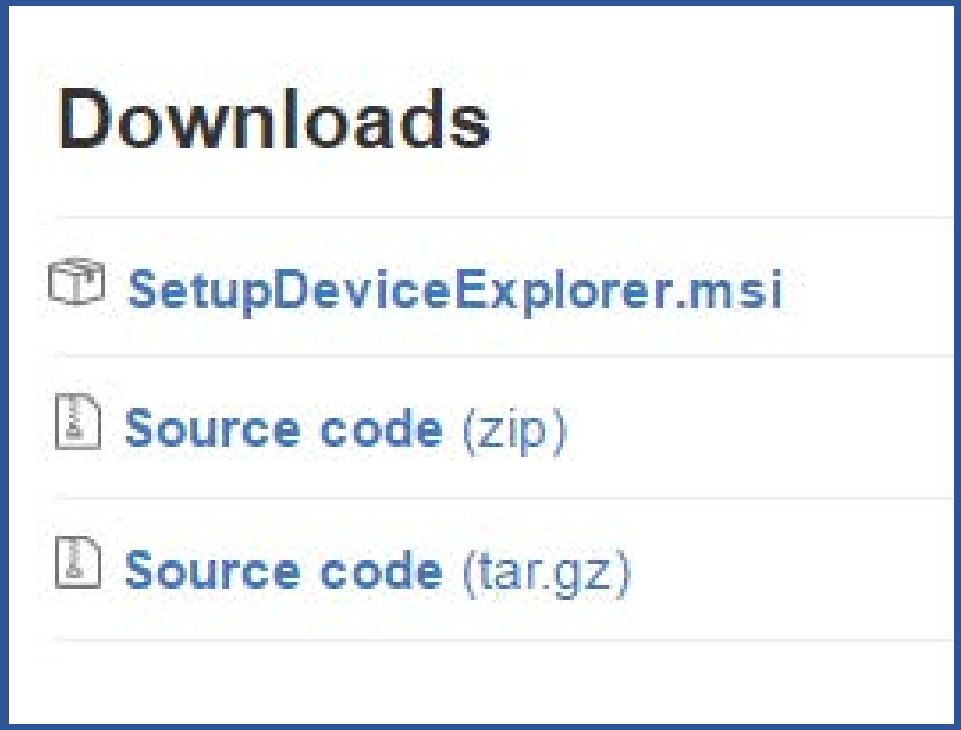
Device Explorer for IoT Hub

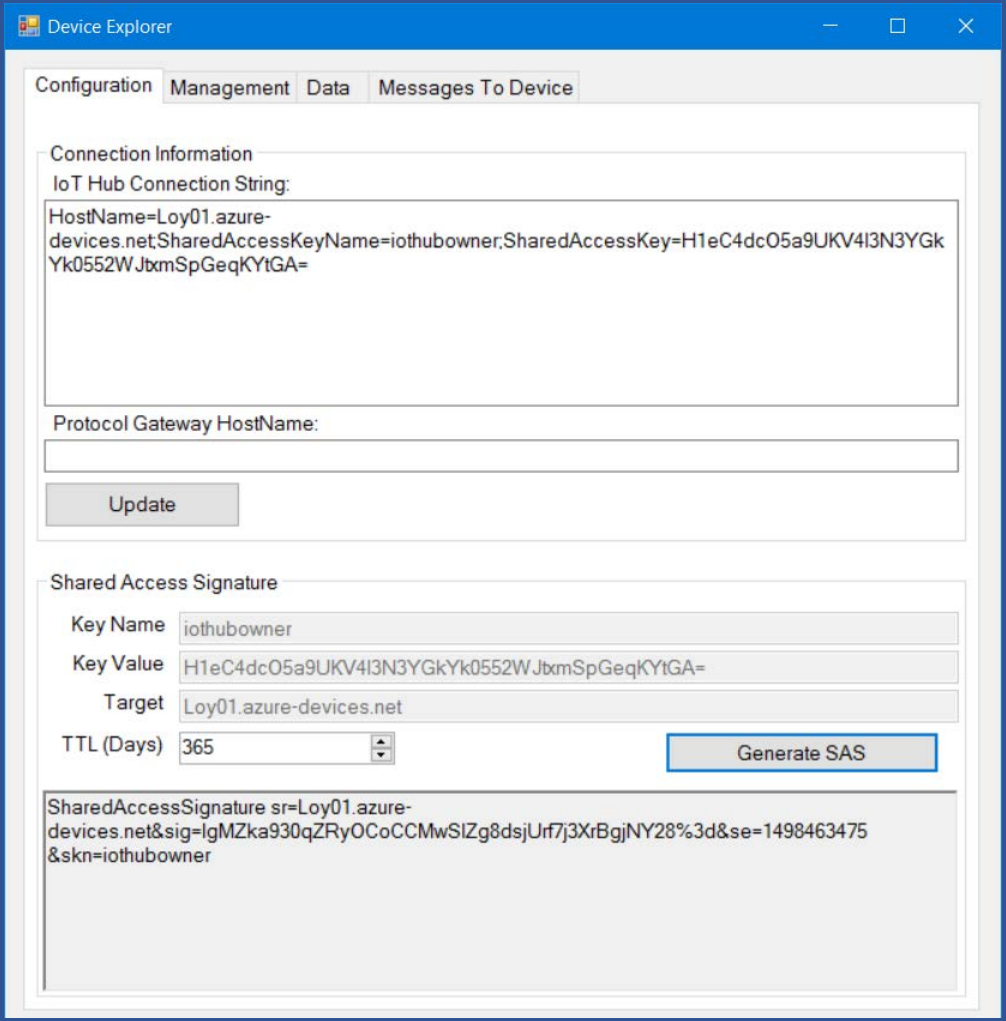
- Getting Device Explorer
- Configure an IoT Hub connection
- Manage devices
- List registered devices
- Create device
- Update device
- Delete device
- Get device connection string or data
- Monitor device-to-cloud events
- Send cloud-to-device messages

Getting Device Explorer

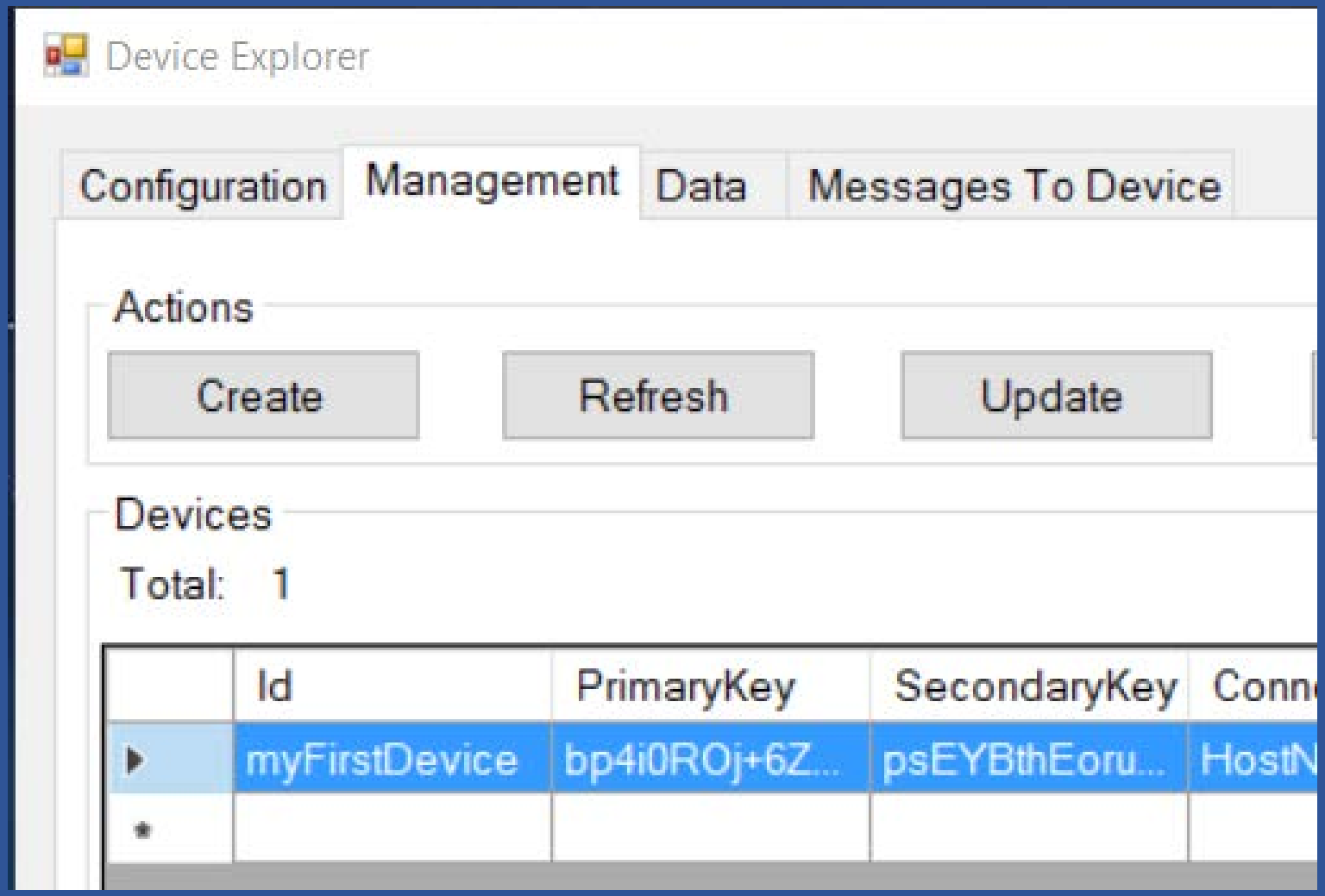
https://github.com/Azure/azure-iot-sdks/releases



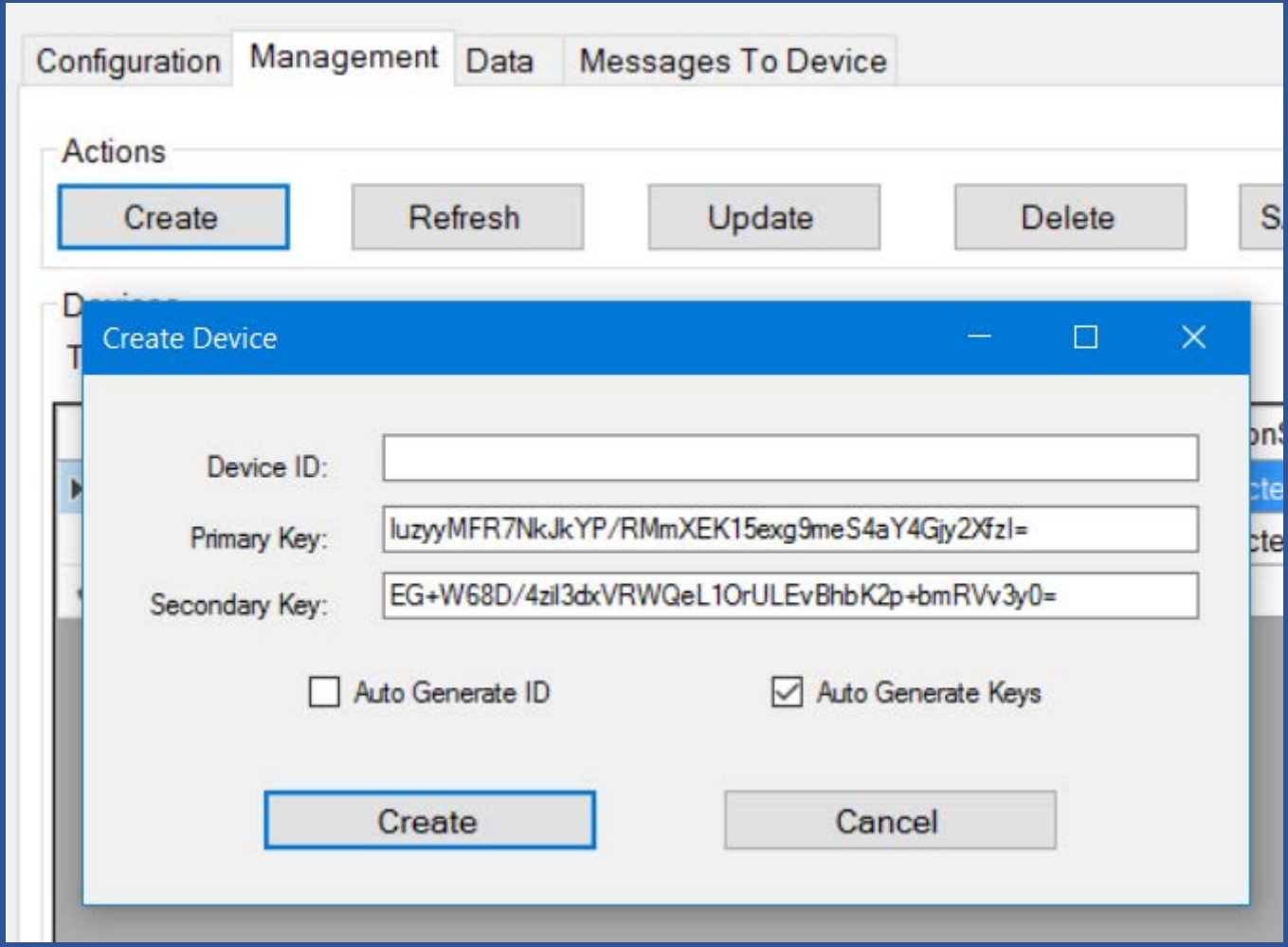
Configure an IoT Hub connection



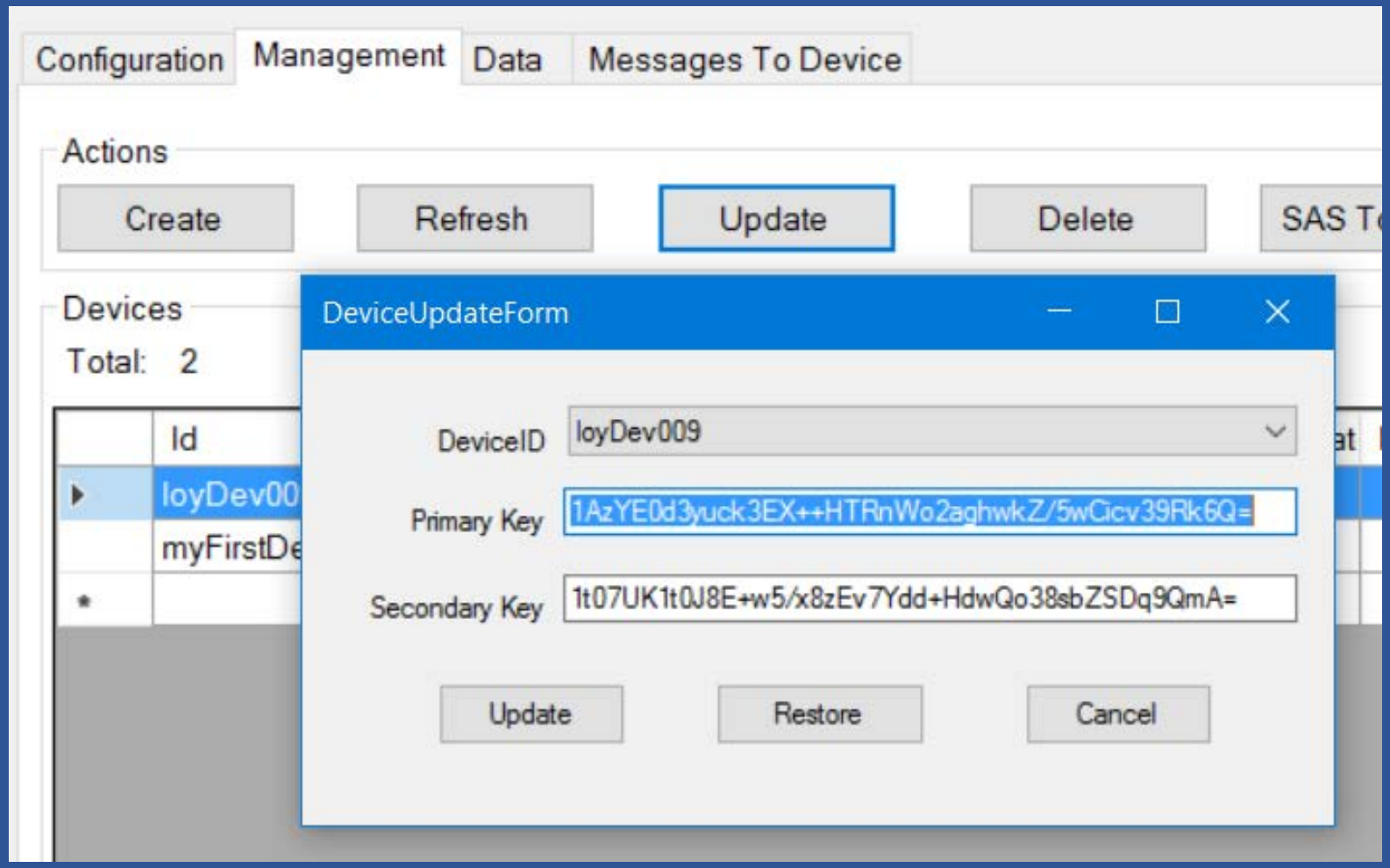
Manage devices



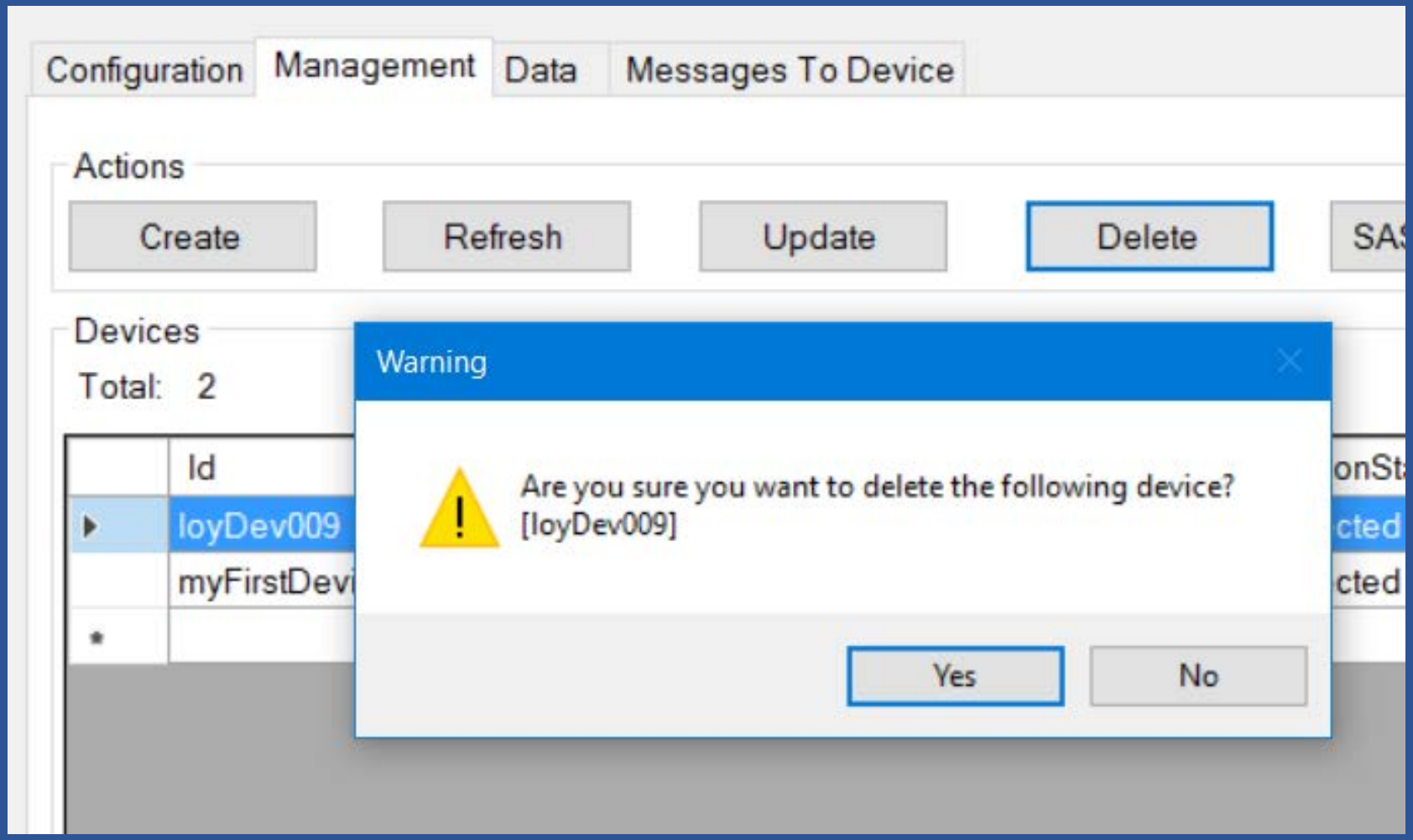
Create device (Add)



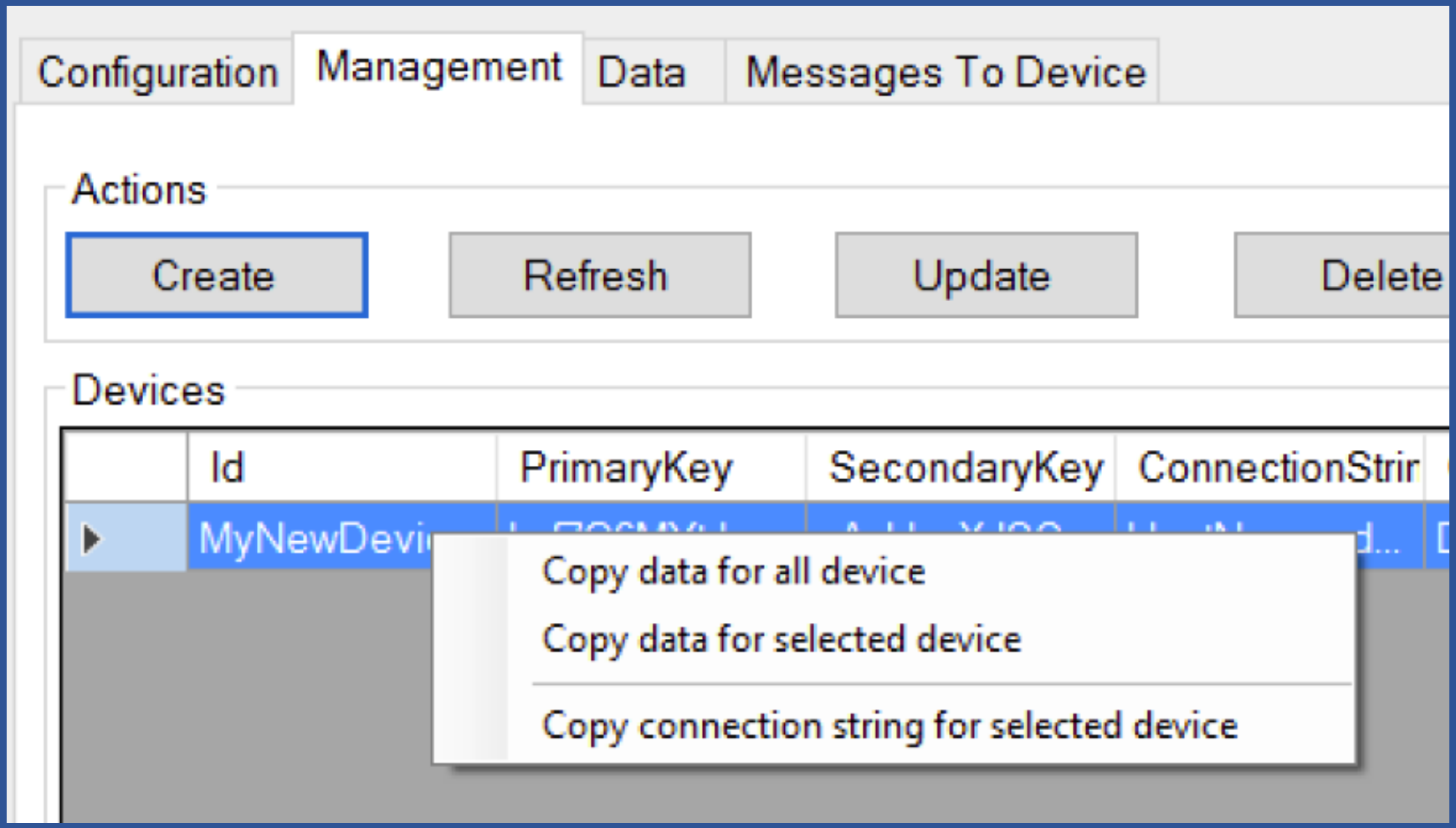
Update device



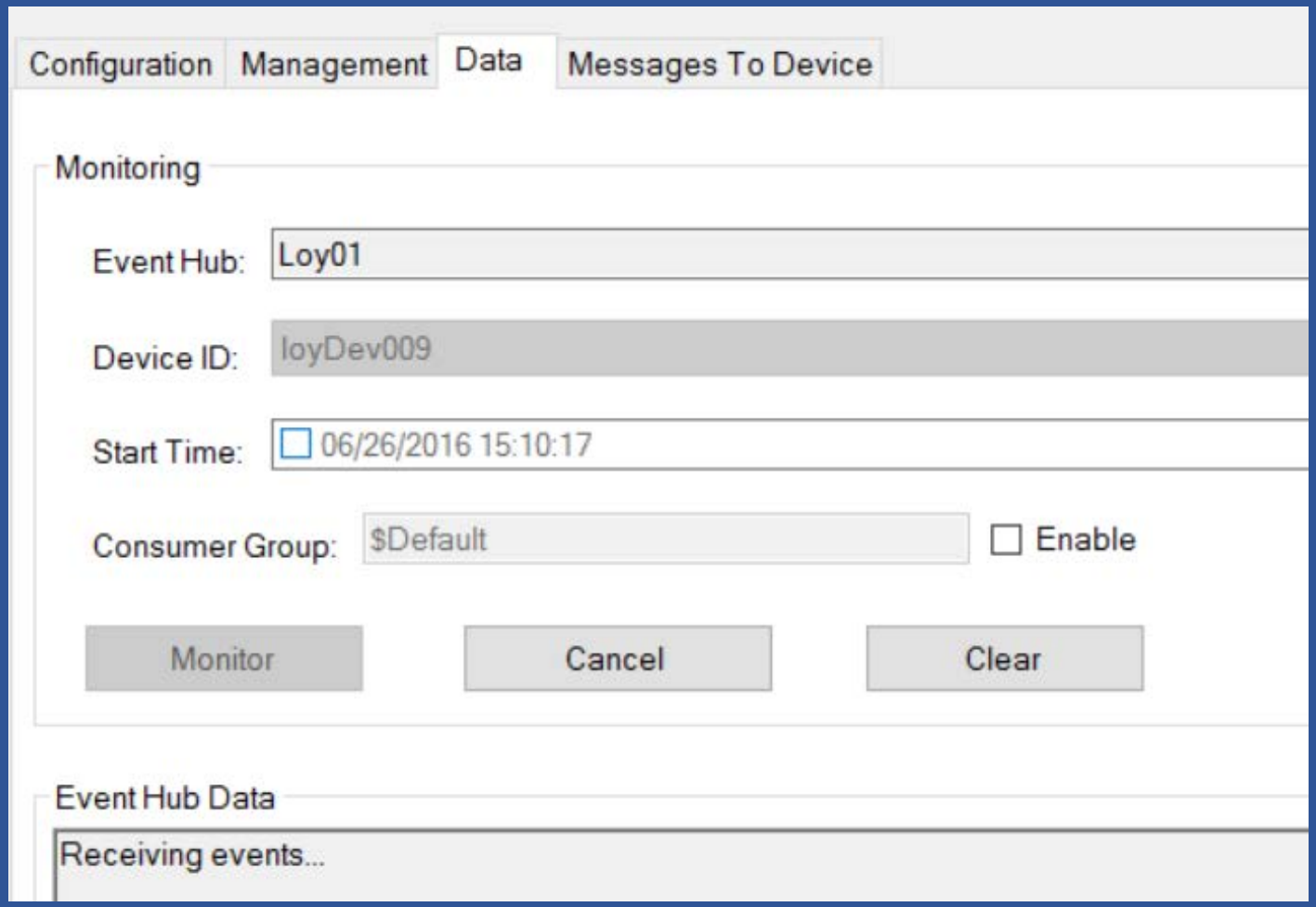
Delete device



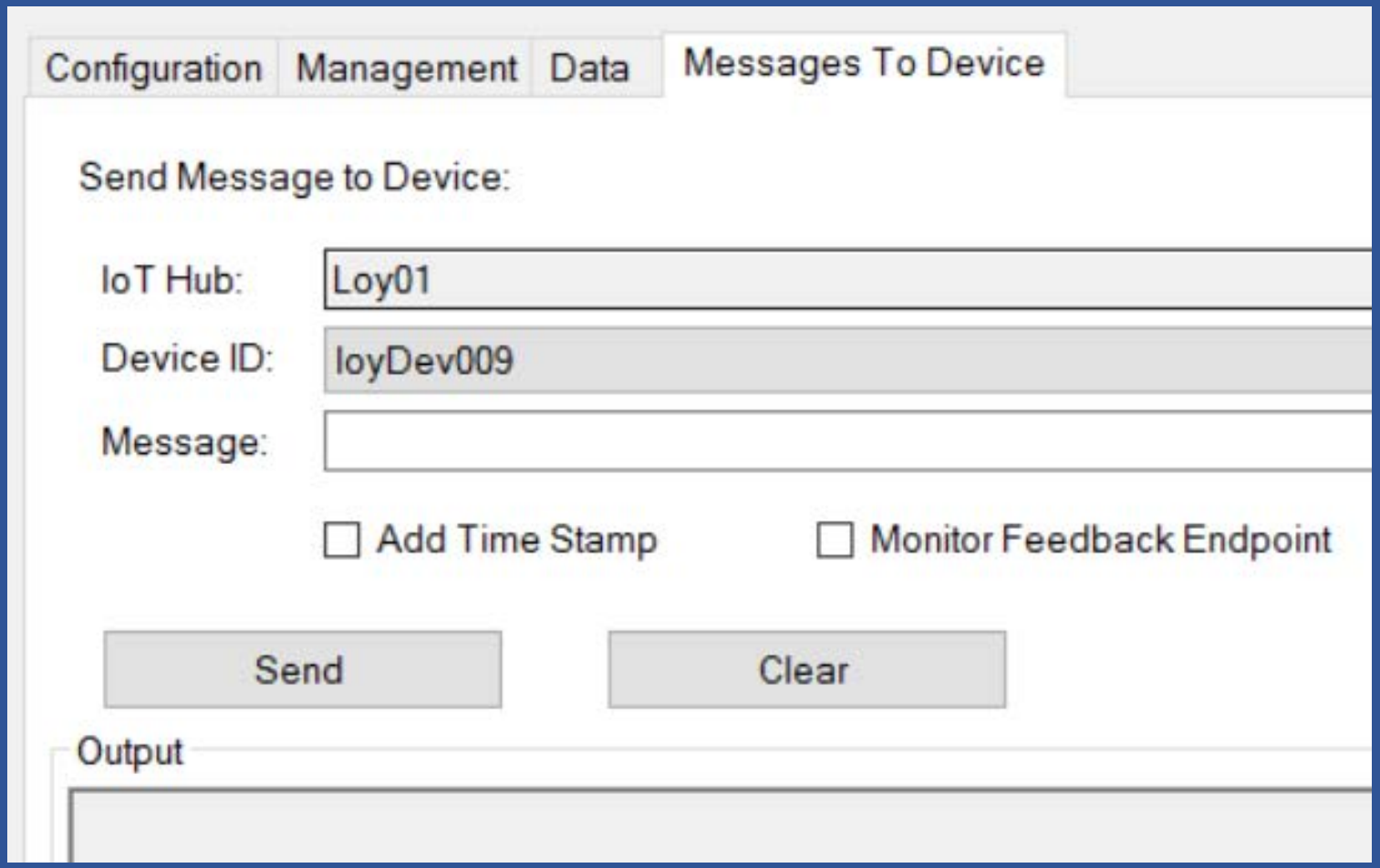
Get device connection string or data



Monitor device-to-cloud events



Send cloud-to-device messages



Creating Device Simulator

- AMQP
- JSON
- Create Device Simulator
- Send Device to Cloud Message
- Receive Cloud to Device Message

AMQP

- The Advanced Message Queuing Protocol (AMQP) is an open standard application layer protocol for message-oriented middleware.
- The defining features of AMQP are message orientation, queuing, routing (including point-to-point and publish-and-subscribe), reliability and security.

JSON

For embedded devices& low bandwidth, an another format is preferred JSON (JavaScript Object Notation)

JSON

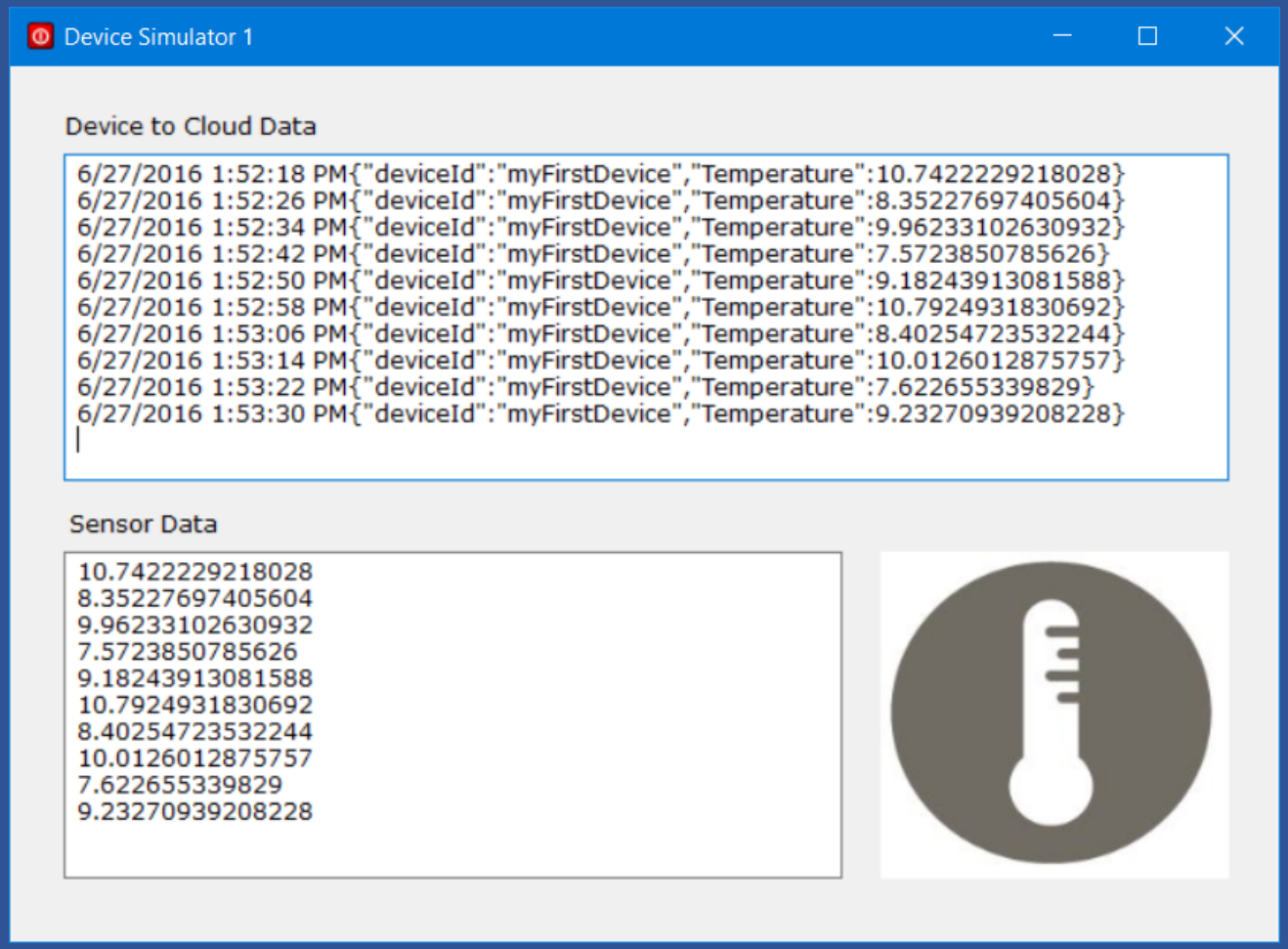
```
{
  "id": 123,
  "title": "Object Thinking",
  "author": "David West",
  "published": {
    "by": "Microsoft Press",
    "year": 2004
  }
}
```

XML

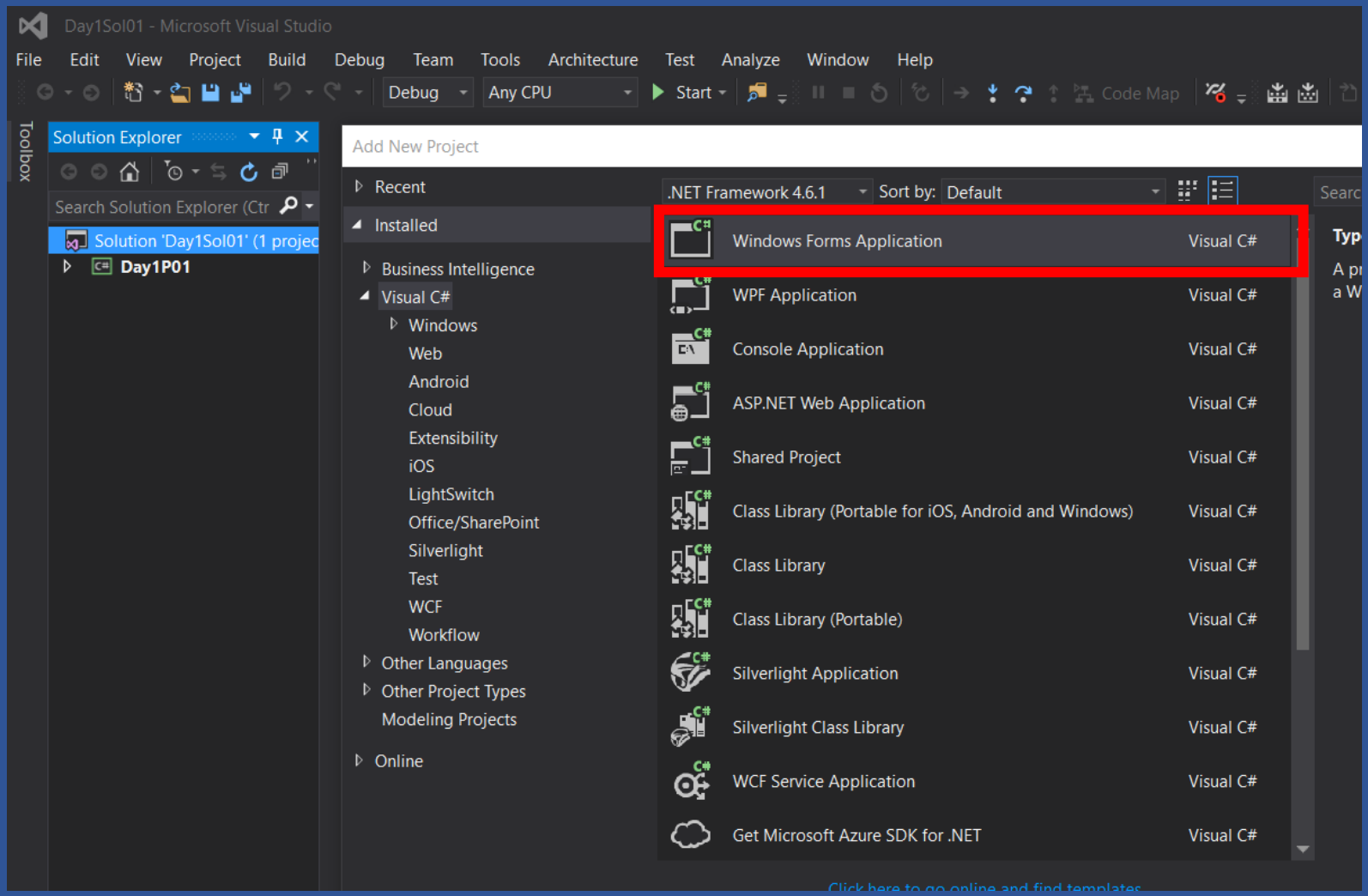
```
<?xml version="1.0"?>
<book id="123">
  <title>Object Thinking</title>
  <author>David West</author>
  <published>
    <by>Microsoft Press</by>
    <year>2004</year>
  </published>
</book>
```

- JSON parser needs to be implemented to analyze server response
- JSON serializer to publish some data on the cloud

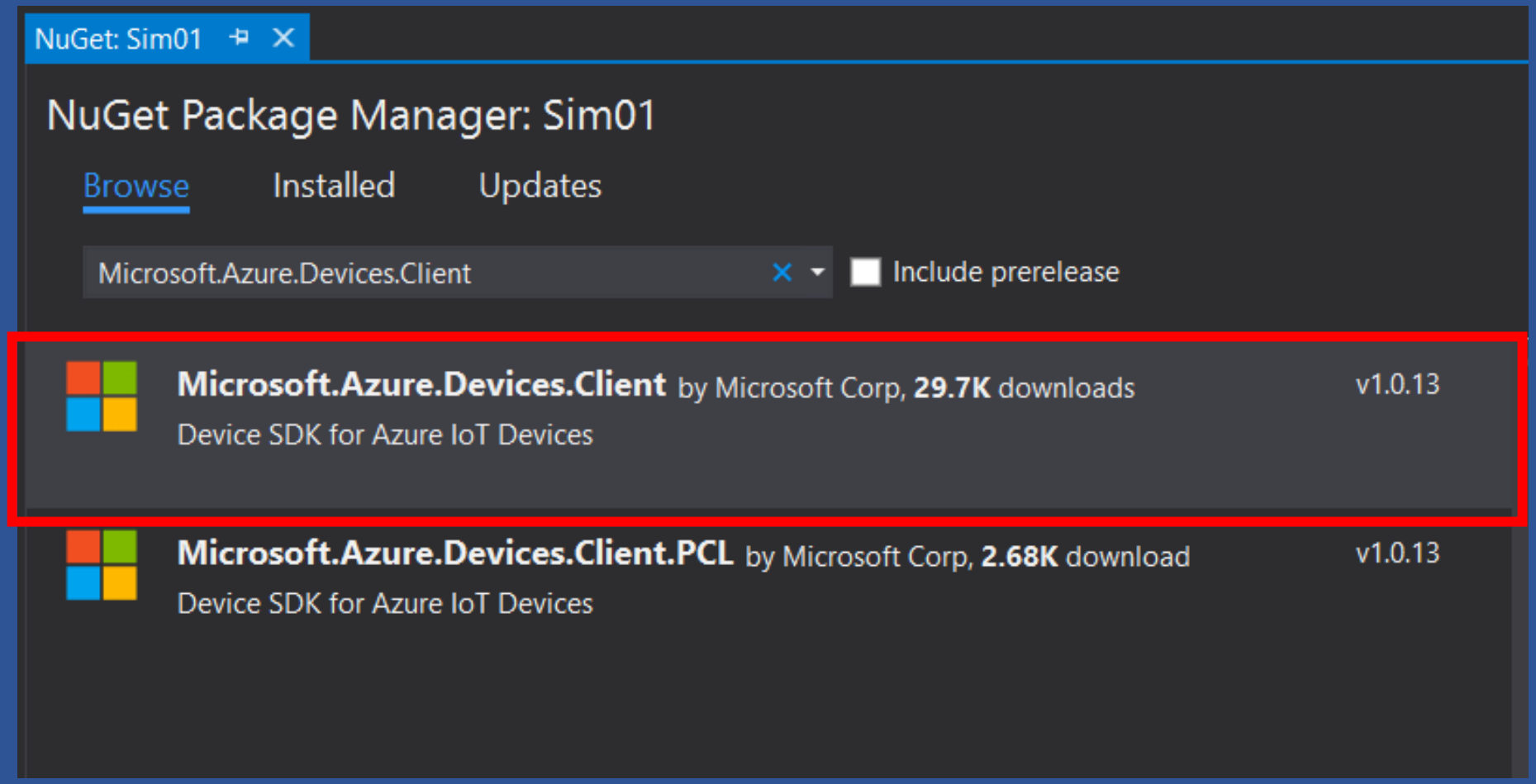
Create Device to Cloud Simulator



1. Create New Windows Forms Application “Sim01”



2. Add NuGet Package: Microsoft.Azure.Devices.Client



3.Add the following using statement at the top of the Form1.cs

Form1.cs*

Sim01

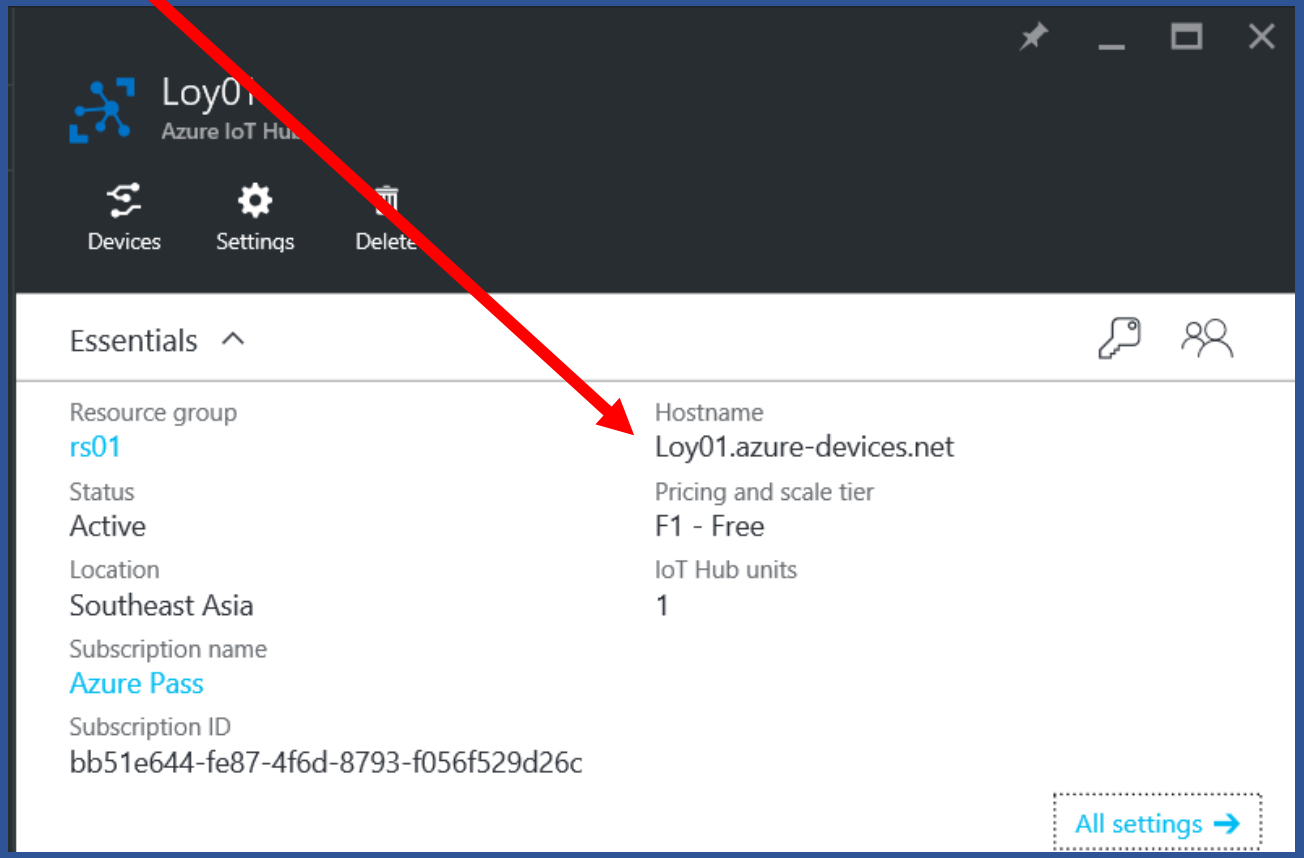
```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

using Microsoft.Azure.Devices.Client;
using Newtonsoft.Json;
```

4. Add the following fields to the Form1 class.
   - Change iotHubUri
   - Change deviceKey

```
public partial class Form1 : Form
{
    DeviceClient deviceClient;
    string deviceId = "myFirstDevice";
    string iotHubUri = "loyHub2.azure-devices.net";
    string deviceKey = "T8m7nbGcHrlJL56AMKT/va3ybdh52BiJ/6pCpnXHMcc=";
    1 reference
    public Form1()
    {
        InitializeComponent();
    }
}
```

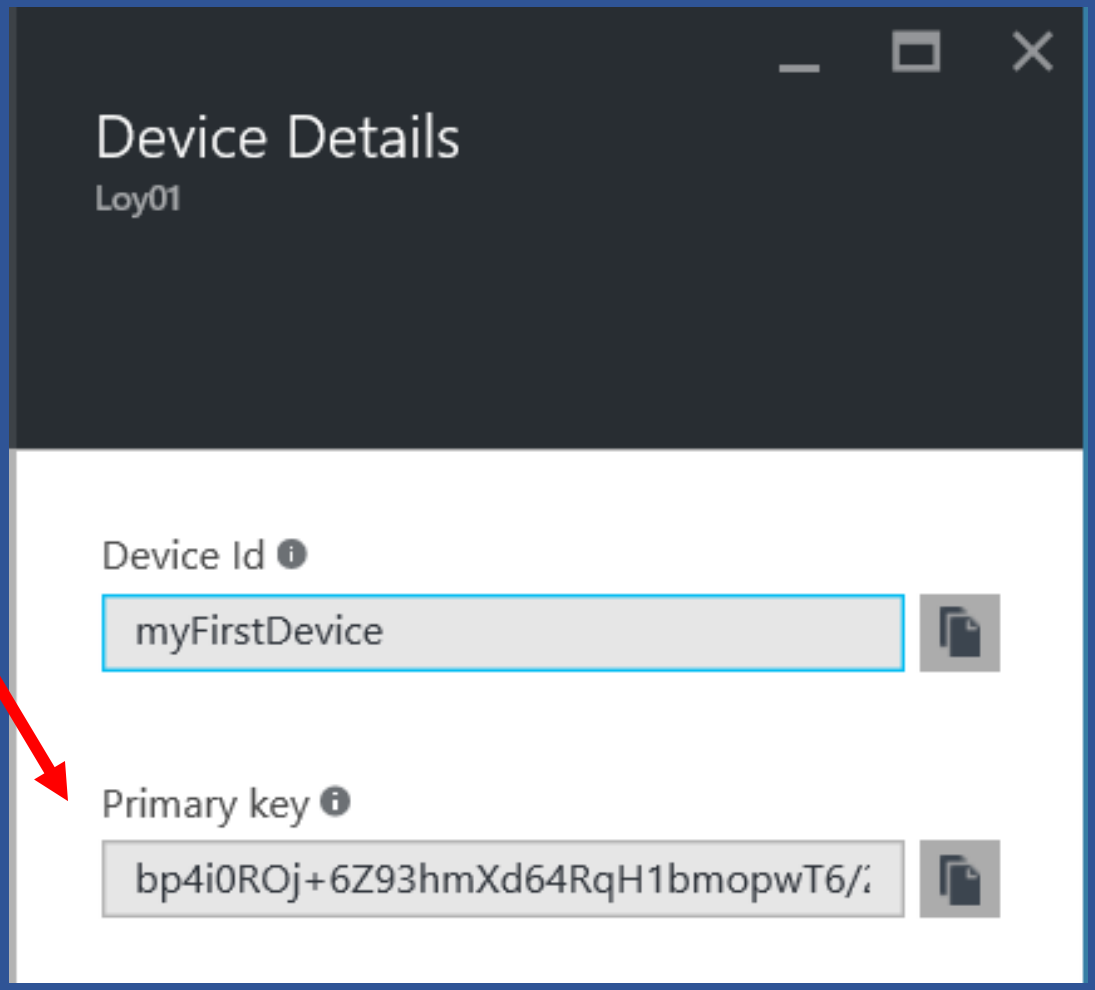
5. Your IoT Hub Home page

iotHubUri

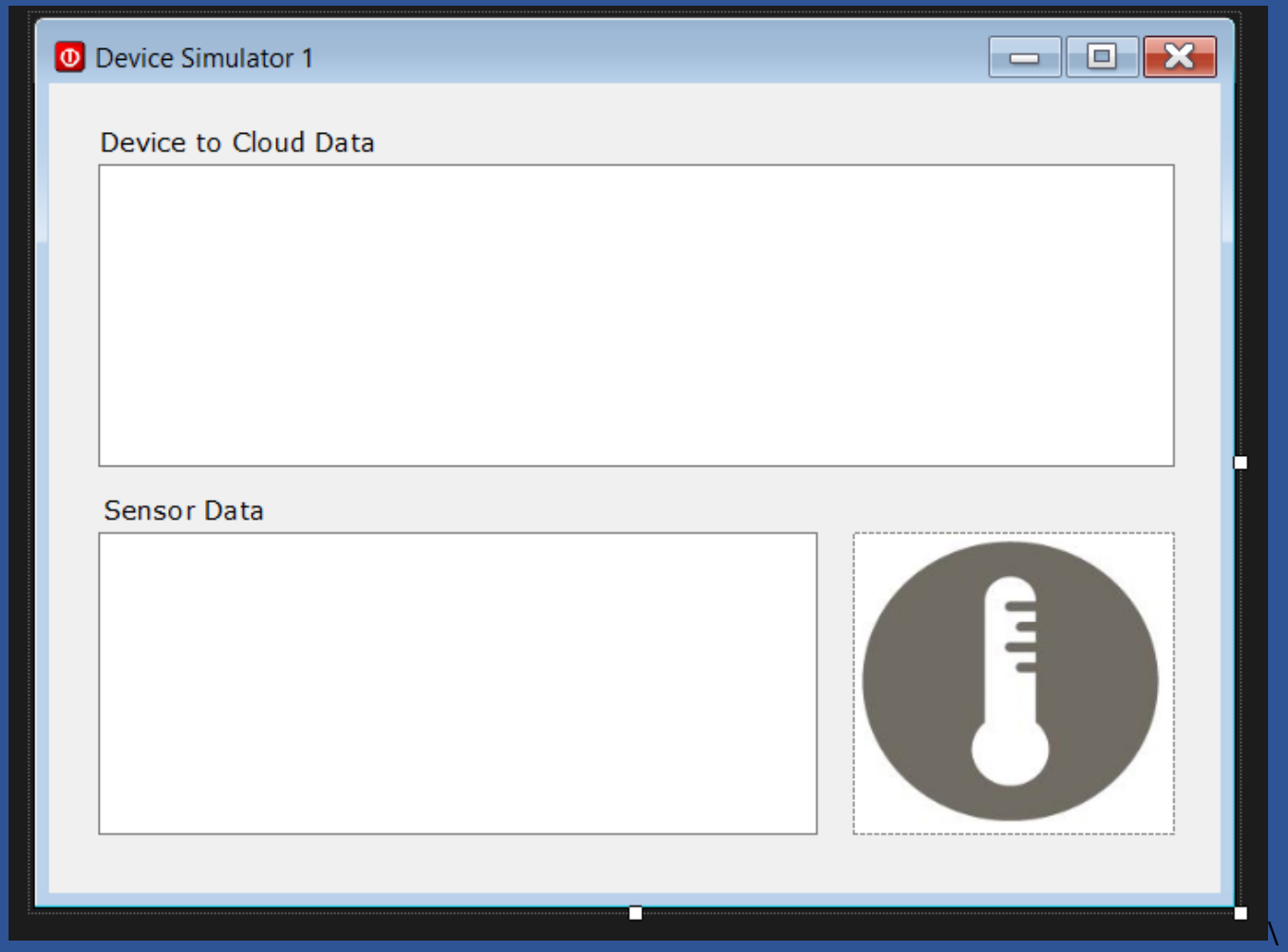


6. Azure IoT Hub/Devices/myFirstDevice/Device Details

device key



7. Add 2 TextBox "textBoxD2C" and "textBoxSensorData"



8. Add below code to Form1 constructor method

```
public Form1()
{
    InitializeComponent();
    deviceClient =
        DeviceClient.Create(iotHubUri,
        new DeviceAuthenticationWithRegistrySymmetricKey
            (deviceId, deviceKey));
    Timer myTimer = new Timer();
    myTimer.Enabled = true;
    myTimer.Interval = 1000;
    myTimer.Tick += MyTimer_Tick;
}
```

9. Add below code to Form1 Class

```
private void MyTimer_Tick(object sender, EventArgs e)
{
    double temper = 10; // m/s
    Random rand = new Random();
    double currentTemper = temper + rand.NextDouble() * 4 - 3;
    string temperStr = currentTemper.ToString();
    textBoxSensorData.Invoke(new Action(() =>
        { textBoxSensorData.AppendText(temperStr + "\r\n"); }));
}
```

10. Test program. Observe the output.

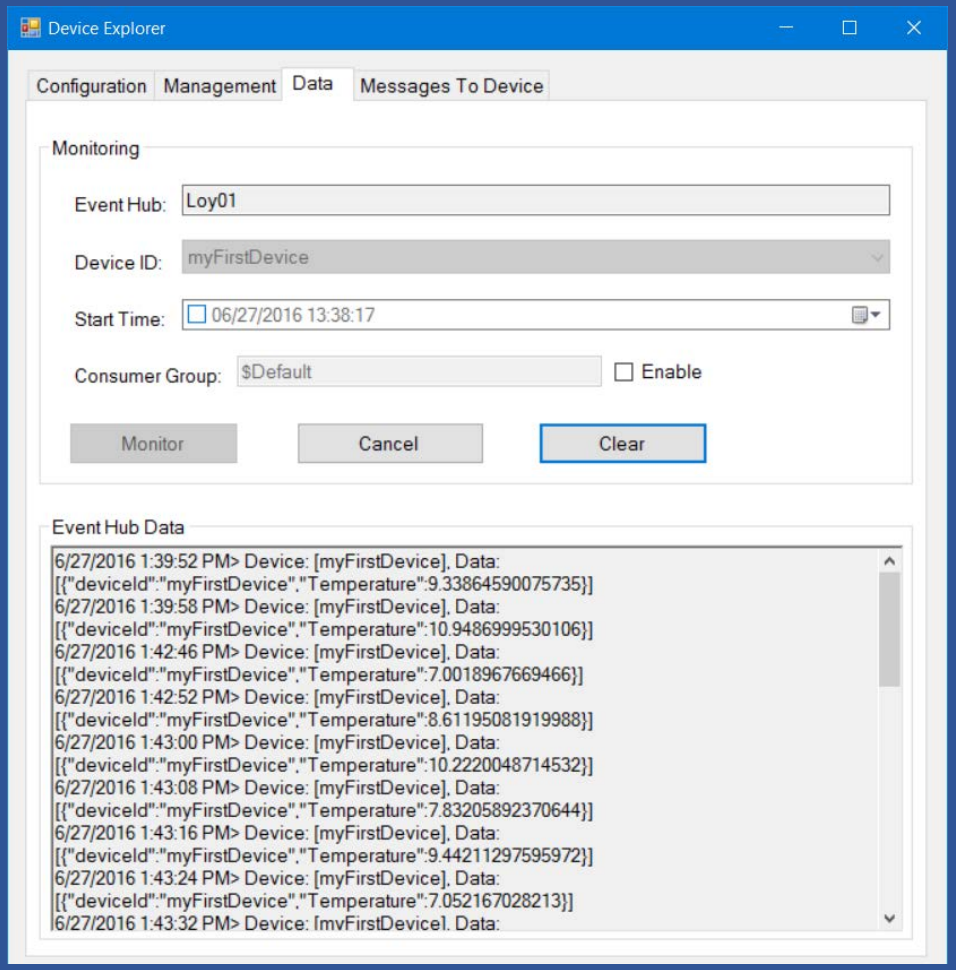
11. Add below code to Form1 Class

```
private async void SendDeviceToCloudMessagesAsync(string data)
{
    var telemetryDataPoint = new
    {
        deviceId = deviceId,
        Temperature = Convert.ToDouble(data)
    };
    var messageString = JsonConvert.SerializeObject(telemetryDataPoint);
    var message = new Microsoft.Azure.Devices.Client.Message
        (Encoding.ASCII.GetBytes(messageString));
    await deviceClient.SendEventAsync(message);
    textBoxD2C.Invoke(new Action(() =>
        { textBoxD2C.AppendText(DateTime.Now + messageString + "\r\n"); }));
}
```

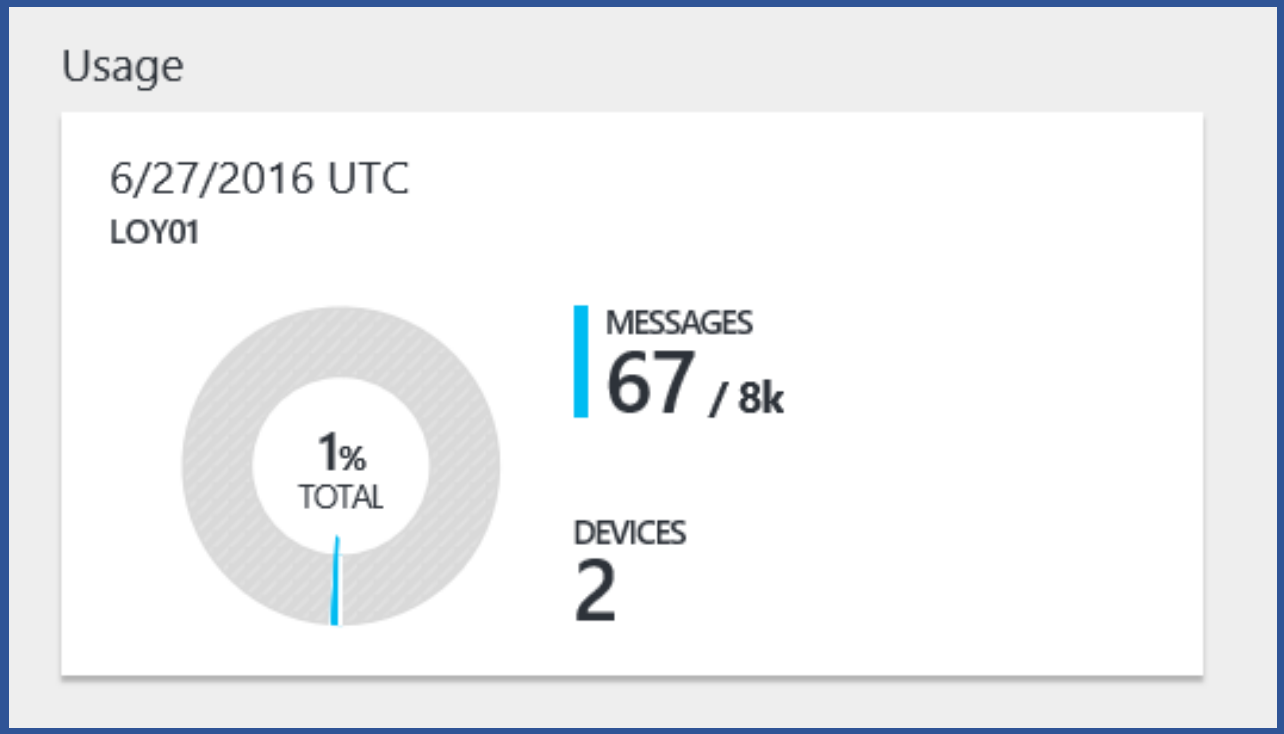
12. Modify MyTimer_Tick method

```
private void MyTimer_Tick(object sender, EventArgs e)
{
    double temper = 10; // m/s
    Random rand = new Random();
    double currentTemper = temper + rand.NextDouble() * 4 - 3;
    string temperStr = currentTemper.ToString();
    textBoxSensorData.Invoke(new Action(() =>
        { textBoxSensorData.AppendText(temperStr + "\r\n"); }));
    SendDeviceToCloudMessagesAsync(temperStr);
}
```

13. Use Device Explorer to inspect Device to Cloud Message



14. Open IoT Hub Home to inspect the Usage



More on Create Simulated Device for Device to Cloud

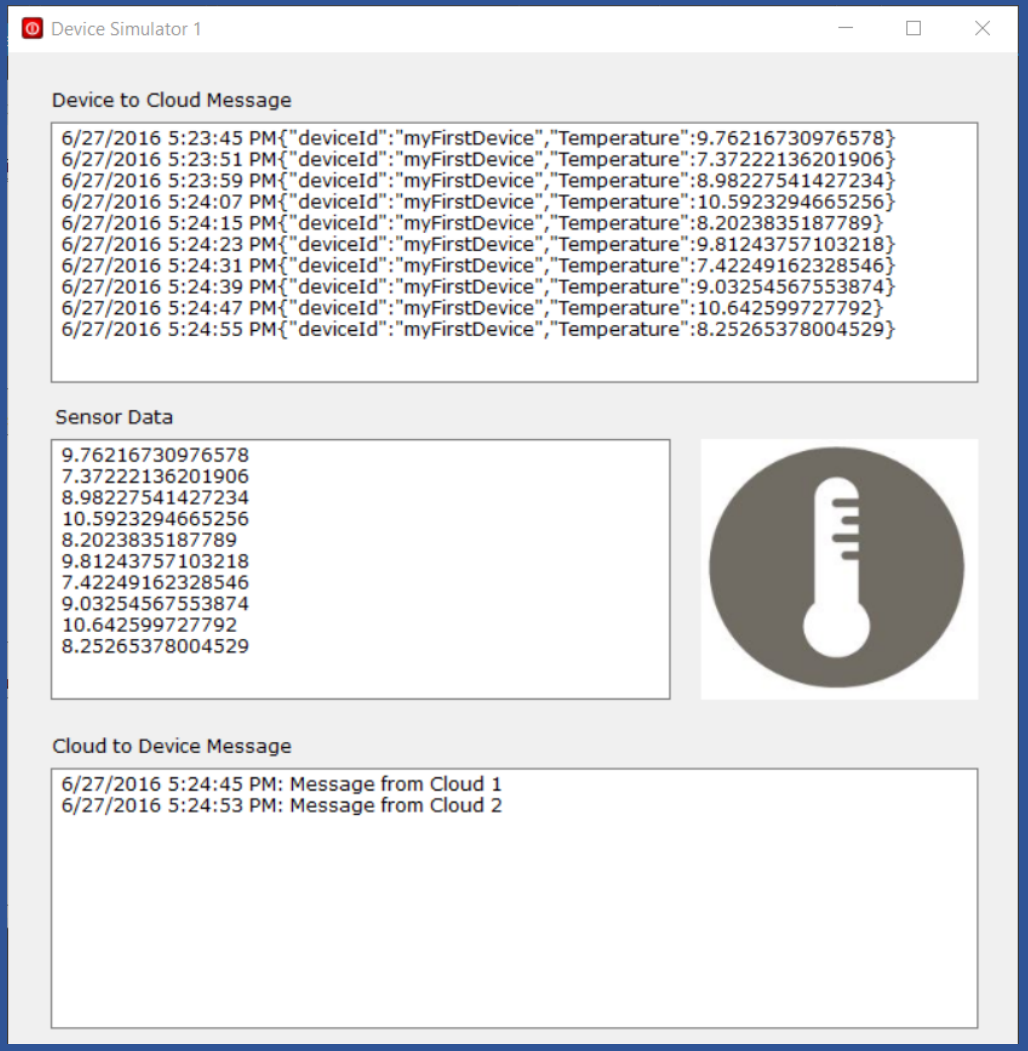
- Get started with Azure IoT Hub for .NET

https://azure.microsoft.com/en-us/documentation/articles/iot-hub-csharp-csharp-getstarted/#create-a-simulated-device-app

- ADD A CUSTOM DEVICE TO THE AZURE IOT SUITE REMOTE MONITORING SOLUTION

http://rickrainey.com/2016/06/22/add-custom-device-azure-iot-suite-remote-monitoring-solution/

Receive message from Cloud



1. Add a TextBox to Form1. Name = “TextBoxC2D”

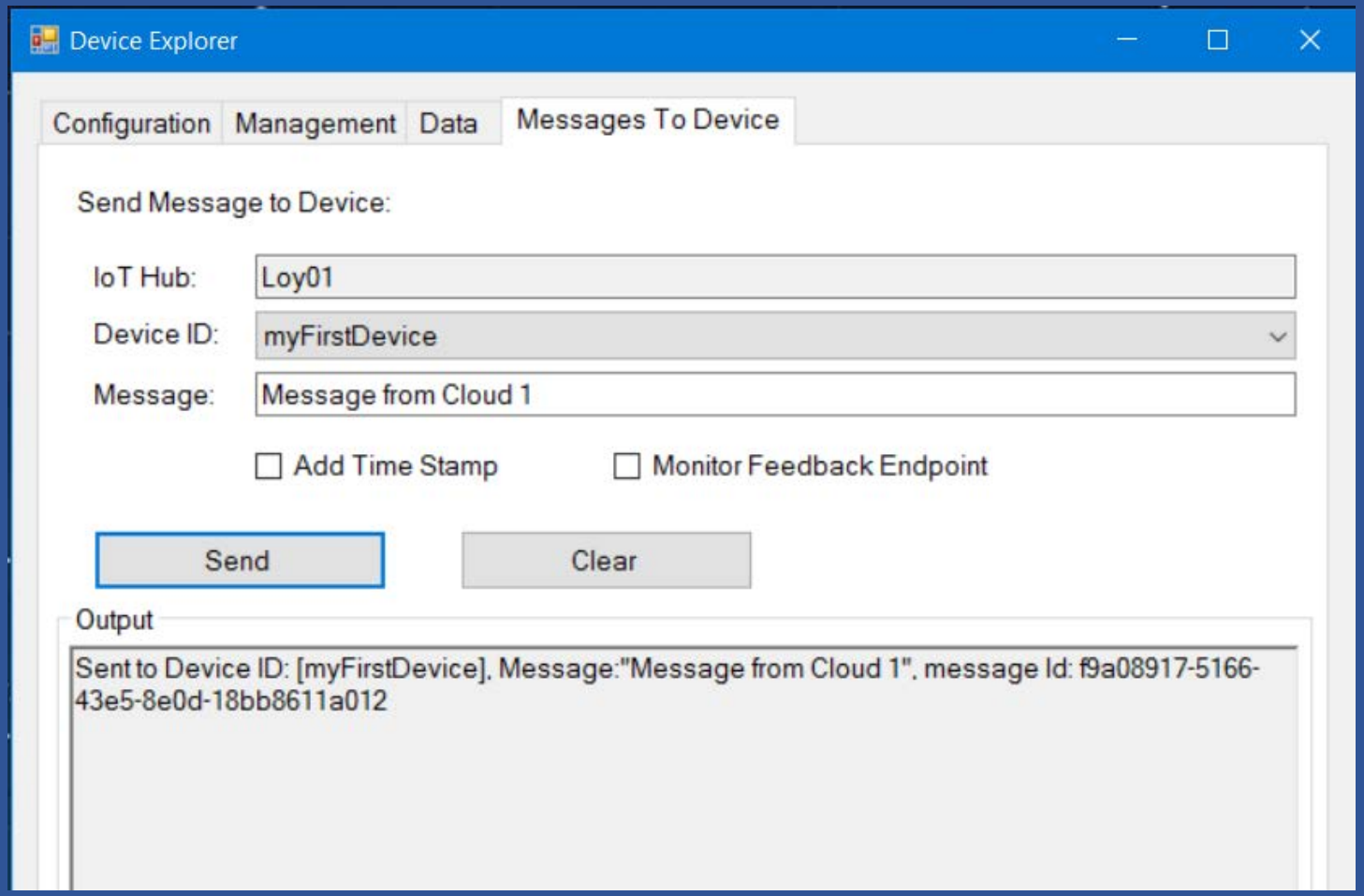
2. Add below method to Form1

```
private async void ReceiveC2dAsync()
{
    string msg = string.Empty;
    while (true)
    {
        Microsoft.Azure.Devices.Client.Message
            receivedMessage = await deviceClient.ReceiveAsync();
        if (receivedMessage == null) continue;
        msg = Encoding.ASCII.GetString(receivedMessage.GetBytes());
        textBoxC2D.Invoke(new Action(() =>
            { textBoxC2D.AppendText(DateTime.Now + ": " + msg + "\r\n"); }));
        await deviceClient.CompleteAsync(receivedMessage);
    }
}
```

3. Modify Form1 Class Constructor

```
public Form1()
{
    InitializeComponent();
    deviceClient =
        DeviceClient.Create(iotHubUri,
        new DeviceAuthenticationWithRegistrySymmetricKey
            (deviceId, deviceKey));
    Timer myTimer = new Timer();
    myTimer.Enabled = true;
    myTimer.Interval = 8000;
    myTimer.Tick += MyTimer_Tick;
    ReceiveC2dAsync();
}
```

4. Test program: Use Device Explorer to send message



More on Cloud to Device Message

- Tutorial: How to send cloud-to-device messages with IoT Hub

https://azure.microsoft.com/en-us/documentation/articles/iot-hub-csharp-csharp-c2d/

- CONNECTING TO THE AZURE IOT HUB USING AN AMQP STACK

https://paolopatierno.wordpress.com/2015/10/24/connecting-to-the-azure-iot-hub-using-an-the-amqp-stack/