

<https://www.halvorsen.blog>



Arduino and ThingSpeak

Hans-Petter Halvorsen

Table of Contents

- Arduino
- ThingSpeak
- Arduino + ThingSpeak
- Arduino Example
 - Write Data to ThingSpeak using a TMP36 Temperature Sensor

<https://www.halvorsen.blog>



Arduino

Hans-Petter Halvorsen

[Table of Contents](#)

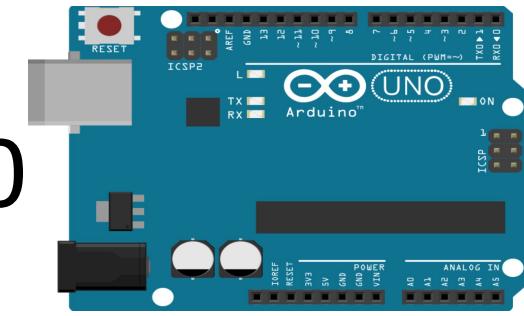
Arduino

- Arduino is an open-source electronics platform based on easy-to-use hardware and software.
- It's intended for anyone making interactive projects, from kids to grown-ups.
- You can connect different Sensors, like Temperature, etc.
- It is used a lots in Internet of Things projects
- Homepage:

<https://www.arduino.cc>

Arduino

- Arduino is a Microcontroller
- Arduino is an open-source platform with Input/Output Pins (Digital In/Out, Analog In and PWM)
- Price about \$20
- Arduino Starter Kit ~\$40-80
 - with Cables, Wires, Resistors, Sensors, etc.

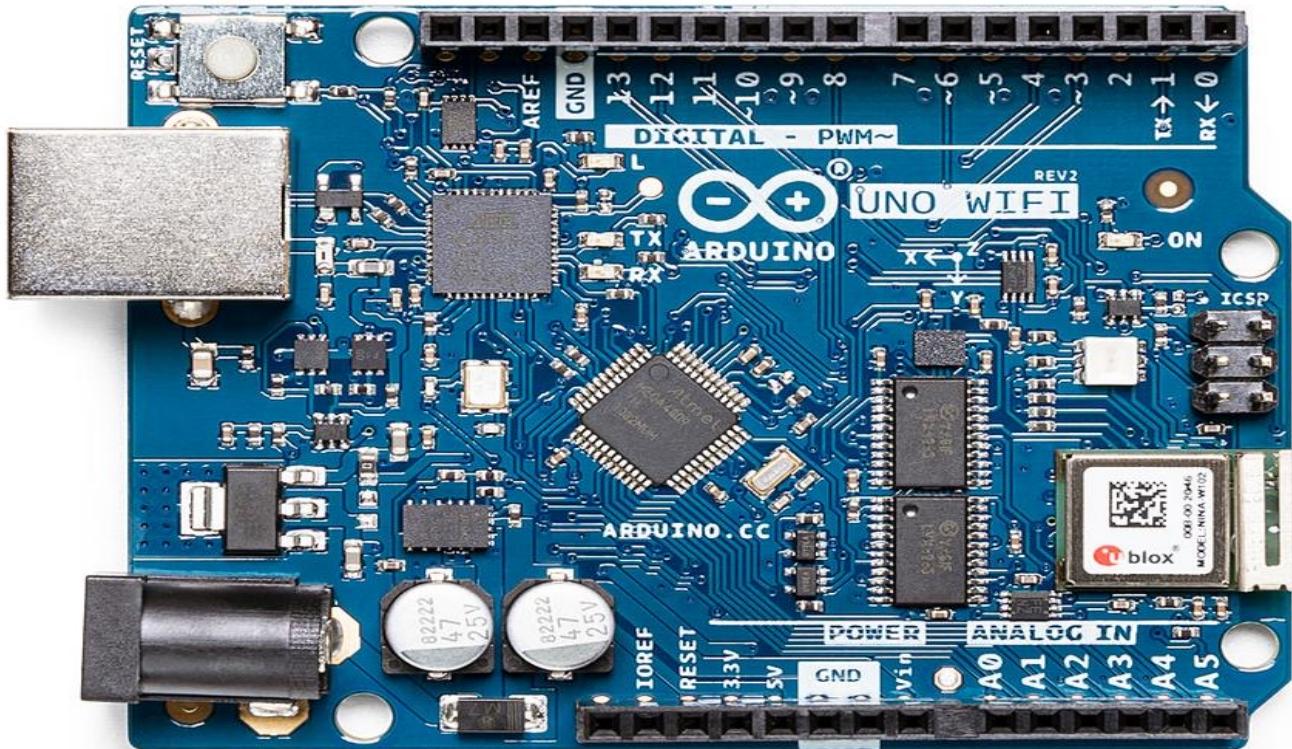


Arduino UNO WiFi Rev 2

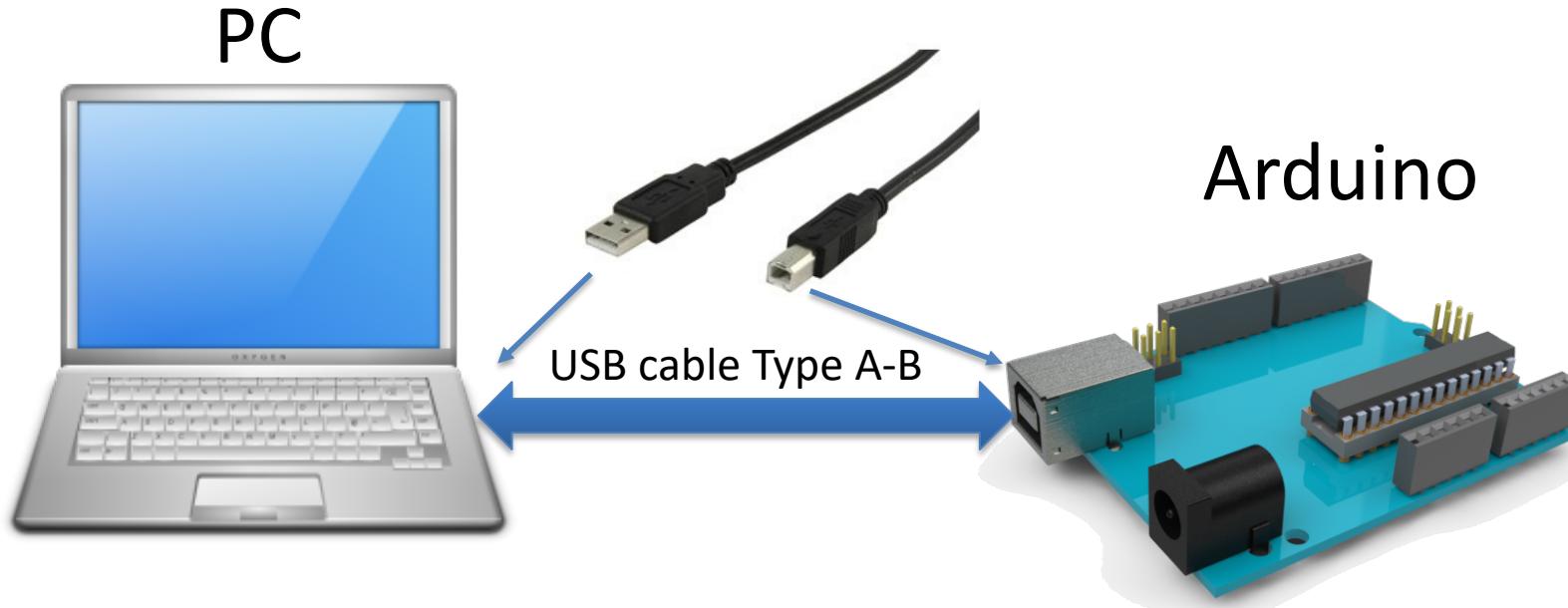
- Lots of different Arduino boards exists
- The basic Arduino UNO don't have WiFi or Ethernet
- We need to use a board with built-in WiFi or Ethernet
- Or we can use a WiFi or Ethernet Shield
- In this Tutorial an “Arduino UNO WiFi Rev 2” is used

Arduino UNO WiFi Rev 2

The Arduino Uno WiFi is functionally the same as the Arduino Uno Rev3, but with the addition of WiFi / Bluetooth and some other enhancements.



Connect Arduino to your PC



Arduino Software

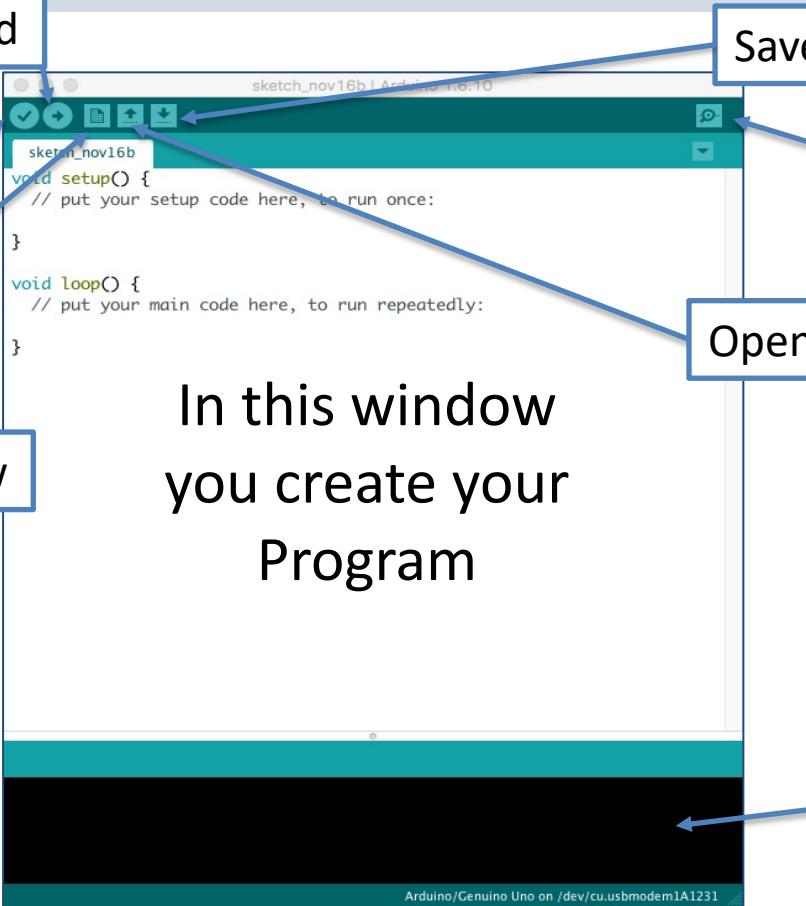
Upload Code to Arduino Board

Compile and Check
if Code is OK

Creates a New Code Window

The software can be
downloaded for free:

www.arduino.cc



Arduino Programs

All Arduino programs must follow the following main structure:

```
// Initialization, define variables, etc.
```

```
void setup()  
{  
    // Initialization  
    ...  
}
```

```
void loop()  
{  
    //Main Program  
    ...  
}
```

<https://www.halvorsen.blog>



ThingSpeak

Hans-Petter Halvorsen

[Table of Contents](#)

ThingSpeak

- ThingSpeak is an IoT analytics platform service that lets you collect and store sensor data in the cloud and develop Internet of Things (IoT) applications.
- ThingSpeak has a free Web Service (REST API) that lets you collect and store sensor data in the cloud and develop Internet of Things applications.
- It works with Arduino, Raspberry Pi, MATLAB and LabVIEW, Python, etc.

<https://thingspeak.com>

ThingSpeak

<https://thingspeak.com>

Work

Channel ID: [REDACTED]
Author: [REDACTED]
Access: Public

temperature

Private View Public View Channel Settings Sharing API Keys Data Import / Export

Add Visualizations Add Widgets Export recent data

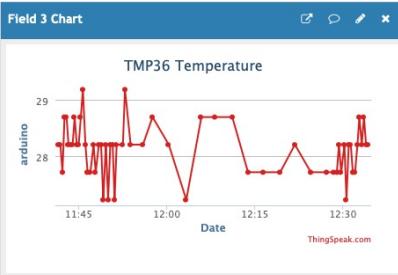
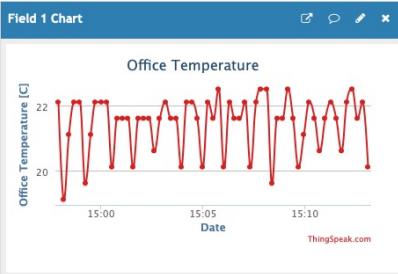
More Information

MATLAB Analysis MATLAB Visualization

Channel 1 of 3 < >

Channel Stats

Created: 4 years ago
Last entry: less than a minute ago
Entries: 242



<https://www.halvorsen.blog>



Arduino + ThingSpeak

Hans-Petter Halvorsen

[Table of Contents](#)

ThingSpeak + Arduino

- Install the “thingspeak” Arduino Library using the Library Manager in your Arduino IDE
- Use e.g., the built-in example "**WriteSingleField**" as a starting point.
- This example is available for different boards and configuration, such as Arduino WiFi rev2 board, Arduino WiFi shield, etc.
- Then you can modify the example to suit your needs

Currently, a single channel can only be
updated once every 15 seconds.

ThingSpeak + Arduino

The screenshot shows the Arduino IDE interface. The top menu bar includes File, Edit, Sketch, Tools, and Help. A toolbar with various icons is located above the code editor. The code editor window has tabs for 'WriteTMP36Data' and 'secrets.h'. The main code area contains C++ code for writing temperature data to a ThingSpeak channel. A search bar in the center says 'thingspeak'. Below it, the 'Library Manager' is open, showing results for 'thingspeak'. The first result is 'ThingSpeak' by MathWorks, Version 1.5.0, which is marked as 'INSTALLED'. It has a brief description and a 'More info' link. The second result is 'ThingSpeak_asukiaaa' by Asuka Kono, Version 1.0.1, with an 'Install' button. The code in the editor includes #include "ThingSpeak.h", #include <WiFiNINA>, and #include "secrets.h". It also defines variables for ssid, pass, keyIndex, client, myCh, myWri, channelField, and SensorPin.

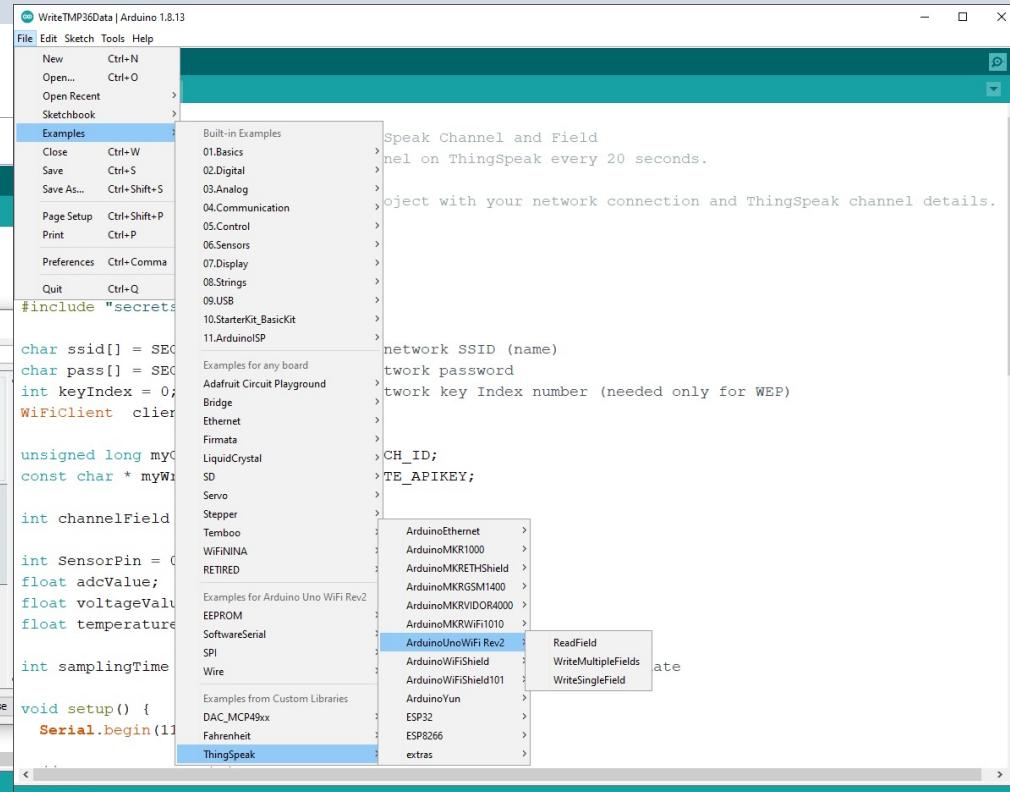
```
/* Write TMP36 Temperature Data to ThingSpeak Channel and Field
Description: Writes a value to a channel on ThingSpeak every 20 seconds.
Hardware: Arduino
Modify the secret file to match your WiFi network information
*/
Write TMP36 Temperature Data to ThingSpeak Channel and Field
Description: Writes a value to a channel on ThingSpeak every 20 seconds.
Hardware: Arduino
Modify the secret file to match your WiFi network information
*/
#include "ThingSpeak.h"
#include <WiFiNINA>
#include "secrets.h"

char ssid[] = SECURE_SSID;
char pass[] = SECURE_PASSWORD;
int keyIndex = 0;
WiFiClient client;

unsigned long myCh;
const char * myWri;

int channelField = 1;

int SensorPin = 0;
```



<https://www.halvorsen.blog>



Arduino Example

Hans-Petter Halvorsen

[Table of Contents](#)

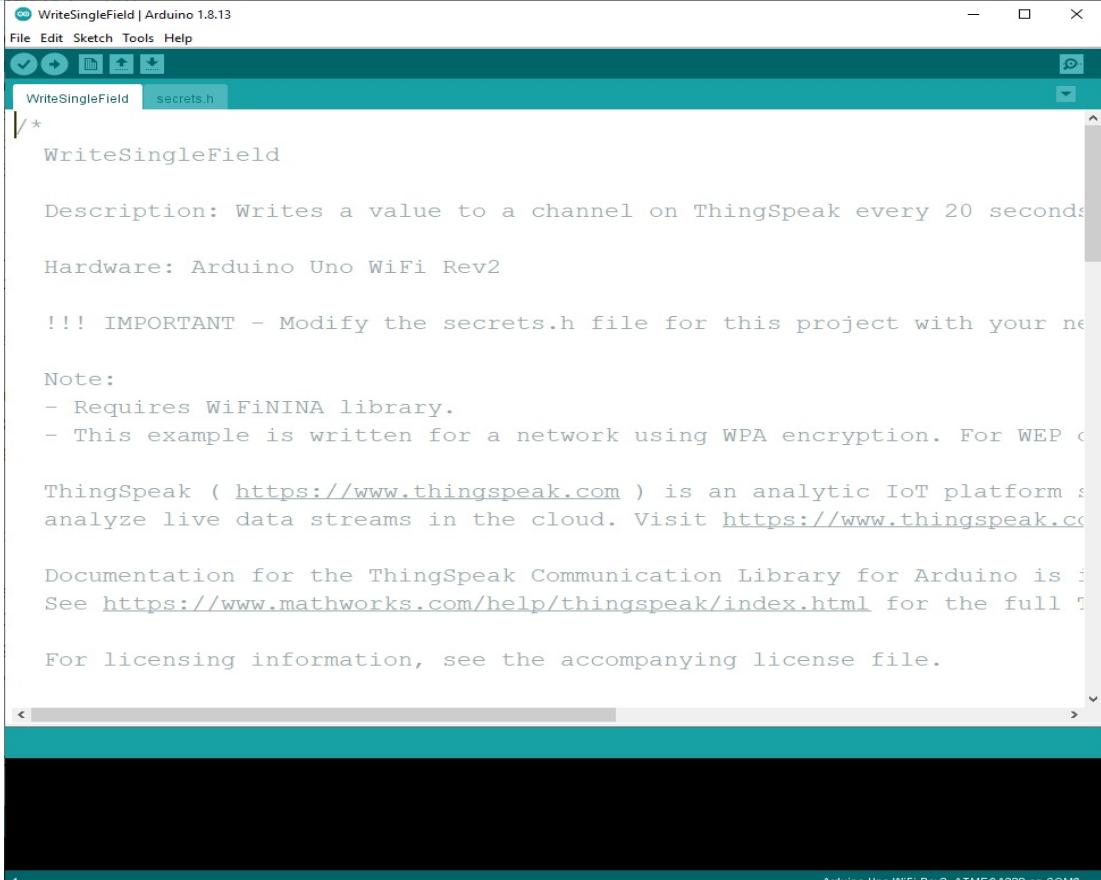
WriteSingleField Example

We use the the “WriteSingleField” Example as a starting point.

We just need to change WiFi information, like Password, etc.

Then we change ThingSpeak Information.

Finally, we add code for reading Temperature values from the TMP36 Temperature Sensor

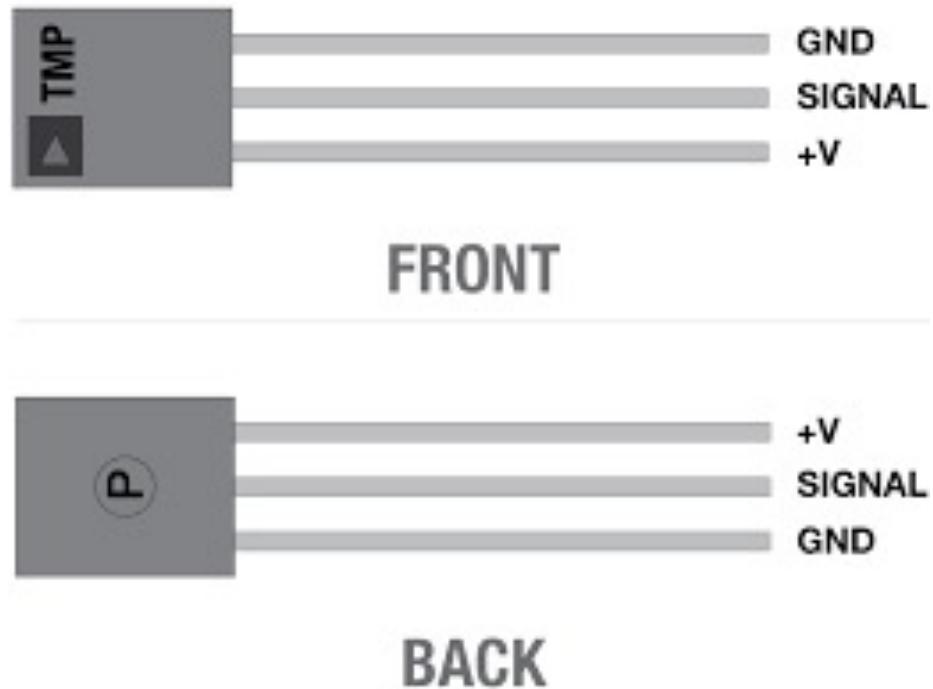
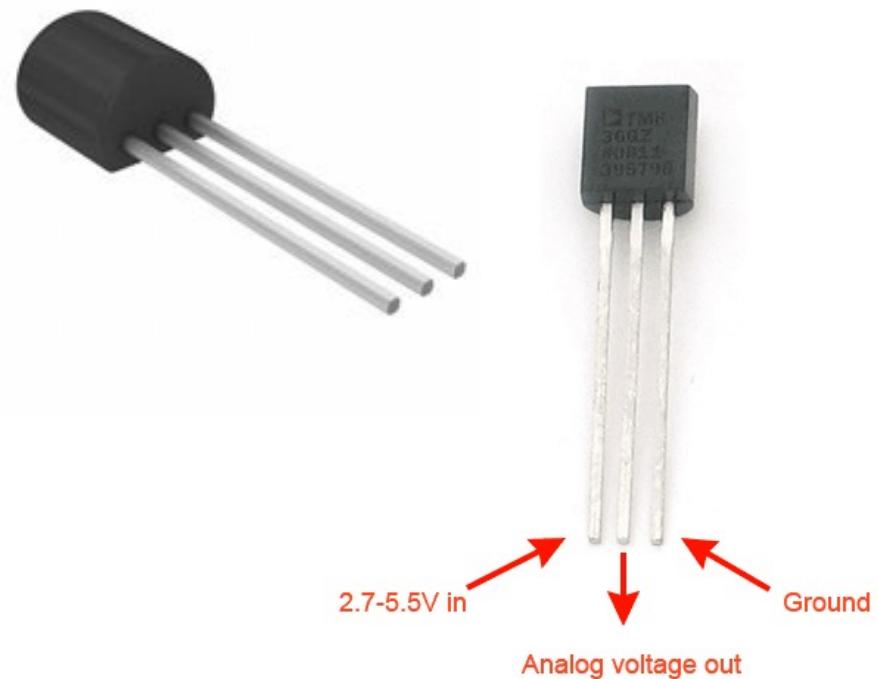


The screenshot shows the Arduino IDE interface with the "WriteSingleField" sketch open. The code is as follows:

```
/*  
 * WriteSingleField  
 *  
 * Description: Writes a value to a channel on ThingSpeak every 20 seconds.  
 *  
 * Hardware: Arduino Uno WiFi Rev2  
 *  
 * !!! IMPORTANT - Modify the secrets.h file for this project with your network  
 * credentials.  
 *  
 * Note:  
 * - Requires WiFiNINA library.  
 * - This example is written for a network using WPA encryption. For WEP or  
 * other network types, modify the WiFi configuration in the secrets.h file.  
 *  
 * ThingSpeak ( https://www.thingspeak.com ) is an analytic IoT platform that lets you  
 * analyze live data streams in the cloud. Visit https://www.thingspeak.com for more information.  
 *  
 * Documentation for the ThingSpeak Communication Library for Arduino is located at  
 * See https://www.mathworks.com/help/thingspeak/index.html for the full documentation.  
 *  
 * For licensing information, see the accompanying license file.  
 */
```

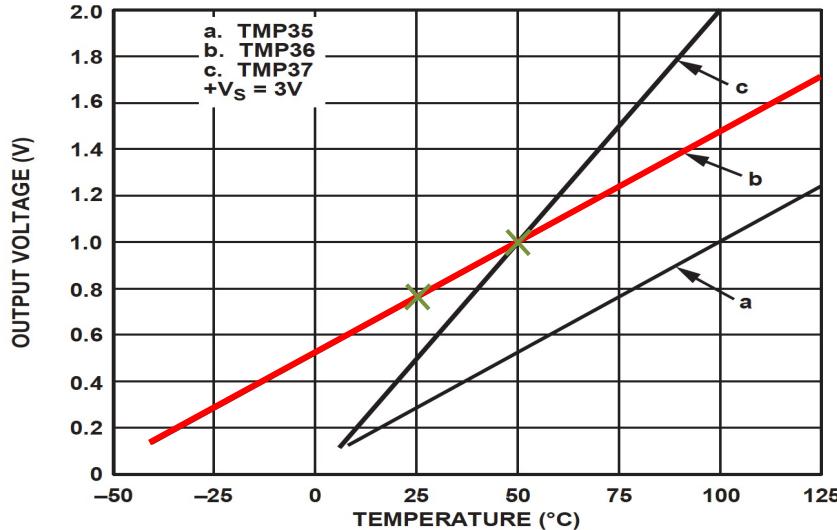
At the bottom of the IDE, it says "Arduino Uno WiFi Rev2, ATMEGA328 on COM6".

TMP36



TMP is a small, low-cost temperature sensor and cost about \$1 (you can buy it “everywhere”)

Linear Scaling



This gives:

$$y - 25 = \frac{50 - 25}{1 - 0.75} (x - 0.75)$$

Then we get the following formula:

$$y = 100x - 50$$

Convert form Voltage (V) to degrees Celsius
From the Datasheet we have:

$$(x_1, y_1) = (0.75V, 25^\circ C)$$
$$(x_2, y_2) = (1V, 50^\circ C)$$

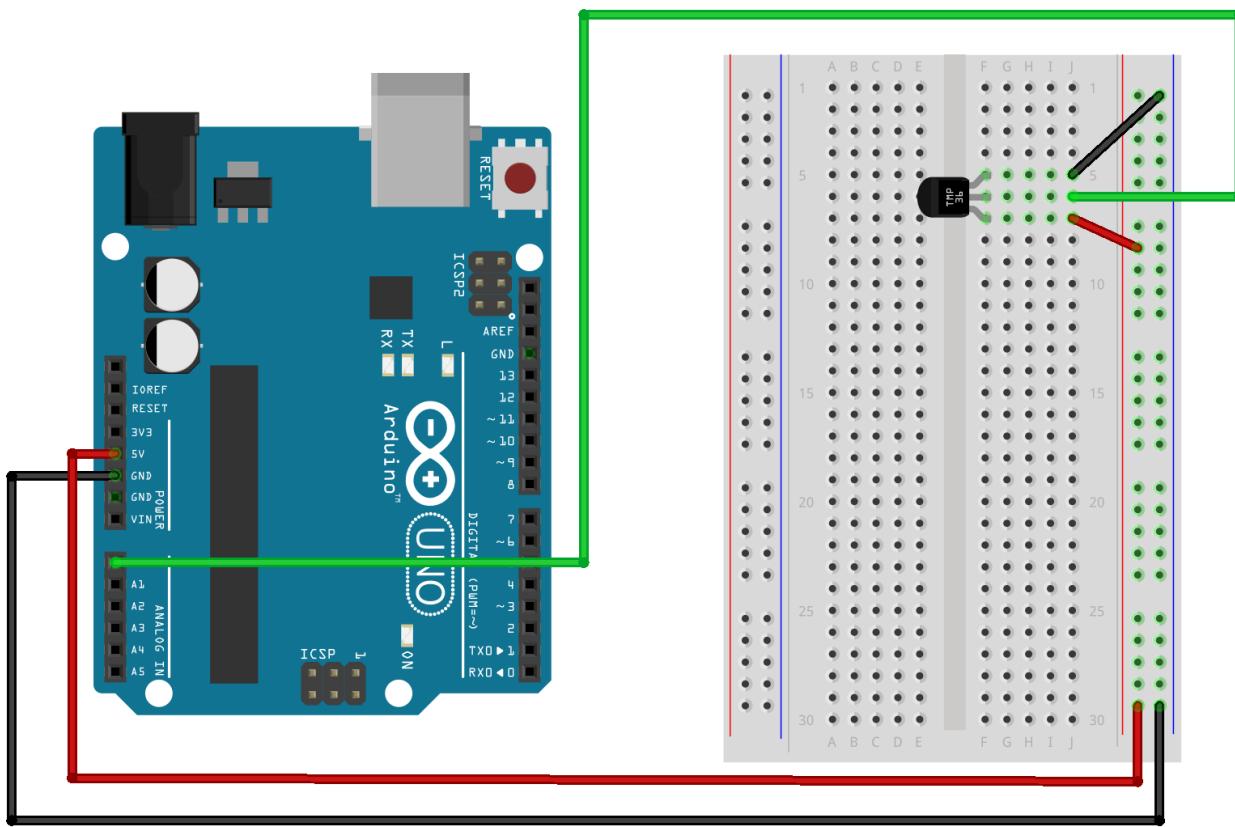
There is a linear relationship between
Voltage and degrees Celsius:

$$y = ax + b$$

We can find a and b using the following
known formula:

$$y - y_1 = \frac{y_2 - y_1}{x_2 - x_1} (x - x_1)$$

Wiring



Temperature Conversion

We want to present the value from the sensor in degrees Celsius:

1. The function `analogRead()` gives a value between 0 and 1023 (Arduino UNO has a built-in 10-bit ADC, $2^{10}=1024$)
2. Then we convert this value to 0-5V.
3. Finally, we convert to degrees Celsius using information from the Datasheet presented on the previous page ($y = 100x - 50$)
4. Then we can, e.g., show the Temperature value in the Serial Monitor

Code

The screenshot shows the Arduino IDE interface with the following details:

- Title Bar:** WriteTMP36Data | Arduino 1.8.13
- Menu Bar:** File Edit Sketch Tools Help
- Toolbar:** Standard icons for file operations.
- Sketch Tabs:** WriteTMP36Data (selected), secrets.h
- Code Area:** Displays C++ code for an Arduino sketch. The code includes comments, #include directives, and variable declarations for WiFi connection and ThingSpeak API.
- Status Bar:** Done uploading.
Sketch uses 18155 bytes (37%) of program storage space. Maximum is 48640 bytes.
Global variables use 470 bytes (7%) of dynamic memory, leaving 5674 bytes for local variables.
avrduude: WARNING: invalid value for unused bits in fuse "fuse5", should be set to 1 ac
- Bottom Status:** Arduino Uno WiFi Rev2 on COM6

```

#include "ThingSpeak.h"
#include <WiFiNINA.h>
#include "secrets.h"
char ssid[] = SECRET_SSID;      // your network SSID (name)
char pass[] = SECRET_PASS;      // your network password
int keyIndex = 0;                // your network key index number (needed only for WEP)
WiFiClient client;
unsigned long myChannelNumber = SECRET_CH_ID;
const char * myWriteAPIKey = SECRET_WRITE_APIKEY;
int channelField = 3;
int SensorPin = 0;
float adcValue;
float voltageValue;
float temperatureValue = 0;
int samplingTime = 20000; // Wait 20 seconds between each channel update
void setup() {
    Serial.begin(115200); // Initialize serial
    if (WiFi.status() == WL_NO_MODULE) {
        Serial.println("Communication with WiFi module failed!");
        // don't continue
        while (true);
    }
    String fv = WiFi.firmwareVersion();
    if (fv != "1.0.0") {
        Serial.println("Please upgrade the firmware");
    }
    ThingSpeak.begin(client); //Initialize ThingSpeak
}
void loop() {
    // Connect or reconnect to WiFi
    if(WiFi.status() != WL_CONNECTED){
        Serial.print("Attempting to connect to SSID: ");
        Serial.println(SECRET_SSID);
        while(WiFi.status() != WL_CONNECTED){
            WiFi.begin(ssid, pass); // Connect to WPA/WPA2 network. Change this line if using open or WEP network
            Serial.print(".");
            delay(5000);
        }
        Serial.println("\nConnected.");
    }
    adcValue = analogRead(SensorPin); // Get Data from Temperature Sensor
    voltageValue = (adcValue*5)/1023;
    temperatureValue = 100*voltageValue - 50;
    Serial.println(temperatureValue);

    // Write to ThingSpeak
    int x = ThingSpeak.writeField(myChannelNumber, channelField, temperatureValue, myWriteAPIKey);
    if(x == 200){
        Serial.println("Channel update successful.");
    }
    else{
        Serial.println("Problem updating channel. HTTP error code " + String(x));
    }
    delay(20000); // Wait 20 seconds to update the channel again
}

```

This Example uses an Arduino WiFi rev2 board.
The Example reads values from TMP36 Temperature Sensor and write the values to ThingSpeak

secrets.h

```

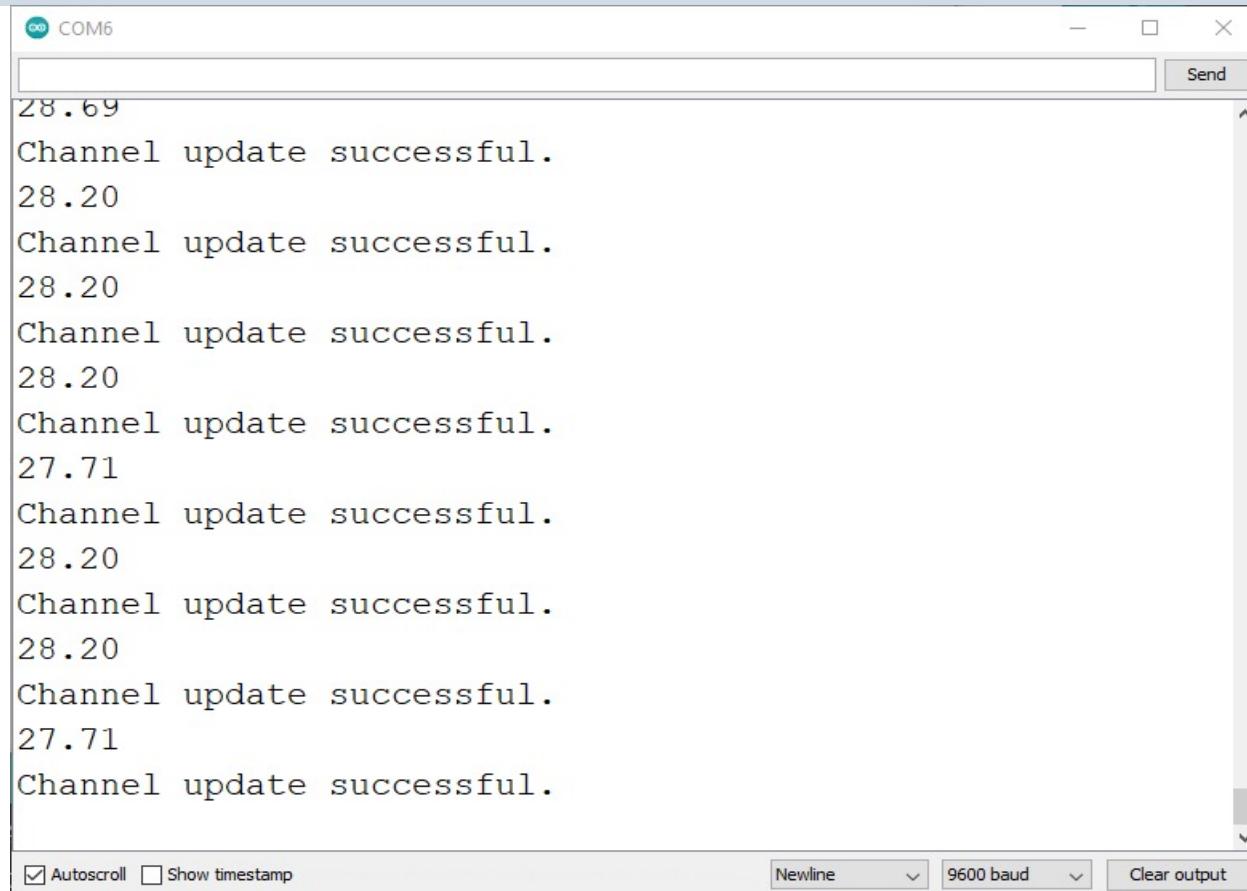
// Use this file to store all of the private credentials
// and connection details

#define SECRET_SSID "MySSID"          // replace MySSID with your WiFi network name
#define SECRET_PASS   "xxxxxx"         // replace MyPassword with your WiFi password

#define SECRET_CH_ID 000000           // replace 000000 with your channel number
#define SECRET_WRITE_APIKEY "XYZ"     // replace XYZ with your channel write API Key

```

Serial Monitor



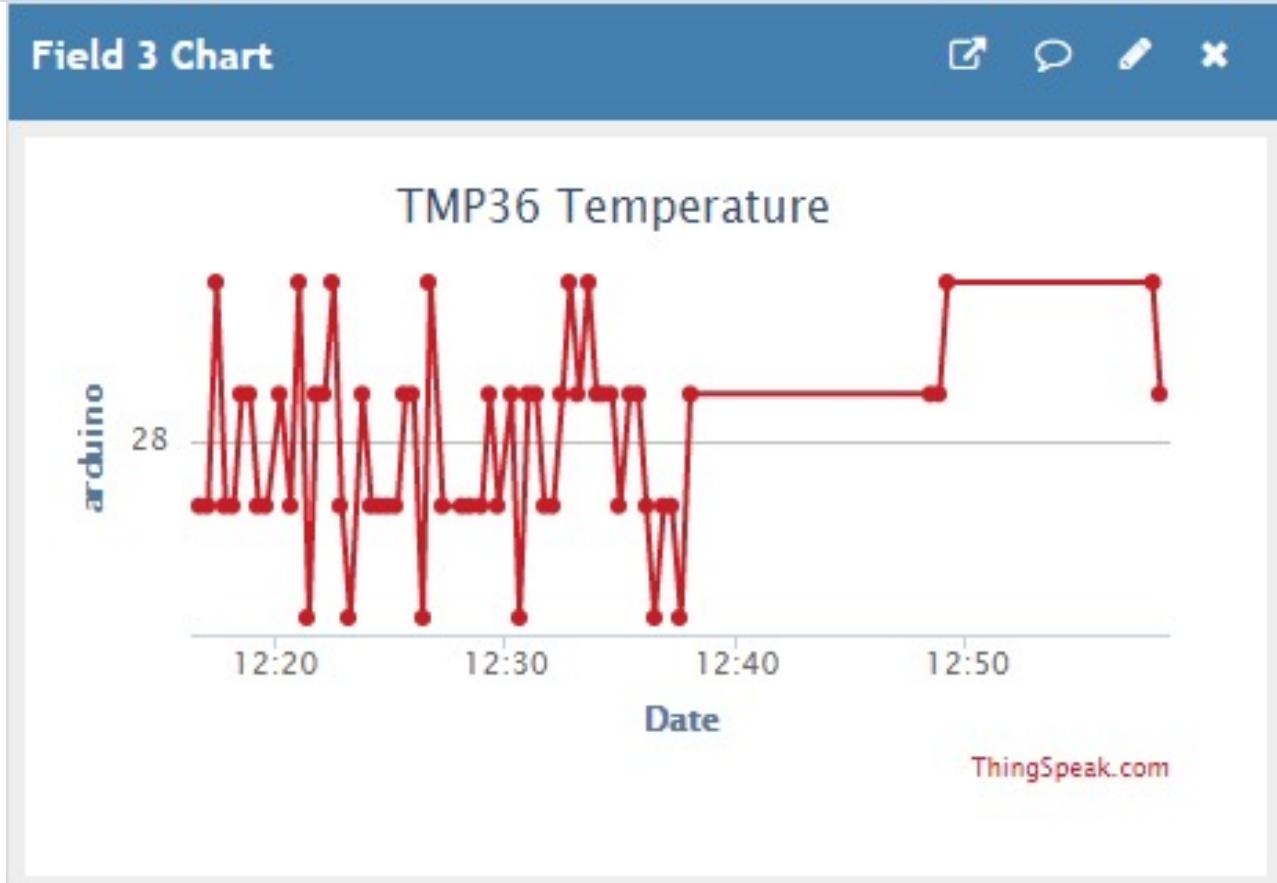
The screenshot shows a Windows-style application window titled "Serial Monitor". The title bar includes standard window controls (minimize, maximize, close) and the port name "COM6". The main area is a text-based terminal window displaying a series of messages. Each message consists of a numerical value followed by the text "Channel update successful.". The values are 28.69, 28.20, 28.20, 28.20, 27.71, 28.20, 28.20, 28.20, 27.71, and 28.69. The window has scroll bars on the right side. At the bottom, there are several configuration options: "Autoscroll" (checked), "Show timestamp" (unchecked), "Newline" (dropdown menu), "9600 baud" (dropdown menu), and "Clear output" (button).

```
28.69
Channel update successful.
28.20
Channel update successful.
28.20
Channel update successful.
28.20
Channel update successful.
27.71
Channel update successful.
28.20
Channel update successful.
28.20
Channel update successful.
27.71
Channel update successful.
```

Autoscroll Show timestamp Newline 9600 baud Clear output

ThingSpeak

We see that the Temperature Data has been successfully written to ThingSpeak



Updated Code

The Code is the same, but it is now structured into different Functions for better readability

secrets.h

```
#define SECRET_SSID "xxxxxx"  
#define SECRET_PASS "xxxxxx"  
  
#define SECRET_CH_ID xxxxx  
  
#define SECRET_WRITE_APIKEY "xxxxxx"
```

```
#include "ThingSpeak.h"  
#include <WiFiNINA.h>  
#include "secrets.h"  
  
WiFiClient client;  
int wait = 20000;  
float temperatureValue = 0;  
  
void setup()  
{  
    Serial.begin(9600);  
    CheckWiFi();  
    ThingSpeak.begin(client);  
}  
  
void loop()  
{  
    ConnectWiFi();  
    ReadTemperature();  
    ThingSpeakWrite();  
    delay(wait);  
}
```

CheckWiFi()

```
void CheckWiFi()
{
    // check for the WiFi module:
    if (WiFi.status() == WL_NO_MODULE) {
        Serial.println("Communication with WiFi module failed!");
        // don't continue
        while (true);
    }

    String fv = WiFi.firmwareVersion();
    if (fv != "1.0.0")
    {
        Serial.println("Please upgrade the firmware");
    }
}
```

ConnectWiFi()

```
void ConnectWiFi()
{
    char ssid[] = SECRET_SSID;
    char pass[] = SECRET_PASS;

    if(WiFi.status() != WL_CONNECTED)
    {
        Serial.print("Attempting to connect to SSID: ");
        Serial.println(SECRET_SSID);
        while(WiFi.status() != WL_CONNECTED)
        {
            WiFi.begin(ssid, pass);
            Serial.print(".");
            delay(5000);
        }
        Serial.println("\nConnected.");
    }
}
```

ReadTemperature()

```
void ReadTemperature()  
{  
    int SensorPin = 0;  
    float adcValue;  
    float voltageValue;  
  
    adcValue = analogRead(SensorPin);  
    voltageValue = (adcValue*5)/1023;  
    temperatureValue = 100*voltageValue - 50;  
    Serial.println(temperatureValue);  
}
```

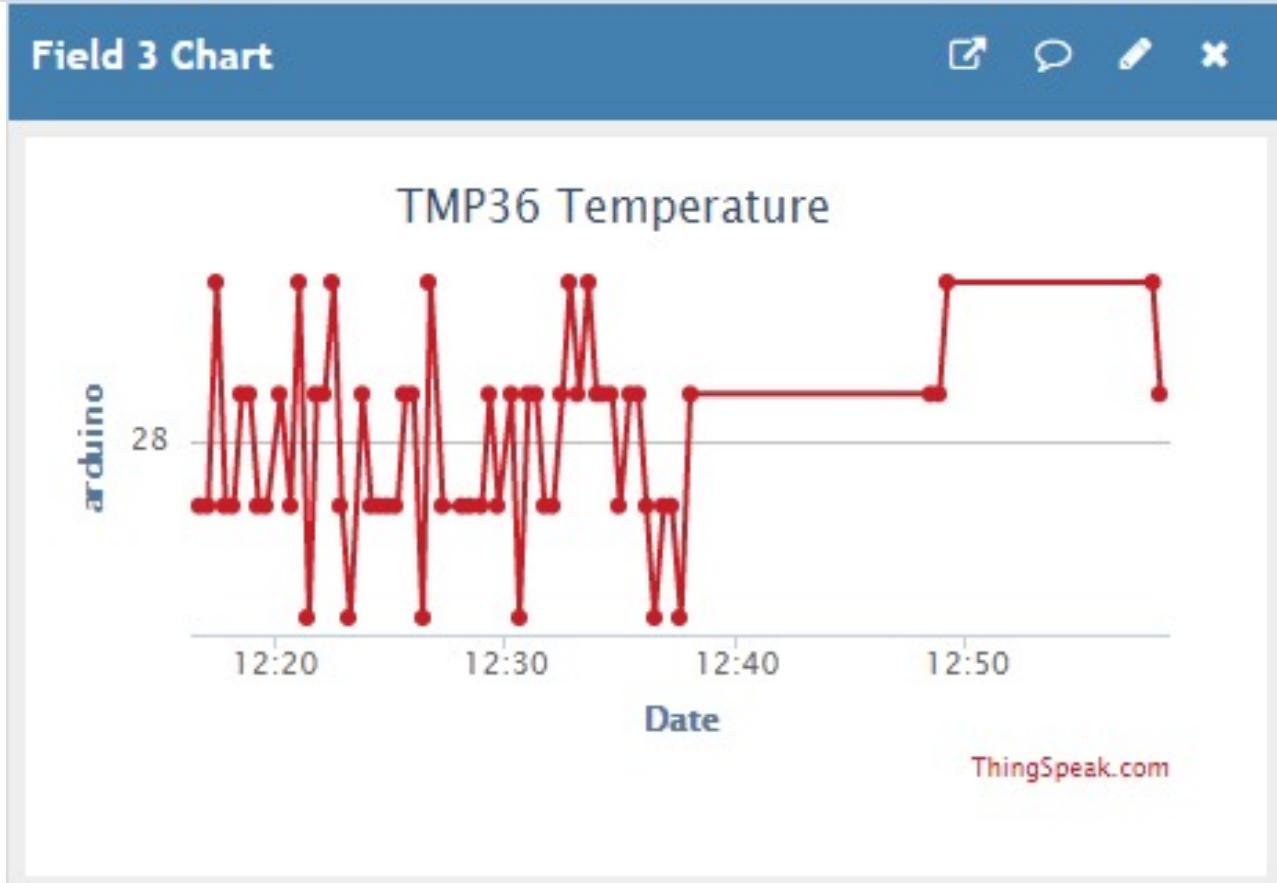
ThingSpeakWrite()

```
void ThingSpeakWrite()
{
    unsigned long myChannelNumber = SECRET_CH_ID;
    const char * myWriteAPIKey = SECRET_WRITE_APIKEY;
    int channelField = 3;

    int x = ThingSpeak.writeField(myChannelNumber, channelField,
        temperatureValue, myWriteAPIKey);
    if(x == 200){
        Serial.println("Channel update successful.");
    }
    else{
        Serial.println("Problem updating channel. HTTP error code " + String(x));
    }
}
```

ThingSpeak

We see that the Temperature Data has been successfully written to ThingSpeak



Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

E-mail: hans.p.halvorsen@usn.no

Web: <https://www.halvorsen.blog>

