2015 Global Azure BOOTCAMP Saturday, April 25, 2015

Hands on Lab

HOL- 6: Adjson - Arduino

In this lab you will run some IOT apps on an Arduino, or Arduino compatible device to post and get some simple telemetry data.

Development Environment

It is assumed that you have an Arduino device with the Ethernet Shield and the Arduino development environment. If you have an 86Duino device and its Arduino development environment you can use that. Whilst you can post some synthetic telemetry data, if you have one the sensors as listed below then you can post actual sensor data. You can modify the program for other sensors quite easily.

Sensors

- Arduino Compatible Barometric and Temperature Sensor Module
  - Eg. http://www.freetronics.com.au/products/barometric-pressure-sensor-module#.VTgEtnkcTIU
- Arduino Compatible Humidity and Temperature Sensor Module
  - Eg http://www.freetronics.com.au/products/humidity-and-temperature-sensor-module#.VTgET3kcTIU

Documentation on the Freetronics site gives wiring etc. details for the sensors.

This lab uses four Arduino projects/apps. One for each of:

- POST Telemetry data: **JsonPostNewTelemetrySensorVersion2.0**
- GET Telemetry data: **JsonParserGetTelemetrySensorValuesVersion2.00**
- PATCH Telemetry Data: **JSONUpdateSensorValue**
- DELETE Telemetry Data: **JSONDeleteSensorValue**

These can be downloaded from the Ardjson Codeplex site: Link

POST Telemetry Data

1. You will use the Ethernet library.
2. Open the POST project in the Arduino or compatible development environment.
3. Connect your via USB device and set the board type Tools→Board
4. Check that your serial port to the device is recognized. Tools→Port
5. Modify the lines to match your AzMS:

```
#define TABLE         "telemetry2"

#define AMW_SERVICE   "sportronicsdj.azure-mobile.net"

#define APP_KEY        "NtcMLPQtuAqWtvXOwrZVQtqHevNUnN27"
```

6. If using a sensor:
    - Set the #define USINGSENSOR to true if using a sensor.
    - The app is configured for the Barometric Sensor
    - If using the Humidity one then
        i. Find the following lines in loop() and swap the commenting

```
ReadBaroSensor();
//ReadDHT();
```

        ii. Find the following lines in SensorsSetup () and swap the commenting:

```
BaroSensor.begin();
//dht.begin();
```

7. Test build the app. Sketch→Verify/Compile
8. Deploy and run the app.
9. Open the serial port and observe the dialog.
10. Examine the table in the Azure Portal after refreshing it.

GET the Telemetry Data

1. Open the GET project.
2. Modify those same 3 lines for your AzMS table
3. Test build the app.
4. Deploy the app.
5. Open the serial port and observe the dialog.
6. Note the id of at least one record.
7. You can easily crash this by having too many records in your table. If this happens, delete records via the Azure portal and add some less values.

Update Telemetry Data

1. Open the PATCH project.
2. Modify those same 3 lines for your AzMS table
3. Set the id to one of the records as observed in GET.
4. Test build the app.
5. Deploy the app.
6. Open the serial port and observe the dialog.
7. Run the GET app again to see the modification or examine the table in the Azure Portal (Refresh it)

Deleting a Record

1. Open the PATCH project.
2. Modify those same 3 lines for your AzMS table
3. Set the id to one of the records as observed in GET.
4. Test build the app.
5. Deploy the app.
6. Open the serial port and observe the dialog
    - ***Issue:*** The app runs as soon as it is downloaded. Opening the serial port terminal just runs it a second time. You could use, say id= 3, go into the portal and truncate the table (clear it) then

using the desktop app or phone add three records. Then reset the board and observe it being deleted.

7. Run the GET app to observe that the record was deleted.
   OR
8. Instead of 6 and 7 just observe the change the Portal after the app is uploaded: Refresh the Portal.