Enzo Unified – Adapter User Guide

Azure IoT Hub

Version 1.6 – RTM

Last Updated On September 2nd 2016

Contents

Pre-Requisites

In order to successfully install any adapter you must install Enzo Unified first. The installation wizard allows you to select the adapter(s) you want to install. You will also need to have an active subscription with the Microsoft Azure and have created an Azure IoT Hub service (the Adapter does not allow you to create the service itself).

Configure the Azure IoT Hub Adapter with Enzo Manager

Introduction

This apadter provides access to the Microsoft Azure IoT Hub and allows you to perform the following functions:

- Create IoT devices
- Send data on behalf of devices
- Simulate a field of devices

Configuration

Configuring the adapter to communicate with the Azure IoT Hub can be done using SQL commands or through Enzo Manager. The following configuration settings are required to communicate with an Azure IoT Hub:

Parameter Name	Type	Comment
Name	string	The name of the IoT Hub
ConnString	string	The connection string to the Hub

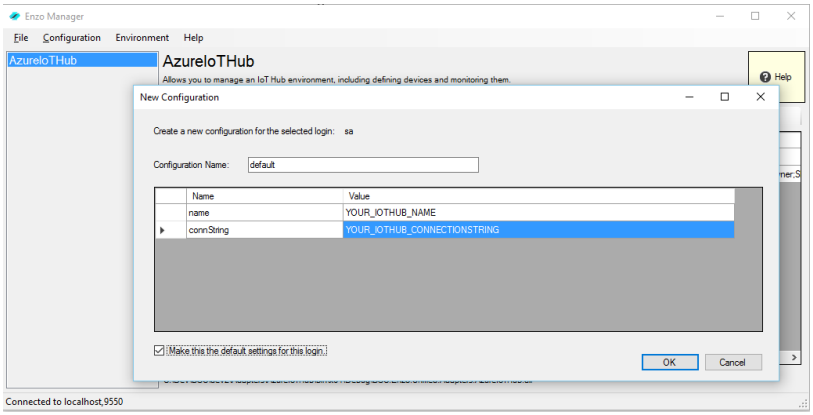
To configure the adapter using a SQL command, follow these steps:

- Open SQL Server Management Studio (SSMS)
- Connect to Enzo Unified using the 'sa' account
    - If Enzo Unified is running on the same machine and was installed with the default settings, the Server name is: localhost,9550
- Run this command to configure the adapter (the IOTHUB names should be replaced by the corresponding values found in the Azure portal on your IoT Hub settings page):

EXEC AzureIoTHub._configCreate 'default', 1, 'IOTHUB_NAME', 'IOTHUB_CONNECTION_STRING'

You can also use Enzo Manager to configure your adapter. Enzo Manager (EnzoManager.exe) is found in the **EnzoUnifiedSvc** directory where Enzo Unified is installed. To use Enzo Manager, follow these steps:

- Start Enzo Manager
- Login to Enzo Unified using the 'sa' account
    - If Enzo Unified is running on the same machine and was installed with the default settings, the Server name is: localhost,9550
- Click on the AzureIotHub adapter in the list of adapters
- Click on New Config
- Enter your Azure IOT Hub settings
    - Make sure to check the "Make this the default settings"



- Click OK

Commands Overview

Once configured, you can use SSMS to interact with the Azure IoT Hub environment. The following represents a subset of the available commands available. To view all available commands, run this command:

EXEC AzureIoTHub.help

Each operation can accept a number of parameters; to obtain further details on the available parameters for the SendMessage command, run this SQL:

EXEC AzureIoTHub.SendMessage help

Operation	SQL Command
Create a Device	EXEC bsc.AzureIoTHub.CreateDevice 'test1'
List Devices	EXEC bsc.AzureIoTHub.ListDevices
Send Telemetry Data from device Test1	EXEC bsc.AzureIoTHub.SendData 'test1', '{"deviceId":"test1", "location":"home", "messurementValue":700, "messurementType":"darkness","localTimestamp":"2016-4-14 16:35:00"}'
Send a Message to a device	EXEC bsc.AzureIoTHub.SendMessage 'test1', 'STOP'
Simulate sending 10 test telemetry data from two devices	EXEC bsc.AzureIoTHub.SendTestData 10, 1000, 'test1,test2', '{"deviceId":"#deviceid()", "location":"home", "messurementValue":700, "messurementType":"darkness","localTimestamp":"#utcnow()"}'

Functions Available

The SendTestData method allows you to specify functions instead of actual values in order to simulate data variations from field devices. The following functions are available:

Function	Description
#deviceid()	The ID of the device
#utcnow()	Current time in UTC
#now()	Current local time
#rndguid()	A random Guid
#rndint(a,b)	A random Integer between a and b
#rnddouble(a,b)	A random double number between a and b
#pick(a,b,c,…)	One of the values provided (a, or b, or c…)

The following example sends data 12 data sets to the IoT Hub from 4 devices:

```
EXEC bsc.AzureIoTHub.SendTestData
  12, -- number of telemetry data sets to send in all
   100, -- delay between each call
   'test1,test2,test3,test4',  -- list of devices to simulate (they must be registered first with CreateDevice)
   '{"deviceId":"#deviceid()", "location":"#pick(home,work,car)",
```

"messurementValue":#rnddouble(400.0,700.0), "messurementType":"darkness", "localTimestamp":"#utcnow()"}'

This command generates 12 JSON documents sent into the Azure IoT Hub: