Cloud-Connected Weather Station with the Arduino Yun & Temboo

Created by Marc-Olivier Schwartz


Last updated on 2023-08-24 04:13:37 PM EDT
# Table of Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Introduction</td>
<td>3</td>
</tr>
<tr>
<td>Connections</td>
<td>4</td>
</tr>
<tr>
<td>Setting up your Temboo account</td>
<td>7</td>
</tr>
<tr>
<td>Arduino sketch</td>
<td>8</td>
</tr>
<tr>
<td>Using the project</td>
<td>9</td>
</tr>
</tbody>
</table>
Introduction

The Temboo service has moved away from Arduino support. The service access in this older guide will no longer work.

In this article, you are going to build a weather measurement station that will automatically send data to the cloud using the Temboo service. When building home automation projects, we love to connect our devices together wirelessly and to the web, especially using WiFi. One solution is to use WiFi breakout boards or shields, for example using the CC3000 WiFi chip, connect it to an Arduino board, and build your own code for the WiFi communication & remote interface. In this article, I will show you another approach using the Arduino Yun & Temboo.

Introduced in 2013 by Arduino, the Arduino Yun is a powerful Arduino board with onboard WiFi & a small Linux machine. Especially, it is really easy to interface with the web service Temboo, which is what we are going to use in this project. We will automatically send data to a Google Docs spreadsheet, which can be accessed from anywhere.
Connections

For this project, you will of course need the Arduino Yun board. You will also need a DHT11 (or DHT22) sensor, along with a 4.7K resistor, for humidity measurements. For pressure & temperature measurements, I used a BMP085 sensor on a simple breakout board, but you can also use the newer Adafruit BMP180 sensor board, which works with the same library. For light levels measurements, I used a photocell with a 10K Ohm resistor. Finally, I used a breadboard and some male-male jumper wires.

On the software side, you will need the Arduino IDE in the latest version (in Beta when this article was written). You will also need the DHT library, the BMP085 or BMP180 library, and the unified sensor library. To install a library, simply put the folder in your /libraries/ folder of your main Arduino folder. This tutorial also assumes that your Arduino Yun is already connected to your WiFi network. If you need help with that, please follow the dedicated tutorials on the Arduino website.

You will also need a Google account for the rest of the project. If you don’t have one yet, please create one, for example by going on the Google Drive page.
The hardware connections for this project are actually quite simple: we have to connect the DHT11 sensor, and then the part responsible for the light level measurement with the photocell. First, connect the Arduino Yun +5V pin to the red rail on the breadboard, and the ground pin to the blue rail.

Then, connect pin number 1 of the DHT11 sensor to the red rail on the breadboard, and pin number 4 the blue rail. Also connect pin number 2 to pin number 8 of the Arduino Yun. To finish up with the DHT11 sensor, connect the 4.7k Ohm between pin number 1 and 2 of the sensor.

For the photocell, first place the cell in series with the 10k Ohm resistor on the breadboard. Then, connect the other end of the photocell to the red rail on the breadboard, and the other end of the resistor to the ground. Finally, connect the common pin to the Arduino Yun analog pin A0.

For the BMP085 or BMP180 sensor, connect the VIN pin to the +5V, GND to Ground, SCL to Arduino Yun pin number 3, and SDA pin to Arduino Yun pin number 2. The following picture summarises the hardware connection:
Before connecting the project to the cloud, we'll test every sensors individually with a test sketch that simply uses the conventional Arduino part of the Yun. The first step in the code is to import the correct libraries:

```cpp
#include "DHT.h"
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP085_U.h>
```

We need to define the pin & type of the DHT sensor:

```cpp
#define DHTPIN 8
#define DHTTYPE DHT11
```

And create the instances for the DHT sensor & BMP sensor:

```cpp
DHT dht(DHTPIN, DHTTYPE);
Adafruit_BMP085_Unified bmp = Adafruit_BMP085_Unified(10085);
```

In the setup() function, we can initialise the BMP sensor:

```cpp
bmp.begin()
```

In the loop() part of the sketch, we make the different measurements, for example humidity, temperature & light level:

```cpp
// Measure the humidity
float humidity = dht.readHumidity();

// Measure light level
int lightLevel = analogRead(A0);

// Measure pressure & temperature from BMP sensor
sensors_event_t event;
bmp.getEvent(&event);
float pressure = event.pressure;
```
These different measurements will then be printed on the Serial monitor. Of course, you can find the complete sketch on our GitHub repository for this project. You can now upload the sketch to the board & open the Serial monitor. This is what you should see:

![Serial Monitor Output](image)

If that works, it means all your hardware connections are correct, and you can move to the next part of the project.

### Setting up your Temboo account

The Temboo service has moved away from Arduino support. The service access in this older guide will no longer work.

Before you can upload data to Google Docs, you need to have a Temboo account. Go over to the [Temboo website](https://temboo.com) and enter your email to start the account creation process:

Insert email address here for one **free** account.

**SIGN UP**

You will also be prompted to enter an account name:

**ACCOUNT NAME**

Your Account Name will be used in your code to call Choreos.

Enter Account Name here

After that step, you will be asked to give a name to your first app, and you will be given an API key for your app. Please keep at hand your account name, app name, and API key, you will need them soon. In case you forgot to write these down, you can
always go in "My account" and click on the "Manage" button next to "Applications" to get it back:

![Activity Dashboard]

You will also need to follow a procedure to create a Google application, and at the end get a Google client ID, client secret & access token. The Temboo website explains this procedure very well, and you can find the whole procedure at:

https://www.temboo.com/library/Library/Google/Spreadsheets/

Arduino sketch

The Temboo service has moved away from Arduino support. The service access in this older guide will no longer work.

Now, we are going to build the Arduino sketch. You need to import all the libraries for the Arduino Yun web & bridge functionalities. You can read more about starting out with Temboo & Yun here: https://www.temboo.com/arduino/yun/

These are all the libraries used in this project:

```cpp
#include <Bridge.h>
#include <Temboo.h>
#include <Process.h>
#include <Wire.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BMP085_U.h>
```

I also placed all my Temboo account information in a separate file, please make sure that you update your personal account information in this file:

```cpp
#include "TembooAccount.h"
```

You also need to set your own informations about your Google account:

```cpp
const String GOOGLE_CLIENT_ID = "your-google-client-id";
const String GOOGLE_CLIENT_SECRET = "your-google-client-secret";
const String GOOGLE_REFRESH_TOKEN = "your-google-refresh-token";
const String SPREADSHEET_TITLE = "your-spreadsheet-title";
```
In the setup() part of the sketch, we initialise the Arduino Bridge library:

```cpp
Bridge.begin();
```

We also start a date process, to automatically send the measurements date to Google Docs:

```cpp
time = millis();
if (!date.running()) {
    date.begin("date");
    date.addParameter("+%D-%T");
    date.run();
}
```

In the loop() part, the most important line is:

```cpp
runAppendRow(humidity, lightLevel, pressure, temperature, altitude);
```

Which calls the function to automatically send the data to the Google Docs spreadsheet. I won’t get into the details of these two functions, but of course you will find all the code in the GitHub repository of this project.

The loop() of the Arduino sketch is repeated every 10 minutes with a delay() function, as these variables are usually changing slowly over time.

### Using the project

It's now time to test our project. First, we'll do some configuration in Google Docs. Create a new spreadsheet in Google Docs, name it (I named mine "Yun"), and set the title of each columns in the first row, like the picture below:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Time</td>
<td>Humidity</td>
<td>Light level</td>
<td>Pressure</td>
<td>Temperature</td>
<td>Altitude</td>
</tr>
</tbody>
</table>

Finally, you are ready to test the project. Upload the code to the Arduino Yun board, open the Google Docs spreadsheet in your browser again, and wait a moment. After a while, the first measurement should appear:

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>03/11/14-06:47:56</td>
<td>34</td>
<td>653</td>
<td>1002.54</td>
<td>23.77</td>
<td>92.43</td>
</tr>
</tbody>
</table>

I also used the built-in plotting functions of Google Docs to plot my data as it comes in. For example, I used the simple "Line" chart type to plot the light level. This is the result over some hours in the morning, showing that the light level is slowly rising:
And over the same period of time, I plotted the evolution of the temperature & humidity:

The really nice thing about this is that it is updated automatically as data comes in, and it can be accessed from anywhere: you just need your Google username & password!

You just learned in this project how to send measurements from your Arduino Yun board to Google Docs, so this data can be accessed from anywhere. You also saw how to build automated email alerts based on the data in your home. You can of course improve this project in many ways. The first one is to add more sensors to the project, for example a wind speed sensor or a smoke sensor. You could even add an USB camera to store pictures on an SD card, using the Yun SD card reader.

You can of course use many Arduino Yun boards, for example in different parts of your home. You can also customise the email alert part: you can build more complex alerts based on the measured data, or set the project to email you the sensor data at a regular time interval. If you have other ideas to improve this project, please share!