there's a link labeled 'Firewall settings' which is highlighted with a red box.

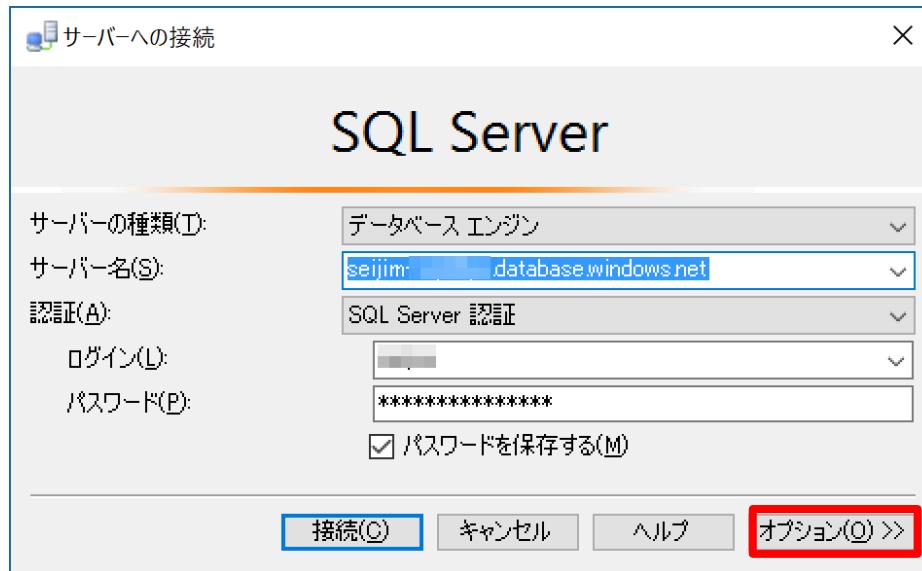
This screenshot shows the 'Firewall settings' blade for the 'seijim-sqldbjw' SQL Server. It includes a message about allowing access from specified IPs, a toggle switch for Azure service access (set to 'On'), and a table for client IP addresses. A new rule has been added with the name 'seijimIP', starting IP '167.220.232.150', and ending IP '167.220.232.150'.

規則名	開始 IP	終了 IP
seijimIP	167.220.232.150	167.220.232.150

# T04\_02 : SQL Server Management Studio の起動

事前にインストールしておいた SQL Server Management Studio を起動します

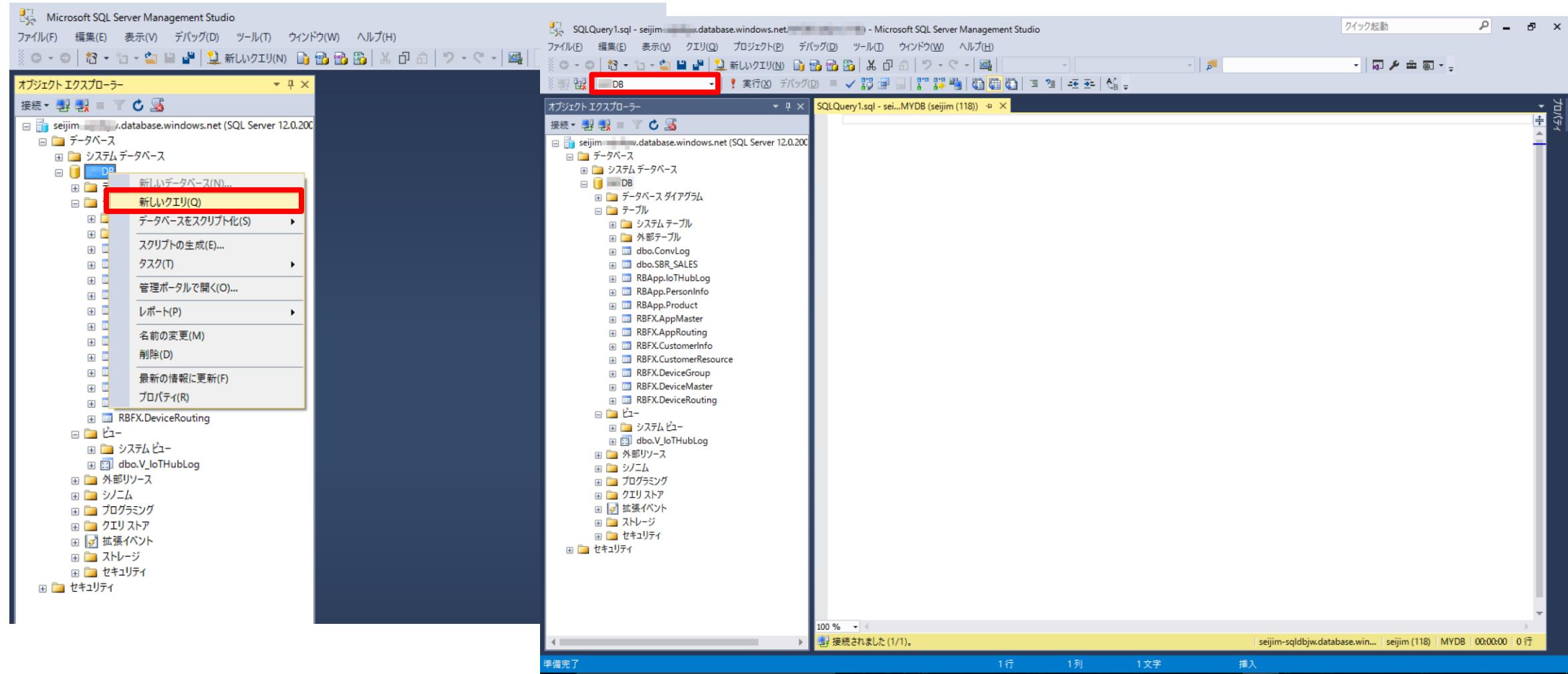
- ログイン画面に、メモ帳に保存しておいた ADO.NET 接続文字列から Server 名をコピペします
- [SQL Server 認証] を選択し、管理者ユーザー ID とパスワードを入力します
- [オプション] を押し、[接続プロパティ] タブで、[サーバー証明書を信頼する] をチェックします
- [接続] ボタンを押します



# T04\_03 : SQL Server Management Studio の操作

SQL Server Management Studio で対象データベースを選択します

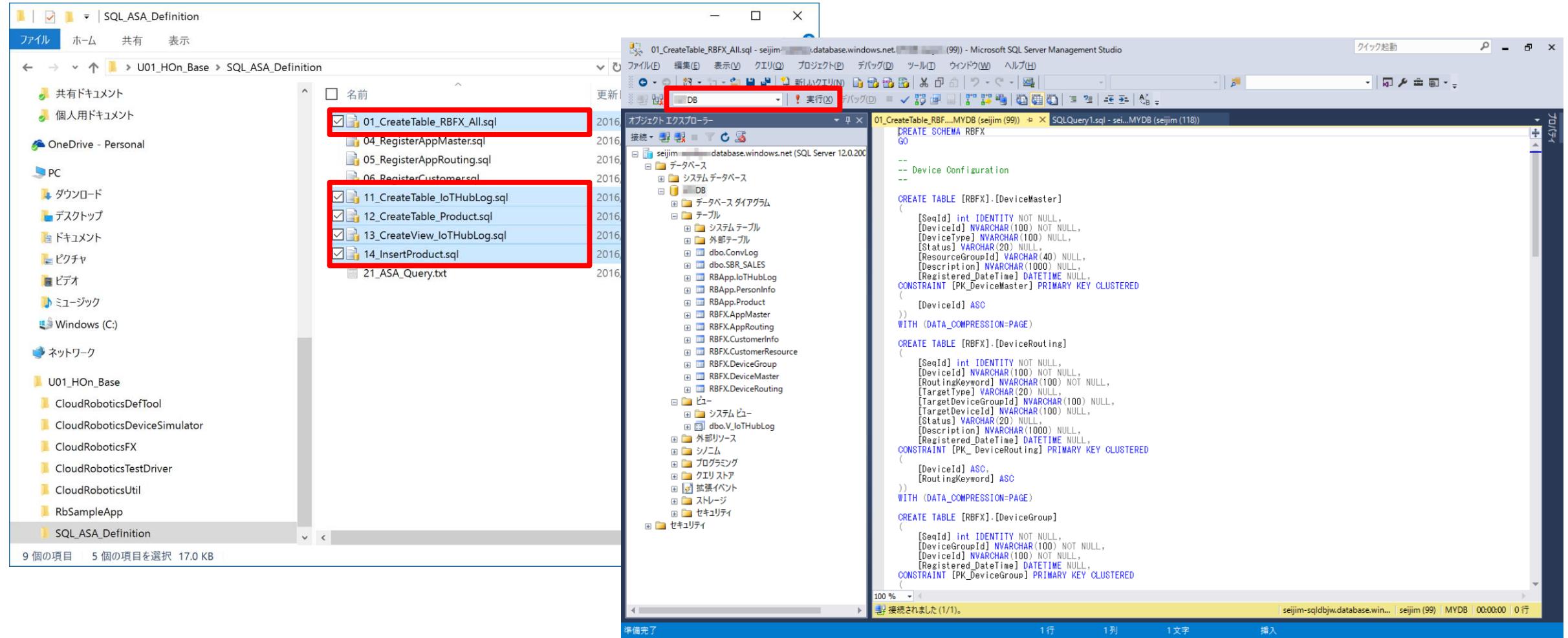
- オブジェクトエクスプローラーから [該当の DB] を選択し、右クリックから、[新しいクエリ] を選択します
- ツールバー上で [該当の DB] が選択されているか確認します



# T04\_04 : SQL スクリプトファイルの実行

SQL Server Management Studio から SQL スクリプトを実行し、Cloud Robotics SDK および アプリ用のテーブルを作成します

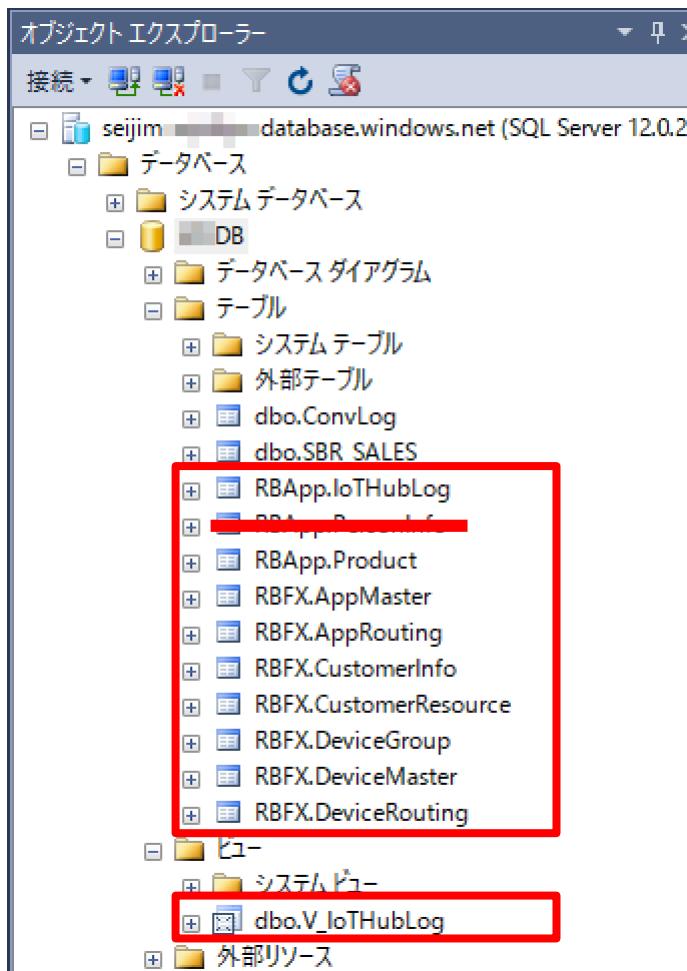
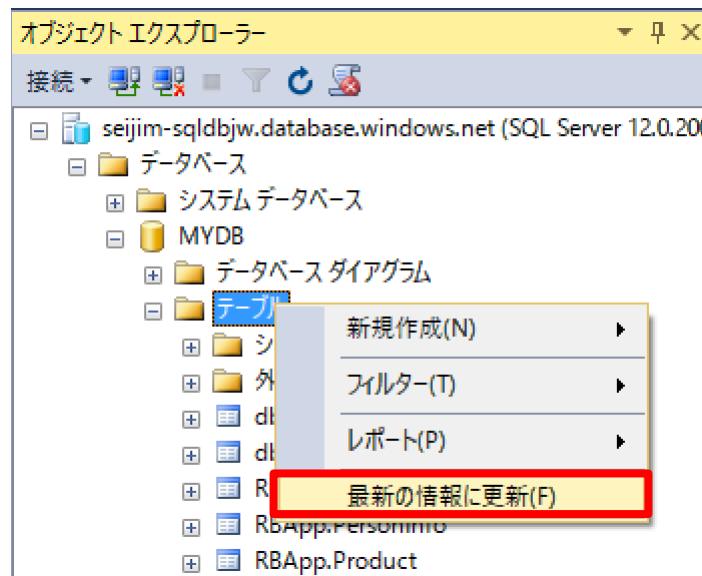
- 事前にダウンロードして頂いた「CRSDKv1.zip」を解凍し、「CRSDKv1\SQL\_ASA\_Definition」に移動します
- 以下囲みをした SQL ファイルをダブルクリックして、Management Studio から [該当の DB] に対して [実行] します



# T04\_05：テーブルの確認

テーブルの作成に成功したか、確認します

- オブジェクトエクスプローラーの [テーブル] を右クリックして、[最新の情報に更新] を押します
- 以下囲んだテーブルが作成されていれば、確認完了です



# Tutorial 05

Cloud Robotics FX V2 本体、および  
関連ソリューションのビルドと設定

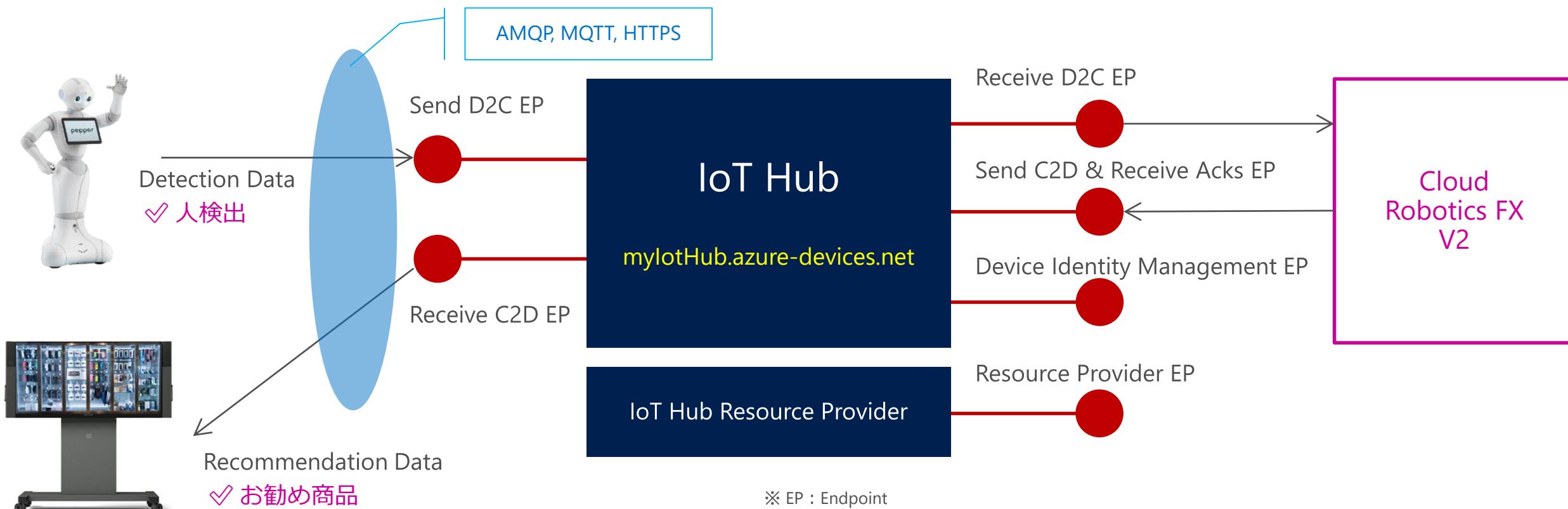
# Azure IoT Hub との通信

## デバイスとバックエンド間の信頼性のある双向通信

- ✓ 通信セキュリティの確保
- ✓ 数百万デバイス接続のスケーラビリティ
- ✓ マルチ プラットフォーム<sup>\*1</sup> / マルチ ランゲージ<sup>\*2</sup> の SDK

\* 1 : RTOS, Linux, Windows, ....etc.

\* 2 : C, .NET, Java, Node.js



※ EP : Endpoint

※参照：<https://azure.microsoft.com/ja-jp/documentation/articles/iot-hub-devguide/>

# 通信フォーマット

# Cloud Robotics FX - D2C(↑) & C2D(↓) Control Format

```
{  
  "RbHeader": {  
    "RoutingType": "CONTROL",  
    "AppId": "PepperShopApp",  
    "AppProcessingId": "ReqAppInfo",  
    "MessageSeqno": "201",  
    "SendDateTime": "2016-04-01 14:05:22.038"  
  },  
  "RbBody": {  
    "visitor": "u001"  
  }  
}
```

**RoutingType** : 設定必須  
→ "CONTROL" 固定  
**AppId** : 設定必須  
→ アプリケーションシステムの特定用途  
**AppProcessingId** : 設定必須  
→ "ReqAppInfo" (アプリケーション情報取得)  
**MessageSeqno** : 設定必須  
→ 指示する側のアプリで設定  
**SendDateTime** : 設定必須  
→ 送信時のローカルタイム (ミリ秒まで)

```
{  
  "RbHeader": {  
    "RoutingType": "CONTROL",  
    "AppId": "PepperShopApp",  
    "AppProcessingId": "ResAppInfo",  
    "MessageSeqno": "202",  
    "SendDateTime": "2016-04-01 14:05:23.425"  
  },  
  "RbBody": {  
    <RBFX.AppMaster の AppInfoDevice 列の内容>  
  }  
}
```

**RoutingType** : "CONTROL" 固定  
**AppId** : 呼び出し元で設定された内容  
**AppProcessingId** : "ResAppInfo" (アプリケーション情報取得への回答)  
**MessageSeqno** : 呼び出し元で設定された内容  
**SendDateTime** : 呼び出し元で設定された内容

# Cloud Robotics FX - D2C(↑) & C2D(↓) App Format

```
{  
  "RbHeader": {  
    "RoutingType": "D2D",  
    "RoutingKeyword": "Default",  
    "AppId": "PepperShopApp",  
    "AppProcessingId": "",  
    "MessageId": "MSG01",  
    "MessageSeqno": "159",  
    "SendDateTime": "2016-04-01 14:05:22.038",  
    "SourceDeviceId": "",  
    "SourceDeviceType": "",  
    "SourceDevRescGroupId": "",  
    "TargetType": "",  
    "TargetDeviceGroupId": "",  
    "TargetDeviceId": "",  
    "ProcessingStack": ""  
  },  
  "RbBody": {  
    ...<アプリ用途で自由に設定>...  
  }  
}
```

## RoutingType : 設定必須

- Robotics FX のデバイス ルーティングやアプリ (DLL) の呼び出しの振る舞いを決める
  - ・デバイス ルーティングが必要な時："D2D"
  - ・アプリ (DLL) 呼び出しのみ必要な時："CALL",
  - ・単なるログ用メッセージの時："LOG",

## RoutingKeyword : 設定自由

- SourceDeviceId + RoutingKeyword によるデバイス ルーティングとなる為、未指定の場合、"Default" が自動設定

## AppId : 設定必須

- アプリケーション システムの特定用途

## AppProcessingId : "CALL" 時は設定必須。"D2D" 時は設定自由。"LOG" 時は設定不要

- 設定時は、Robotics FX が RBFX.AppRouting から <AppId>+<AppProcessingId> でアプリ (DLL) を特定し、呼び出す

## MessageId : 設定自由

- アプリ (DLL) 内や "D2D" の連携時に自由に利用

## MessageSeqno : 設定必須

- 通信連番を設定。Robotics FX やアプリ (DLL) で問題が発生した場合の追跡用に必要

## SendDateTime : 設定必須

- 送信時のローカルタイム (ミリ秒まで)

## SourceDeviceId : 設定不要

- Robotics FX が IoT Hub のデバイス ID を自動設定

## SourceDeviceType : 設定不要

- Robotics FX が RBFX.DeviceMaster から <SourceDeviceId> で特定し、自動設定

## SourceDevRescGroupId : 設定不要

- Robotics FX が RBFX.DeviceMaster から <SourceDeviceId> で特定し、<ResourceGroupId> の値を自動設定

## TargetType : アプリ (DLL) で上書き可

- 値は、"Device" or "DeviceGroup"

- Robotics FX が RBFX.DeviceRouting から <SourceDeviceId>+<RoutingKeyword> で送信先を特定し、自動設定

## TargetDeviceGroupId : アプリ (DLL) で上書き可

- 複数デバイスへの同時送信用途

- Robotics FX が RBFX.DeviceRouting から <SourceDeviceId>+<RoutingKeyword> で送信先を特定し、自動設定

## TargetDeviceId : アプリ (DLL) で上書き可

- 単一デバイスへの送信用途

- Robotics FX が RBFX.DeviceRouting から <SourceDeviceId>+<RoutingKeyword> で送信先を特定し、自動設定

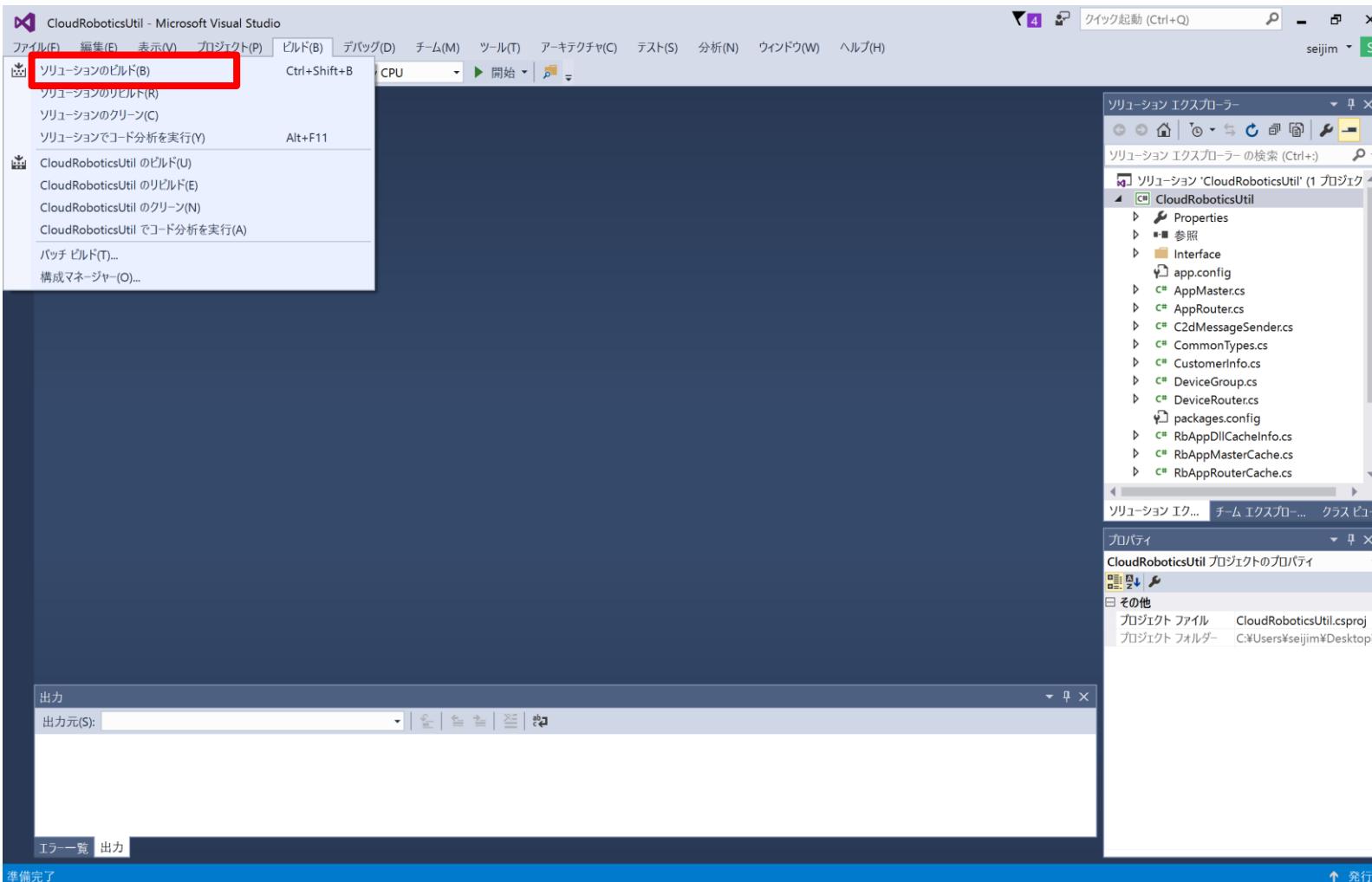
## ProcessingStack : 設定不要

- Robotics FX がどの DLL を呼び出したかの結果や、例外発生時の内容を自動設定

# T05\_01 : CloudRoboticsUtil.dll のビルド

Cloud Robotics Utility (共通機能 DLL) をビルドします

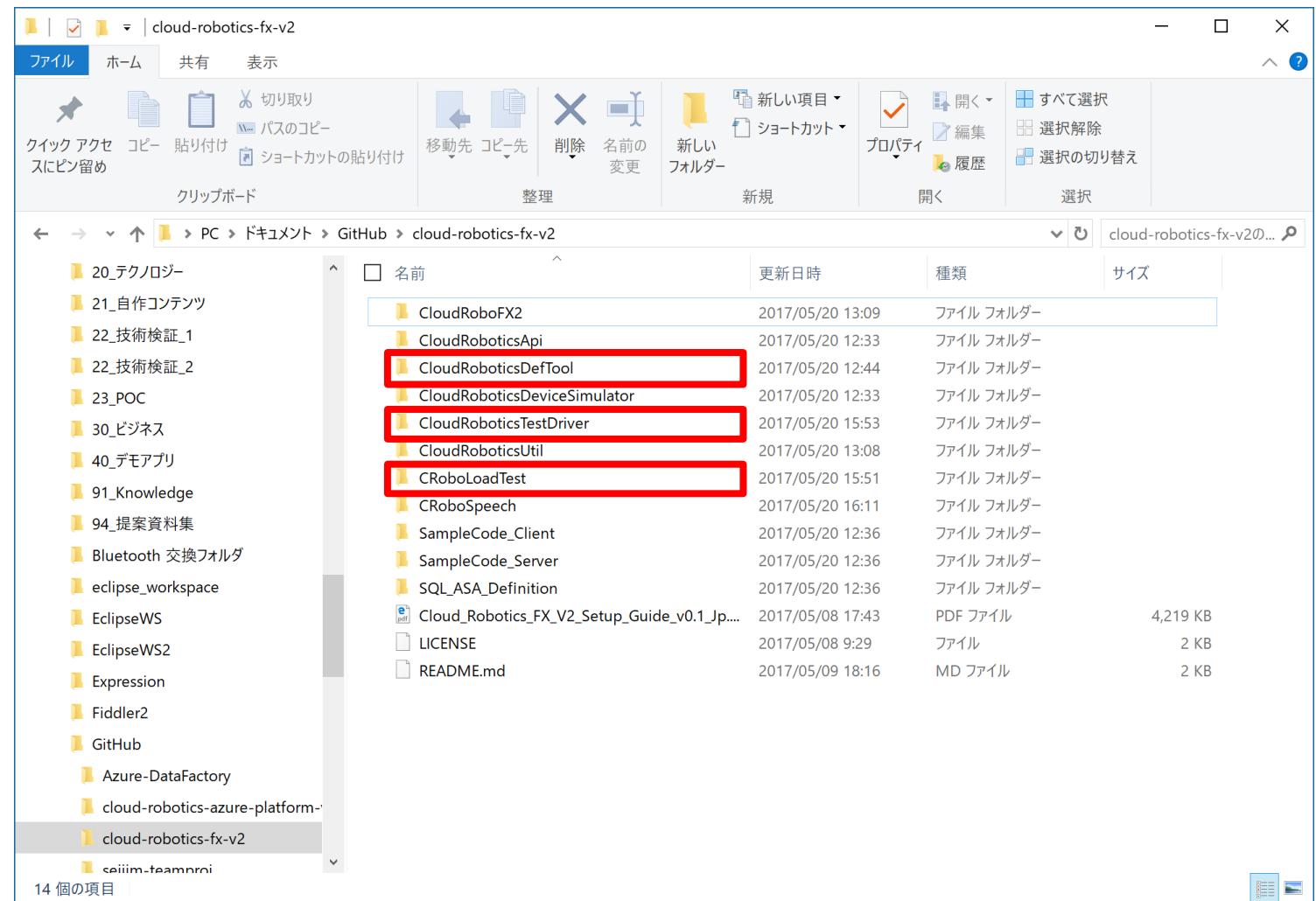
- 「<base directory>\CloudRoboticsUtil」 フォルダーを開き、「CloudRoboticsUtil.sln」をダブルクリックします
- Visual Studio 2015 のメニューバーから [ビルド]→[ソリューションのビルド] を選択します



# T05\_02：各ソリューションファイルのビルド

同様に、以下のソリューションファイルのビルドを行います

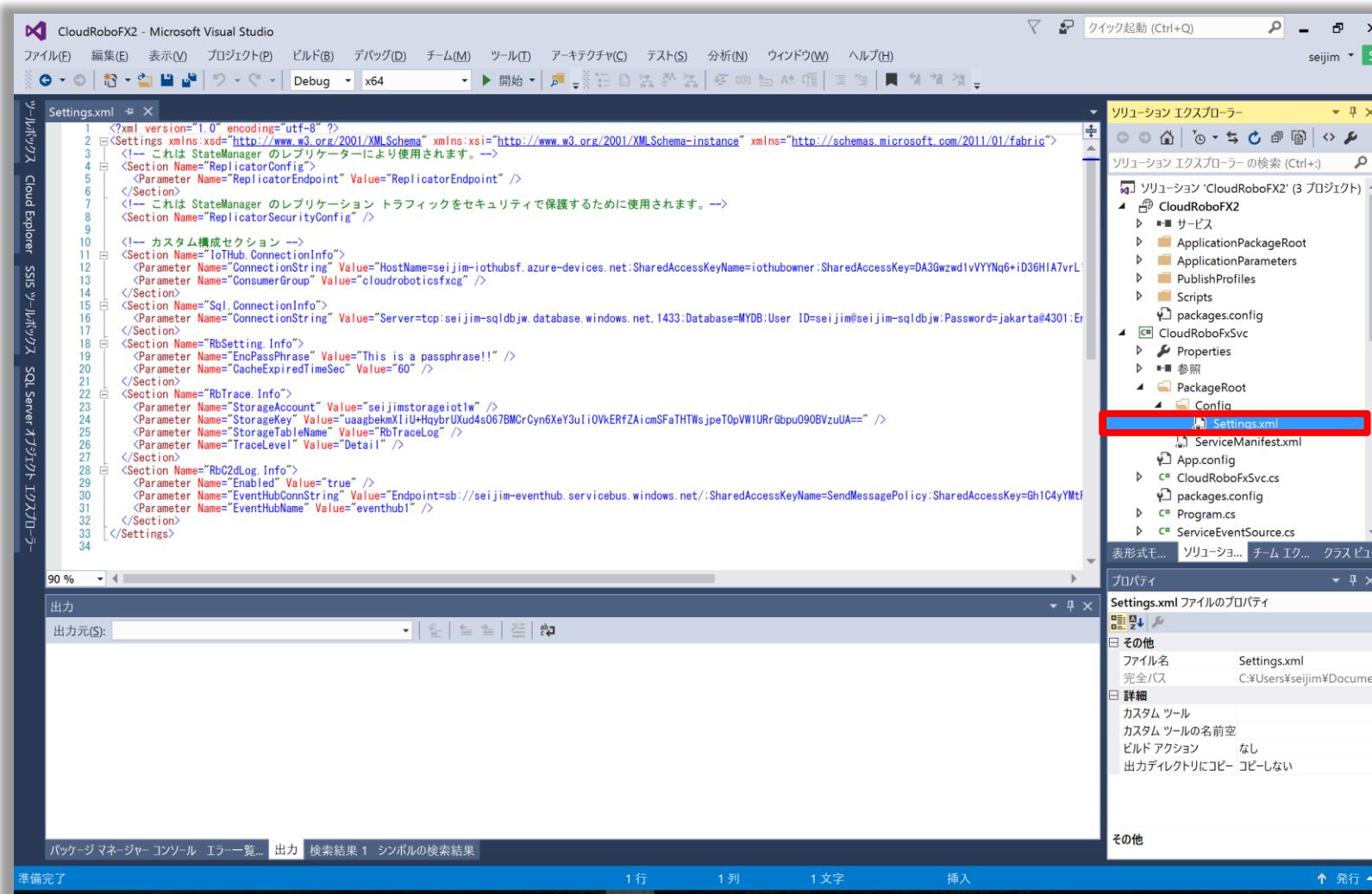
- CloudRoboticsDefTool (Definition & Management Tool)
- CloudRoboticsTestDriver (Test Driver)
- CRoboLoadTest (Load Test Tool)



# T05\_02：初期化パラメータの設定

Cloud Robotics FX V2 ソリューション ファイルを開き、アプリケーションの初期化パラメータ設定を行います

- 「CloudRoboFX2¥CloudRoboFX2.sln」を開きます
- 次に [CloudRoboFxSvc] ⇒ [PackageRoot] ⇒ [Config] ⇒ [Setting.xml] を開きます



# T05\_03：初期化パラメータの設定

Setting.xml の以下パラメータを設定します (V1 に比べると、シンプルになっています)

- [IoTHub.ConnectionInfo] ⇒ [ConnectionString]  
: [IoT Hub 接続文字列] を設定
- [IoTHub.ConnectionInfo] ⇒ [ConsumerGroup]  
: [IoT Hub 消費者グループ名] を設定
- [Sql.ConnectionInfo] ⇒ [ConnectionString]  
: [ADO.NET (SQL 認証)] 接続文字列を設定
- [RbSetting.Info] ⇒ [EncPassPhrase]  
: <任意の暗号化パスフレーズ> を設定
- [RbSetting.Info] ⇒ [CacheExpiredTimeSec]  
: RBFX.AppMaster, AppRouting のキャッシュ秒数を設定
- [RbTrace.Info] ⇒ [StorageAccount]  
: FX トレースログ用ストレージアカウント名を設定
- [RbTrace.Info] ⇒ [StorageKey]  
: FX トレースログ用ストレージキーを設定
- [RbTrace.Info] ⇒ [StorageTableName]  
: FX トレースログ用テーブル名「RbTraceLog」を設定
- [RbTrace.Info] ⇒ [TraceLevel]  
: FX トレースログ用出力モード「Detail」(1 メッセージ読み取り毎のログ出力) or 未設定
- [RbC2dLog.Info] ⇒ [Enabled] (option)  
: Cloud Robotics FX からの送信メッセージのロギング有効化 ("true" or "false")
- [RbC2dLog.Info] ⇒ [EventHubConnString] (option)  
: Event Hubs の接続文字列
- [RbC2dLog.Info] ⇒ [EventHubName] (option)  
: Event Hubs のエンティティ名

# T05\_04：サービスレプリカの初期設定

ソリューションファイルを開き、Service Fabric の設定を行います

- 「CloudRoboFX2\CloudRoboFX2.sln」を開きます
- 次に [CloudRoboFx2] ⇒ [ApplicationPackageRoot] ⇒ [ApplicationManifest.xml] を開きます
- サービスレプリカが担当するパーティションの全体数と範囲 (LowKey, HighKey) を設定します

CloudRoboFX2 - Microsoft Visual Studio (管理者)

ApplicationManifest.xml

```
<?xml version="1.0" encoding="utf-8"?>
<ApplicationManifest xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" ApplicationTypeName="CloudRoboFX2" xmlns="http://schemas.microsoft.com/2011/01/fabric">
  <Parameters>
    <Parameter Name="CloudRoboFxSvc_MinReplicaSetSize" DefaultValue="3" />
    <Parameter Name="CloudRoboFxSvc_PartitionCount" DefaultValue="4" />
    <Parameter Name="CloudRoboFxSvc_TargetReplicaSetSize" DefaultValue="3" />
  </Parameters>
  <!-- Import the ServiceManifest from the ServicePackage. The ServiceManifestName and ServiceManifestVersion should match the Name and Version attributes of the ServiceManifest element defined in the ServiceManifest.xml file. -->
  <ServiceManifestImport>
    <ServiceManifestRef ServiceManifestName="CloudRoboFxSvcPkg" ServiceManifestVersion="1.0.0" />
    <ConfigOverrides />
  </ServiceManifestImport>
  <DefaultServices>
    <!-- The section below creates instances of service types, when an instance of this application type is created. You can also create one or more instances of service type using the ServiceFabric PowerShell module. -->
    <Service Name="CloudRoboFxSvc">
      <StatefulService ServiceTypeName="CloudRoboFxSvcType" TargetReplicaSetSize="[CloudRoboFxSvc_TargetReplicaSetSize]" MinReplicaSetSize="[CloudRoboFxSvc_MinReplicaSetSize]" PartitionCount="[CloudRoboFxSvc_PartitionCount]" LowKey="0" HighKey="3" />
    </StatefulService>
  </DefaultServices>
</ApplicationManifest>
```

ソリューション エクスプローラー

ソリューション エクスプローラー の検索 (Ctrl+.)

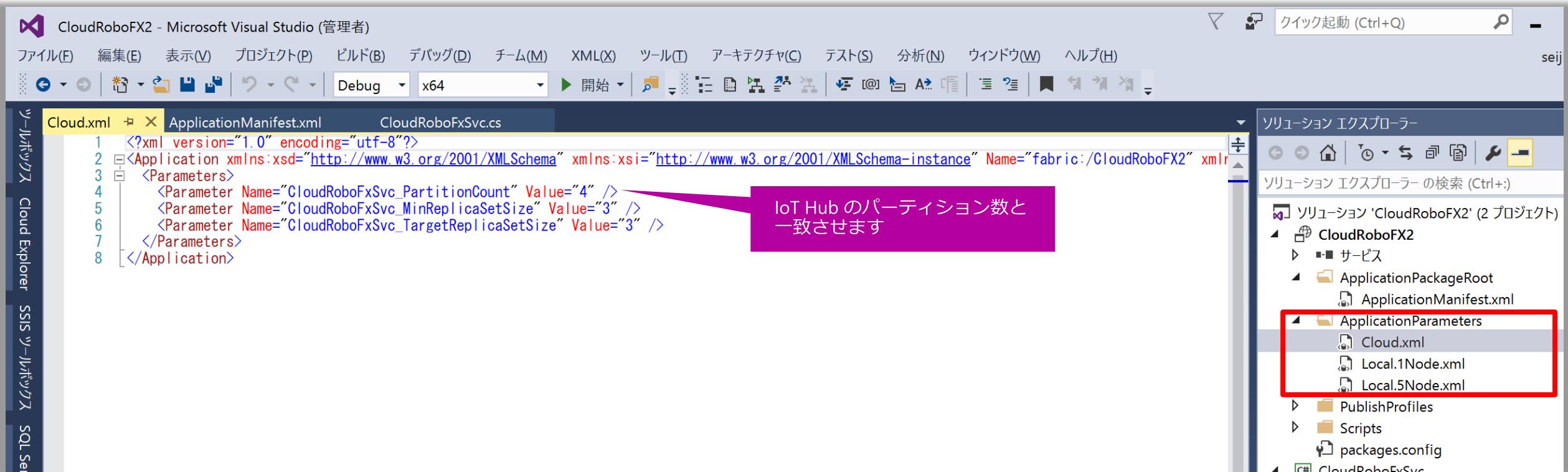
ソリューション 'CloudRoboFX2' (2 プロジェクト)

- CloudRoboFX2
  - サービス
  - ApplicationPackageRoot
    - ApplicationManifest.xml
    - ApplicationParameters
    - PublishProfiles
    - Scripts
    - packages.config
  - CloudRoboFxSvc
    - Properties
    - 参照
    - PackageRoot
      - App.config
      - CloudRoboFxSvc.cs
      - packages.config
      - Program.cs
      - ServiceEventSource.cs

# T05\_05：サービスレプリカの起動時パラメータ設定

ソリューションファイルを開き、Service Fabric の設定を行います

- 「CloudRoboFX2\CloudRoboFX2.sln」を開きます
- 次に、環境に合わせて、[CloudRoboFx2] ⇒ [ApplicationParameters] 以下のファイルを開きます
- サービスレプリカのパーティション数の設定を行います (以下の場合、 $4 \times 3$  (Primary:1, Secondary:2) = 6 個の Instance が作成される)



# T05\_06 : Cloud Robotics FX V2 のビルド

ソリューション ファイルを開き、ビルドを行います

- 「CloudRoboFX2\CloudRoboFX2.sln」を開き、「ビルド」を実行します

The screenshot shows the Microsoft Visual Studio interface with the following details:

- Title Bar:** CloudRoboFX2 - Microsoft Visual Studio
- Menu Bar:** ファイル(F)、編集(E)、表示(V)、プロジェクト(P)、ビルド(B)、デバグ(D)、チーム(M)、ツール(T)、アーキテクチャ(C)、テスト(S)、分析(N)、ウィンドウ(W)、ヘルプ(H)
- Toolbar:** Standard icons for file operations.
- Code Editor:** The main window displays the code for `CloudRoboFxService.cs`. The code is part of the `CloudRoboticsFX` namespace and defines a `CloudRoboFxService` class that inherits from `StatefulService`. It includes several private fields and methods, such as `OffsetDictionaryName`, `EpochDictionaryName`, and `appException`.
- Solution Explorer:** Shows the project structure for `'CloudRoboFX2' (2 プロジェクト)`. The `CloudRoboFX2` project contains `CloudRoboFx2` (サービス), `ApplicationPackageRoot`, `ApplicationParameters`, `PublishProfiles`, `Scripts`, and `CloudRoboFxSvc`. The `CloudRoboFxSvc` project contains `Properties`, `参照`, `PackageRoot`, `App.config`, `CloudRoboFxService.cs`, `packages.config`, `Program.cs`, and `ServiceEventSource.cs`.
- Properties Explorer:** Shows the properties for the `CloudRoboFxService.csproj` project.
- Task List:** Shows the tasks available for the current file.
- Output Window:** Shows the message "準備完了" (Prepared).

# T05\_07 : Cloud Robotics FX の設定 (Option)

Cloud Robotics FX のパラメータで、 [RbC2dLog.Info] の [Enabled] = "true" にした場合は、 Event Hubs を構成します

- ① Event Hubs の作成
- ② Shared access policies (send 専用) の作成
- ③ Entities の作成

The screenshot shows three overlapping Azure portal windows illustrating the configuration steps:

- Left Window (New Resource):** Shows the 'Event Hubs' service selected under 'モノのインターネット'. A red box highlights the 'Event Hubs' icon.
- Middle Window (Shared access policies):** Shows the 'Send' policy being created. A red box highlights the '+ Add' button.
- Right Window (Event Hubs Overview):** Shows the newly created 'eventhub1' hub. A red box highlights the '+ Event Hub' button.

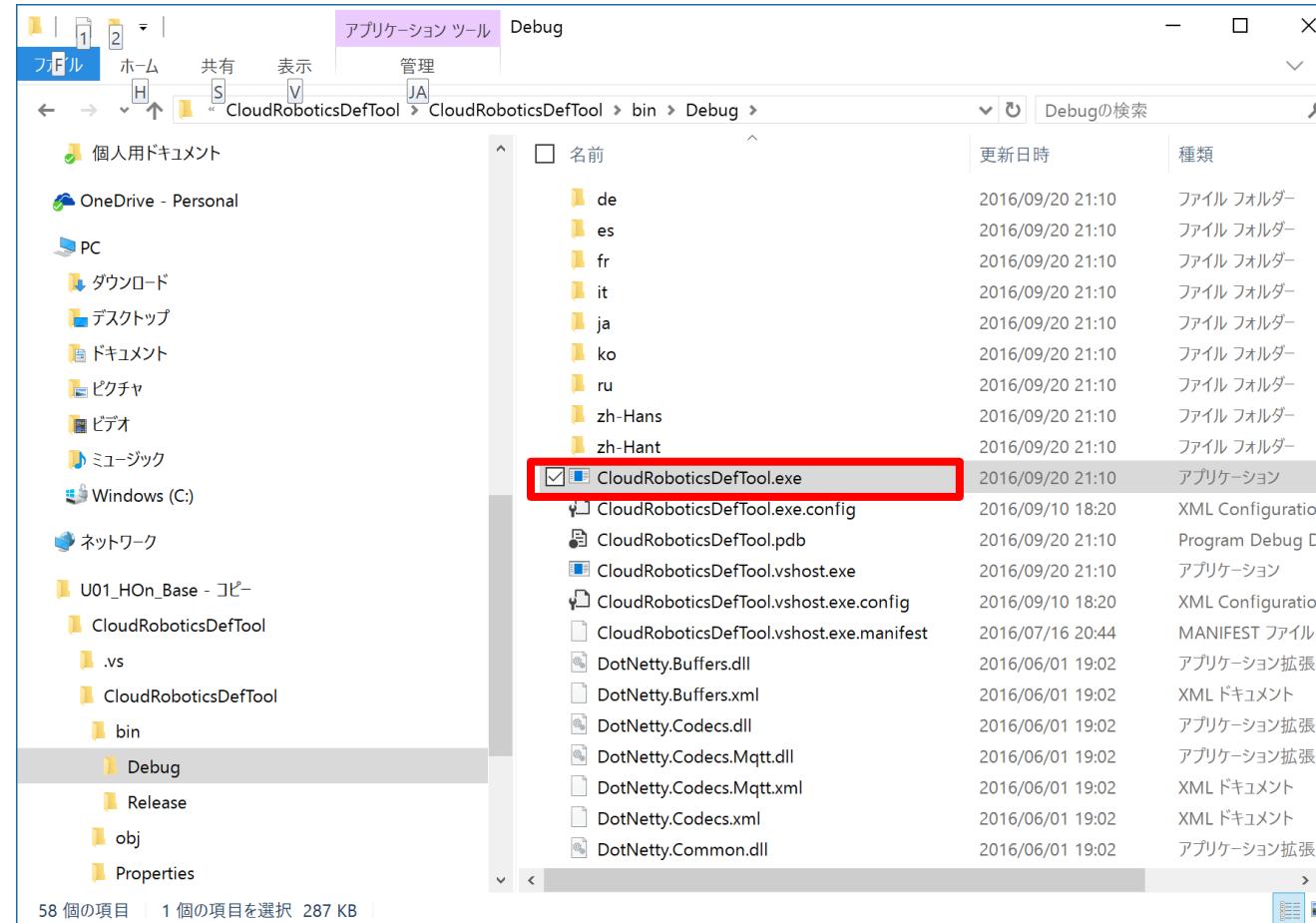
# Tutorial 06

## デバイス間通信の為の設定

# T06\_01 : Cloud Robotics Definition Tool へのショートカット

T05\_02 でビルドした Cloud Robotics Definition Tool (Management Tool) へのショートカットを作成します

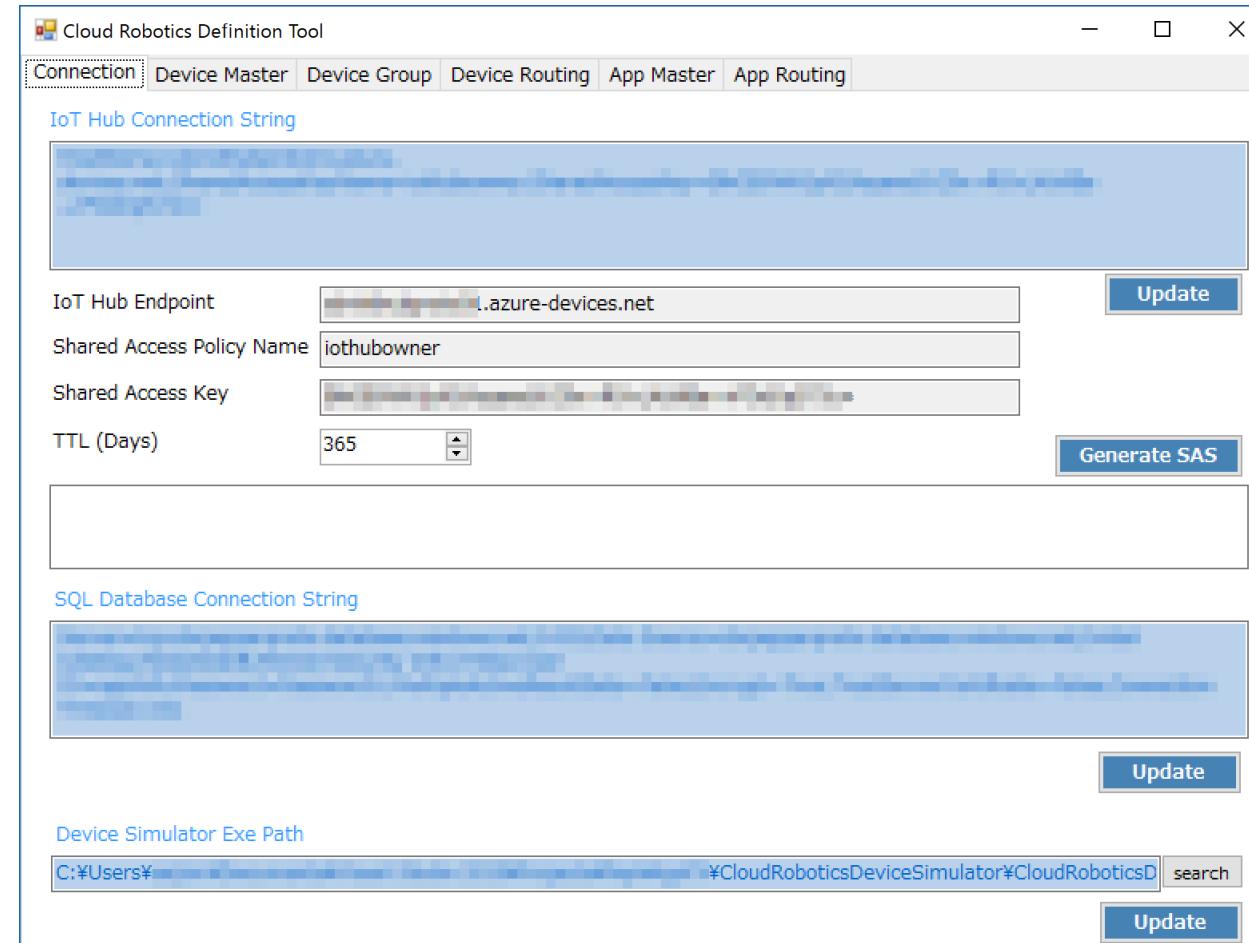
- 「CRSDKv1¥CloudRoboticsDefTool¥CloudRoboticsDefTool¥bin¥Debug」に移動します
- 「CloudRoboticsDefTool.exe」のショートカットをデスクトップなどに作成してください



# T06\_02 : Cloud Robotics Definition Tool の初期設定

Cloud Robotics Definition Tool (Management Tool) をショートカットから起動し、以下を設定して [Update] していきます

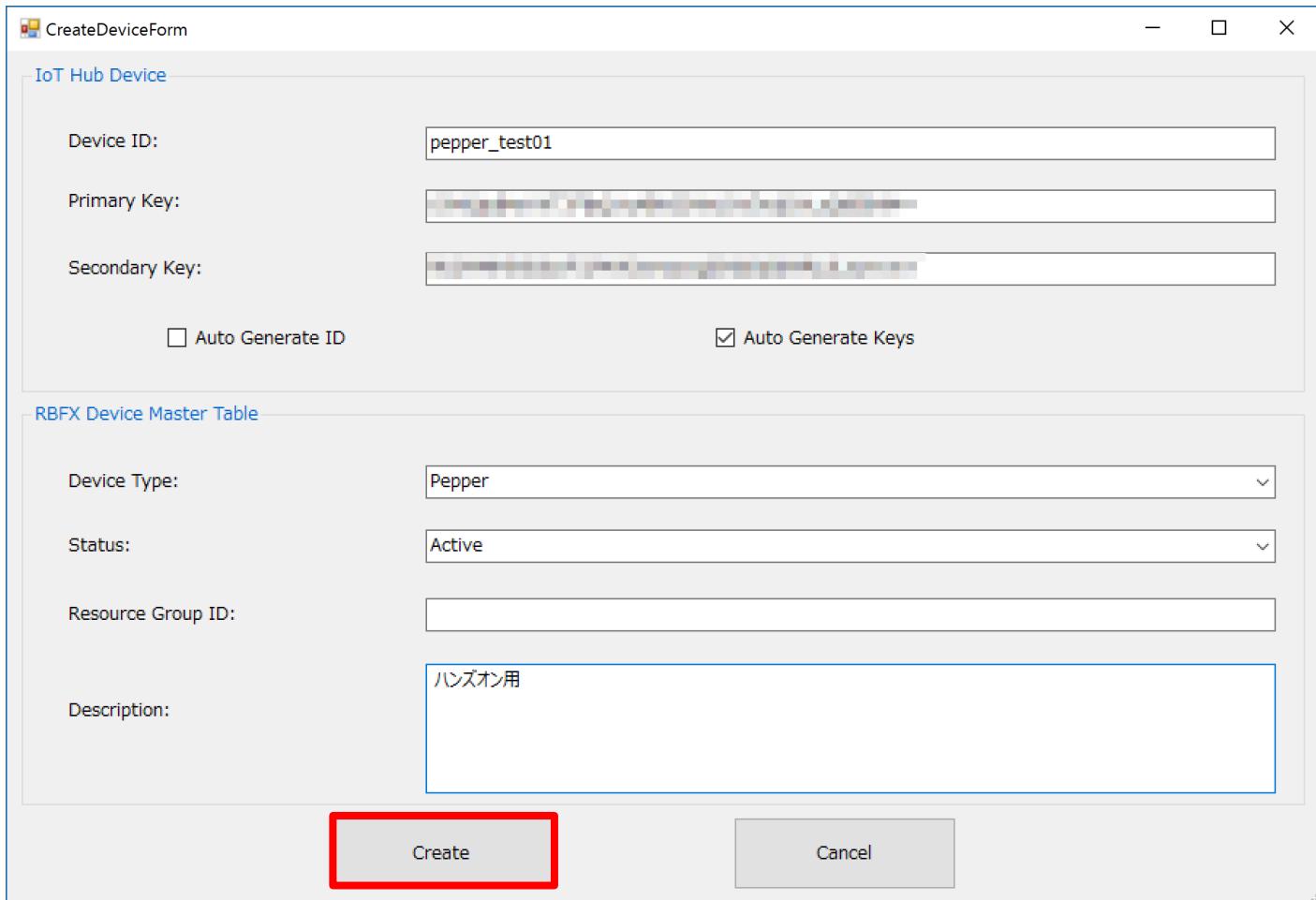
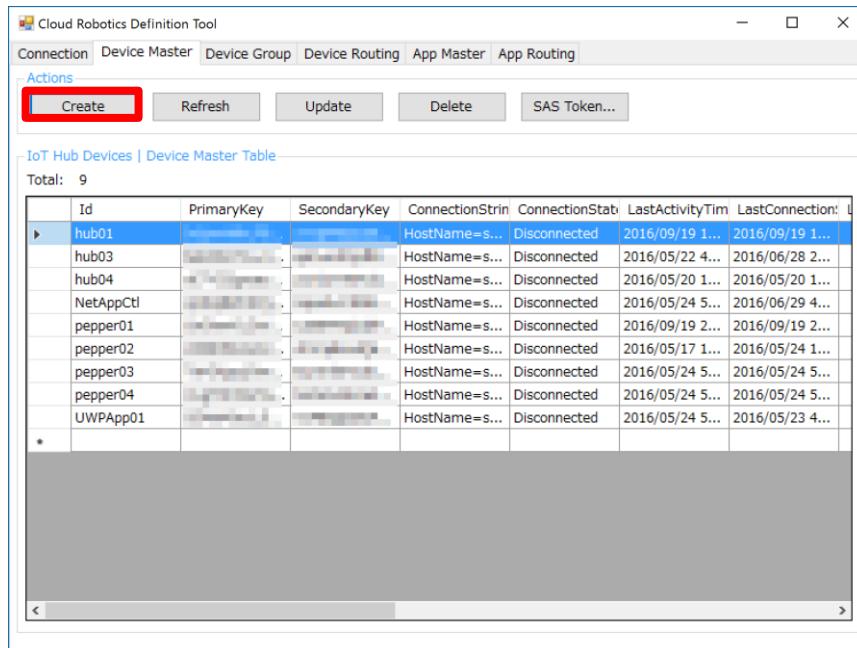
- [IoT Hub Connection String] : T02\_03 でメモ帳に保存した [IoT Hub 接続文字列] を設定
- [SQL Database Connection String] : T03\_05 でメモ帳に保存した [ADO.NET (SQL 認証)] 接続文字列を設定
- Device Simulator Exe Path : ここは設定する必要がありません



# T06\_03：IoT Hub & Device Masterへのデバイス登録

Cloud Robotics Definition Tool (Management Tool) の [Device Master] タブ→[Create] から以下のデバイスを登録します

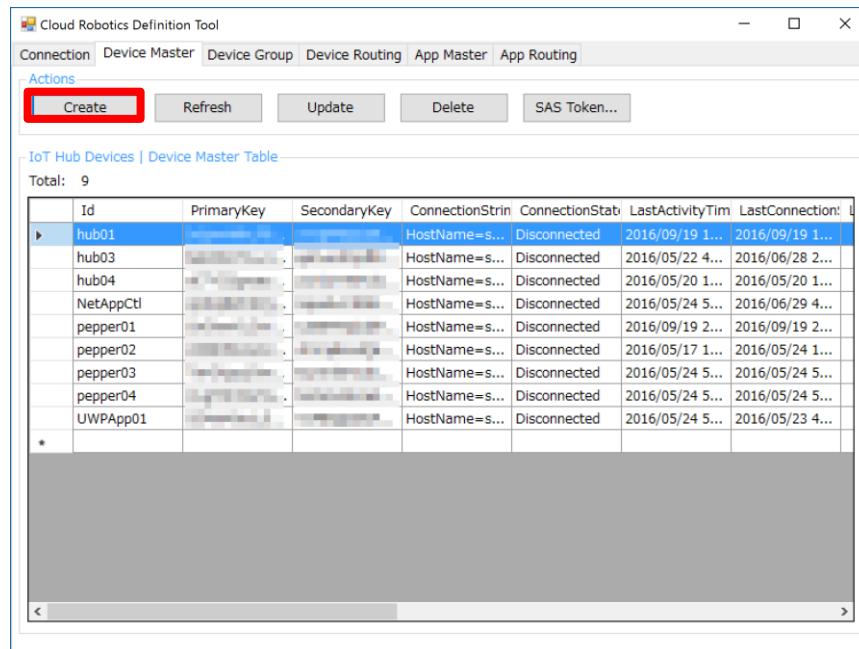
- [Device Id] : 「pepper\_test01」(Pepper 用デバイス)
  - [Device Type] : 「Pepper」を選択
  - [Status] : 「Active」を選択



# T06\_04 : IoT Hub & Device Masterへのデバイス登録

Cloud Robotics Definition Tool (Management Tool) の [Device Master] タブ→[Create] から以下のデバイスを登録します

- [Device Id] : 「hub\_test01」(Surface Hub 用デバイス)
- [Device Type] : 「Surface Hub」を選択
- [Status] : 「Active」を選択

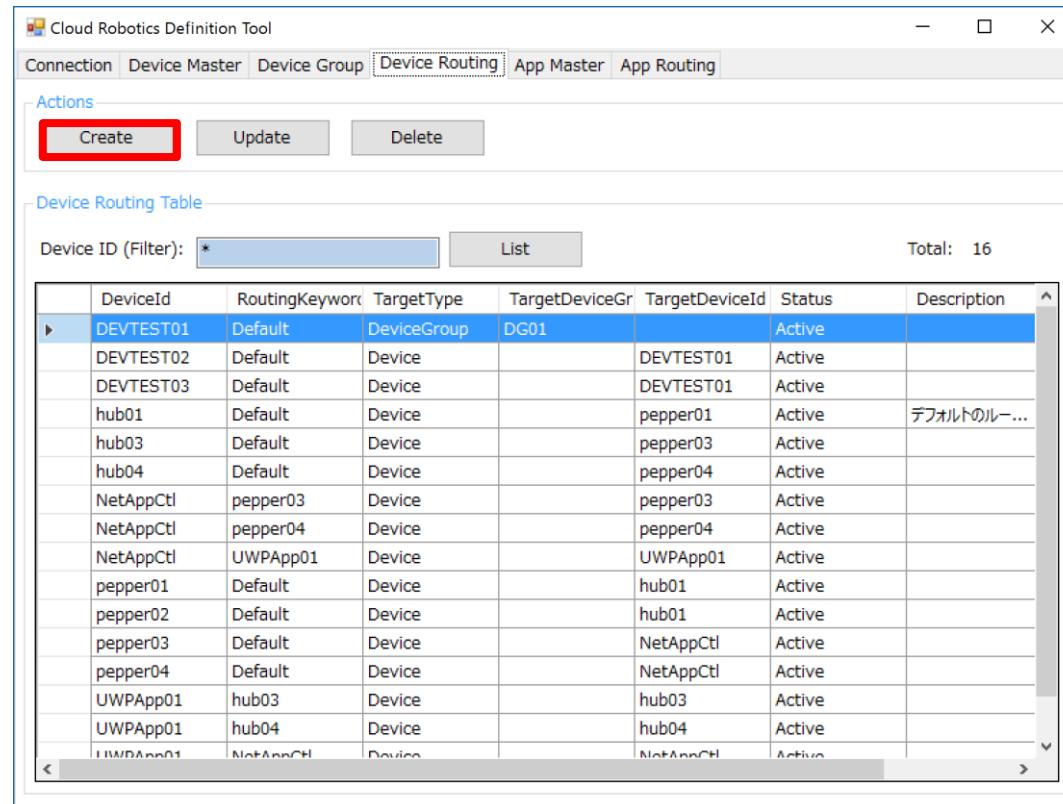


The screenshot shows the 'CreateDeviceForm' dialog box for creating an IoT Hub Device. The 'IoT Hub Device' section contains fields for 'Device ID' (set to 'hub\_test01'), 'Primary Key' (redacted), and 'Secondary Key' (redacted). There are checkboxes for 'Auto Generate ID' (unchecked) and 'Auto Generate Keys' (checked). The 'RBFX Device Master Table' section contains fields for 'Device Type' (set to 'Surface Hub'), 'Status' (set to 'Active'), 'Resource Group ID' (empty), and 'Description' (empty). At the bottom right are 'Create' and 'Cancel' buttons, with the 'Create' button highlighted with a red box.

# T06\_05 : Device Routingへのルーティング登録

Cloud Robotics Definition Tool (Management Tool) の [Device Routing] タブ→[Create] から以下のルーティングを登録します

- [Device ID] : 「pepper\_test01」を入力
- [Target Device Type] : 「Device」を選択
- [Target Device ID] : 「hub\_test01」を入力



The screenshot shows the 'EditDevRoutingForm' dialog box for creating a new device routing entry. It contains fields for 'Device ID' (set to 'pepper\_test01'), 'Routing Keyword' (set to 'Default'), 'Target Device Type' (set to 'Device'), 'Target Device Group ID' (empty), 'Target Device ID' (set to 'hub\_test01'), 'Status' (set to 'Active'), and a 'Description' field (empty). At the bottom, there are buttons for 'Create', 'Update', and 'Cancel', with the 'Create' button highlighted with a red box.

# T06\_06 : Device Routingへのルーティング登録

Cloud Robotics Definition Tool (Management Tool) の [Device Routing] タブ→[Create] から以下のルーティングを登録します

- [Device ID] : 「hub\_test01」を入力
- [Target Device Type] : 「Device」を選択
- [Target Device ID] : 「pepper\_test01」を入力

The screenshot shows the 'Device Routing' tab of the Cloud Robotics Definition Tool. At the top, there's a toolbar with tabs: Connection, Device Master, Device Group, Device Routing (which is selected and highlighted with a dotted border), App Master, and App Routing. Below the toolbar is an 'Actions' bar with three buttons: 'Create' (highlighted with a red box), 'Update', and 'Delete'. The main area is titled 'Device Routing Table' and contains a table with columns: DeviceID, RoutingKeyword, TargetType, TargetDeviceGr, TargetDeviceId, Status, and Description. A filter input 'Device ID (Filter): \*' and a 'List' button are at the top of the table. The table shows 16 entries. One entry in the 'Description' column for row 15 is partially visible as 'デフォルトのルー...'. The table has a header row and several data rows.

The screenshot shows the 'EditDevRoutingForm' dialog box for creating a new device routing entry. The form is titled 'RBFX Device Routing Table'. It contains the following fields:

- Device ID: hub\_test01
- Routing Keyword: Default
- Target Device Type: Device (selected from a dropdown)
- Target Device Group ID: (empty text field)
- Target Device ID: pepper\_test01
- Status: Active (selected from a dropdown)
- Description: (empty text area)

At the bottom of the dialog are three buttons: 'Create' (highlighted with a red box), 'Update', and 'Cancel'.

# T06\_07 : Device Simulator の初期設定

CloudRoboticsDeviceSimulator の通信メッセージテンプレートの設定を行います

- Cloud Robotics Definition Tool (Management Tool) の [Device Master] タブで、所定のデバイスを選択し、右クリックします
- [Launch Device Simulator] を選択して、Device Simulator を起動します
- [Launch Message Edit Form] ボタンを押して、通信メッセージテンプレート編集フォームを開きます
- 「<base directory>\CloudRoboticsDefTool\DeviceSimulatorMessages.txt」をメモ帳などで開き、コピペして、[Save] します

The screenshots illustrate the steps to configure the communication message template:

- The first screenshot shows the 'Device Master' table in the Cloud Robotics Definition & Management Tool. The row for 'dev01' is selected, and the 'Actions' menu is open, showing options like 'Launch Device Simulator' and 'Launch Message Edit Form'. The 'Launch Device Simulator' option is highlighted.
- The second screenshot shows the 'Device Simulator' window. It displays a simulated device interface and lists actions for the selected device 'dev01'. The 'Launch Message Edit Form' button is highlighted.
- The third screenshot shows the 'Json Message Edit Form' dialog box. It contains a JSON message template with fields for RbBody, Ttyle, MessageType, RbMessage, RbHeader, and RbBody. The 'Save' button at the bottom right is highlighted.

```
JSON Message Template (Visible in the third screenshot):
```

```
{
  "RbBody": {
    "visitor": "u001",
    "gender": "f",
    "age": "29",
    "product": "S01_0840076146642",
    "quantity": "1",
    "stock": "1"
  },
  "Ttyle": "Call RbSampleApp",
  "MessageType": "CALL",
  "RbMessage": [
    {
      "RbHeader": {
        "RoutingType": "CALL",
        "RoutingKeyword": "Default",
        "AppId": "PepperShopApp",
        "AppProcessingId": "RbSampleApp",
        "MessageId": "SayHello",
        "MessageSeqno": "",
        "SendDateTime": ""
      }
    },
    {
      "RbBody": {
        "visitor": "u002"
      }
    }
  ]
}
```

# Tutorial 07

デバイス間通信のテスト  
(Cloud Robotics FX V2 のデバッグ実行)

# Cloud Robotics FX トレースログの確認

事前にインストールしていた Microsoft Azure Storage Explorer を利用するか、Cloud Robotics Definition Tool を利用します

- Azure ポータル同様に、ログインを行います
- T05\_03 で設定した Cloud Robotics FX のトレースログ用ストレージアカウントを開きます
- [Tables] 配下の「RbTraceLog」をダブルクリックして、メッセージにエラーが出ていないことを確認します
- ※ 「RbTraceLog」は、Cloud Robotics FX を最初に起動した時に作成されます

The screenshot shows the Microsoft Azure Storage Explorer interface. On the left, the 'Resource Types' sidebar is selected, and the 'Tables' section is expanded, showing the 'RbTraceLog' table highlighted with a red box. The main pane displays a grid of log entries with columns: PartitionKey, RowKey, Timestamp, and LocalDateTime. The status bar at the bottom indicates 'Showing 3,601 to 3,698 of 3,698 loaded items.'

The screenshot shows the 'Cloud Robotics Definition & Management Tool' window. The top navigation bar has tabs: Connection, Device Master, Device Group, Device Routing, App Master, App Routing, and FX Trace Log, with 'FX Trace Log' highlighted by a red box. Below the tabs, there are fields for 'Storage Account' and 'Table Name' (set to 'RbTraceLog'). A 'UTC Date Filter' field contains '20161021'. The main area is a table listing log entries with columns: PartitionKey, RowKey, LocalDateTime, HostName, ThreadId, Category, andAppName. The table shows multiple entries for the date 20161021, all categorized as 'Info'.

# Cloud Robotics FX のトラブルシュート

デバッグ実行した時、ブレイクポイントにカーソルが来ない場合、或いは、デバイス間通信うまく行かない場合

- Microsoft Azure Storage Explorer を起動します
- Cloud Robotics FX のトレースログ「RbTraceLog」にエラーが出ていないかを確認します
- T05\_03 での FX 初期化パラメータの間違い、CloudRoboticsDefTool と FX との暗号化パスフレーズの不一致を疑います

Microsoft Azure Storage Explorer

Microsoft Azure

Resource Types

Search for resources

Storage Accounts

- (Development)
- (Service SAS)
- CloudRoboticsFXCG (highlighted with a red box)
- eventhub.info
- ...
- seijim- (highlighted with a red box)
  - Queues
  - Tables
- ...

Name Last Modified Blob Type Content Type Size

Fri, 29 Apr 2016 07:29:57 GMT Block Blob application/octet-stream 91 B

Actions Properties

Showing 1 to 2 of 2 loaded items.

URL: https://[REDACTED].w.blob  
Type: Blob Container  
Last Modified: Fri, 29 Apr 2016 07:29:56 GMT  
Public Read Access: Off

What do you like about this tool?  
What don't you like or feel is missing?

Microsoft Azure Storage Explorer

Microsoft Azure

Resource Types

Search for resources

Storage Accounts

- (Development)
- (Service SAS)
- ...
- Blob Containers
- Queues
- Tables (highlighted with a red box)
  - RbTraceLog (highlighted with a red box)
- ...

PartitionKey	RowKey	Timestamp	LocalData
20160920	201609201419000479_77c48b36-eaa5-4dd6-8097-810914d06e50	2016-09-20T14:19:00.884Z	2016-0
20160920	201609201419000979_3216080d-d8b0-4d35-a20d-e0d612860c3a	2016-09-20T14:19:01.303Z	2016-0
20160920	201609201419010026_a1cb0a9e-d3f-4050-af15-f9d0dbc22c75	2016-09-20T14:19:01.301Z	2016-0
20160920	201609201419010026_b4918b6e-d8a0-4a5f-9595-3fe5b81d461e	2016-09-20T14:19:01.408Z	2016-0
20160920	201609201419010042_26cdf77a-db3e-4ffd-9398-643dc51ef478	2016-09-20T14:19:01.326Z	2016-0
20160920	201609201419010158_5ef0821d-5cc0-4b82-b6da-f872915c4da4	2016-09-20T14:19:01.476Z	2016-0
20160920	201609201419010180_02eed73a-3857-4fd8-a4e9-865e6b8bc7e	2016-09-20T14:19:01.558Z	2016-0
20160920	201609201419020097_1f69537a-7230-4988-b5cb-9a3e6859720c	2016-09-20T14:19:02.414Z	2016-0
20160920	201609201419020097_2965a1f5-3636-49e4-8c1a-e8bb48d7f0fa	2016-09-20T14:19:02.501Z	2016-0
20160920	201609201419020175_e11552da-7b4b-43fd-8ec1-4e3a468785e7	2016-09-20T14:19:02.461Z	2016-0
20160920	201609201419050871_2d35e664-124d-458a-a98f-eb6ecc18d50	2016-09-20T14:19:06.202Z	2016-0
20160920	201609201419070246_f96178b5-27ca-485f-afe6-f91b249043ce	2016-09-20T14:19:07.552Z	2016-0
20160920	201609201419410822_55a9e7e5-38f8-4668-bfe0-38f73a3b71bb	2016-09-20T14:19:42.144Z	2016-0
20160920	201609201418050187_24099774-9f26-4d3f-a454-75ee0e5018b2	2016-09-20T14:18:05.869Z	2016-0

Showing 3,601 to 3,698 of 3,698 loaded items.

Actions Properties

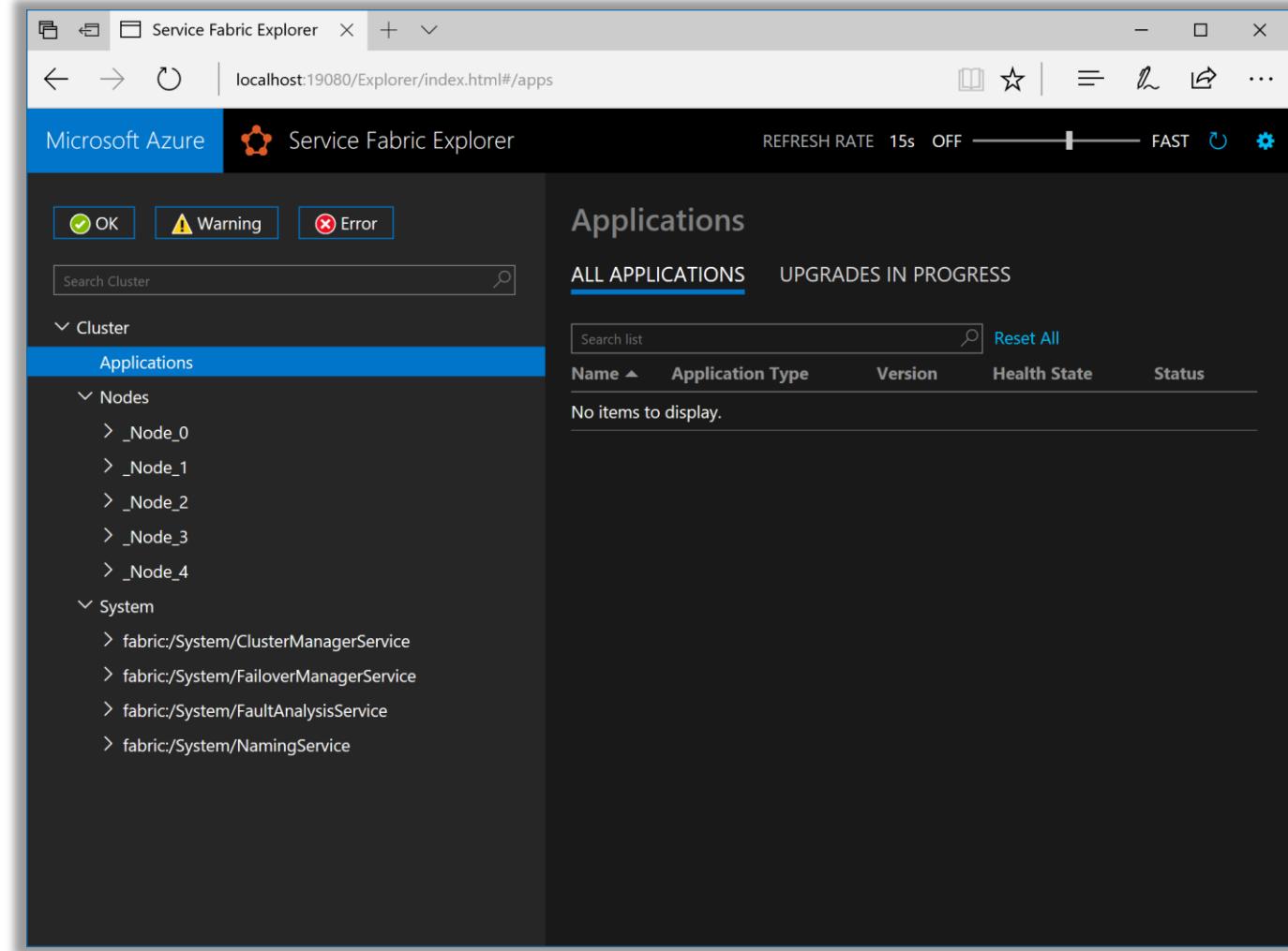
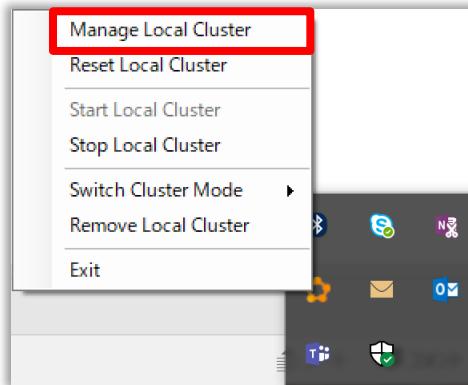
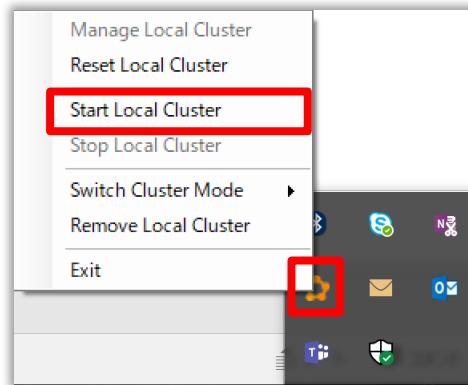
URL: https://[REDACTED].table.core.windows.net  
Type: Table

Activity Log

What do you like about this tool?  
What don't you like or feel is missing?

# T07\_01 : Service Fabric ローカル クラスターの実行

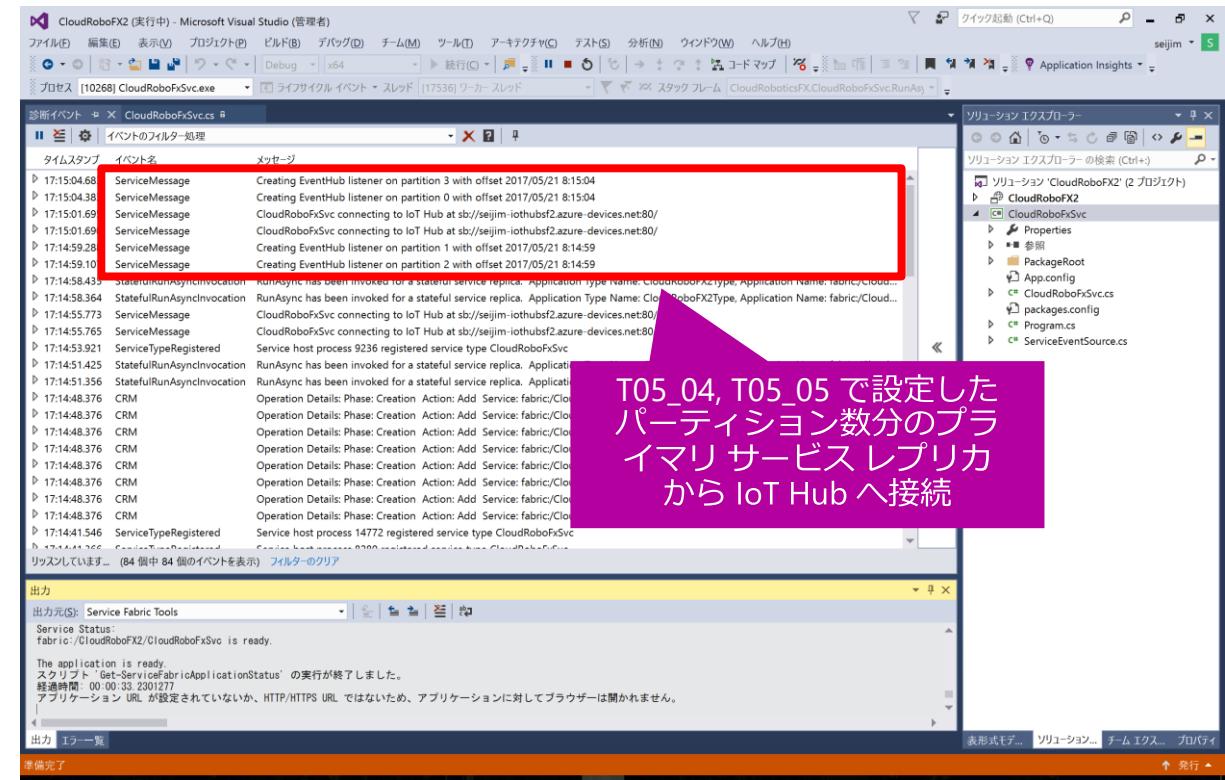
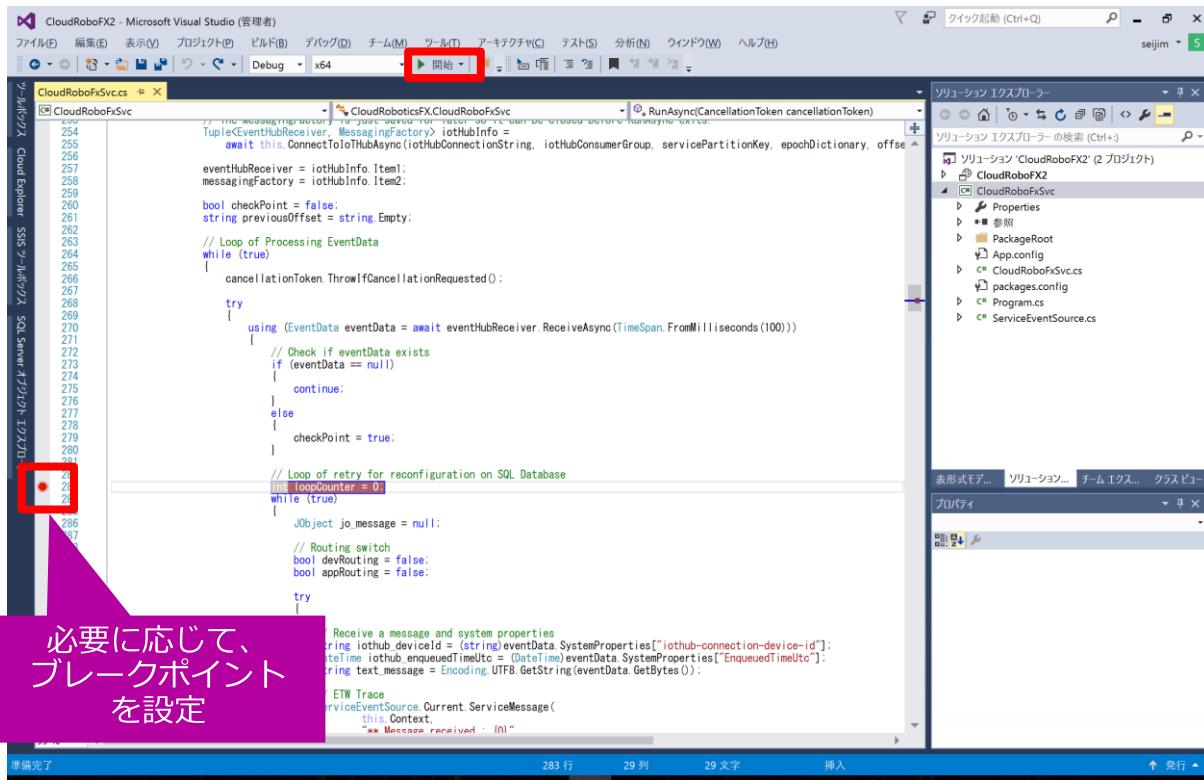
- タスクトレイに入っている Service Fabric ローカル クラスターを開始 (Start Local Cluster) します
- 開始が完了すると、[Manage Local Cluster] が Active になりますので、クリックします
- ローカル環境の Service Fabric Explorer で、クラスターの状態を確認することができます



# T07\_02 : Cloud Robotics FX V2 のデバッグ実行

Visual Studio を「管理者として実行」し、アプリケーションを Service Fabric ローカル クラスターへの配置を行います

- 「CloudRoboFX2\CloudRoboFX2.sln」を開きます
- もし、Cloud Robotics FX V1 が使っている IoT Hub と同一のものを利用する場合、クラウドサービスを一度停止してください
- [開始] ボタンをクリックして、Service Fabric クラスターへの配置を行います
- [診断イベント] で、設定したパーティション数分のプライマリ サービス レプリカが RunAsync を開始し、IoT Hub に接続したことが分かります
- セカンダリ サービス レプリカでは、RunAsync は実行されません



必要に応じて、ブレークポイントを設定

T05\_04, T05\_05 で設定したパーティション数分のプライマリ サービス レプリカから IoT Hub へ接続

# T07\_03 : Service Fabric Explorer による状態監視

- タスクトレイから Service Fabric を右クリックして、[Manage Local Cluster] をクリックします
- 以下のように、ドリルダウンして行きます
- この例では、4つのパーティションが存在し、その中にそれぞれ、3つのサービス レプリカが稼働していることが分かります

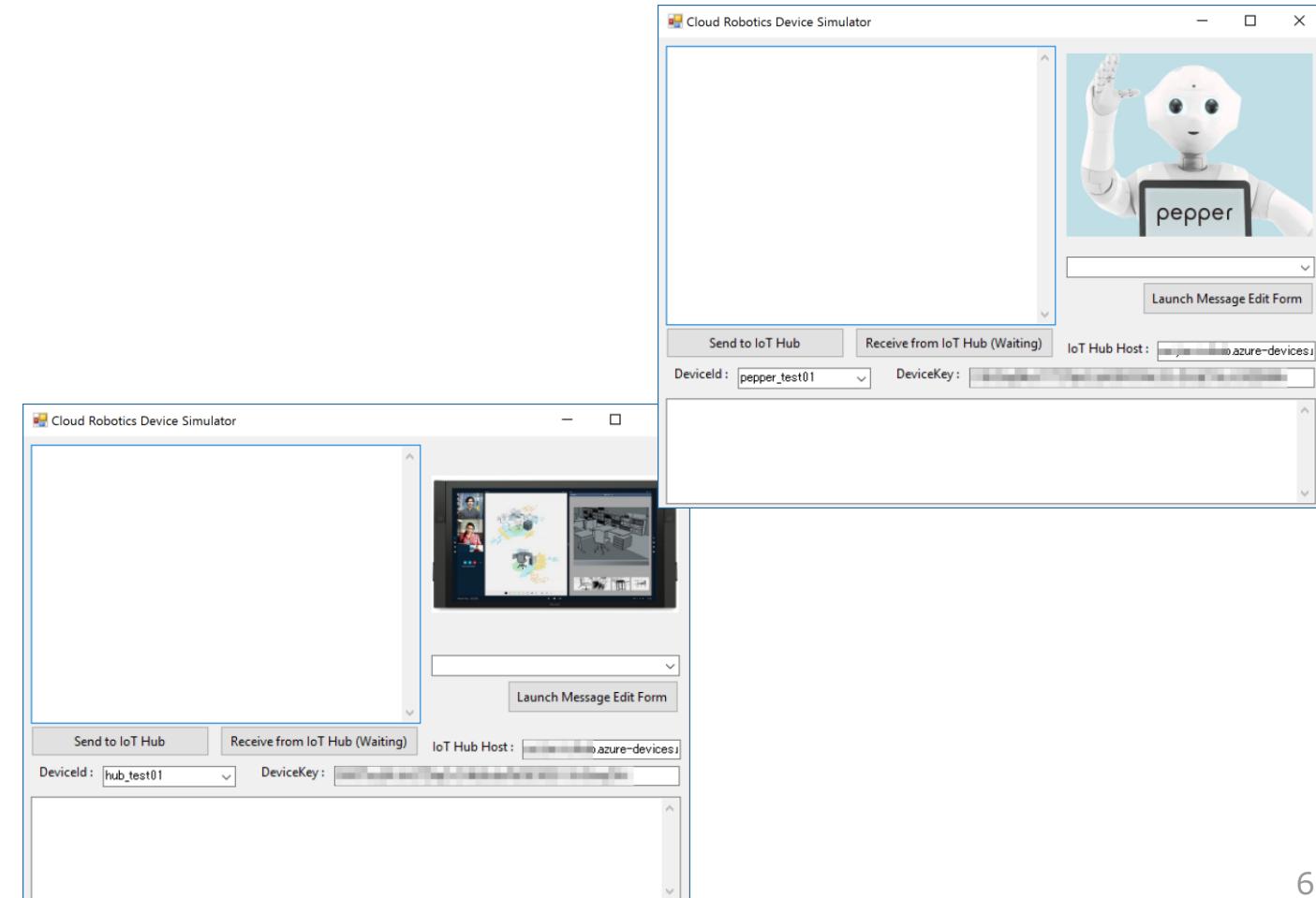
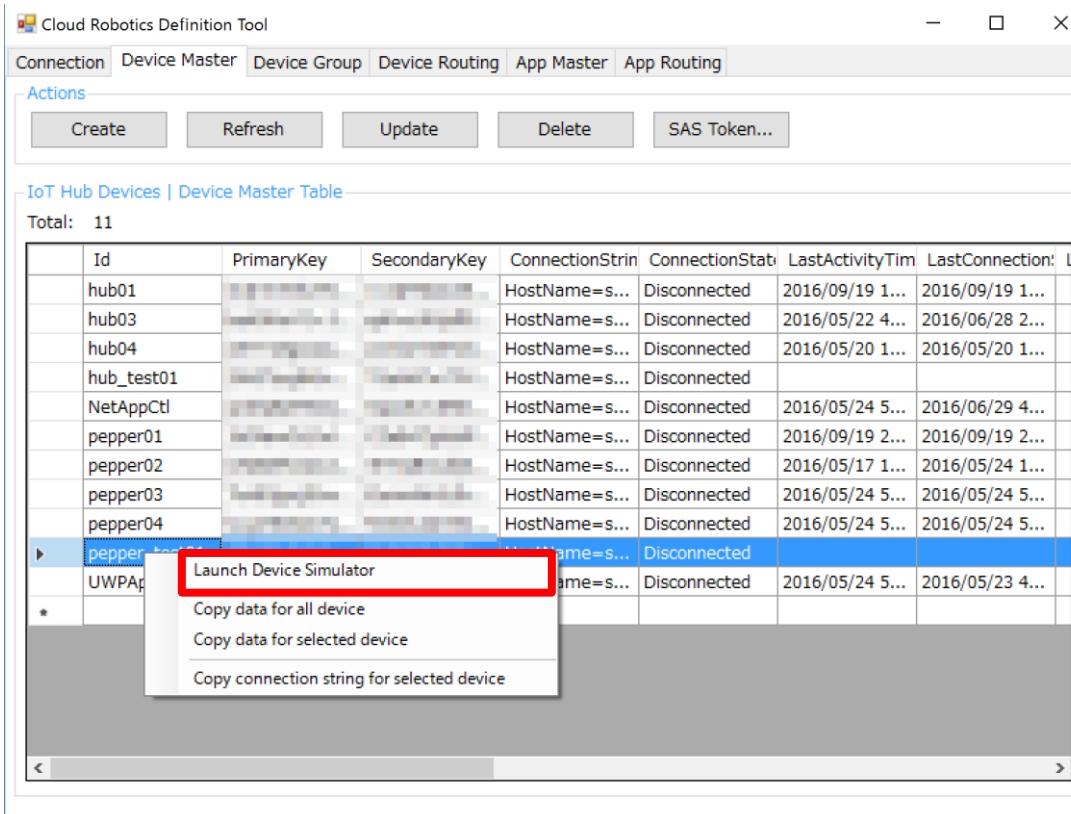
The screenshot shows the Service Fabric Explorer interface for monitoring a local cluster. The main window displays the details of a specific partition, "Partition 12af0653-ac26-4486-a8eb-33d2cfaaa15d". The "ESSENTIALS" tab is selected, showing the partition's ID, kind (Int64Range), health state (OK), and other configuration details like minimum and target replica sizes. Below this, the "REPLICAS" section is highlighted with a red box, showing three replicas: one primary and two active secondaries, all in an "OK" state and labeled "Ready". The left sidebar shows a tree view of the cluster structure, including applications like "CloudRoboFX2Type" and "fabric:/CloudRoboFX2/CloudRoboFxFvc".

Id	Node Name	Replica Role	Health State	Status
131398280687110607	_Node_2	Primary	OK	Ready
131398280883860201	_Node_1	ActiveSecondary	OK	Ready
131398280883860200	_Node_4	ActiveSecondary	OK	Ready

# T07\_04 : Device Simulator によるテスト

Cloud Robotics Definition Tool (Management Tool) をショートカットから起動します

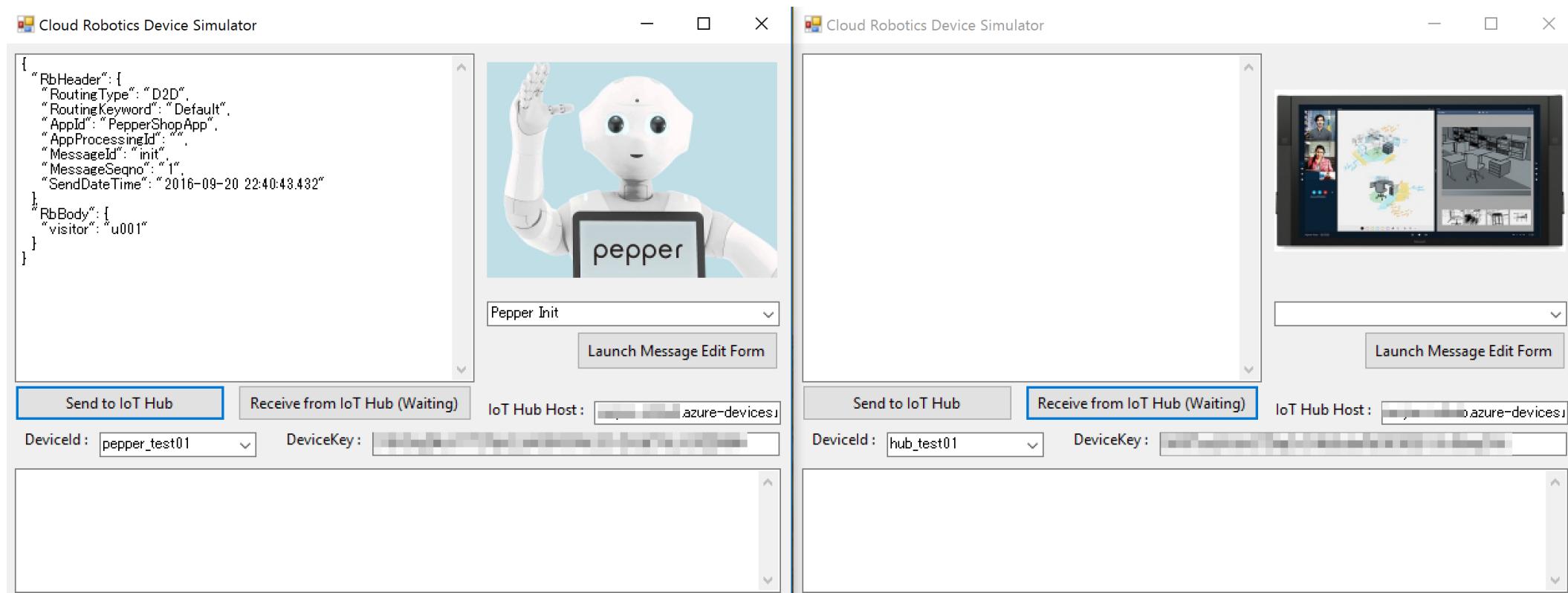
- [Device Master] タブに移動します
- [id]=「pepper\_test01」を選択し、その後、右クリックをして、「Launch Device Simulator」をクリックします
- 同様に、[id]=「hub\_test01」を選択し、その後、右クリックをして、「Launch Device Simulator」をクリックします



# T07\_05 : Device Simulator によるテスト

Cloud Robotics Device Simulator を使って、デバイス ルーティングのテストを実施します

- 「hub\_test01」(Surface Hub) のシミュレーターで、[Receive from IoT Hub (Waiting)] ボタンを押します
- 「pepper\_test01」(Pepper) のシミュレーターで、[Receive from IoT Hub (Waiting)] ボタンを押します
- 「pepper\_test01」(Pepper) のシミュレーターで、通信メッセージテンプレートから「Pepper Init」を選択します
- 「pepper\_test01」(Pepper) のシミュレーターで、[Send to IoT Hub] ボタンを押します



# Tutorial 08

Cloud Robotics FX V2 のデイプロイ

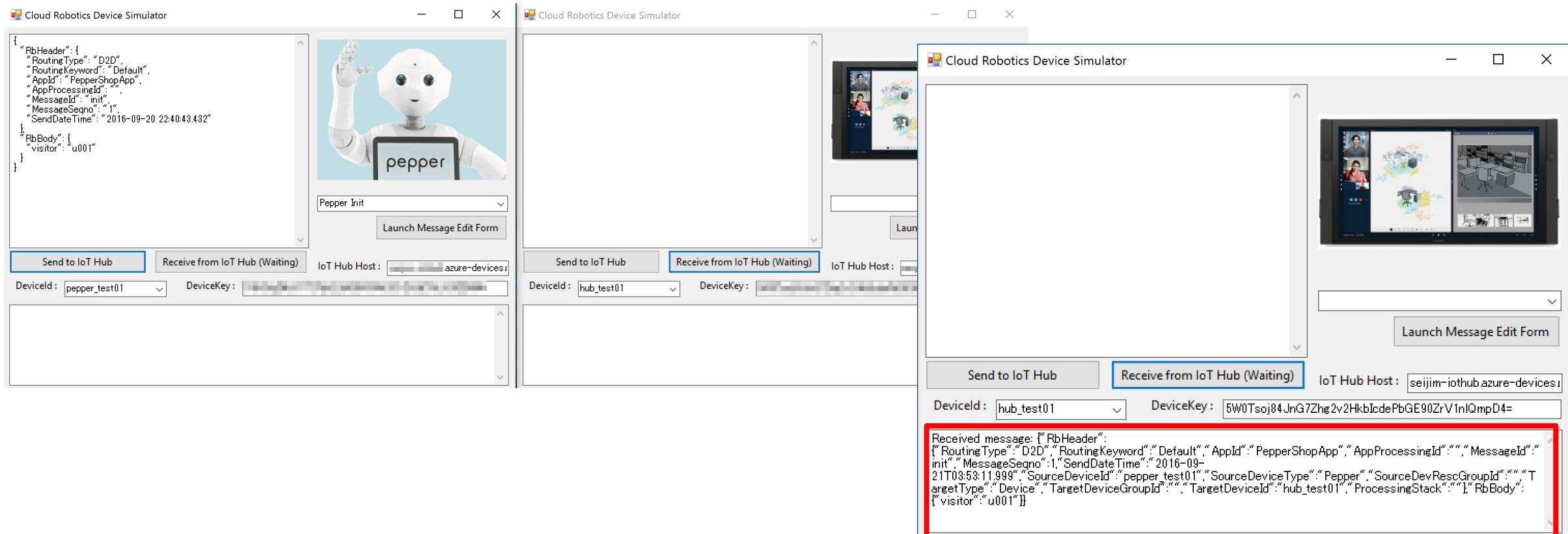
# T08\_01 : Cloud Robotics FX V2 のディプロイ設定

Coming soon

# T08\_0x：ディプロイ後のテスト

Cloud Robotics Device Simulator を使って、デバイス ルーティングのテストを実施します

- 「hub\_test01」(Surface Hub) のシミュレーターで、[Receive from IoT Hub (Waiting)] ボタンを押します
- 「pepper\_test01」(Pepper) のシミュレーターで、[Receive from IoT Hub (Waiting)] ボタンを押します
- 「pepper\_test01」(Pepper) のシミュレーターで、通信メッセージテンプレートから「Pepper Init」を選択します
- 「pepper\_test01」(Pepper) のシミュレーターで、[Send to IoT Hub] ボタンを押します
- 「hub\_test01」(Surface Hub) のシミュレーターにメッセージが届けば、成功です



# Tutorial 09

Cloud Robotics FX 上で、アプリを実行

# T09\_01 : Hello World アプリ (DLL) のビルド

RbSampleApp ソリューションファイルを開きます

- 「CRSDKv1¥SampleCode\_Server¥RbSampleApp¥RbSampleApp.sln」を開き、ビルドを実施します

```
using System;
using Newtonsoft.Json;
using Newtonsoft.Json.Linq;
using CloudRoboticsUtil;

namespace RbSampleApp
{
    public class SayHello : MarshalByRefObject, IAppRouterDll
    {
        public JArrayString ProcessMessage(RbAppMasterCache rbappmc, RbAppRouterCache rbapprc, RbHeader rbh, string rbBodyString)
        {
            JArray ja_messages = new JArray();
            AppBody appbody = new AppBody();
            appbody.Hello = "Hello World !!!!!!";

            RbMessage message = new RbMessage();
            message.RbHeader = rbh;
            message.RbBody = appbody;

            string json_message = JsonConvert.SerializeObject(message);
            JObject jo = (JObject)JsonConvert.DeserializeObject(json_message);
            ja_messages.Add(jo);
            JArrayString jaString = new JArrayString(ja_messages);

            return jaString;
        }
    }
}
```

[RBFX].[AppRouter] テーブルの情報を  
FX 側でキャッシュ (※1)

[RBFX].[AppMaster] テーブルの情報を  
FX 側でキャッシュ (※1)

デバイスへの出力メッセージを複数セット可能  
(=複数のデバイスに異なるメッセージを送信可能)

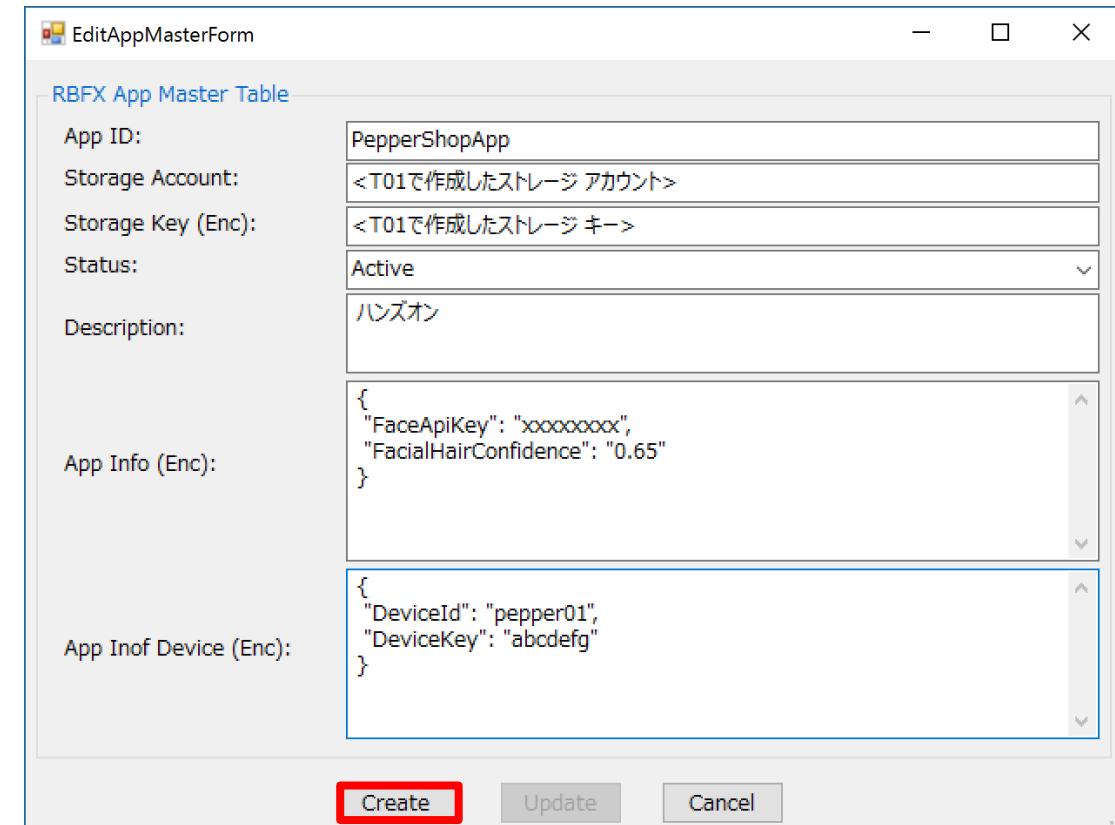
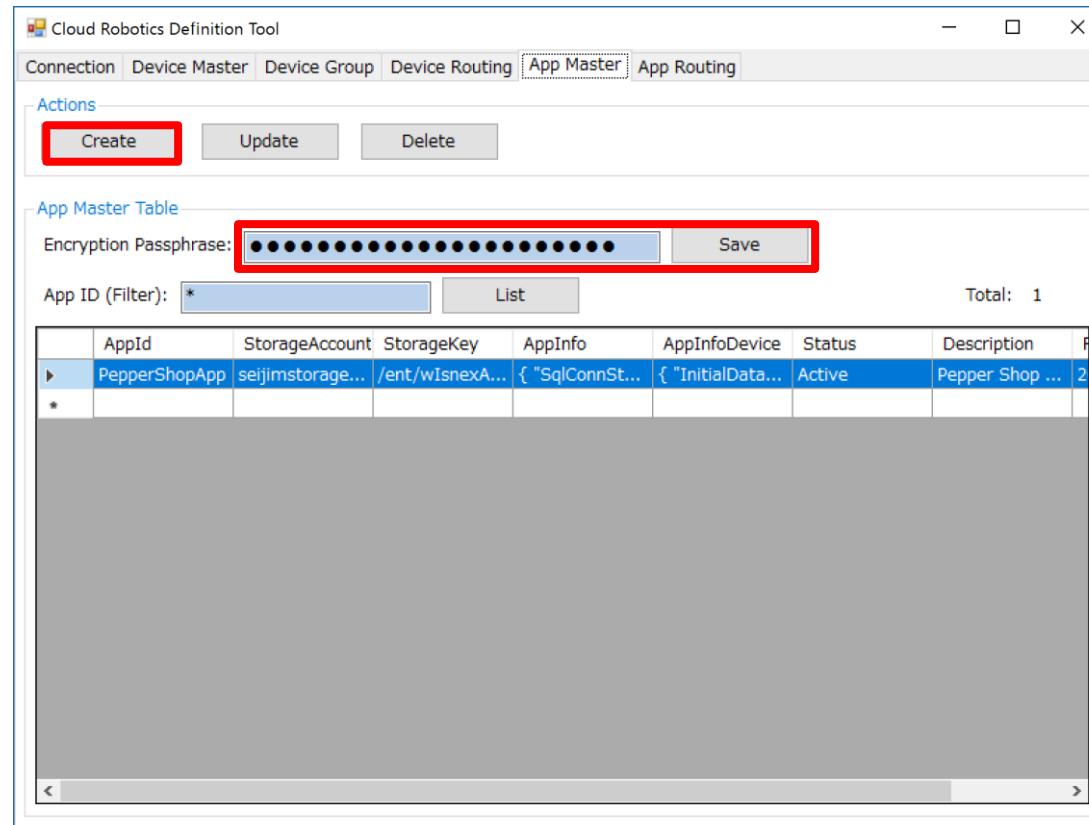
※1 : FX 側の cscfg ファイルで設定。キャッシュされる時間のデフォルト値は 60 秒

※2 : DLL 自体は、BLOB から一度読み込まれると、[RBFX].[AppRouter] の Registered\_DateTime が変更されない限りは、FX のカレントディレクトリにキャッシュされたまま

# T09\_02 : App Master の設定

Cloud Robotics Definition Tool (Management Tool) をショートカットから起動します

- [App Master] タブを開きます
- T05\_04 で設定した Cloud Robotics FX の暗号化用パスフレーズを [Encryption Passphrase] に設定し、[Save] します
- [Create] ボタンを押し、App Master 編集フォームで以下のように入力し、[Create] ボタンを押します



# T09\_03：アプリ (DLL) 配置用 BLOB コンテナーの作成

Azure ポータルを開きます

- [ストレージアカウント]→[該当のストレージアカウント]→[概要]→[BLOB] を選択します
- [+コンテナー] をクリックし、コンテナーの [名前]={ appdll }, [アクセスの種類]={ プライベート } を選択し、[作成] します

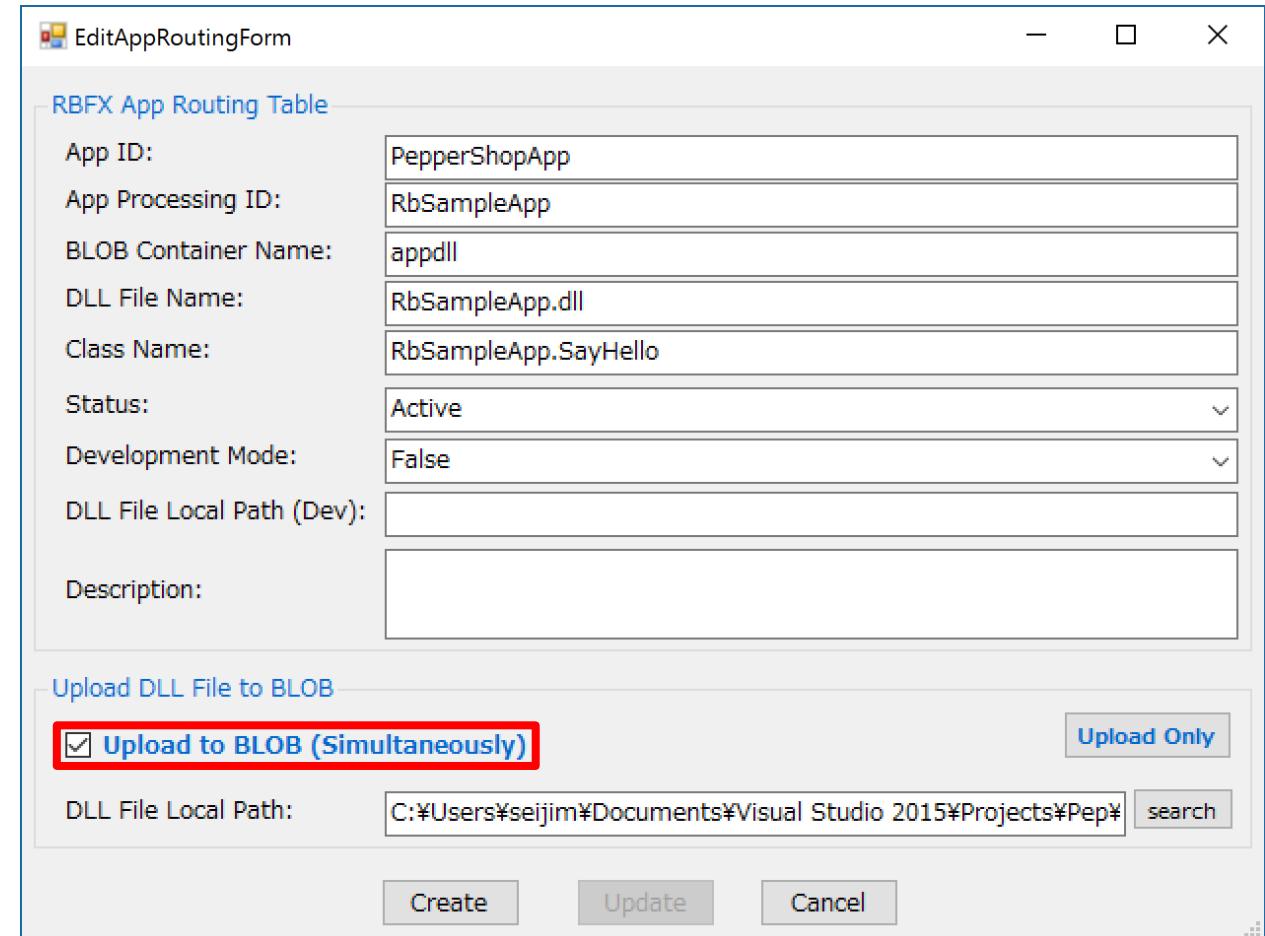
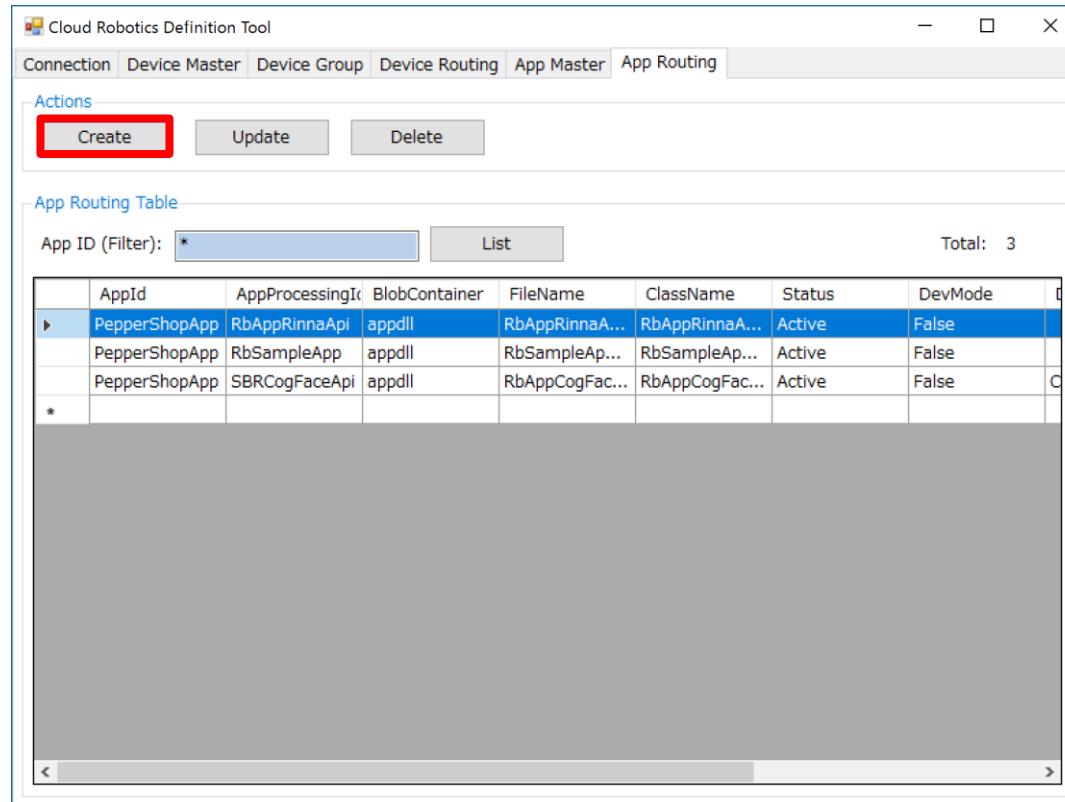
This screenshot shows the Microsoft Azure portal interface. On the left, the navigation menu is visible with 'Storage Account' selected under 'Storage Accounts'. In the main content area, the 'Overview' tab is selected for the storage account 'sejim...'. A red box highlights the 'Overview' tab in the top navigation bar. Another red box highlights the 'BLOB' service icon in the bottom navigation bar.

This screenshot shows the 'Create New Container' blade in the Azure portal. At the top, there is a 'New Container' button highlighted with a red box. The 'Name' field contains 'appdll' and the 'Access Type' dropdown is set to 'Private', also highlighted with red boxes. At the bottom right, there is a large 'Create' button.

# T09\_04 : App Routing の設定とアプリのディプロイ

Cloud Robotics Definition Tool の [App Routing] タブを開きます

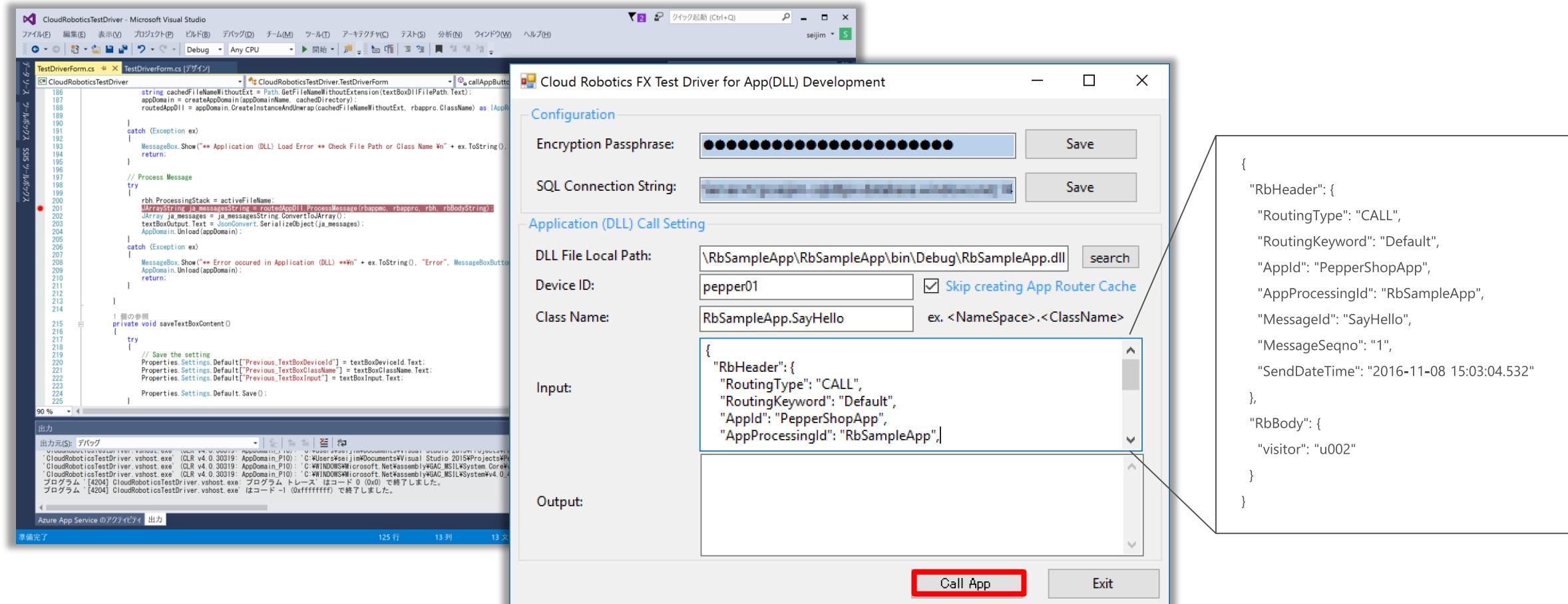
- [App Routing] タブで、[Create] ボタンを押します
- App Routing 編集フォームで以下のように入力します
- RbSampleApp.dll ファイルを [search] で選択し、[Upload to BLOB (Simultaneously)] をチェックし、[Create] ボタンを押します



# T09\_05：アプリ (DLL) の単体テスト

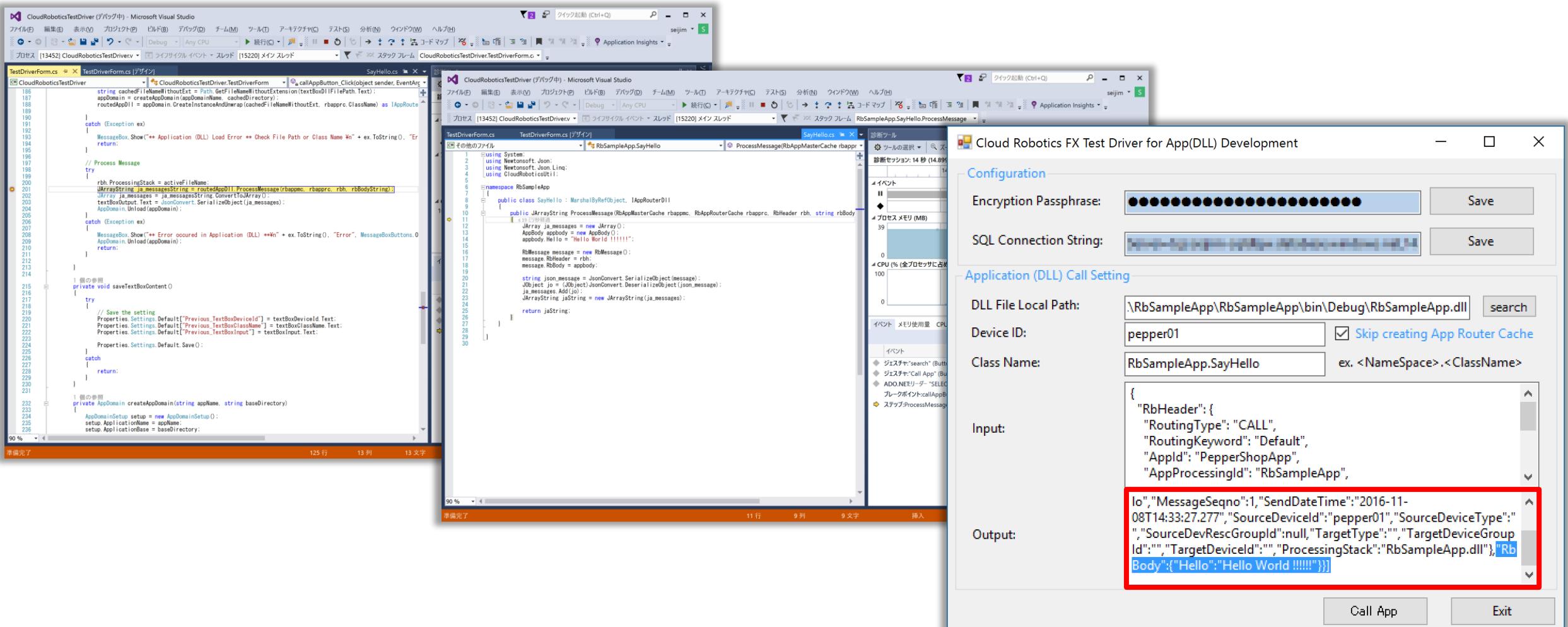
CloudRoboticsTestDriver ソリューションファイルを開きます

- 「CRSDKv1¥CloudRoboticsTestDriver¥CloudRoboticsTestDriver.sln」を開き、ビルドを実施します
- 「JArrayList ja\_messagesString = routedAppDll.ProcessMessage(...)」行にブレイクポイントを設定します
- 「▶開始」ボタンを押します
- T05\_04 で設定した Cloud Robotics FX の暗号化用パスフレーズを [Encryption Passphrase] に設定し、[Save] します
- T05\_04 で設定した Cloud Robotics FX の SQL 接続文字列を [SQL Connection String] に設定し、[Save] します
- [DLL File Local Path], [Device ID], [Class Name], [Input] を設定し、[Call App] ボタンを押します



# T09\_05：アプリ (DLL) の単体テスト

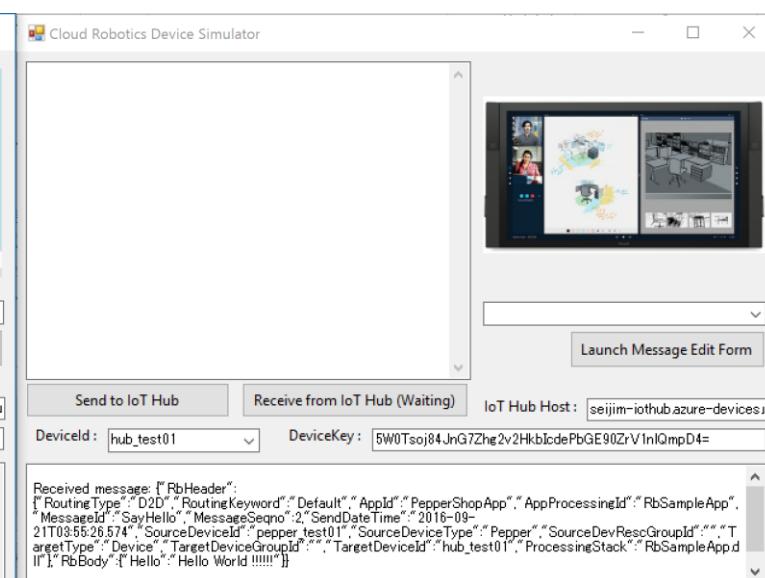
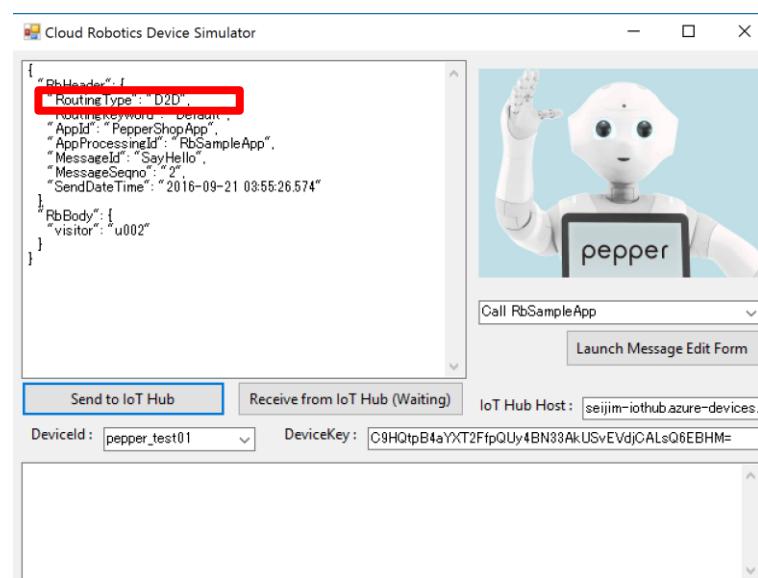
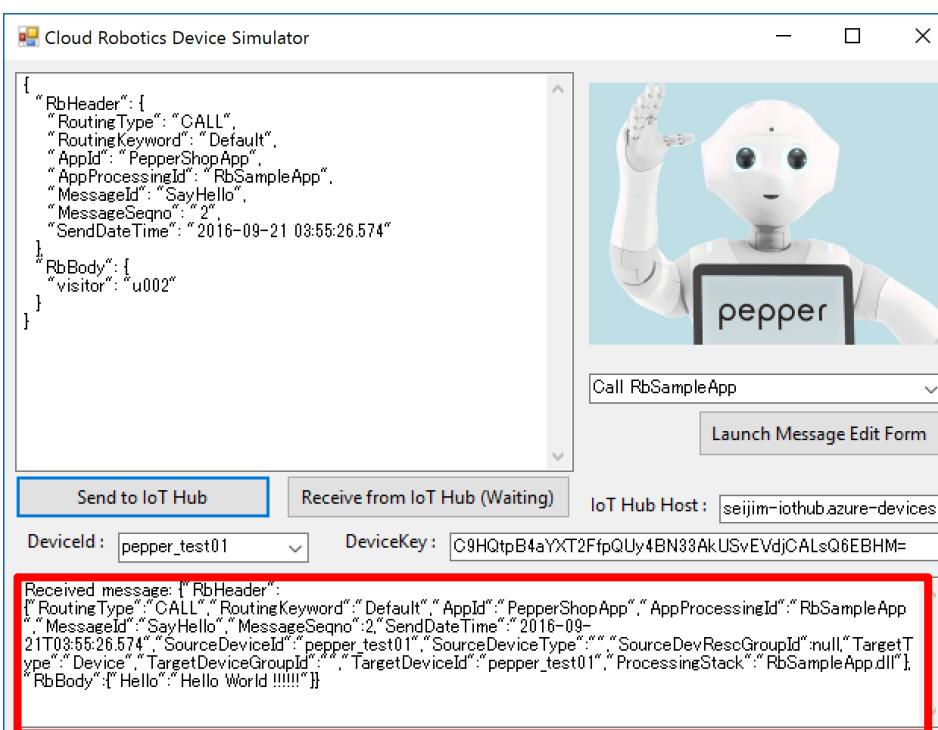
- ブレイクポイントまで実行されたら、[F11] を押し、Hello World アプリにステップインします
- Hello World アプリのソースコードをデバッグします
- 最後まで実行されると、[Output] 欄に結果が返ります
- 「Hello World !!!!!」が表示されれば、成功です



# T09\_06：ディプロイしたアプリ (DLL) のテスト

Cloud Robotics Definition Tool から Device Simulator を起動します

- [Device Master] タブで「pepper\_test01」を選択して、Device Simulator を起動します
- [Receive from IoT Hub (Waiting)] ボタンを押します
- 通信メッセージテンプレートから「Call RbSampleApp」を選択し、[Send to IoT Hub] ボタンを押します
- 「Hello World」メッセージが返ってくれば、成功です
- 通信メッセージの “RoutingType” の値を “CALL” から “D2D” に変更すると、デバイス間通信になります
- その場合、Hello World アプリは、デバイス間通信の間で、フィルターアプリケーションのような動作となります



# Tutorial 10

Stream Analytics の設定 (option)

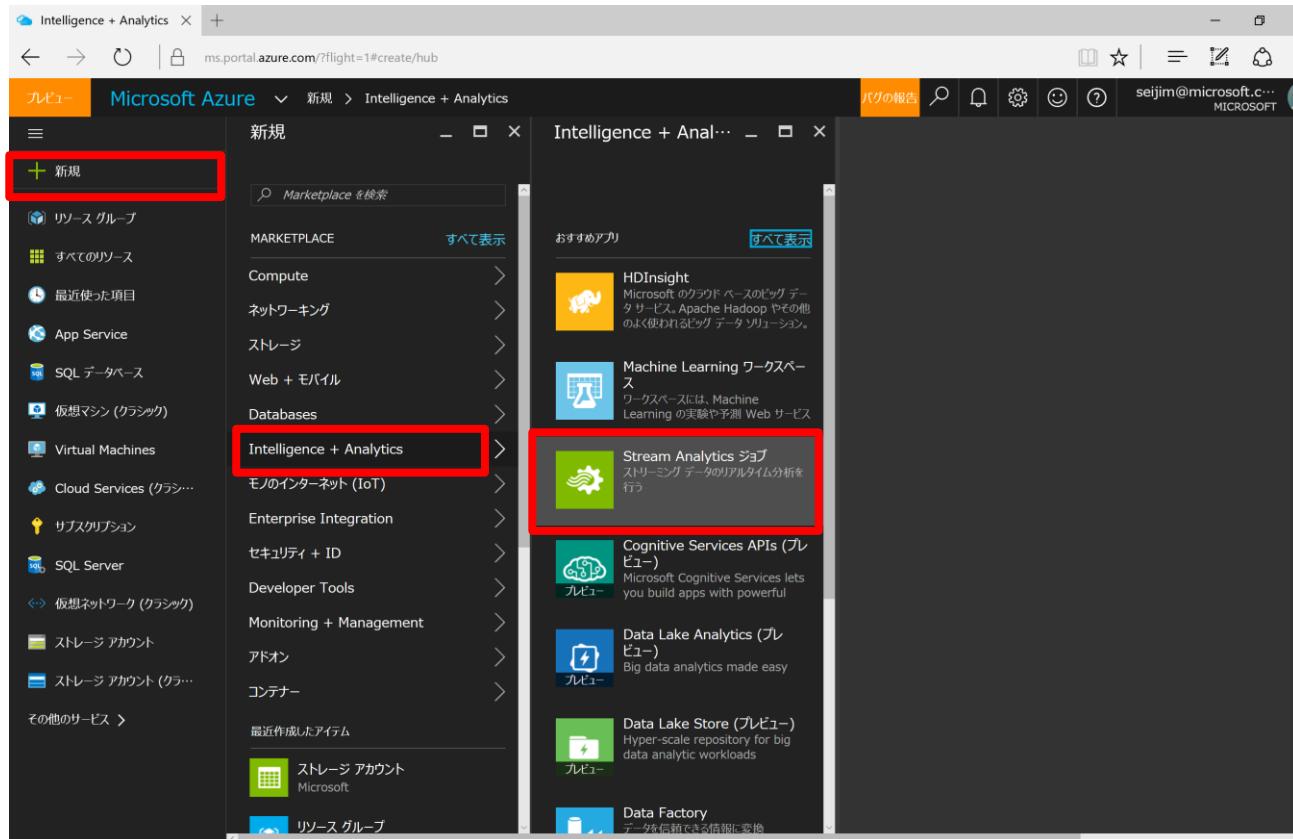
# T10\_01 : Stream Analytics ジョブの作成

Azure ポータルを開きます

- [+新規]→[Intelligence + Analytics]→[Stream Analytics ジョブ] を選択します

以下設定で、Stream Analytics ジョブを新規作成します

- [ジョブ名]={ ASACRoboticsFX }
- [リソースグループ]={ T01\_02 の設定値 }
- [場所]={ 東日本 or 西日本 }



# T10\_02 : Stream Analytics ジョブの入力設定

Azure ポータルから [Stream Analytics ジョブ] を開きます

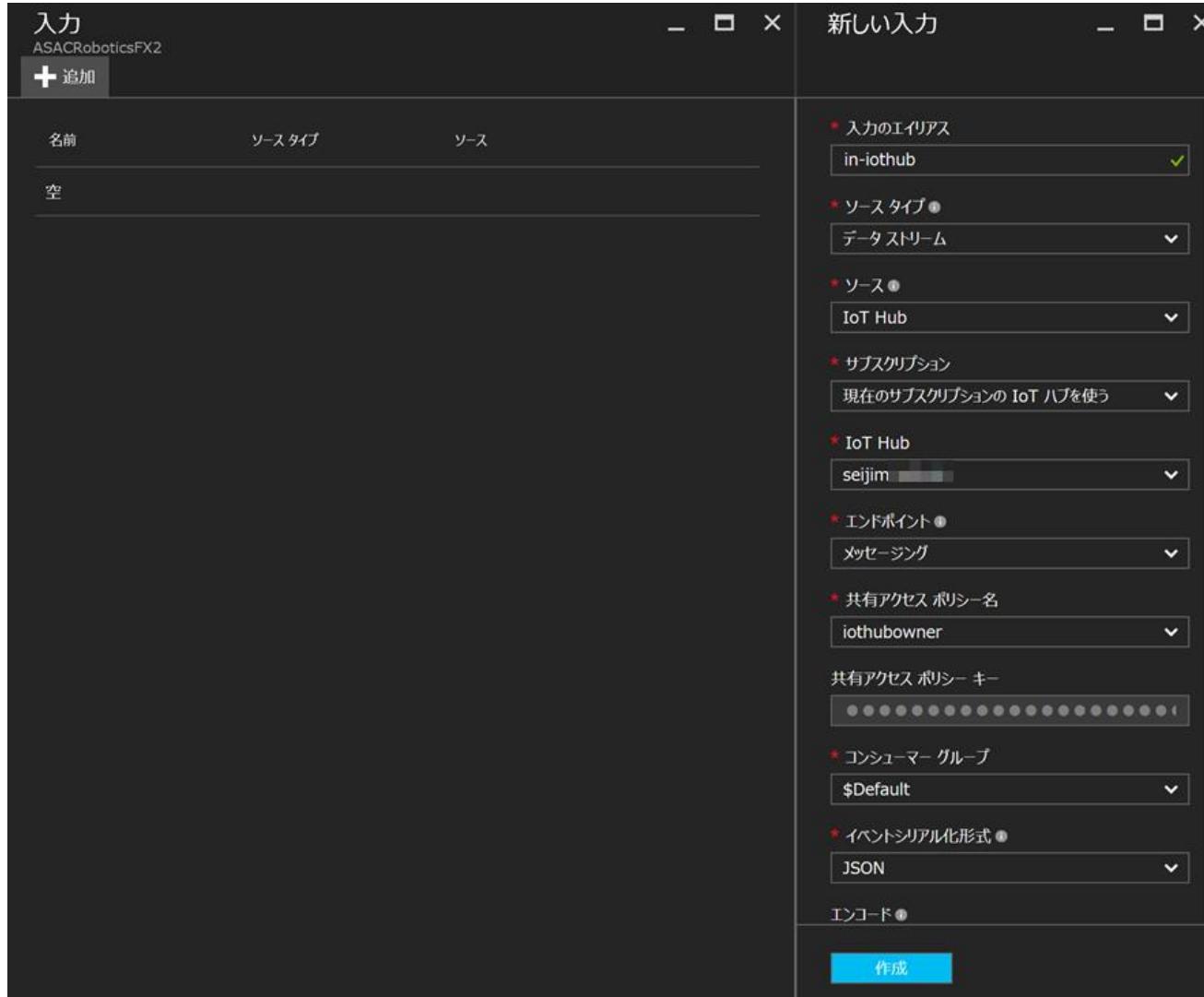
- 該当のジョブを選択して、[概要]→[入力] を選択します

The screenshot shows the Microsoft Azure portal interface for managing Stream Analytics jobs. On the left, the navigation menu lists various Azure services like Resource Groups, App Service, and SQL Database. In the center, the main content area displays the 'Stream Analytics ジョブ' (Stream Analytics Job) for 'ASACRoboticsFX2'. A red box highlights the '概要' (Overview) tab in the top navigation bar of the right-hand panel. Another red box highlights the '入力' (Input) section under the '構成' (Configuration) tab, which currently shows '結果がありません。' (No results). The top right corner shows the user's email 'seijim@microsoft.com'.

# T10\_03 : Stream Analytics ジョブの入力設定

[+追加]を押し、入力作成用のパラメータを設定し、[作成] ボタンを押します

- [入力のエイリアス]={ in-iothub }
- [ソースタイプ]={ データストリーム }
- [ソース]={ IoT Hub }
- 以降はデフォルトのまま



# T10\_04 : Stream Analytics ジョブの出力設定 1

Azure ポータルから [Stream Analytics ジョブ]→[概要]→[出力]→[+追加]を押します

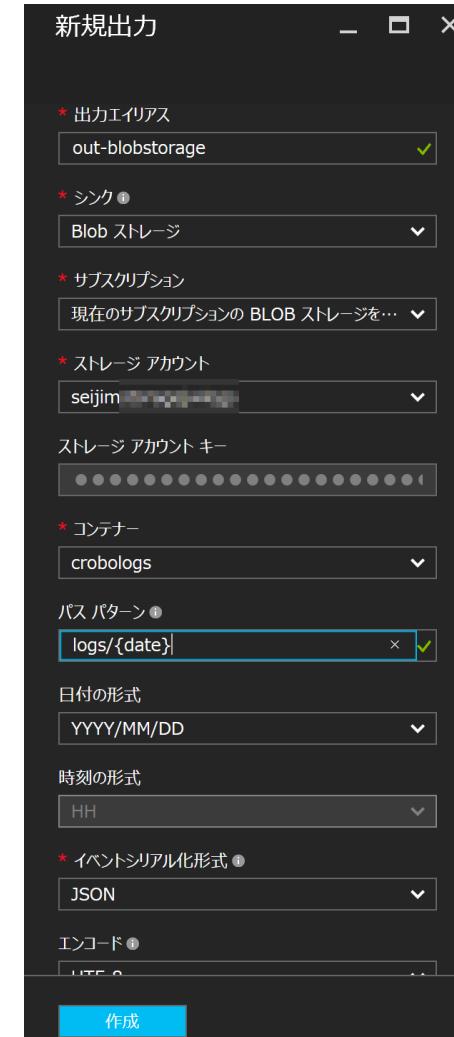
- 事前に T01 で作成したストレージアカウントに BLOB コンテナー「crobologs」を作成しておきます

BLOB への出力パラメータを設定し、[作成] ボタンを押します

- [出力エイリアス]={ out-blobstorage }
- [シンク]={ Blob ストレージ }
- [コンテナー]={ crobologs }
- [パス パターン]={ logs/{date} }

The screenshot shows the Azure Stream Analytics job overview page for 'ASACRoboticsFX2'. On the left, there's a sidebar with navigation links like '検索', '監査', 'アクセス制御 (IAM)', 'タグ', 'ロック', 'クォート', '入力', '関数', 'クエリ', '出力', 'スケール', 'ロゲール', 'イベント順序', and 'エラー ポリシー'. The main area displays the job details: 'リソース グループ: Resc-Seyen-Public', '状態: 作成済み', '場所: Japan West', 'リファレンス名: seiim\_azure', and 'リファレンス ID: 10'. Below this, the 'ジョブトポロジ' section shows '入力: 1' (in-iothub) and '出力: 0' (結果がありません). A modal window titled '新規出力' is open on the right, containing the configuration for the new output:

設定	値
* 出力エイリアス	out-blobstorage
* シンク	Blob ストレージ
* サブスクリプション	現在のサブスクリプションの BLOB ストレージを…
* ストレージ アカウント	seijim [選択]
ストレージ アカウント キー	(複数のドットマーク)
* コンテナー	crobologs
パス パターン	logs/{date}
日付の形式	YYYY/MM/DD
時刻の形式	HH
* イベントシリアル化形式	JSON
エンコード	UTF-8



# T10\_05 : Stream Analytics ジョブの出力設定 2

Azure ポータルから [Stream Analytics ジョブ]→[概要]→[出力]→[+追加]を押します  
SQLDB への出力パラメータを設定し、[作成] ボタンを押します

- [出力エイリアス]={ out-sqldb }
- [シンク]={ SQL データベース }
- [データベース], [ユーザー名], [パスワード]={ T03 で作成した SQLDB }
- [テーブル]={ RBApp.IoTHubLog }

The screenshot shows the Azure Stream Analytics job overview page for 'ASACRoboticsFX2'. On the right, under the 'Output' section, there is a table with one input row ('in-iothub') and one output row ('out-blobstorage'). The 'out-blobstorage' row is highlighted with a blue border. The 'Output' column contains the value 'out-blobstorage'.



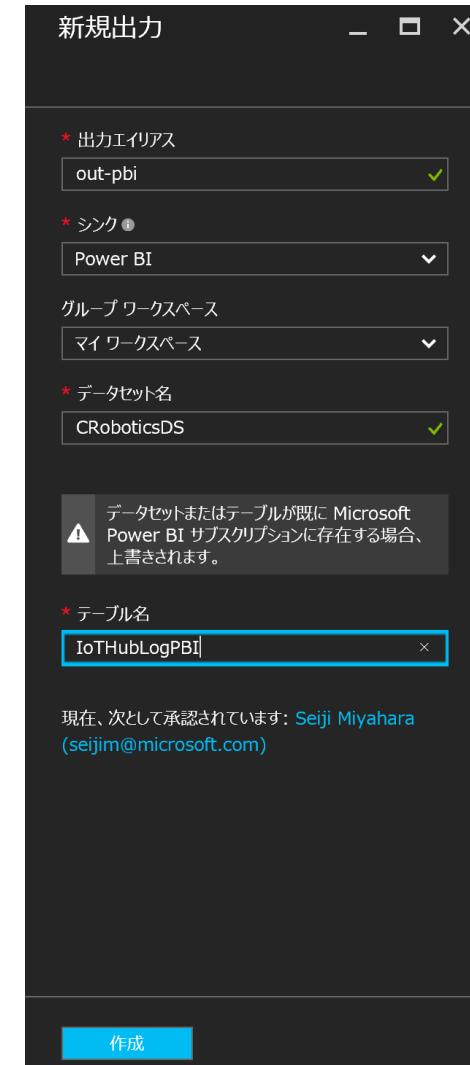
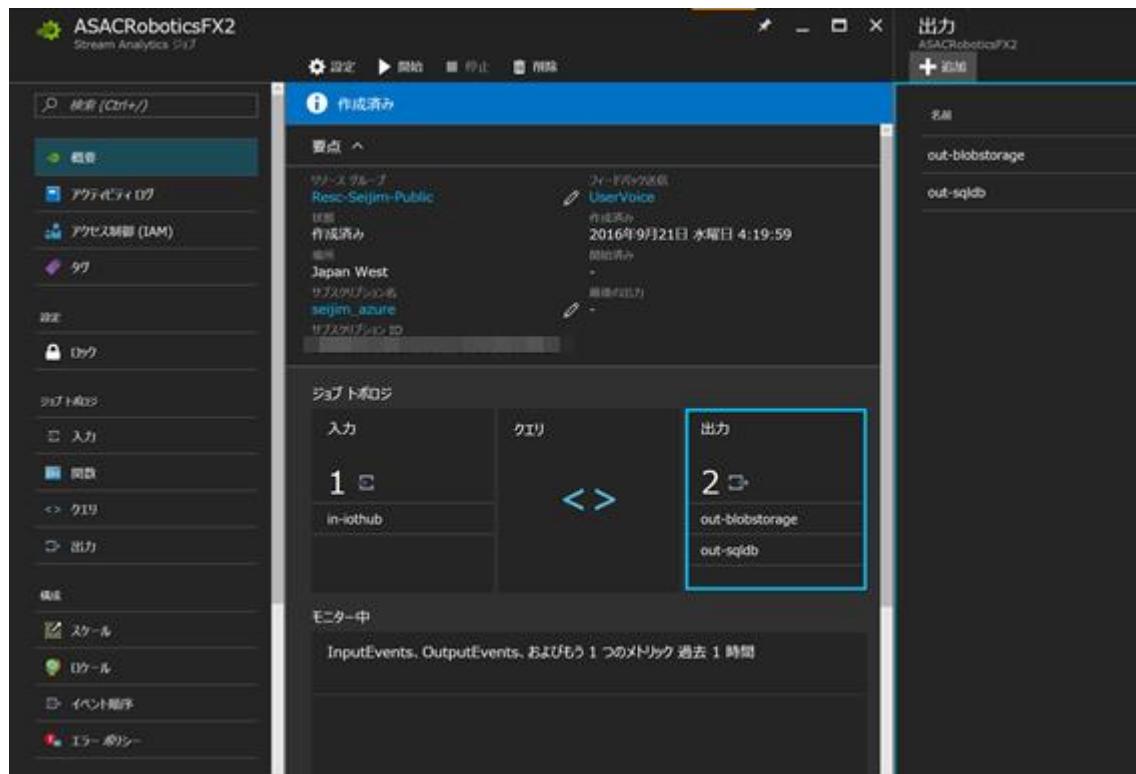
# T10\_06 : Stream Analytics ジョブの出力設定 3

Azure ポータルから [Stream Analytics ジョブ]→[概要]→[出力]→[+追加]を押します  
Power BI への出力パラメータを設定します

- [出力エイリアス]={ out-pbi }
- [シンク]={ Power BI }

Power BI への認証を実施後、以下を設定し、 [作成] ボタンを押します

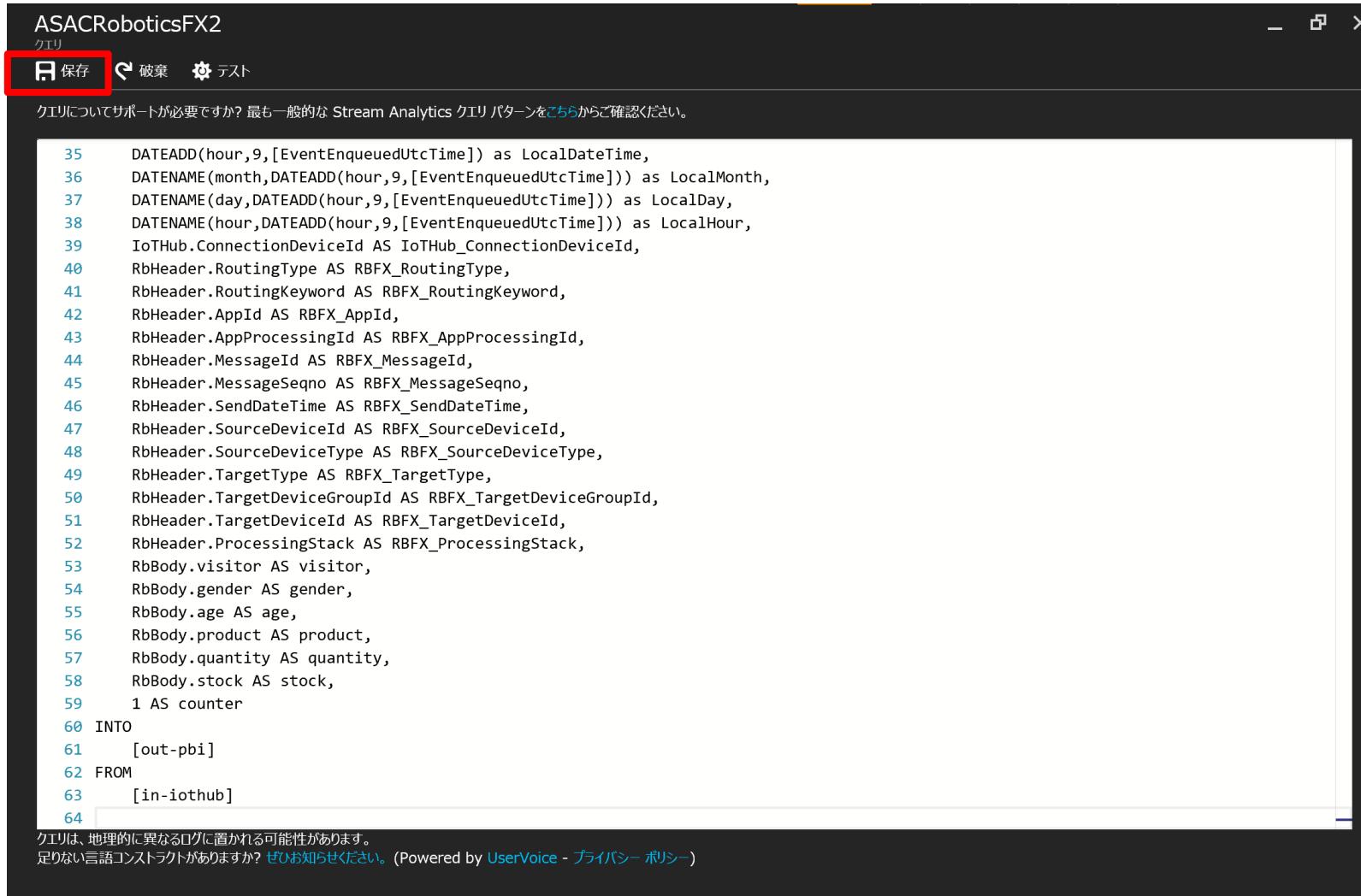
- [データセット名]={ CRoboticsDS }
- [テーブル名]={ IoTHubLogPBI }



# T10\_07 : Stream Analytics ジョブのクエリ設定

Azure ポータルから [Stream Analytics ジョブ]→[概要]→[クエリ] を押します

- 「CRSDKv1¥SQL\_ASA\_Definition¥21\_ASA\_Query.txt」をメモ帳などで開き、コピペし、[保存] を押します



The screenshot shows the Azure Stream Analytics Query editor interface. The title bar says "ASACRoboticsFX2" and "クエリ". Below the title bar are three buttons: "保存" (Save) with a red box around it, "破棄" (Delete), and "テスト" (Test). A message below the buttons says "クエリについてサポートが必要ですか? 最も一般的な Stream Analytics クエリ パターンを[こちら](#)からご確認ください。". The main area contains a large block of T-SQL code with line numbers from 35 to 64. The code defines a query that processes data from an IoT Hub source and inserts it into an output PBI destination. At the bottom of the code area, there is a note: "クエリは、地理的に異なるログに置かれる可能性があります。足りない言語構造がありますか? ぜひお知らせください。 (Powered by UserVoice - プライバシー ポリシー)".

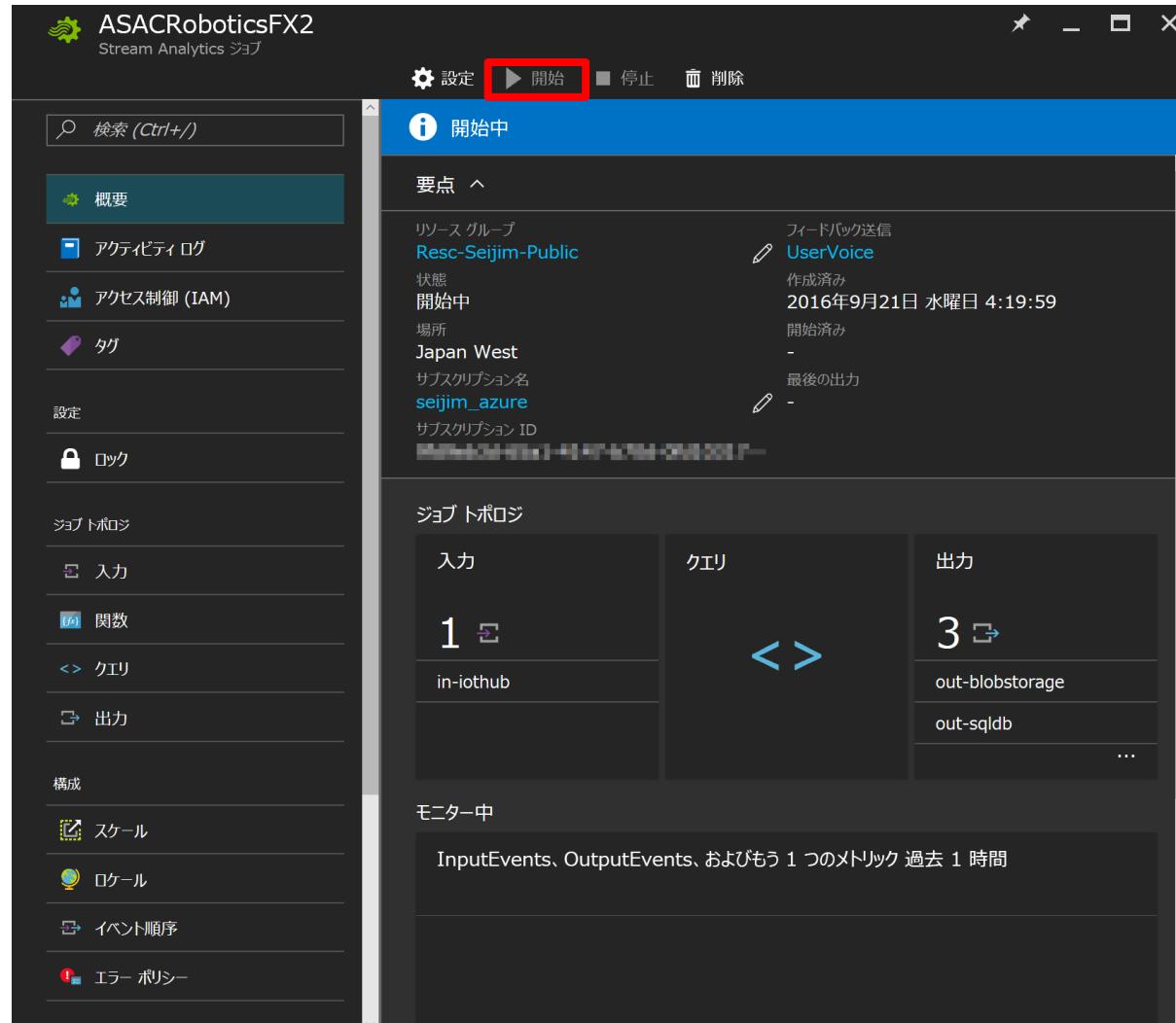
```
35     DATEADD(hour,9,[EventEnqueuedUtcTime]) as LocalDateTime,
36     DATENAME(month,DATEADD(hour,9,[EventEnqueuedUtcTime])) as LocalMonth,
37     DATENAME(day,DATEADD(hour,9,[EventEnqueuedUtcTime])) as LocalDay,
38     DATENAME(hour,DATEADD(hour,9,[EventEnqueuedUtcTime])) as LocalHour,
39     IoTHub.ConnectionDeviceId AS IoTHub_ConnectionDeviceId,
40     RbHeader.RoutingType AS RBFX_RoutingType,
41     RbHeader.RoutingKeyword AS RBFX_RoutingKeyword,
42     RbHeader.AppId AS RBFX_AppId,
43     RbHeader.AppProcessingId AS RBFX_AppProcessingId,
44     RbHeader.MessageId AS RBFX_MessageId,
45     RbHeader.MessageSeqno AS RBFX_MessageSeqno,
46     RbHeader.SendDateTime AS RBFX_SendDateTime,
47     RbHeader.SourceDeviceId AS RBFX_SourceDeviceId,
48     RbHeader.SourceDeviceType AS RBFX_SourceDeviceType,
49     RbHeader.TargetType AS RBFX_TargetType,
50     RbHeader.TargetDeviceGroupId AS RBFX_TargetDeviceGroupId,
51     RbHeader.TargetDeviceId AS RBFX_TargetDeviceId,
52     RbHeader.ProcessingStack AS RBFX_ProcessingStack,
53     RbBody.visitor AS visitor,
54     RbBody.gender AS gender,
55     RbBody.age AS age,
56     RbBody.product AS product,
57     RbBody.quantity AS quantity,
58     RbBody.stock AS stock,
59     1 AS counter
60 INTO
61     [out-pbi]
62 FROM
63     [in-iothub]
64
```

クエリは、地理的に異なるログに置かれる可能性があります。  
足りない言語構造がありますか? ぜひお知らせください。 (Powered by UserVoice - プライバシー ポリシー)

# T10\_08 : Stream Analytics ジョブの開始

Azure ポータルから [Stream Analytics ジョブ]→[概要]→[開始] を押します

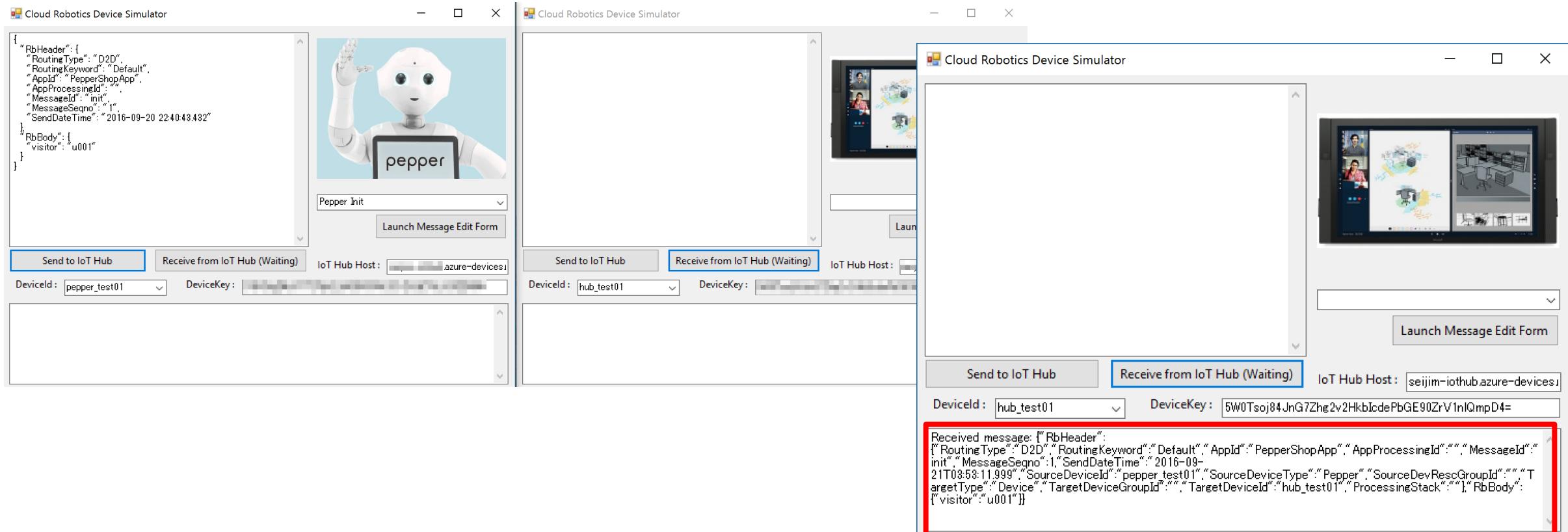
- IoT Hub の Read Point を聞かれますので、最初の起動時は「開始時刻」、2回目以降は「最終停止時刻」を選択します



# T10\_09 : Stream Analytics のテスト

Cloud Robotics Device Simulator を使って、デバイス間通信を実施します

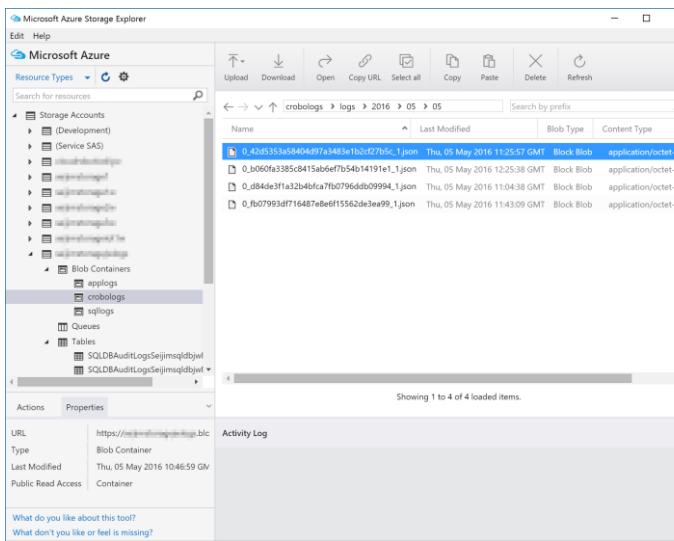
- 「hub\_test01」(Surface Hub) のシミュレーターで、[Receive from IoT Hub (Waiting)] ボタンを押します
- 「pepper\_test01」(Pepper) のシミュレーターで、[Receive from IoT Hub (Waiting)] ボタンを押します
- 双方のシミュレーターから色々なメッセージを [Send to IoT Hub] ボタンで送信します



# T10\_10 : Stream Analytics の出力結果の確認

BLOB 出力、SQLDB 出力、Power BI 出力について、結果を確認します

- BLOB : Microsoft Azure Storage Explorer で、出力対象のストレージ アカウント、BLOB コンテナーの内容を確認します
- SQLDB : SQL Server Management Studio で、出力対象のテーブルの内容を確認します
- Power BI : powerbi.com にサインインして、データセットが作成されているかを確認します



This screenshot shows the Microsoft SQL Server Management Studio interface. It displays a file browser window showing database objects like 'dbo.RoboLogs' and 'dbo.RoboLog'. Below it is a 'SQL Query' window with the following T-SQL script and results:

```
SELECT TOP 100 [id]
,[EventProcessedUtcTime]
,[PartitionId]
,[IsHub_ConnectionDeviceId]
,[RBFX_RoutingType]
,[RBFX_RoutingKeyword]
,[RBFX_Approved]
,[RBFX_Approving]
,[RBFX_ApprovedOn]
,[RBFX_ApprovedBy]
,[RBFX_MessageStatus]
,[RBFX_SourceDeviceId]
,[RBFX_TargetDeviceId]
,[RBFX_ProcessingStack]
,[value]
,[age]
,[emotion]
,[gender]
,[product]
,[quant]
,[v]
```

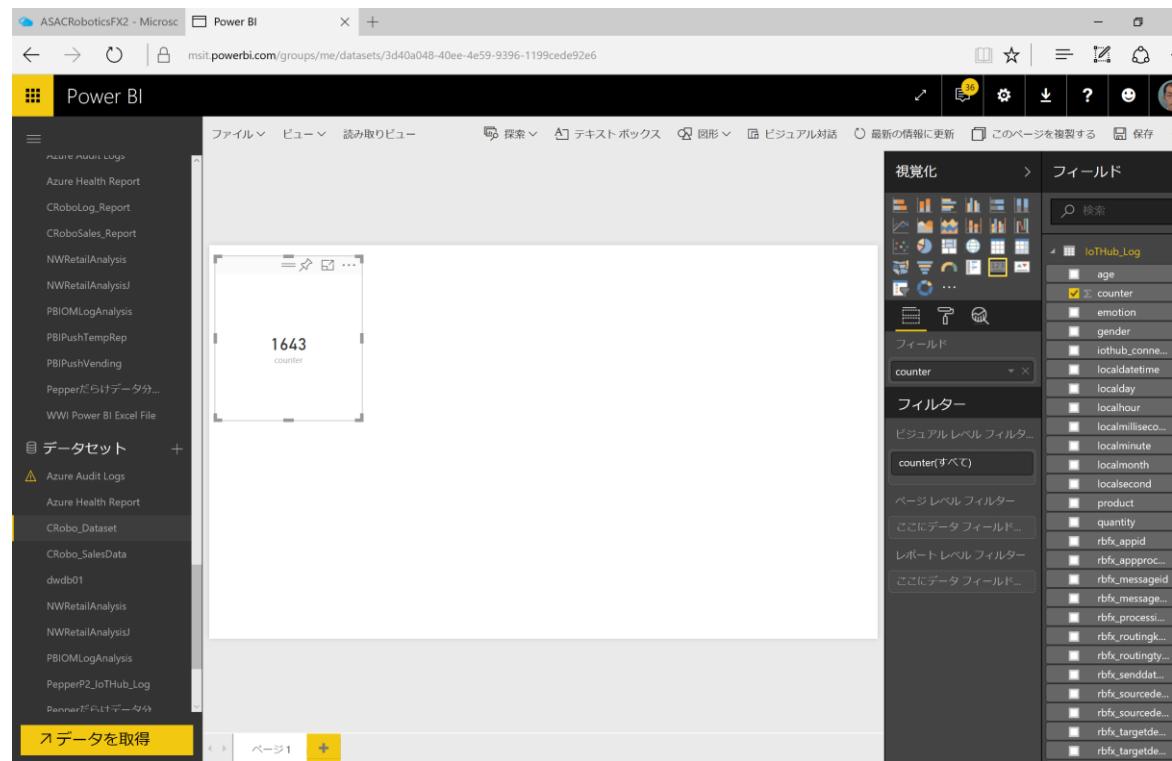
	EventProcessedUtcTime	PartitionId	IsHub_ConnectionDeviceId	RBFX_RoutingType	RBFX_RoutingKeyword	RBFX_Approved	RBFX_Approving	RBFX_ApprovedOn	RBFX_ApprovedBy	RBFX_MessageStatus	RBFX_SourceDeviceId	RBFX_TargetDeviceId	RBFX_ProcessingStack	v	age	emotion	gender	product	quant	v
1	2016-05-05 11:04:22.070000	2016-05-05 11:04:31.057142	7	pepper@1	DOD	NULL	PepperShopApp													
2	2016-05-05 11:04:34.094000	2016-05-05 11:04:33.205183	6	hub@1	DOD	NULL	PepperShopApp													
3	2016-05-05 11:04:34.930000	2016-05-05 11:04:41.049000	5	pepper@1	DOD	NULL	PepperShopApp													
4	2016-05-05 11:04:35.000000	2016-05-05 11:04:41.049000	6	hub@1	DOD	NULL	PepperShopApp													
5	2016-05-05 11:04:35.040000	2016-05-05 11:22:53.095913	6	hub@1	DOD	NULL	PepperShopApp													
6	2016-05-05 11:22:53.095913	2016-05-05 11:22:53.095913	6	hub@1	DOD	NULL	PepperShopApp													
7	2016-05-05 11:22:53.095913	2016-05-05 11:22:53.095913	6	hub@1	DOD	NULL	PepperShopApp													
8	2016-05-05 11:22:53.095913	2016-05-05 11:22:53.095913	6	hub@1	DOD	NULL	PepperShopApp													
9	2016-05-05 11:22:53.095913	2016-05-05 11:22:53.095913	7	pepper@1	DOD	NULL	PepperShopApp													
10	2016-05-05 11:22:54.760000	2016-05-05 11:24:45.770583	6	hub@1	DOD	NULL	PepperShopApp													
11	2016-05-05 11:22:55.409000	2016-05-05 11:24:45.770499	6	hub@1	DOD	NULL	PepperShopApp													
12	2016-05-05 11:22:55.760000	2016-05-05 11:25:54.804814	6	hub@1	DOD	NULL	PepperShopApp													
13	2016-05-05 11:22:55.804814	2016-05-05 11:25:54.804814	6	hub@1	DOD	NULL	PepperShopApp													
14	2016-05-05 11:22:55.804814	2016-05-05 11:41:37.127185	6	hub@1	DOD	NULL	PepperShopApp													

This screenshot shows a Power BI report titled 'ASACRoboticsFX2 - Microsoft'. It displays a visual with the value '1643' and the word 'counter'. The report includes a navigation bar with links like 'Azure Health Report', 'ClobLog\_Report', 'ClobLogs\_Report', 'NWRetailAnalysis', 'PBIMLogAnalysis', 'PBIPushTempRep', 'PBIPushVending', 'Pepperだらけデータ...', 'WW Power BI Excel File', and a 'Data Set' section for 'Clobo\_Dataset' and 'Clobo\_SalesData'. The right side of the screen shows the 'Visualizations' and 'Fields' panes, with various fields listed under 'IoTHub\_Log' such as 'age', 'counter', 'emotion', 'gender', 'iothub\_connec...', 'localsettime', 'localday', 'localhour', 'localmilliseco...', 'localminute', 'localmonth', 'localsecond', 'product', 'quantity', and several 'rbf...' fields.

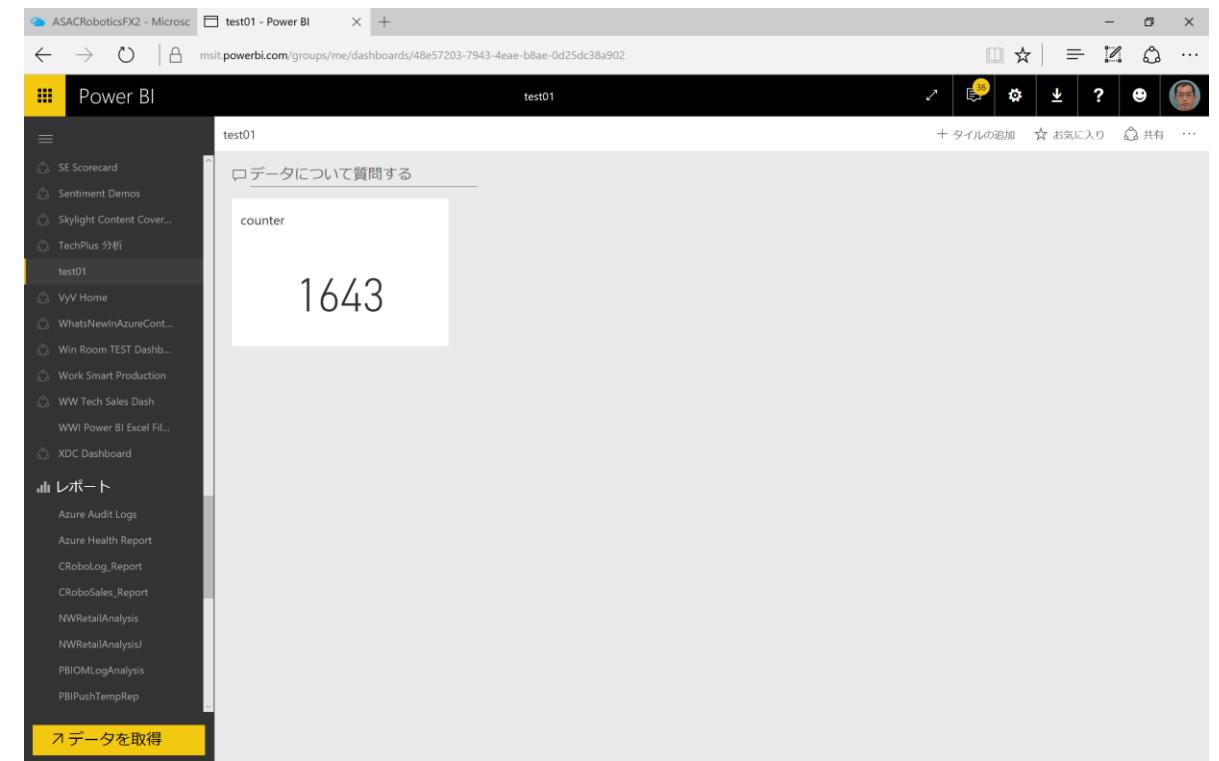
# T10\_11 : Power BI リアルタイム ダッシュボード

該当のデータセットを選択すると、レポート作成画面が表示されます

- フィールドから「counter」を選択し、視覚表現を「カード」にします
- このレポートに名前を付けて保存し、カードの上部にある「ピン」ボタンを押します
- [新しいダッシュボード] を選択して、名前を付けます
- シミュレーターで送受信を行うと、ダッシュボードの counter 値がリアルタイムに変化するのが確認できます



The screenshot shows the Power BI report editor interface. On the left, the 'Data Set' pane lists various datasets like 'Azure Audit Logs', 'Azure Health Report', and 'CRoboDataset'. In the center, a card visual displays the value '1643' with the label 'counter'. The right side shows the 'Fields' pane with a selected field 'counter' under the 'IoTHub.Log' category. A yellow bar at the bottom indicates 'Data is loaded'.



The screenshot shows a Power BI dashboard titled 'test01'. It features a card visual with the text 'データについて質問する' and 'counter' below it, displaying the value '1643'. The dashboard sidebar lists various reports and datasets, including 'SE Scorecard', 'Sentiment Demos', and 'TechPlus Analysis'. A yellow bar at the bottom indicates 'Data is loaded'.



- 本書に記載した情報は、本書各項目に関する発行日現在の Microsoft の見解を表明するものです。Microsoftは絶えず変化する市場に対応しなければならないため、ここに記載した情報に対していかなる責務を負うものではなく、提示された情報の信憑性については保証できません。
- 本書は情報提供のみを目的としています。Microsoft は、明示的または暗示的を問わず、本書にいかなる保証も与えるものではありません。
- すべての当該著作権法を遵守することはお客様の責務です。Microsoftの書面による明確な許可なく、本書の如何なる部分についても、転載や検索システムへの格納または挿入を行うことは、どのような形式または手段（電子的、機械的、複写、レコーディング、その他）、および目的であっても禁じられています。  
これらは著作権保護された権利を制限するものではありません。
- Microsoftは、本書の内容を保護する特許、特許出願書、商標、著作権、またはその他の知的財産権を保有する場合があります。Microsoftから書面によるライセンス契約が明確に供給される場合を除いて、本書の提供はこれらの特許、商標、著作権、またはその他の知的財産へのライセンスを与えるものではありません。

© 2016 Microsoft Corporation. All rights reserved.

Microsoft, Windows, その他本文中に登場した各製品名は、Microsoft Corporation の米国およびその他の国における登録商標または商標です。

その他、記載されている会社名および製品名は、一般に各社の商標です。