GoPiGo3 Cloud Data with ThingSpeak

Contents

ThingSpeak Sensor Data Example

http://www.billthecomputerguy.com/gopigo

Create ThingSpeak Account

ThingSpeak.com is a free cloud service that can be used to collect and display data from the GoPiGo3. You can create a maximum of 4 channels with 8 data fields per channel.

1. Go to www.thingspeak.com Create a free account.
2. Go to **My Profile**. Edit and change your time zone to your local time zone.

Setup ThingSpeak Channel

1. Logon to your ThinkSpeak account.
2. Click **New Channel** to create a new channel. Give it a name.
3. Field 1: **Distance Sensor** Click **Save Channel**.
4. Click the **API Keys** tab. Copy the **Write API Key**. We will use this key to upload data to this channel.

Create ThingSpeak Python Program

We are going to upload Distance Sensor Data to our ThingSpeak channel.

Create the following file to hold your **Write API Key** for the channel you are using. You can keep multiple API keys in this file, just be sure to give each one a different name.

```
# thingspeak_api_key.py
# ThingSpeak channel write api keys

THINGSPEAK_API_KEY = 'insert your api key here'
```

```python
#!/usr/bin/env python3
"""
    Name:    thingspeak_distance_sensor.py
    Author:  William A Loring
    Created: 10/17/21 Revised:
    Purpose: Example of uploading data to a ThingSpeak Channel
"""
# This uses the EasyGoPiGo3 library
# https://gopigo3.readthedocs.io/en/master/api-basic/easygopigo3.html#easygopigo3

# Import the time library for the sleep function
import time
import requests
from easygopigo3 import EasyGoPiGo3  # Import GoPiGo3 library

# Substitute your api key in this file for updating your ThingSpeak channel
import thingspeak_api_key
TS_KEY = thingspeak_api_key.THINGSPEAK_API_KEY

# ThingSpeak data dictionary
ts_data = {}    # Thingspeak data dictionary

# Create an instance of the GoPiGo3 class
gpg = EasyGoPiGo3()

# Initialize a Distance Sensor object
my_distance_sensor = gpg.init_distance_sensor()


def main():
    while True:
        # ================================================================
        # field1: Read the distance sensor data into millimeters
        mm = str(my_distance_sensor.read_mm())

        # ================================================================
        # field2: Read the distance sensor data into inches
        inches = str(my_distance_sensor.read_inches())

        # Print the values of the sensor to the console for debugging
        print("Distance Sensor Reading: " + inches + " inches " + mm + " mm")

        # Send sensor data to ThingSpeak
        thingspeak_send(mm, inches)

        # 15 seconds is the minimum amount of time between uploads
        # Sleep is set to 15 seconds for testing purposes
        time.sleep(15)


```

```
51 def thingspeak_send(mm, inches):
52     """
53         Update the ThingSpeak channel using the requests library
54     """
55     print("Update Thingspeak Channel")
56 
57     # Each field number corresponds to a field in ThingSpeak
58     params = {
59         "api_key": TS_KEY,
60         "field1": mm,
61         "field2": inches
62     }
63 
64     # Update data on Thingspeak
65     ts_update = requests.get(
66         "https://api.thingspeak.com/update", params=params)
67 
68     # Was the update successful?
69     if ts_update.status_code == requests.codes.ok:
70         print("Data Received!")
71     else:
72         print("Error Code: " + str(ts_update.status_code))
73 
74     # Print ThngSpeak response to console
75     # ts_update.text is the thingspeak data entry number in the channel
76     print("ThingSpeak Channel Entry: " + ts_update.text)
77 
78 
79 # If a standalone program, call the main function
80 # Else, use as a module
81 if __name__ == '__main__':
82     main()
```

Upload the Sensor Data

Run the program. Move the GoPiGo around by hand or by a remote control program. Go to your ThingSpeak channel. Your data should show up almost immediately.

What's Next?

Work with other sensors to read and upload the data.