## 環境準備

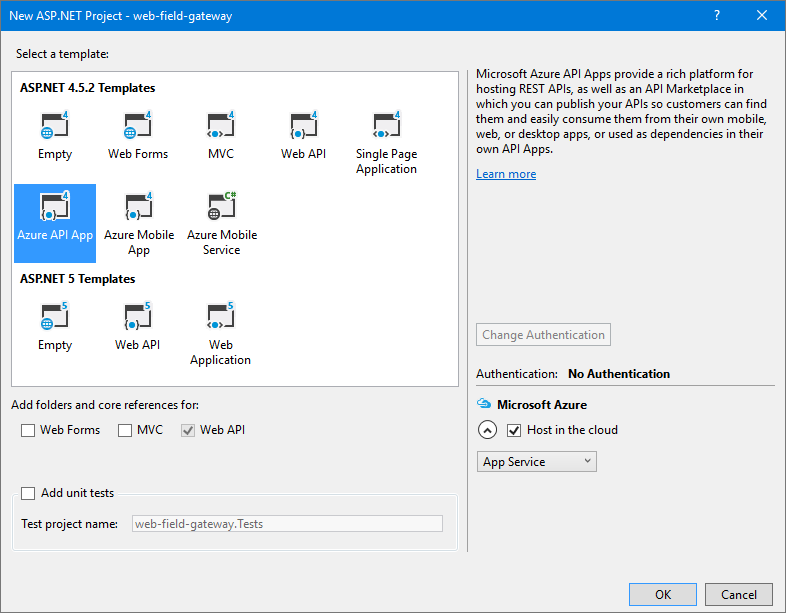
* Visual Studio 2015
* Azure SDK：<http://go.microsoft.com/fwlink/?linkid=518003&clcid=0x404>
* 已經建立Azure IOT Hub

## 說明

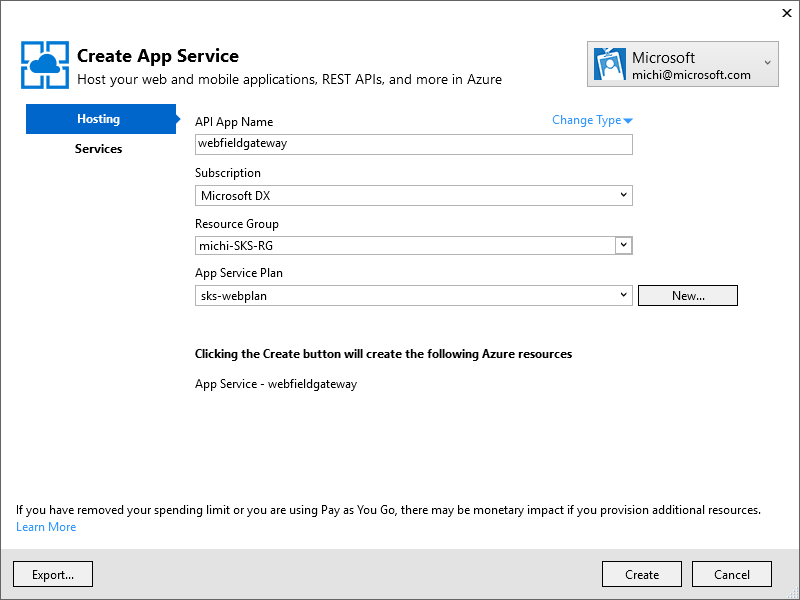
* 在這個Lab我們會建立一個ASP.Net MVC網站模擬Field Gateway
  + 請注意，實際上的Field Gateway可能不是以此形式存在；這個Lab僅僅是為了模擬SKS device連線到Gateway傳送訊息
* 由於Device會透過Field Gateway將資訊傳遞到IOT Hub；在Field Gateway必須處理Device authentication、Command Pattern(如果需要)等等；為求簡化，這個Lab並未處理這些部分。

## 步驟

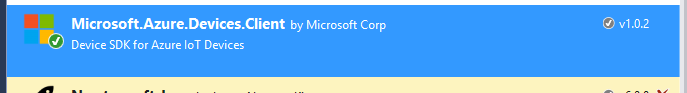
* 開啟Visual Studio
* 建立Web專案，選擇Azure API App



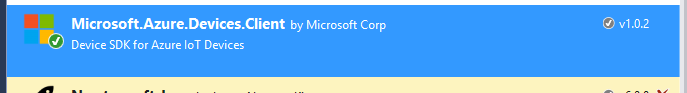
* 輸入發布資訊，之後按下確定開始建立專案



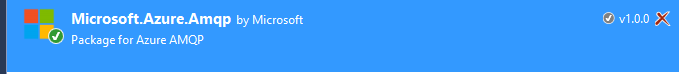
* 加入以下的NUGet參考
* Microsoft.Azure.Devices.Client



* Microsoft.Azure.Devices



* Microsoft.Azure.Amqp



* 打開APP\_Start下的SwaggerConfig.cs；找到以下這一段

/\*

})

.EnableSwaggerUi(c =>

{

\*/

* 將他Unmark，程式碼看起來應該像這樣

//c.ResolveConflictingActions(apiDescriptions => apiDescriptions.First());

// \*\*\*\*\* Uncomment the following to enable the swagger UI \*\*\*\*\*

})

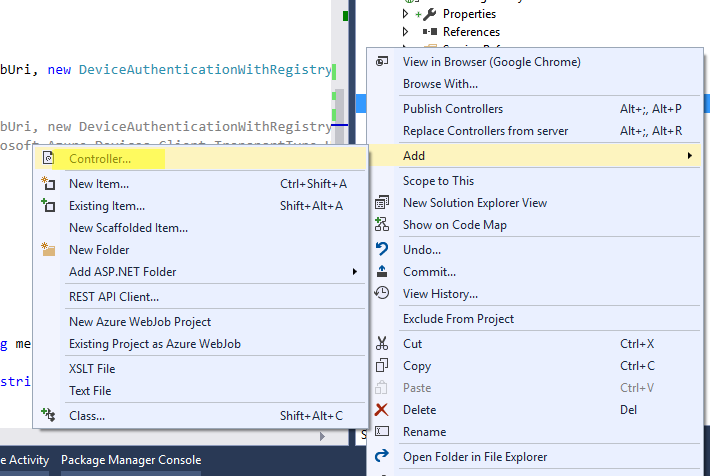
.EnableSwaggerUi(c =>

{

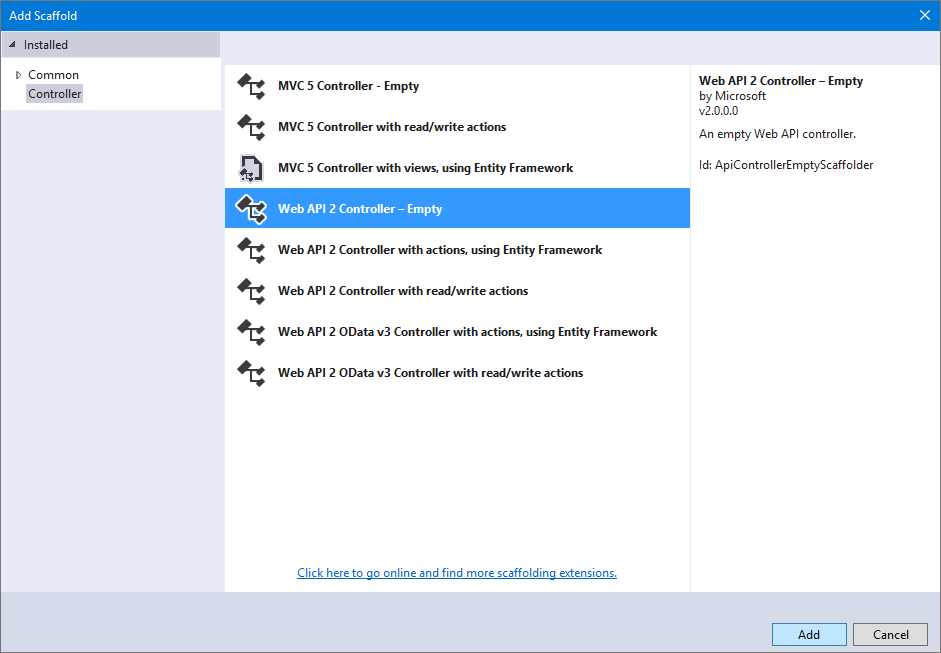
* 打開APP\_START\WebApiConfig.cs，修改routeTemplate如下

routeTemplate: "api/{controller}/{action}/{id}",

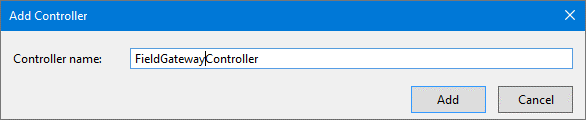
* 新增一個新的Controller



* 選擇Web API2



* 命名為FieldGatewayController



* 在FieldGatewayControll類別中加入下列宣告

static string connectionString = "<your IOT Hub connection string>";

static string iotHubUri = "<iot hub url>";

* 加入以下的namespace

using System.Threading.Tasks;

using System.Web.Http;

using Microsoft.Azure.Devices;

using Microsoft.Azure.Devices.Client.Exceptions;

using System.Web;

using web\_field\_gateway.Models;

using Microsoft.Azure.Devices.Client;

using Newtonsoft.Json;

using System.Text;

using System.IO;

* 加入Register()如下

public async Task<string> Register(string deviceId)

{

Device device = null;

RegistryManager registryManager = RegistryManager.CreateFromConnectionString(connectionString);

try {

device = await registryManager.AddDeviceAsync(

new Device(deviceId)

);

}

catch (DeviceAlreadyExistsException)

{

device = await registryManager.GetDeviceAsync(deviceId);

}

if(device != null)

{

var deviceKey =device.Authentication.SymmetricKey.PrimaryKey;

SaveDeviceIdentity(deviceId, deviceKey);

return deviceKey;

}

return string.Empty;

}

* 加入方法存取local store中的DeviceKey

private void SaveDeviceIdentity(string deviceId, string deviceKey)

{

var fn = Path.Combine(HttpContext.Current.Server.MapPath("~/APP\_DATA"),

deviceId);

File.WriteAllText(fn, deviceKey);

}

private string GetDeviceKey(string deviceId)

{

var fn = Path.Combine(HttpContext.Current.Server.MapPath("~/APP\_DATA"),

deviceId);

if (File.Exists(fn))

{

return File.ReadAllText(fn);

}

else

{

return string.Empty;

}

}

private void DeleteDeviceKey(string deviceId)

{

var fn = Path.Combine(HttpContext.Current.Server.MapPath("~/APP\_DATA"),

deviceId);

if (File.Exists(fn))

{

File.Delete(fn);

}

}

* 加入Unregister()方法

public async Task Unregister(string deviceId)

{

RegistryManager registryManager = RegistryManager.CreateFromConnectionString(connectionString);

var device = await registryManager.GetDeviceAsync(deviceId);

await registryManager.RemoveDeviceAsync(device);

}

* 在Models目錄中新增一個TelemetryData.cs，內容如下

//流水號,timestamp,類別,主機號,UID,DC/AC,ADSL/3G,Msg

public class TelemetryData

{

static Random random = new System.Random();

public string SeqNo { get; set; }

public DateTime Timestamp { get; set; }

public TelemetryTypes Type { get; set; }

public string DeviceId { get; set; }

public string UID { get; set; }

public DCAC DCorAC { get; set; }

public string ADSLor3G { get; set; }

public string Message { get; set; }

public static TelemetryData Random(string deviceID,string seqNo, string msg)

{

var ret = new TelemetryData()

{

SeqNo = seqNo,

Timestamp = DateTime.UtcNow,

Type = (TelemetryTypes)random.Next(0, 2),

DeviceId = deviceID,

UID = "UID-" + Guid.NewGuid().ToString(),

DCorAC = (DCAC)random.Next(0, 1),

ADSLor3G = random.Next(100) >= 50 ? "ADSL" : "3G",

Message = msg

};

return ret;

}

}

public enum TelemetryTypes

{

A= 0,B= 1,C = 2

}

public enum DCAC

{

DC = 0,AC = 1

}

* 新增以下的method¸前端程式(Sensor)會透過這個方法傳遞資料到IOT Hub

public async Task SendTelemetry(TelemetryData telemetry)

{

DeviceClient dc = DeviceClient.Create(iotHubUri,

new DeviceAuthenticationWithRegistrySymmetricKey(

telemetry.DeviceId,

GetDeviceKey(telemetry.DeviceId)

));

var text = JsonConvert.SerializeObject(telemetry);

var buffer = Encoding.UTF8.GetBytes(text);

await dc.SendEventAsync(new Microsoft.Azure.Devices.Client.Message(buffer));

}

* 新增以下的method¸前端程式(Sensor)會透過這個方法接收雲端送給裝置的命令；在這個Workshop中，當雲端傳送命令給前端時，Field Gateway會將收到的指令存成檔案放在APP\_DATA目錄下。

[HttpGet]

public HttpResponseMessage ReceiveCommand([FromUri]string deviceId)

{

try {

var fn = HttpContext.Current.Server.MapPath($"~/App\_Data/{deviceId}.txt");

if (File.Exists(fn))

{

var text = File.ReadAllText(fn);

Trace.WriteLine($"{fn}::{text}");

File.Delete(fn);

return Request.CreateResponse<string>(HttpStatusCode.OK, text);

}

else

{

System.Diagnostics.Trace.WriteLine($"Receive File {fn}...Not exists");

return Request.CreateResponse(HttpStatusCode.OK);

}

}catch(Exception exp)

{

return Request.CreateErrorResponse(HttpStatusCode.InternalServerError, exp.Message);

}

}

* Build Solution並且執行
* 瀏覽到[http://localhost:<port>/swagger](http://localhost:%3cport%3e/swagger)進行測試

