



# Enzo Unified – Adapter User Guide

Azure IoT Hub

Version 1.6 – RTM

Last Updated On September 2<sup>nd</sup> 2016

## Contents

Pre-Requisites .....	1
Configure the Azure IoT Hub Adapter with Enzo Manager .....	1
Introduction .....	1
Configuration .....	2
Commands Overview .....	3
Functions Available .....	3

## 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 adapter 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
<b>Name</b>	string	The name of the IoT Hub
<b>ConnString</b>	string	The connection string to the Hub

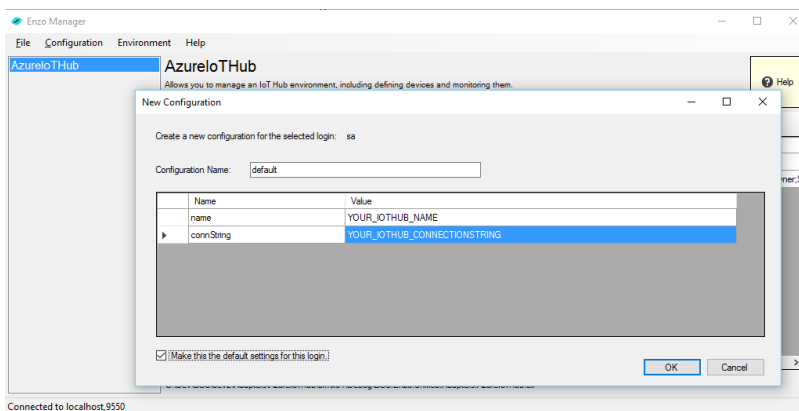
To configure the adapter using a SQL command, follow these steps:

- Open SQL Server Management Studio (SSMS)
- Connect to Enzo Unified using the 'sa' account
  - o 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
  - o 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 AzureIoT Hub adapter in the list of adapters
- Click on New Config
- Enter your Azure IOT Hub settings
  - o 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
<b>Create a Device</b>	EXEC bsc.AzureIoTHub.CreateDevice 'test1'
<b>List Devices</b>	EXEC bsc.AzureIoTHub.ListDevices
<b>Send Telemetry Data from device Test1</b>	EXEC bsc.AzureIoTHub.SendData 'test1', '{"deviceId":"test1", "location":"home", "messurementValue":700, "messurementType":"darkness","localTimestamp":"2016-4-14 16:35:00"}'
<b>Send a Message to a device</b>	EXEC bsc.AzureIoTHub.SendMessage 'test1', 'STOP'
<b>Simulate sending 10 test telemetry data from two devices</b>	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:

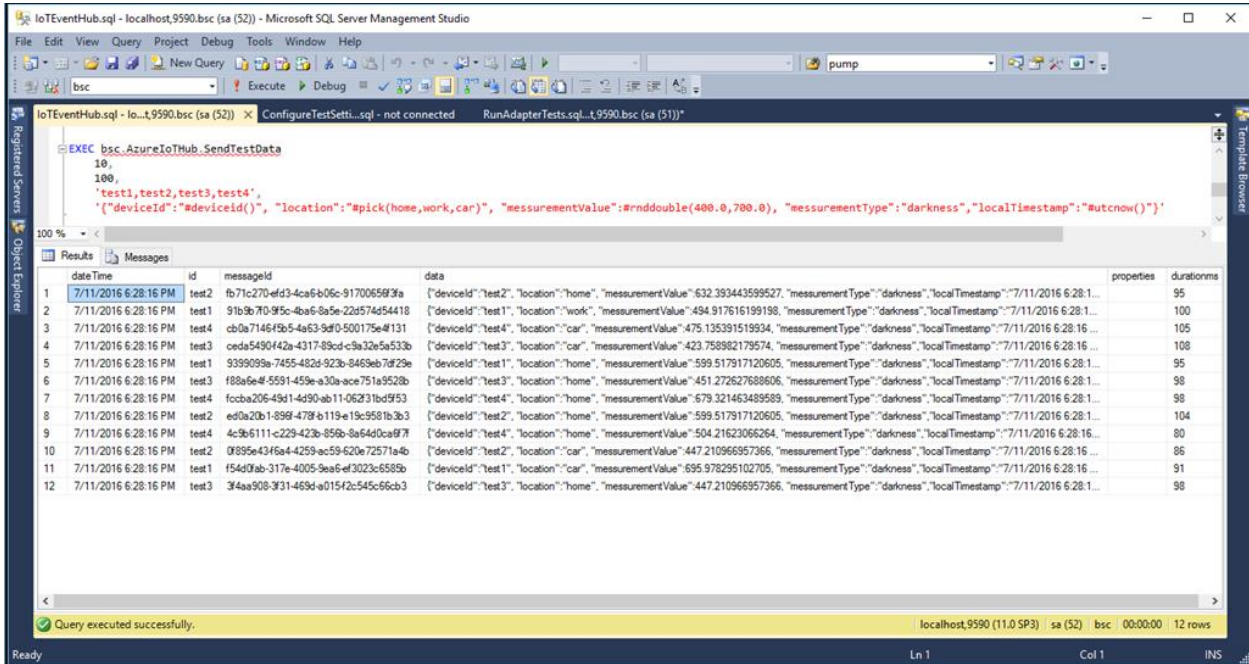
Function	Description
<b>#deviceId()</b>	The ID of the device
<b>#utcnow()</b>	Current time in UTC
<b>#now()</b>	Current local time
<b>#rndguid()</b>	A random Guid
<b>#rndint(a,b)</b>	A random Integer between a and b
<b>#rnddouble(a,b)</b>	A random double number between a and b
<b>#pick(a,b,c,...)</b>	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)"
```

```
"measurementValue":#rnddouble(400.0,700.0), "measurementType":"darkness",
"localTimestamp": "#utcnow()}"'
```

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



The screenshot shows the following SQL query executed in Microsoft SQL Server Management Studio:

```
EXEC bsc.AzureIoTHub_SendTestData
10,
100,
'test1,test2,test3,test4',
'{"deviceId":"#deviceId()", "location":"#pick(home,work,car)", "measurementValue":#rnddouble(400.0,700.0), "measurementType":"darkness", "localTimestamp": "#utcnow()}"'
```

The results table contains 12 rows of data:

dateTime	id	messageId	data	properties	durationms
7/11/2016 6:28:16 PM	test2	fb71c270-ef43-4ca6-b06c-9170065983fa	["deviceId":"test2", "location":"home", "measurementValue":632.393443599527, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		95
7/11/2016 6:28:16 PM	test1	91b9b7f0-9fc5-4ba6-8a5e-22d574d54418	["deviceId":"test1", "location":"work", "measurementValue":494.917616199198, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		100
7/11/2016 6:28:16 PM	test4	cb0a7146f9b5-4a63-9af0-500175e4f131	["deviceId":"test4", "location":"car", "measurementValue":475.135391519934, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		105
7/11/2016 6:28:16 PM	test3	ceda549042a-4317-89cd-c9a32e5a533b	["deviceId":"test3", "location":"car", "measurementValue":423.758982179574, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		108
7/11/2016 6:28:16 PM	test1	9399099a-7455-482d-923b-8469eb7af29e	["deviceId":"test1", "location":"home", "measurementValue":599.517917120605, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		95
7/11/2016 6:28:16 PM	test3	f88a6e4f-5591-459e-a30a-ace751a9528b	["deviceId":"test3", "location":"home", "measurementValue":451.272627688606, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		98
7/11/2016 6:28:16 PM	test4	fccba206-49d1-4d90-ab11-06231bd9f53	["deviceId":"test4", "location":"home", "measurementValue":679.321463489589, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		98
7/11/2016 6:28:16 PM	test2	ed0a20b1-896f-478f-b119-e19c9581b3b3	["deviceId":"test2", "location":"home", "measurementValue":599.517917120605, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		104
7/11/2016 6:28:16 PM	test4	4c9b6111-c229-423b-856b-8a64d0ca9f7f	["deviceId":"test4", "location":"home", "measurementValue":504.21623066264, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		80
7/11/2016 6:28:16 PM	test2	0895e43f6a-4259-ac59-620e72571a4b	["deviceId":"test2", "location":"car", "measurementValue":447.210966957366, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		86
7/11/2016 6:28:16 PM	test1	f54d0ab-317e-4005-9ea5-ef3023c6589b	["deviceId":"test1", "location":"car", "measurementValue":695.978295102705, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		91
7/11/2016 6:28:16 PM	test3	3f4aa908-3f31-469d-a01542c545c66cb3	["deviceId":"test3", "location":"home", "measurementValue":447.210966957366, "measurementType":"darkness", "localTimestamp":"7/11/2016 6:28:16 PM"]		98

The status bar at the bottom indicates: Query executed successfully. localhost:9590 (11.0 SP3) sa (52) bsc 00:00:00 12 rows