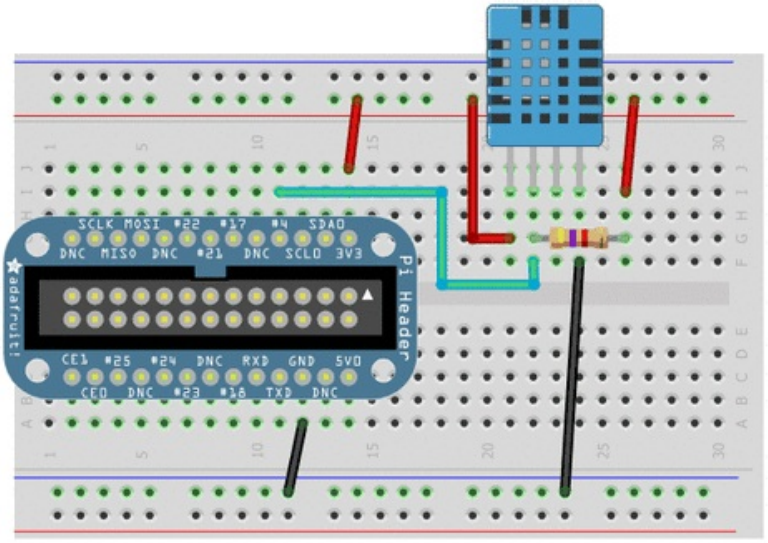
adafruit learning system

DHT Humidity Sensing on Raspberry Pi or Beaglebone Black with GDocs Logging

Created by lady ada

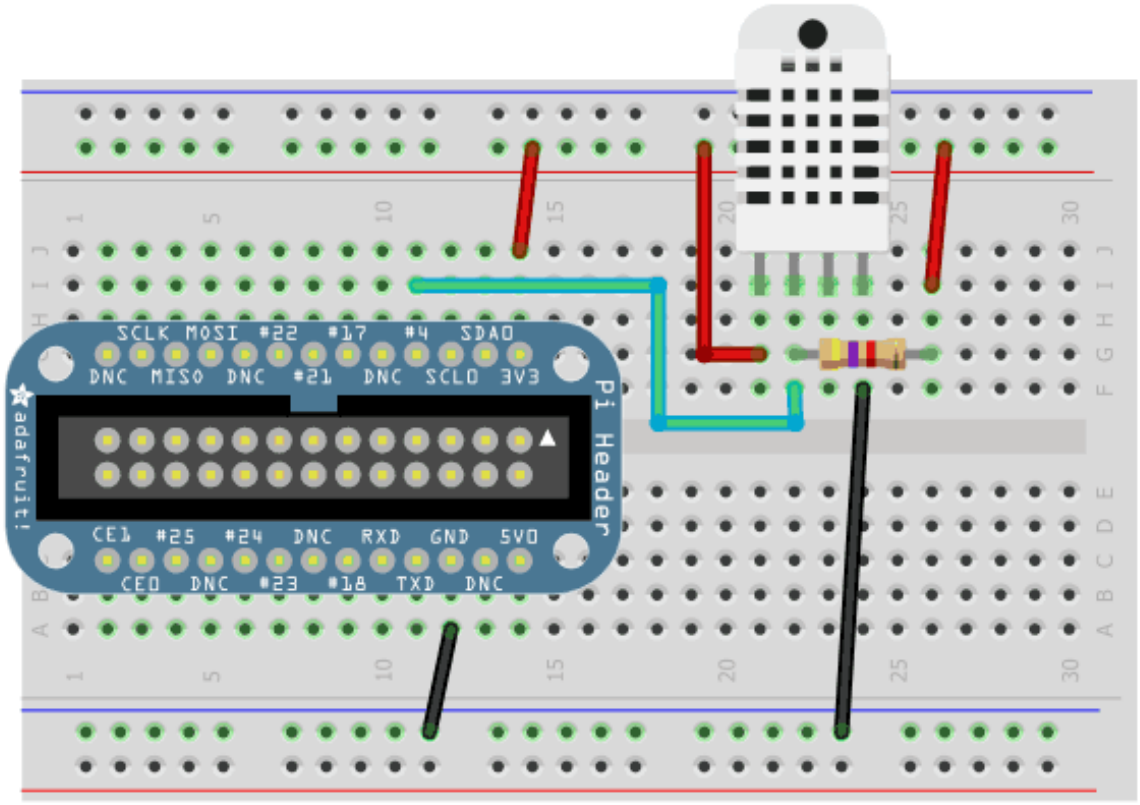


Last updated on 2015-04-26 02:30:10 PM EDT

Guide Contents

Overview

This tutorial is a first attempt to develop a DHT interface driver. It is not guaranteed to work, and is for experimentation and hacking!



Time to start exploring more sensors with the Raspberry Pi and Beaglebone Black! Today we'll be checking out the DHT11 (http://adafru.it/386), DHT22 (http://adafru.it/385) and AM2302 (http://adafru.it/393) humidity and temperature sensors available from Adafruit

In this tutorial we'll be showing how to install a DHT sensor Python library which utilizes C for high-speed GPIO polling to handle bit-banged sensor output. Many low cost sensors have unusual output formats, and in this case, a "Manchester-esque" output that is not SPI, I2C or 1-Wire compatible must be polled continuously by the Pi to decode. Luckily, the C GPIO libraries are fast enough to decode the output.

Once we have that working, we add the fun of Python to update a google spreadsheet live with the temperature/humidity data. This project would be the great basis for home or garden automation!

You can check out our spreadsheet here, it wont be updated live after Aug 24 2012 but it will show

you the format of data you get (http://adafru.it/aOU)

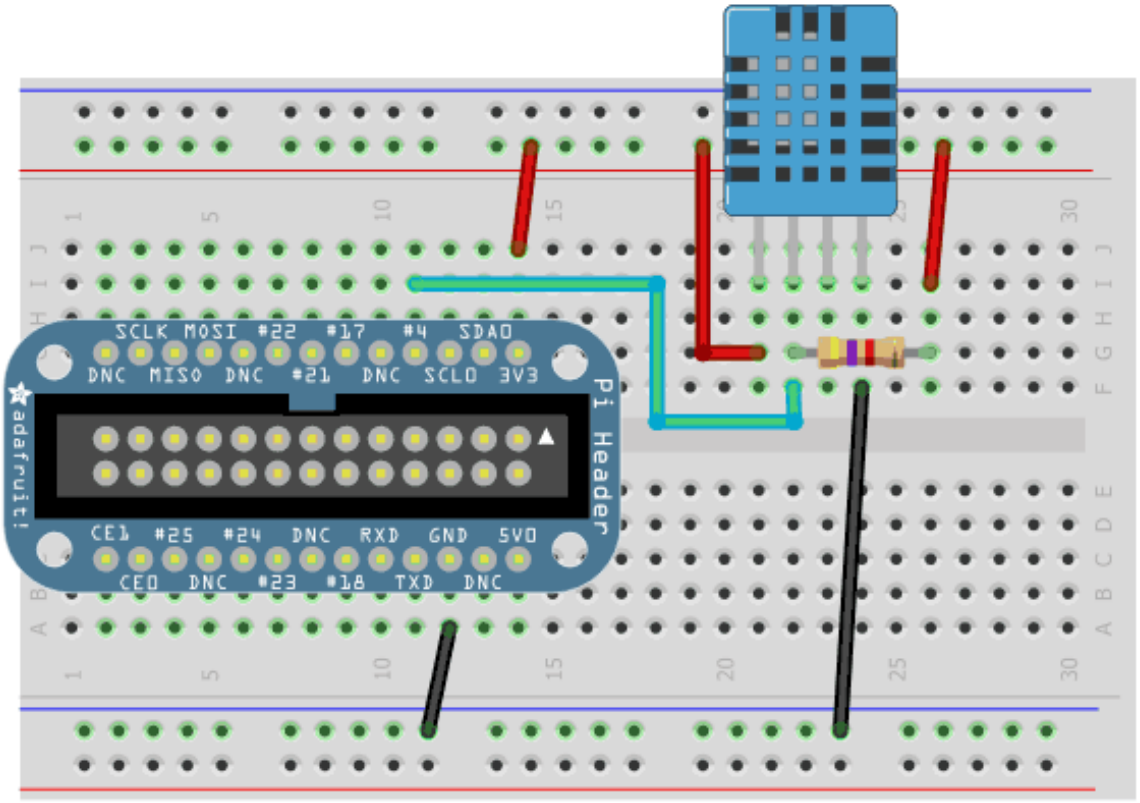
Wiring
Wiring up DHT humidity sensors

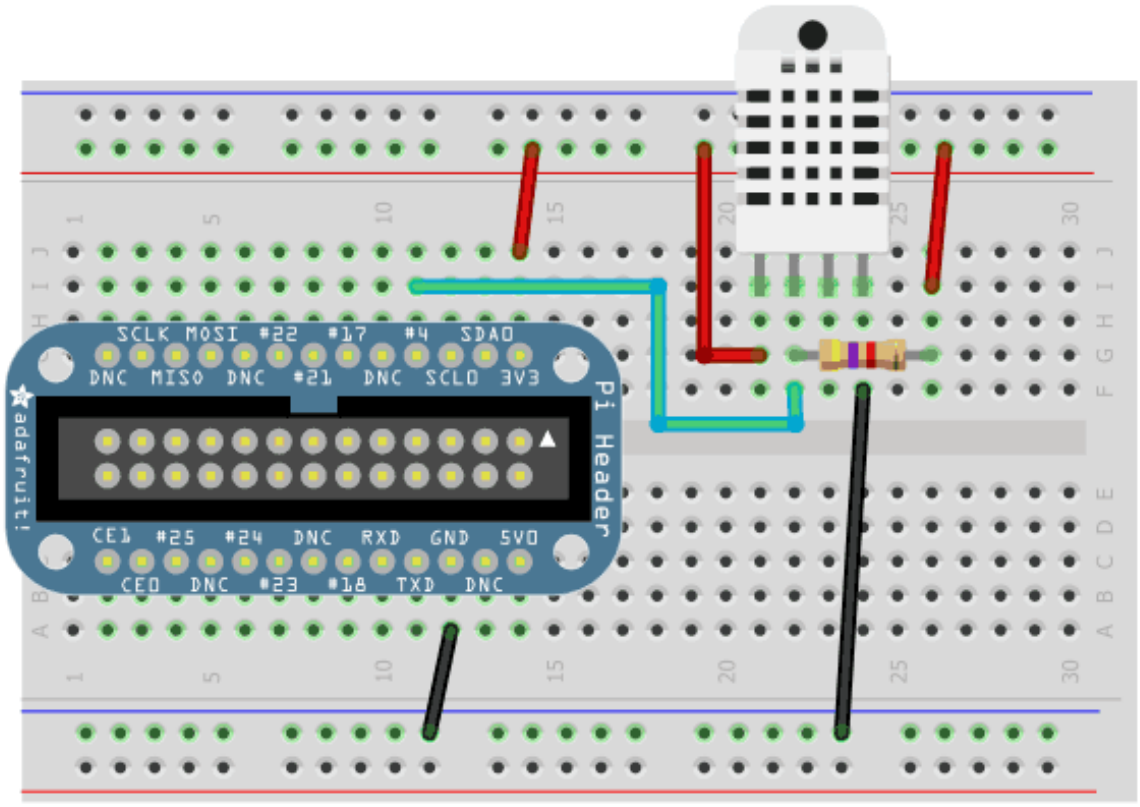
Raspberry Pi

Its easy to connect these sensors to your Raspberry Pi. Our code can use any GPIO pin, but we'll be using GPIO #4 for our diagrams and code. Once you have it working, you can simply adapt the code to change to any other GPIO pin (e.g. pin #**18**). You can also have as many DHT sensors as you want **but** they cannot share the data pin - each sensor needs a unique data pin!

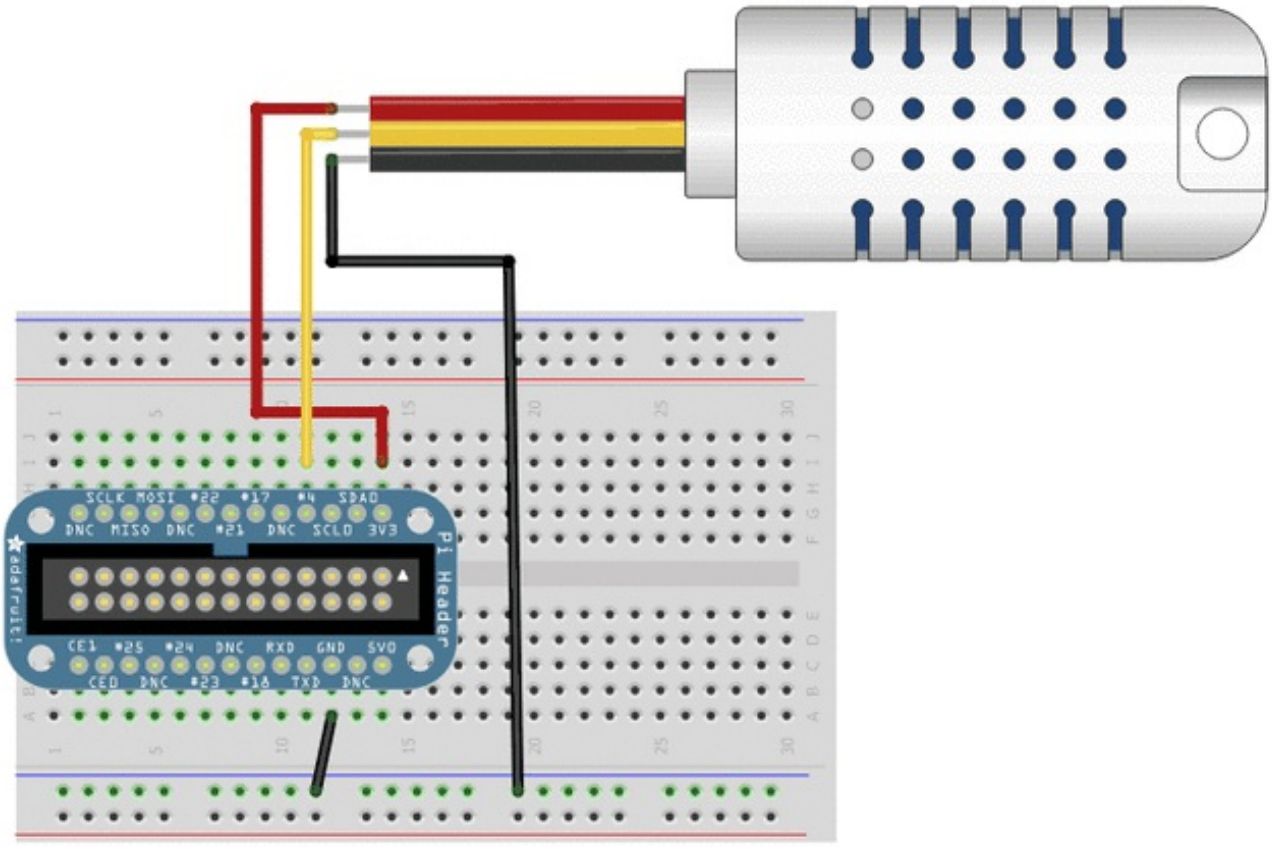
For DHT11 and DHT22 sensors, don't forget to connect a 4.7K - 10K resistor from the data pin to VCC

& if 4.7K doesnt work, try 10K





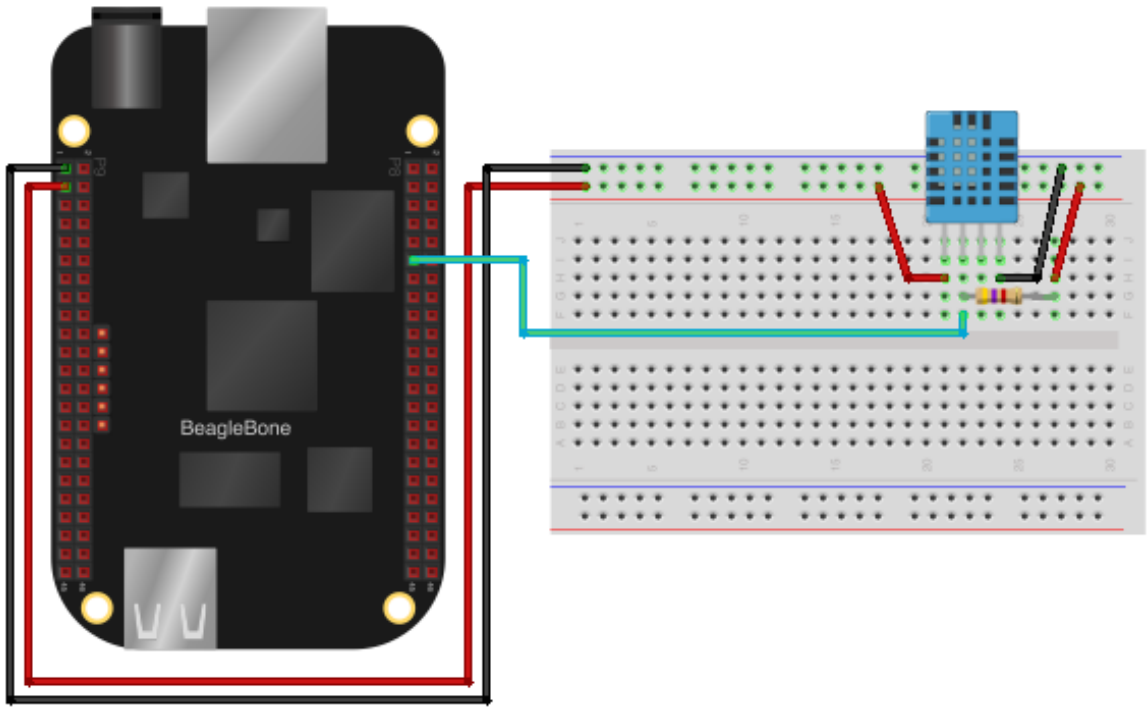
If your sensor has a white wire, leave it disconnected

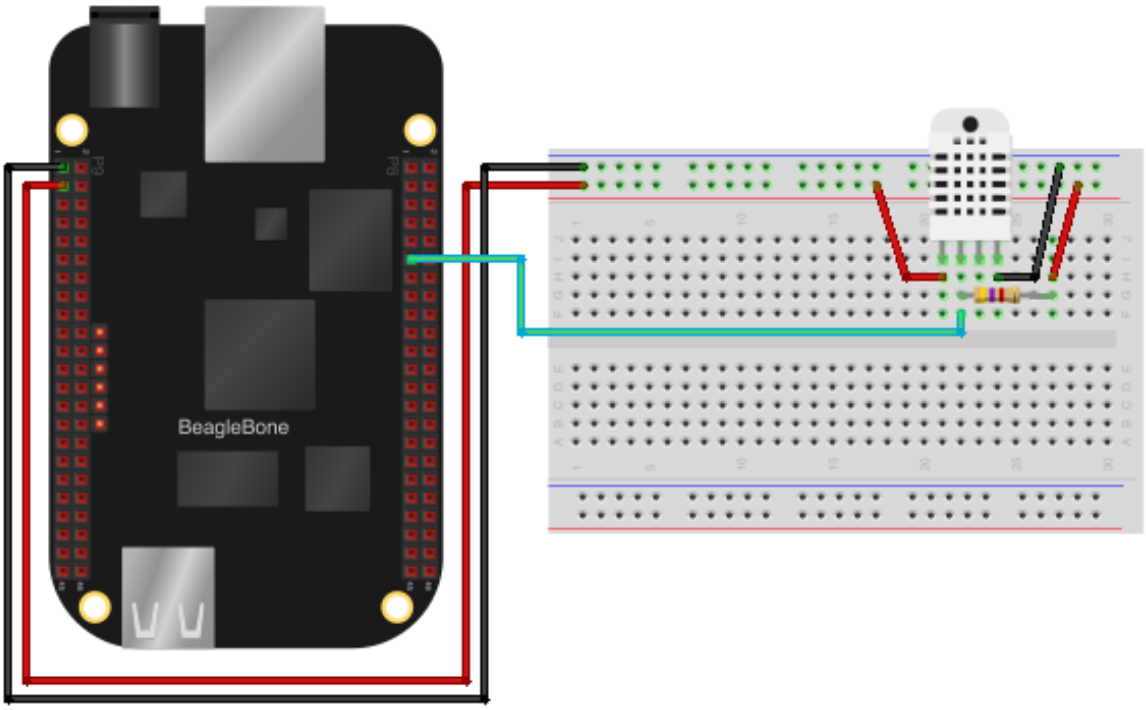


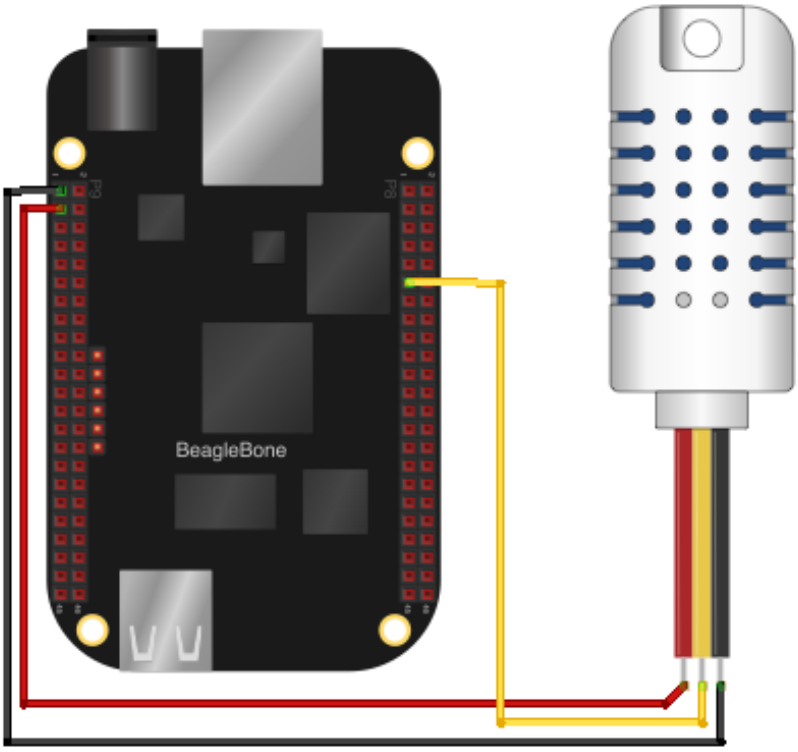
Beaglebone Black

Connecting a DHT sensor to the Beaglebone Black is just as simple as connecting to a Raspberry Pi. In this example pin P8_11 will be used, but you can use any free digital GPIO pin. If you aren't familiar with how pins are numbered on the Beaglebone Black, check out the tutorial on Beaglebone Black GPIO (http://adafru.it/dCI).

Don't forget to connect the 4.7k-10k resistor from the data pin to VCC, like with the Raspberry Pi wiring above.


BeagleBone


BeagleBone


BeagleBone

Software Install (Updated)

The Python and C code to work with Adafruit's DHT sensors is available on Github at https://github.com/adafruit/Adafruit_Python_DHT (http://adafru.it/dCJ).

We use some C code to talk to the DHT sensors since they require extremely fast timing to read, and then wrap the C code in a simple Python library for easy integration into your own programs.

(http://adafru.it/dCK)Downloading the Code from Github

The easiest way to get the code onto your Pi or Beaglebone Black is to hook up an Ethernet cable, and clone it directly using 'git', which is installed by default on most distros. Simply run the following commands from an appropriate location (ex. "/home/pi"):

```
git clone https://github.com/adafruit/Adafruit_Python_DHT.git
cd Adafruit_Python_DHT
```

Installing the Library

To install the Python library on either the Raspberry Pi or Beaglebone Black you will first need a few dependencies. Execute the following command to install these dependencies (assuming you're using Raspbian/Occidentalis (http://adafru.it/dpb) on the Pi and Debian (http://adafru.it/dvh) on the Beaglebone Black):

```
sudo apt-get update
sudo apt-get install build-essential python-dev
```

If you see an error that a package is already installed or at the latest version, don't worry you can ignore it and move on.

Next, to install the library execute:

```
sudo python setup.py install
```

This should compile the code for the library and install it on your device so any Python program can access the Adafruit_DHT python module.

Testing the Library

To test the Python library you can run some of the example programs in the examples folder. The AdafruitDHT.py example is a simple program which takes from the command line parameters the type of sensor (11, 22, or 2302) and GPIO pin connected to the sensor, and displays a single

reading from the sensor.

First navigate to the examples folder by executing:

```
cd examples
```

Now to run the example on a Raspberry Pi with an AM2302 sensor connected to GPIO #4, execute:

```
sudo ./AdafruitDHT.py 2302 4
```

Make sure to run the command as root with the sudo command as shown above or else the program will fail to run (root access is required for reading and writing the GPIO pins).

Alternatively, to run the example on a Beaglebone Black with a DHT22 sensor connected to pin P8_11, execute:

```
sudo ./AdafruitDHT.py 22 P8_11
```

Again make sure the command is run as root with sudo!

After the program executes you should see both the temperature and humidity displayed once. If you see an error that the sensor could not be read, double check you have the right GPIO pin connected to the data line of the DHT sensor and specified in the last parameter.

Note that sometimes you might see an error that the sensor can't be read and to try again, even if you have your connections setup correctly. This is a limitation of reading DHT sensors from Linux--there's no guarantee the program will be given enough priority and time by the Linux kernel to reliably read the sensor. If this occurs, run the program again (or call the read function again in your code) to try to get a new reading. In testing on both the Raspberry Pi & Beaglebone Black, about 75% of the read requests should generally succeed and return a result (assuming the board is not under heavy load).

Examine the source code for AdafruitDHT.py and simpletest.py to see simple examples of reading the DHT sensors from Python code.

C Language Code

If you'd like to access the DHT sensors using C/C++, you can use the C code that powers the Python library in your own program. Look at the **source/Raspberry_Pi** or **source/Beaglebone_Black** directory to see the C code for the Raspberry Pi or Beaglebone Black respectively. The code exposes a single function like **pi_dht_read** which performs the bitbang read of the DHT sensor.

Feel free to copy this code into your own program to read DHT sensors. Note that you will need to

include all the .c & .h files in the directory, and the common_dht_read.c & common_dht_read.h files in the parent source directory.

Connecting to Googles Docs (Updated)

Google sometimes will update their API and cause issues with the gspread library. Consult this thread for information on converting a spreadsheet to an old-style spreadsheet if you have problems accessing your sheet:

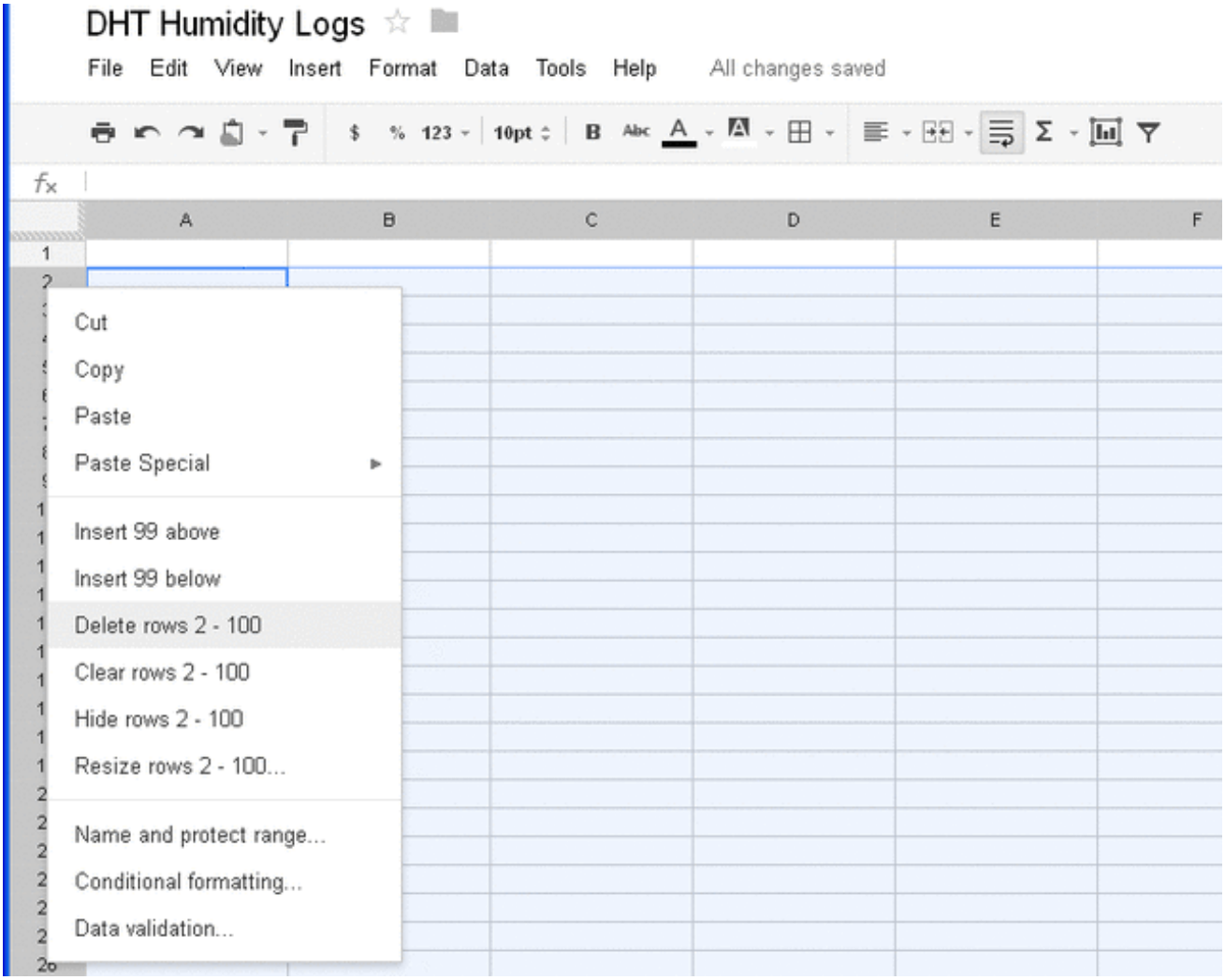
RE: DHT 22 Temperature and Humidity sensor with adafruit code (http://adafru.it/doW)

As of April 2015 Google has deprecated an old authentication interface for updating Google Sheets. You must carefully read the new steps below to make your Google Sheets work with the new OAuth2 authentication scheme.

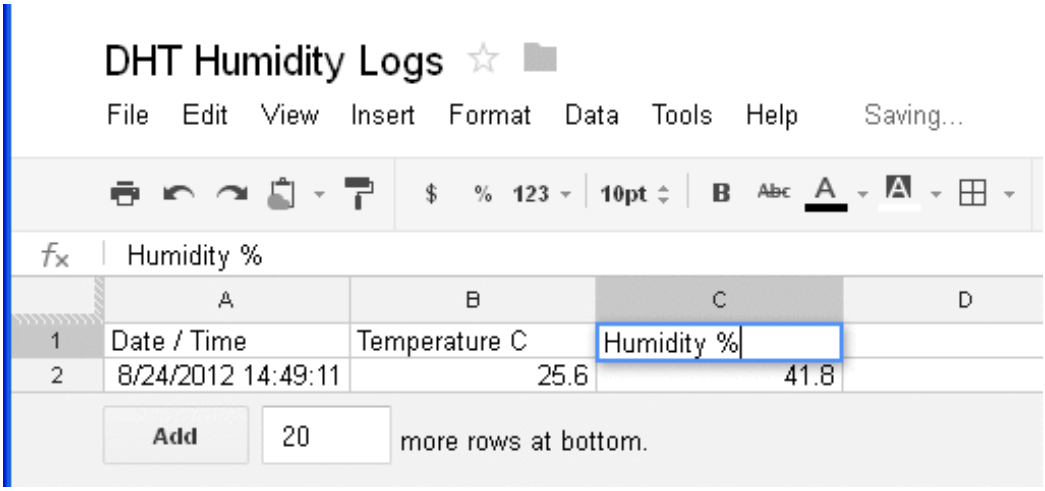
Create and prepare spreadsheet

First up you will need to sign up for Google Docs and create a spreadsheet. We're going to call ours DHT Humidity Logs.

Once you've created it, delete all but one line (since we don't want 1000 empty rows):



Then make the one remaining line a header with row names:



Get OAuth2 credentials

As of April 2015 Google has deprecated the older simple authentication interface for accessing

Google spreadsheet data. You must carefully follow the steps below to enable OAuth2 access to your Google spreadsheet. Unfortunately these steps are somewhat complex, so go through them very carefully to make sure you don't miss a step. If you run into problems try consulting the gspread python library (http://adafru.it/f2l) that this script uses.

To get your OAuth2 credentials follow the steps on this page:

- gspread - Using OAuth2 for Authorization (http://adafru.it/f2m)

After you follow the steps in the document above you should have downloaded a .json file, like SpreadsheetData-(gibberish).json. **Place this .json file in the same directory as the google_spreadsheet.py example.** If you don't place this file in the same directory then authentication will fail and you will not be able to update your spreadsheet!

One last step that **must be completed** is to share your Google spreadsheet to the email address associated with the OAuth2 credentials. Open the .json file and search for the **"client_email":** line that looks like this (but with a different email address):

```
"client_email": "149345334675-md0qff5f0kib41meu20f7d1habos3qcu@developer.gserviceaccount.com",
```

Take note of that email address value and go to your Google spreadsheet in a web browser. Using the **File -> Share...** menu item share the spreadsheet with **read and write access** to the email address found above. **Make sure to share your spreadsheet or you will not be able to update it with the script!**

Run Python Code

First up we will have to install the **gspread** python library, which will do the heavy lifting of connecting to google docs and updating the spreadsheet! With your board connected and online, run the following:

```
sudo apt-get update
sudo apt-get install python-pip
sudo pip install gspread oauth2client
```

Next, in the **examples** directory again, edit **google_spreadsheet.py** and adjust the configuration values towards the top of the file:

```
# Type of sensor, can be Adafruit_DHT.DHT11, Adafruit_DHT.DHT22, or Adafruit_DHT.AM2302.
DHT_TYPE = Adafruit_DHT.DHT22

# Example of sensor connected to Raspberry Pi pin 23
DHT_PIN  = 23
# Example of sensor connected to Beaglebone Black pin P8_11
#DHT_PIN  = 'P8_11'

# Google Docs OAuth credential JSON file.  Note that the process for authenticating
# ...
GDOCS_OAUTH_JSON       = 'your SpreadsheetData-*.json file name'

# Google Docs spreadsheet name.
GDOCS_SPREADSHEET_NAME = 'your google docs spreadsheet name'
```

Make sure **DHT_TYPE** is set to the type of sensor you are using (either **Adafruit_DHT.DHT11**, **Adafruit_DHT.DHT22**, or **Adafruit_DHT.AM2302**), and **DHT_PIN** is set to the GPIO pin number which is connected to your DHT sensor.

In the example above a Raspberry Pi GPIO pin #23 is shown, however commented below it is an example of a Beaglebone Black using GPIO pin P8_11.

Next make sure to set the **GDOCS_OAUTH_JSON** to the name of the SpreadsheetData-*.json file in the same directory as the google_spreadsheet.py file. If you don't have a SpreadsheetData-*.json file then you accidentally missed the steps above. **Go back and carefully follow the OAuth2 credential steps (http://adafru.it/f2n) to get an OAuth2 credential .json file before continuing!**

Finally set **GDOCS_SPREADSHEET_NAME** to the name of your spreadsheet, like 'DHT Humidity Logs'.

Save the file and execute the Python script by running:

```
sudo ./google_spreadsheet.py
```

You should see the program run and after about 30 seconds a humidity and temperature measurement is displayed and written to the spreadsheet. The program will continue to run and log a measurement every 30 seconds until you force it to quit by pressing **Ctrl-C**.

The measurement frequency can be adjusted by changing the **FREQUENCY_SECONDS** configuration in the python code.

Open the spreadsheet on Google's site and you should see measurements added in real time!

You can also see our spreadsheet here, it wont be running live after Aug 24, 2012 but it gives you an idea of the data format (http://adafru.it/aOU)