## 說明

* 本節會透過Web Job專案撰寫一隻程式，此程式會監控指定的Service Bus Queue中是否有訊息，如果有，則將此訊息取回並發送命令給指定的設備。

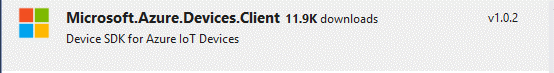
## 注意

* Service Bus Queue中的訊息須為Json格式，並包含DeviceID欄位表示設備代號。

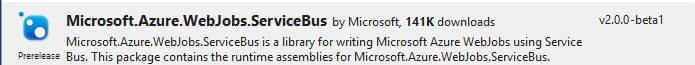
## 步驟

* 新增一個WebJob專案，並加入以下的NUGet參考

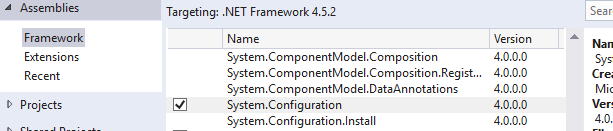








* 加入以下參考



* 在App.config加入Storage Account連線字串，用來儲存Azure Web Job Dashboard與Log資訊
* <connectionStrings>
* <!-- The format of the connection string is "DefaultEndpointsProtocol=https;AccountName=NAME;AccountKey=KEY" -->
* <!-- For local execution, the value can be set either in this config file or through environment variables -->
* <add name="AzureWebJobsDashboard" connectionString="｛Storage for Dashboard｝" />
* <add name="AzureWebJobsStorage" connectionString="{Storage for Log}" />
* <add name="AzureWebJobsServiceBus" connectionString="{Alarm Service Bus Queue connection string}" />
* </connectionStrings>
* 修改Functions.cs如下；其中
  + iotHubOwnerConnectionString為IOT Hub的Owner權限之連線字串

using System;

using System.Collections.Generic;

using System.IO;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using Microsoft.Azure.WebJobs;

using Microsoft.Azure.Devices.Client;

using Microsoft.Azure.Devices;

using System.Configuration;

using Newtonsoft.Json;

using Microsoft.ServiceBus.Messaging;

namespace c2d\_alarm

{

public class Functions

{

// This function will get triggered/executed when a new message is written

// on an Azure Queue called queue.

public static void ProcessQueueMessage([ServiceBusTrigger("c2dalarm")] BrokeredMessage message, TextWriter log)

{

var text = GetMessagePayload(message);

dynamic obj = JsonConvert.DeserializeObject(text);

var registryMgr = RegistryManager.CreateFromConnectionString(ConfigurationManager.AppSettings["iotHubConnectionString"]);

try

{

var device = registryMgr.GetDeviceAsync((string)obj.DeviceId).GetAwaiter().GetResult();

var deviceConnString = $"HostName={ConfigurationManager.AppSettings["iotHubUri"]};DeviceId={device.Id};SharedAccessKey={device.Authentication.SymmetricKey.PrimaryKey}";

var dc = DeviceClient.CreateFromConnectionString(deviceConnString);

ServiceClient sc = ServiceClient.CreateFromConnectionString(ConfigurationManager.AppSettings["iotHubConnectionString"]);

sc.SendAsync(device.Id,

new Microsoft.Azure.Devices.Message(Encoding.UTF8.GetBytes($"Command:{text}"))).Wait();

sc.CloseAsync().Wait();

message.Complete();

}

catch (Exception exp)

{

System.Diagnostics.Debug.WriteLine($"Exception:{exp.Message}");

}

}

private static string GetMessagePayload(BrokeredMessage message)

{

string ret = string.Empty;

//try

//{

// var o = message.GetBody<System.IO.Stream>();

// using (var r = new StreamReader(o))

// {

// ret = r.ReadToEnd();

// }

//}

//catch

//{

// ret = message.GetBody<string>();

//}

//return ret;

if (message.Properties.Keys.Contains("message-source") && (string)message.Properties["message-source"] == "evh")

{

var o = message.GetBody<System.IO.Stream>();

using (var r = new StreamReader(o))

{

ret = r.ReadToEnd();

}

}

else

{

ret = message.GetBody<string>();

}

return ret;

}

}

}

* 將WebJob佈署到Web Host中並執行