Aerospace & Defence | Lighting | Power Management | Internet of Things | Transportation

Components – EMEA

ESC – Software Solutions

Send sensor data of ST SensorTile.box to the Azure IoT Central via Shiratech IoT Box

Revision 1.0, 20.03.2020

Quick Start Guide

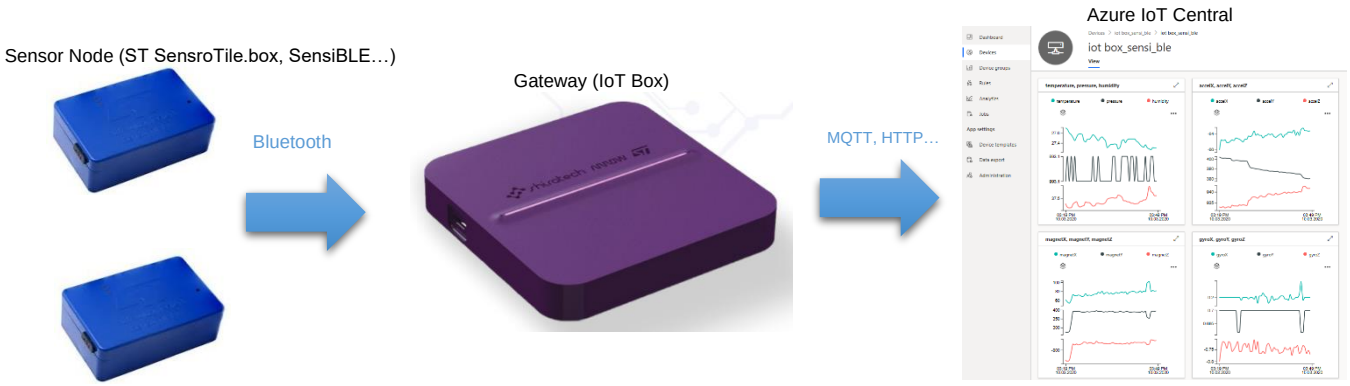
esc@arroweurope.com

Table of Contents

List of Figures

Introduction

This document provides step-by-step guidance to help you connect your ST SensorTile.box to IoT Box and send sensor data to Azure IoT Central. The demo principle is shown in the figure below:



Hardware Requirements

- ST SensorTile.box (STEVAL-MKSBOX1V1)
- Shiratech IoT Box

Software Requirements

- “dps-keygen” Software, download link: https://github.com/ArrowElectronics/IoTBOX/blob/master/Azure%20IoT%20Central%20Guide/SW/dps-keygen.zip

- ST BlueSTSDK_Python, download link: https://github.com/STMicroelectronics/BlueSTSDK_Python

- Azure IoT SDK for Python, download link: https://github.com/Azure/azure-iot-sdk-python

- Example code link: https://github.com/ArrowElectronics/IoTBOX/tree/master/Azure%20IoT%20Central%20Guide/SW/Sample%20Code

Step 1: Sign Up to Azure IoT Central

Follow the instructions here to sign up to the Azure IoT Central service.
To use Azure, you will need to register Microsoft Azure Account at first and then sign up to Azure.
Go to the link: https://azure.microsoft.com/de-de/services/iot-central/ , click the button “First Steps”.
After signing up, you can access your applications website.
Create a new application, click the “Build” button, choose “Custom apps”. In the “Create an application” website configure your application as following:
1. Choose a payment plan: select “Trial”. Your trial is expiring in 7 days, you can convert to Pay-As-You-Go after your application is created.
3. Configure your application: choose the application name and URL as you wish.
4. Contact information: your name, Email, Phone number, Country/Region

Then click the “create” button.

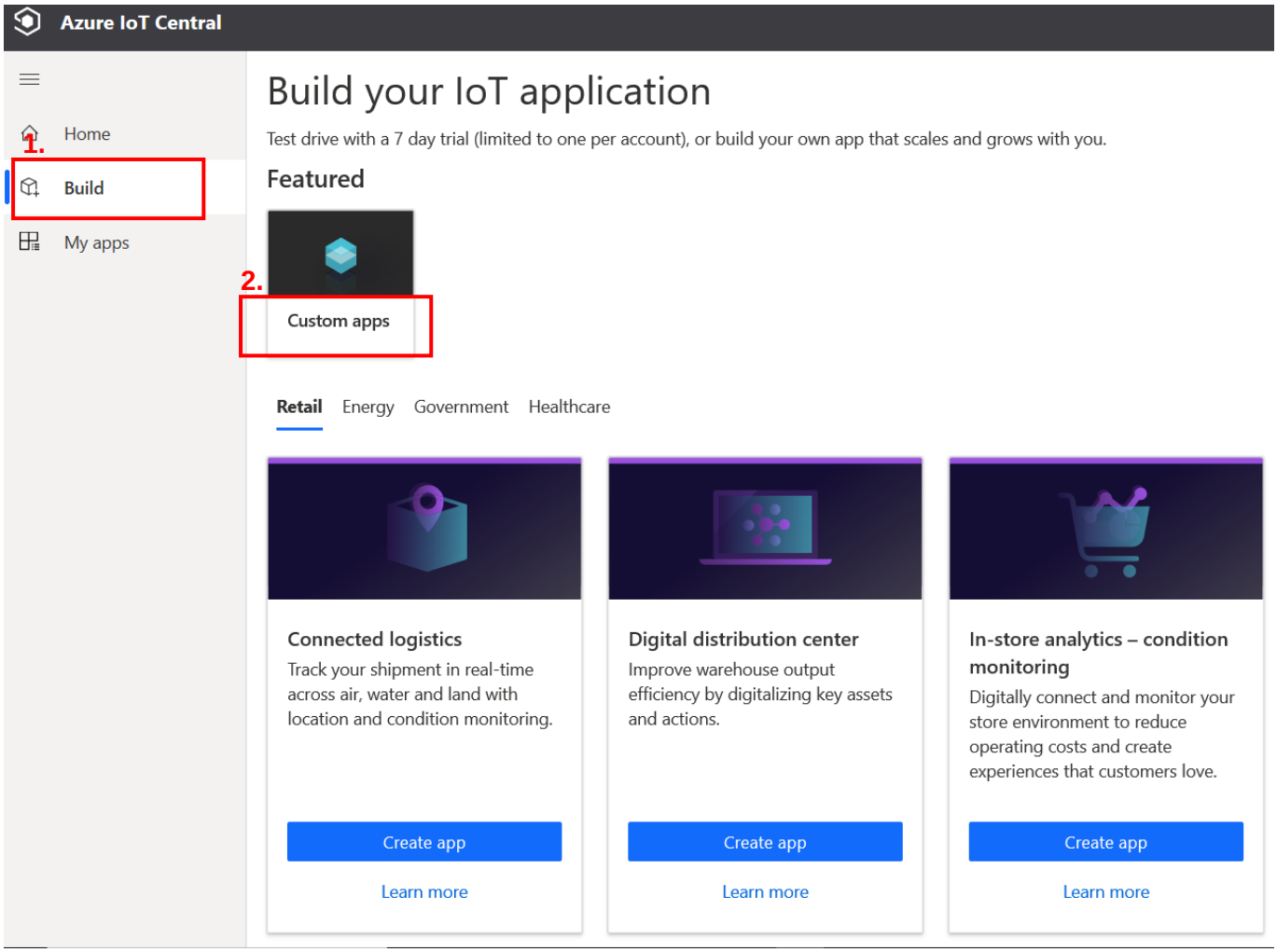


Figure 1: My applications

Now this is your Azure IoT Central application dashboard, then you need to create a new device in your application depends on the device template SensorTile.box to enable the connection with the real device.
After your new device is created, you can see the device with the default settings in your application.

Click the “Device Explorer” button, you can find the newly created device in the Explorer and the provisioning status of the device. Since the device template in IoT Central is not connected to the real device, the provisioning status of the device is now “Registered”.

Click “your device name” in the Explorer to go to the device page. In the device page, click the “Connect” button in the top of the device page.

In the “Device Connection” pop-up windows. Copy the “Scope ID”, “Device ID” and “Primary Key” and save them in notepad for later use.

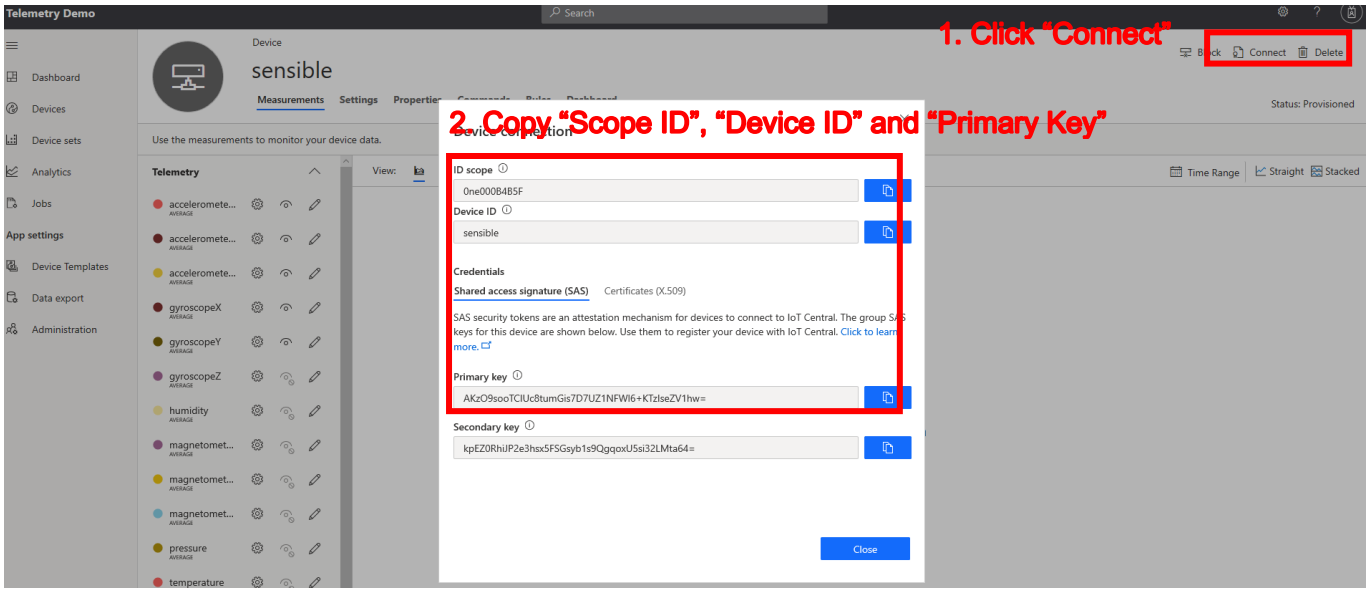


Figure 2: Device Connection

Step 2: Register Device

In this section, you will register your device using dps-keygen on Windows.

Click here to download dps-keygen package, then unzip the dps-keygen package.

In the windows command line go to the location where "dps_cstr.exe" located, default in the folder "\dps-keygen\bin\windows\dps_cstr".

Run the command "dps_cstr" with "Scope ID", "Device ID" and "Primary Key" (see Figure 2: Device Connection) that you saved before. Wait for the exe to complete, then copy and save the generated Connection String. When this step is completed, the provisioning status of your device is now "Provisioned".

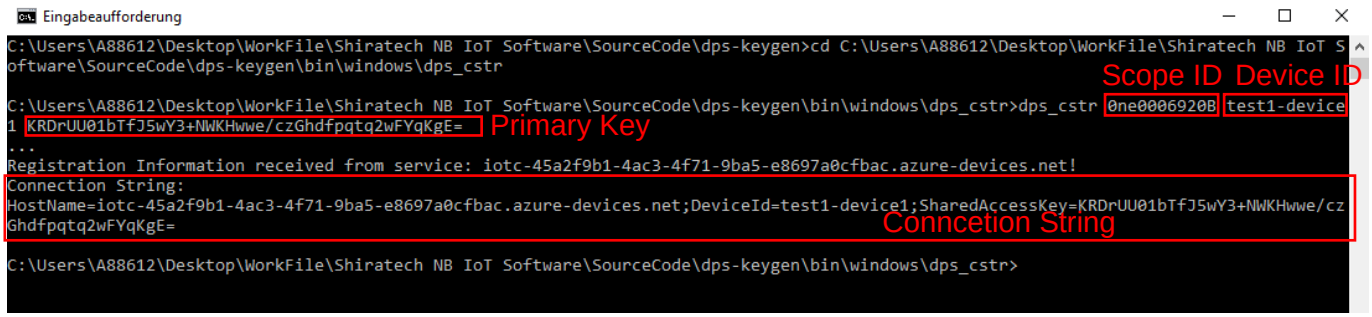


Figure 3: dps_cstr

Step 3: Prepare your IoT Box for Connection

1. Initialize your IoT Box first, the initial steps are shown in the manual, the manual download link: https://www.shiratech-solutions.com/software/IoT-Box_Quick_Start_Guide.pdf
2. Install ST BlueST SDK on your IoT Box

Installation steps: https://github.com/STMicroelectronics/BlueSTSDK_Python

3. Install Azure IoT SDK for Python on your IoTBox

Installation steps: https://github.com/Azure/azure-iot-sdk-python

After the software package ST BlueST SDK and Azure IoT SDK for Python are installed correctly, go to the folder “/blue_st_examples” under the ST BlueST SDK installation path and go to Downloads page to download the example code “sensible_to_azure.py” for the demo and put it in this folder.
In the “sensible_to_azure.py” file. Replace the `CONNECTION_STRING = ""` with your connection string what you got from step 2.

```
55 from blue_st_sdk.feature import FeatureListener
56 from blue_st_sdk.features.audio.adpcm.feature_audio_adpcm import FeatureAudioADPCM
57 from blue_st_sdk.features.audio.adpcm.feature_audio_adpcm_sync import FeatureAudioADPCMSync
58
59 # The sample connects to a device-specific MQTT endpoint on your IoT Hub.
60 from azure.iot.device import IoTHubDeviceClient, Message
61 import json
62
63 CONNECTION_STRING = "HostName=testiotbox.azure-devices.net;DeviceId=iotbox;SharedAccessKey=LFkFBgLj+aEyw/cucrpkdXcWQy3kNJc1kMGQ/lPI19Q="
64 message = {}
65
66 # PRECONDITIONS
67 #
```

Figure 4: Replace Connection String

Save your changes and run “python3 sensible_to_azure.py”

Go to the Azure IoT Central, according to the type of sensor, you should be able to see the sensor value in your device template.

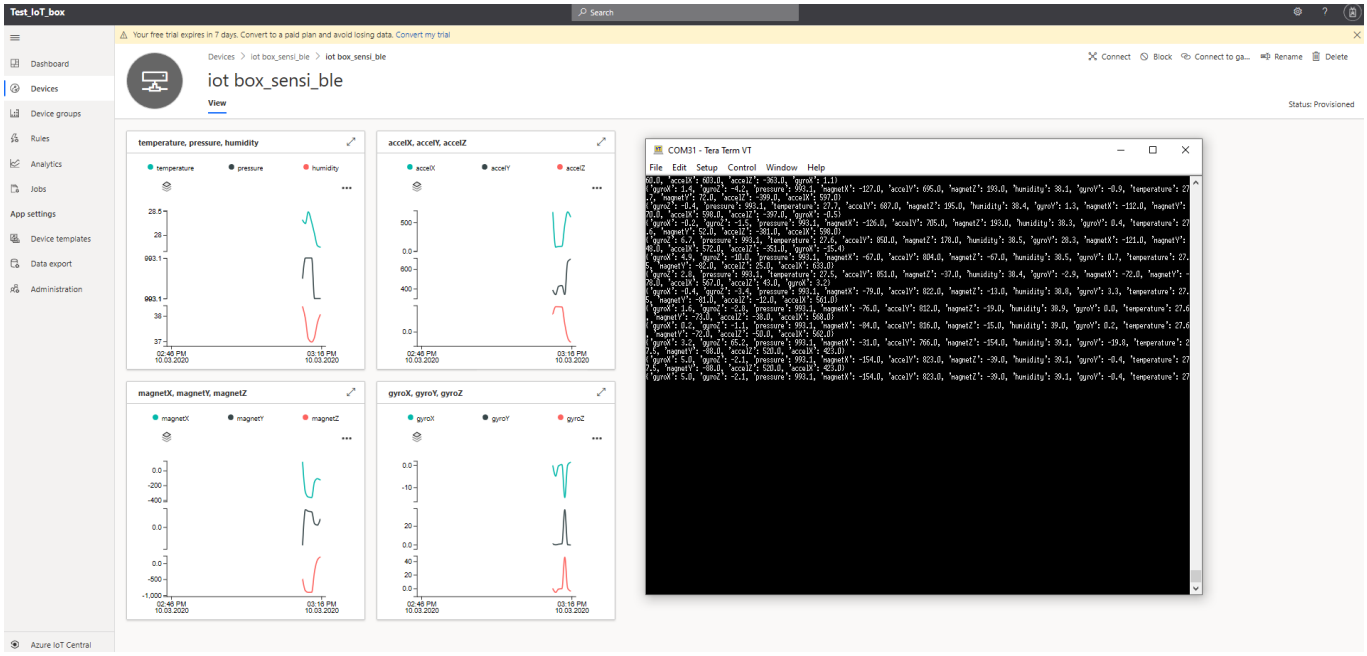


Figure 5: Demo Result