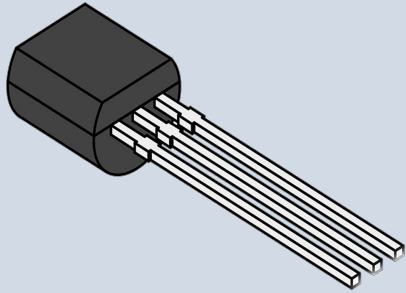


<https://www.halvorsen.blog>



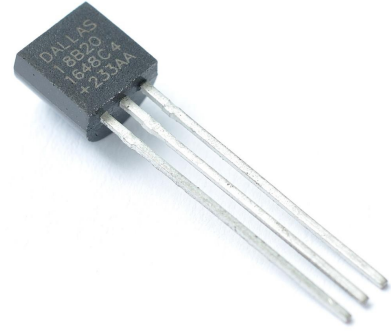
Arduino and DS18B20

1-Wire Temperature Sensor

Hans-Petter Halvorsen

Contents

- Introduction to Arduino
- DS18B20 Temperature Sensor
- Arduino Examples
 - Read Temperature Data from DS18B20 Sensor
 - Write Temperature Data to ThingSpeak





Arduino

Hans-Petter Halvorsen

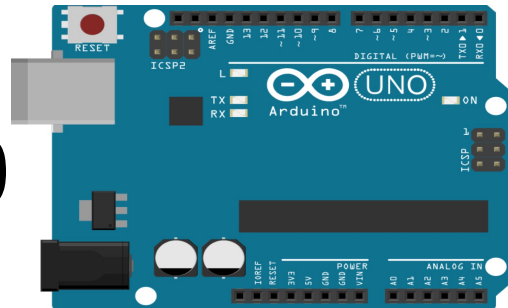
[Table of Contents](#)

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

Digital ports (2-13)

Reset button

3

RESET

USB for PC connection

2



External Power Supply

1



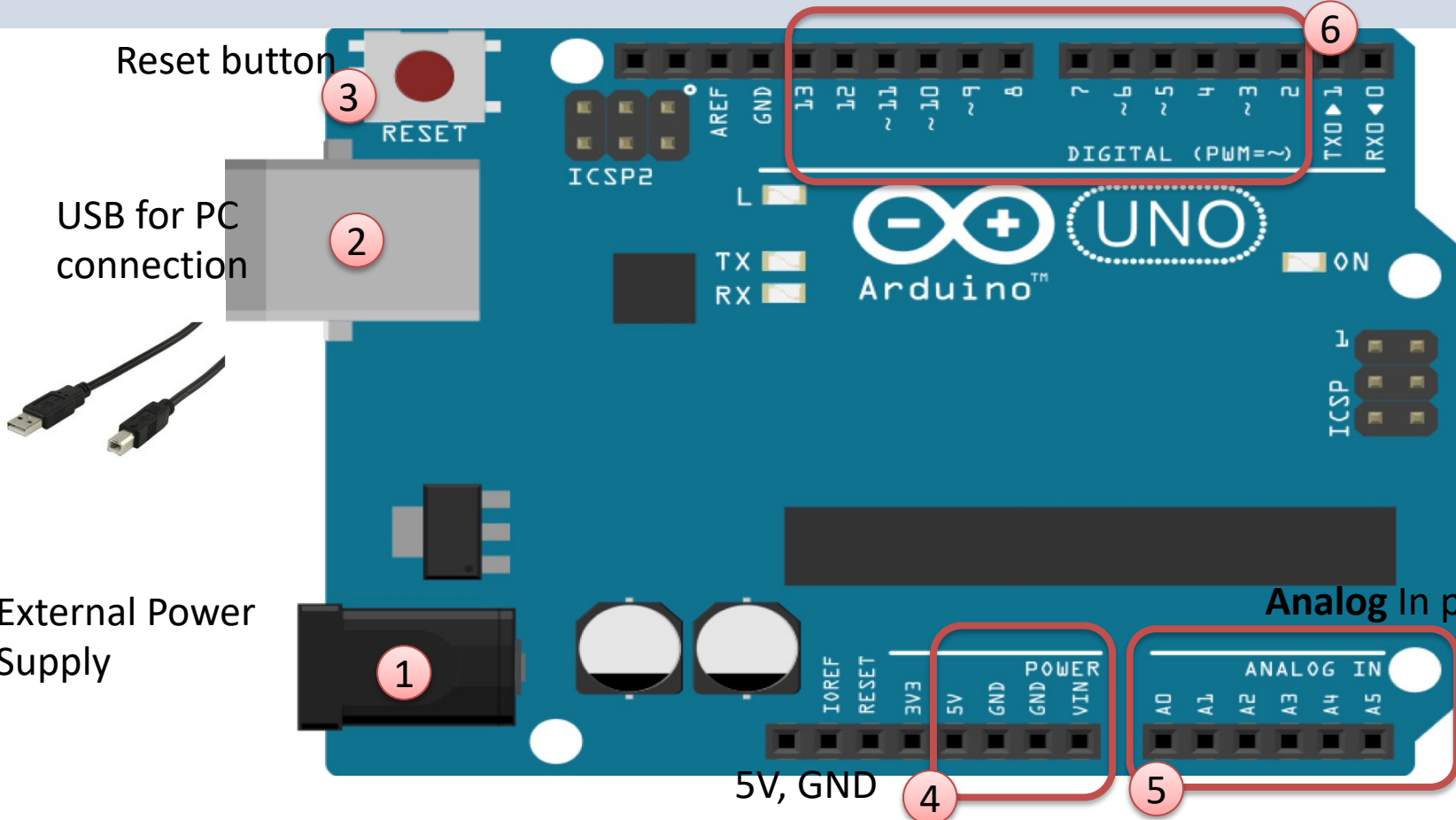
5V, GND

4

5

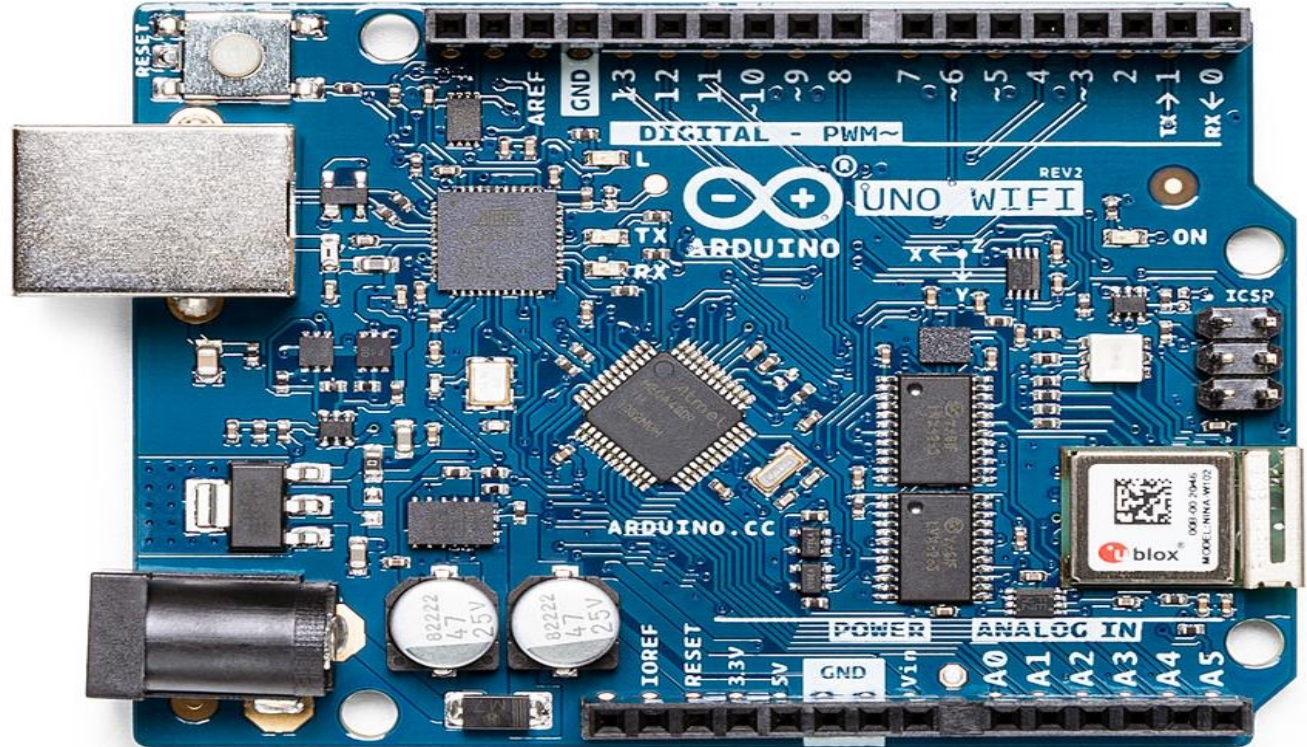
Analog In ports (0-5)

6



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.



Arduino Software

Upload Code to Arduino Board

Save

Open Serial Monitor

Compile and Check
if Code is OK

Open existing Code

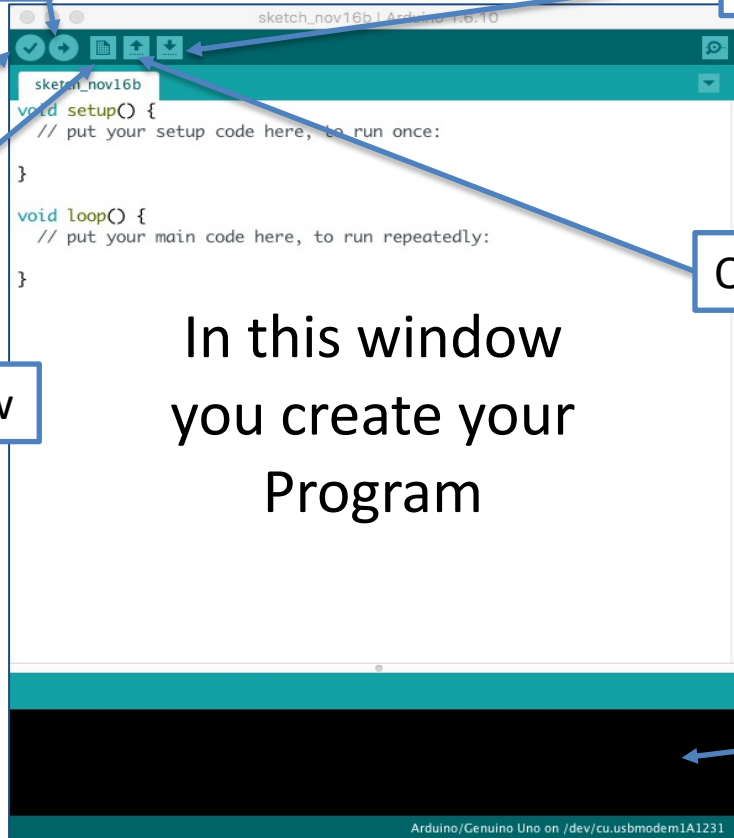
Creates a New Code Window

In this window
you create your
Program

The software can be
downloaded for free:

www.arduino.cc

Error Messages
can be seen here





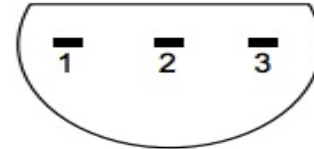
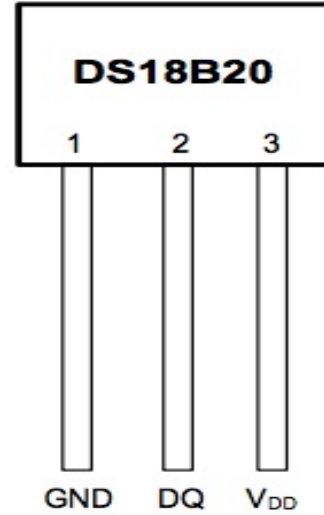
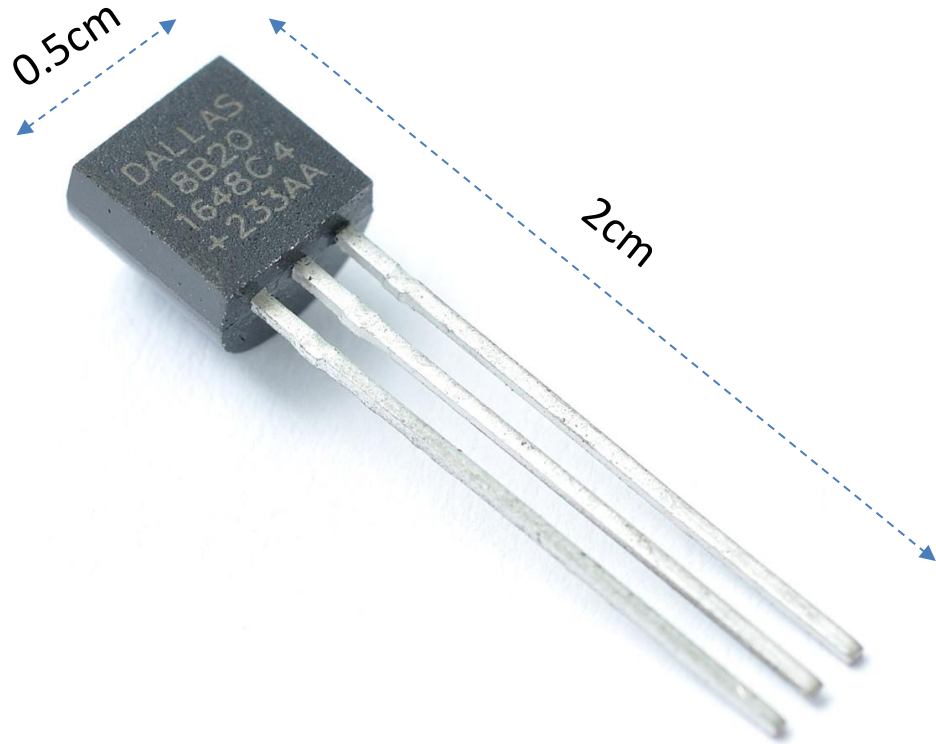
DS18B20

1-Wire Temperature Sensor

Hans-Petter Halvorsen

[Table of Contents](#)

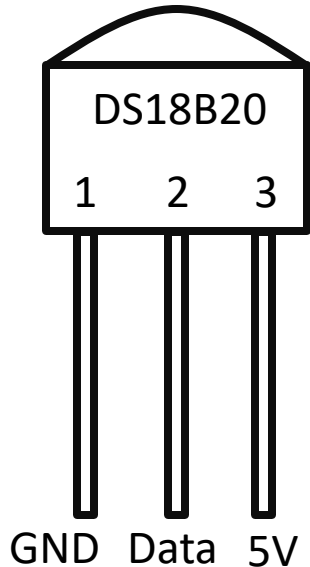
DS18B20 1-Wire Temperature Sensor



BOTTOM VIEW

TO-92
(DS18B20)

DS18B20 1-Wire Temperature Sensor



- DS18B20 is a 1-Wire Digital Temperature Sensor, this means the Sensor only need 1 Pin for Communication (+ one pin for GND and one pin for 5V)
- **Accuracy** $\pm 0.5^{\circ}\text{C}$
- 9 to 12-bit resolution (Programmable)
- **Temperature range** -55°C to $+125^{\circ}\text{C}$
- Price: About \$4
- Datasheet:

<https://datasheets.maximintegrated.com/en/ds/DS18B20.pdf>

The Data pin goes to a Digital Input pin on the Arduino

Digital vs. Analog Temperature Sensors

- Digital temperature sensors like the DS18B20 differ from analog thermistors in several important ways.
- In thermistors, changes in temperature cause changes in the resistance of a ceramic or polymer semiconducting material.
- Usually, the thermistor is set up in a voltage divider, and the voltage is measured between the thermistor and a known resistor.
- The voltage measurement is converted to resistance and then converted to a temperature value by the microcontroller.
- Digital temperature sensors are typically silicon based integrated circuits.
- They contain the temperature sensor, an analog to digital converter (ADC), memory to temporarily store the temperature readings, and an interface that allows communication between the sensor and a microcontroller.
- Unlike analog temperature sensors, calculations are performed by the sensor, and the output is an actual temperature value (in degrees Celsius) – so no conversion is needed.
- The DS18B20 communicates with the “One-Wire” communication protocol, a proprietary serial communication protocol that uses only one wire to transmit the temperature readings to the microcontroller.

<https://www.circuitbasics.com/raspberry-pi-ds18b20-temperature-sensor-tutorial/>

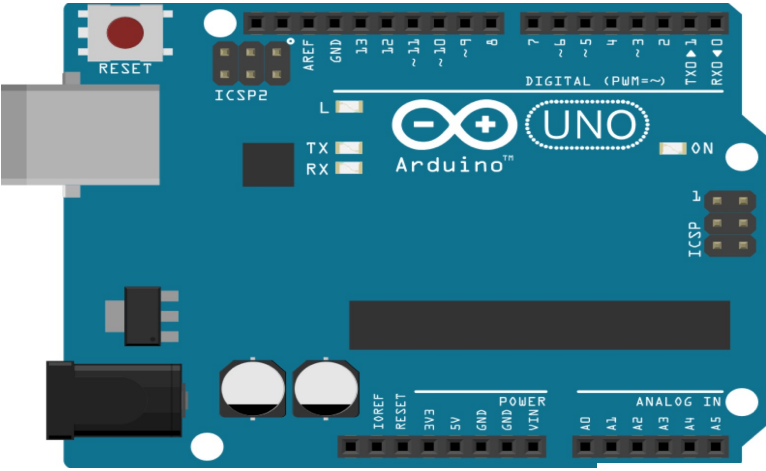


Arduino Examples



Read Temperature Data

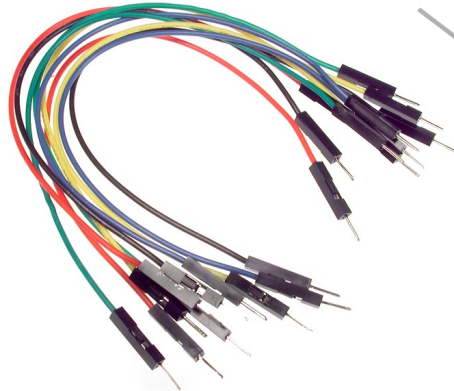
Equipment



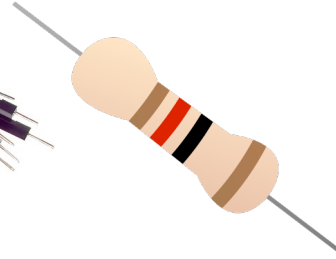
Arduino



Breadboard



Wires



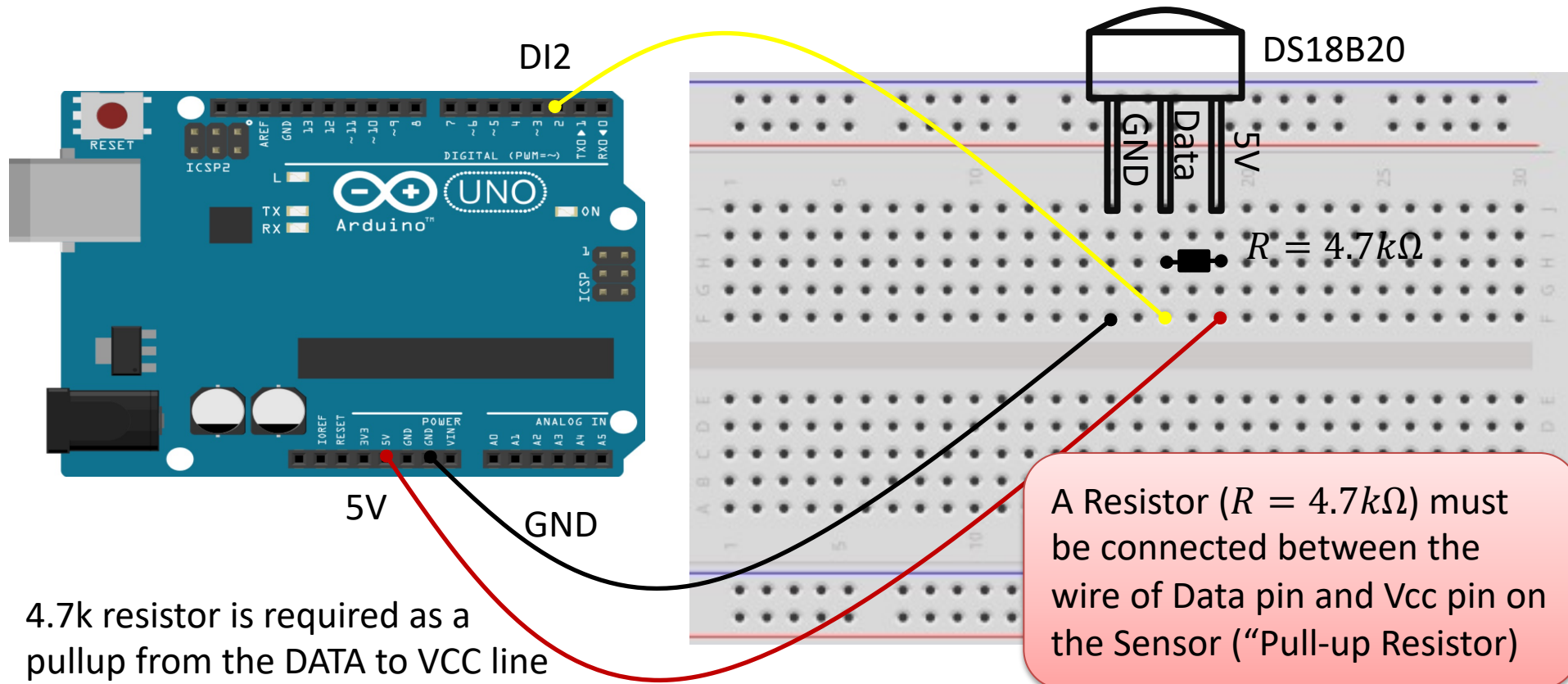
Resistor $R = 4.7k\Omega$



DS18B20

Wiring

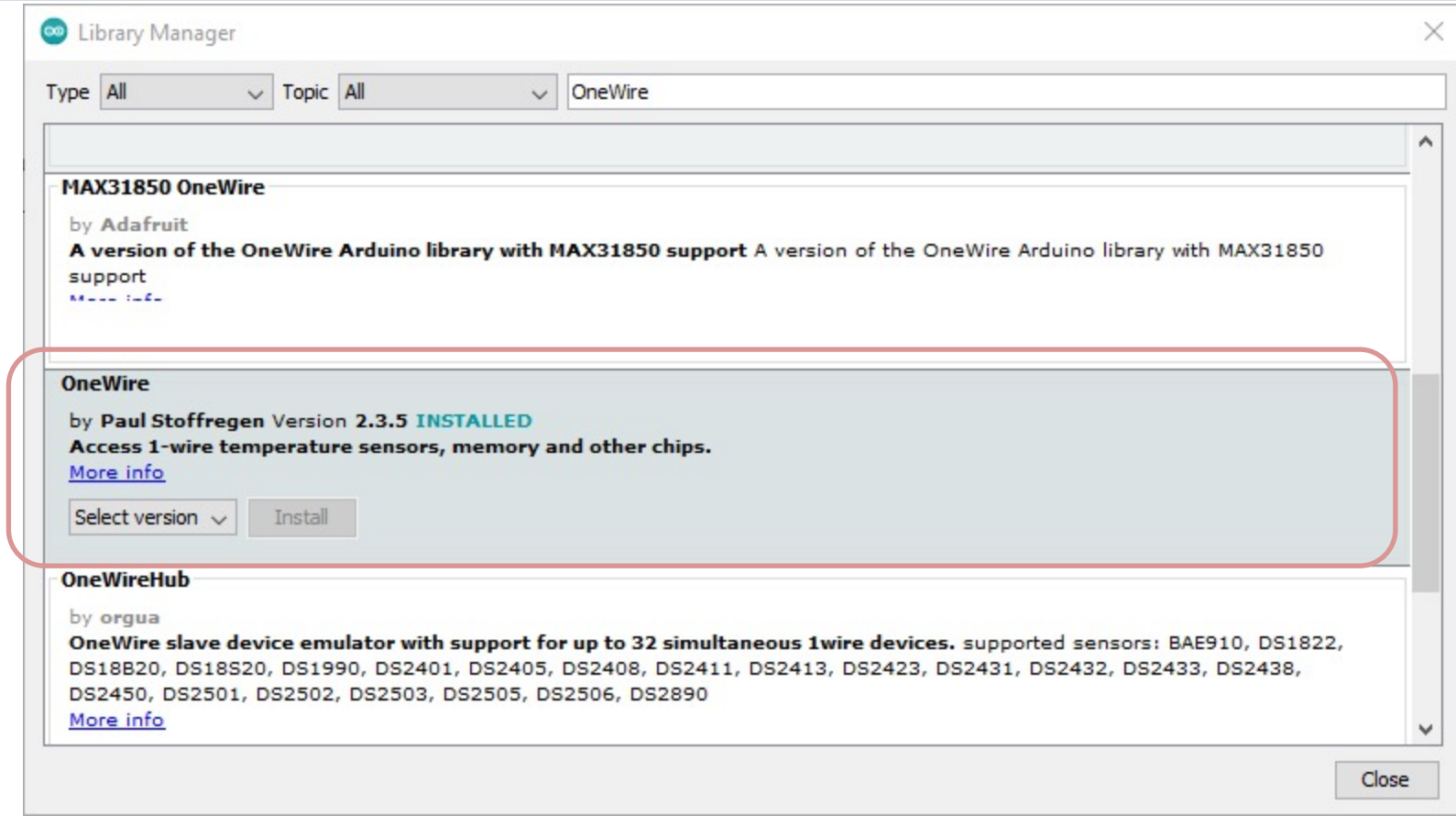
Be careful to get the DS18B20 the right way around. The curved edge should be to placed as shown in the figure below. If you put it the wrong way around, it will get hot and then break.



Arduino Code Example

- We use some existing Libraries
 - **OneWire** Library
 - **DallasTemperature** Library
- By using these Libraries, it requires just a few lines of code in order to get the Temperature Value in degrees Celsius

OneWire Library



The screenshot shows the Arduino IDE Library Manager window. At the top, the title bar reads "Library Manager" with a close button on the right. Below the title bar, there are two dropdown menus for "Type" and "Topic", both set to "All". To the right of these is a search bar containing the text "OneWire".

The main area of the window displays a list of libraries. The "OneWire" library by Paul Stoffregen is highlighted with a red rounded rectangle. The details for this library are as follows:

- MAX31850 OneWire**
by Adafruit
A version of the OneWire Arduino library with MAX31850 support A version of the OneWire Arduino library with MAX31850 support
[More info](#)
- OneWire**
by Paul Stoffregen Version 2.3.5 **INSTALLED**
Access 1-wire temperature sensors, memory and other chips.
[More info](#)
Select version
- OneWireHub**
by orgua
OneWire slave device emulator with support for up to 32 simultaneous 1wire devices. supported sensors: BAE910, DS1822, DS18B20, DS18S20, DS1990, DS2401, DS2405, DS2408, DS2411, DS2413, DS2423, DS2431, DS2432, DS2433, DS2438, DS2450, DS2501, DS2502, DS2503, DS2505, DS2506, DS2890
[More info](#)

At the bottom right of the window, there is a "Close" button.

DallasTemperature Library

The screenshot shows the Arduino IDE Library Manager window. At the top, the title bar reads "Library Manager" with a close button on the right. Below the title bar, there are filters for "Type" (set to "All"), "Topic" (set to "All"), and a search field containing "DallasTemperature". The search results are displayed in a list. The first result, "DallasTemperature", is highlighted with a red rounded rectangle. This entry is by Miles Burton and is described as an "Arduino Library for Dallas Temperature ICs" that supports DS18B20, DS18S20, DS1822, and DS1820 sensors. It includes a "More info" link and an "Install" button. The version "3.9.0" is shown in a dropdown menu next to the "Install" button. The second result, "DS18B20Events", is by Ihar Yakimush and is described as "Arduino temperature changed events for DS18B20 and other DallasTemperature compatible sensors". It also has a "More info" link. At the bottom right of the window, there is a "Close" button.

Library Manager

Type All Topic All DallasTemperature

DallasTemperature
by Miles Burton
Arduino Library for Dallas Temperature ICs Supports DS18B20, DS18S20, DS1822, DS1820
[More info](#)
Version 3.9.0 Install

DS18B20Events
by Ihar Yakimush
Arduino temperature changed events for DS18B20 and other DallasTemperature compatible sensors Arduino temperature changed events for DS18B20 and other DallasTemperature compatible sensors
[More info](#)

Close

Example

sketch_oct01a | Arduino 1.8.16

File Edit Sketch Tools Help

New Ctrl+N
Open... Ctrl+O
Open Recent >
Sketchbook >
Examples >
Close Ctrl+W
Save Ctrl+S
Save As... Ctrl+Shift+S
Page Setup Ctrl+Shift+P
Print Ctrl+P
Preferences Ctrl+Comma
Quit Ctrl+Q

06.Sensors >
07.Display >
08.Strings >
09.USB >
10.StarterKit_BasicKit >
11.ArduinoISP >

run once:

Examples for any board
Adafruit Circuit Playground >
Bridge >
Ethernet >
Firmata >
LiquidCrystal >
SD >
Servo >
Stepper >
Tombuo >
WiFiNINA >
RETIRED >

run repeatedly:

Examples for Arduino Uno WiFi Rev2

EEPROM >
SoftwareSerial >
SPI >
Wire >

Examples from Custom Libraries

Adafruit Unified Sensor >
Control >
DAC_MCP49xx >
DallasTemperature >
DHT sensor library >
Fahrenheit >
MCP_DAC >
OneWire >

Alarm
AlarmHandler
ExternalPullup
Multibus_simple
Multiple
oneWireSearch
readPowerSupply
SaveRecallScratchPad
SetUserData
Simple
Single
Tester
Timing
TwoPin_DS18B20
UserDataDemo
UserDataWriteBatch
WaitForConversion
WaitForConversion2

Simple | Arduino 1.8.16

File Edit Sketch Tools Help

Simple

```
void setup(void)
{
  // start serial port
  Serial.begin(9600);
  Serial.println("Dallas Temperature IC Control Library Demo");

  // Start up the library
  sensors.begin();
}

/*
 * Main function, get and show the temperature
 */
void loop(void)
{
  // call sensors.requestTemperatures() to issue a global temperature
  // request to all devices on the bus
  Serial.print("Requesting temperatures...");
  sensors.requestTemperatures(); // Send the command to get temperatures
  Serial.println("DONE");
  // After we got the temperatures, we can print them here.
  // We use the function ByIndex, and as an example get the temperature from
  float tempC = sensors.getTempCByIndex(0);
```

We use the built-in Example as a starting point

Arduino Code

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 2

OneWire oneWire(ONE_WIRE_BUS);

DallasTemperature sensors(&oneWire);

float tempCelcius=0;

void setup(void)
{
  Serial.begin(9600);
  sensors.begin();
}

void loop(void)
{
  sensors.requestTemperatures();
  tempCelcius = sensors.getTempCByIndex(0);
  Serial.print("T = ");
  Serial.print(tempCelcius);
  Serial.println("°C");

  delay(1000);
}
```

Serial Monitor

The screenshot shows a Windows Serial Monitor window titled "COM6". The window contains a text area displaying a series of temperature readings: "T = 24.25°C", "T = 24.25°C", "T = 24.31°C", "T = 24.25°C", "T = 24.31°C", "T = 24.31°C", "T = 24.25°C", "T = 24.31°C", "T = 24.31°C", "T = 24.31°C", "T = 25.25°C", "T = 27.12°C", "T = 28.25°C", and "T = 28.31°C". At the top of the window is an input field and a "Send" button. At the bottom, there are checkboxes for "Autoscroll" (checked) and "Show timestamp" (unchecked), along with dropdown menus for "Newline" and "9600 baud", and a "Clear output" button.

COM6

Send

T = 24.25°C
T = 24.25°C
T = 24.31°C
T = 24.25°C
T = 24.31°C
T = 24.31°C
T = 24.25°C
T = 24.31°C
T = 24.31°C
T = 24.31°C
T = 25.25°C
T = 27.12°C
T = 28.25°C
T = 28.31°C

Autoscroll Show timestamp

Newline 9600 baud Clear output



Log Sensor Data to ThingSpeak

Hans-Petter Halvorsen

[Table of Contents](#)

Log Data to ThingSpeak

- In this Example we will read Temperature data from the DS18B20 Sensor
- Then we will the Temperature data to the ThingSpeak Cloud Service

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>

ThingSpeak™ ☰

Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

+ Add Visualizations

+ Add Widgets

MATLAB Analysis

Export recent data

MATLAB Visualization

More Information

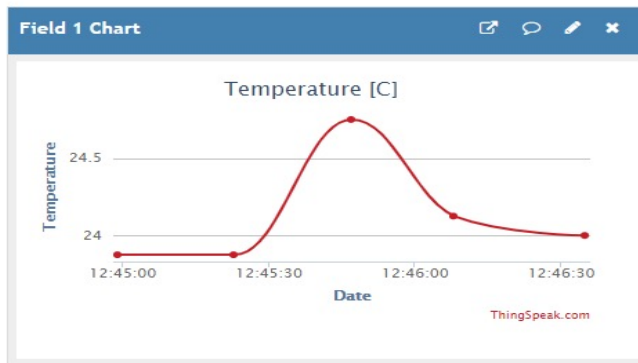
Channel 1 of 3 < >

Channel Stats

Created: 4 years ago

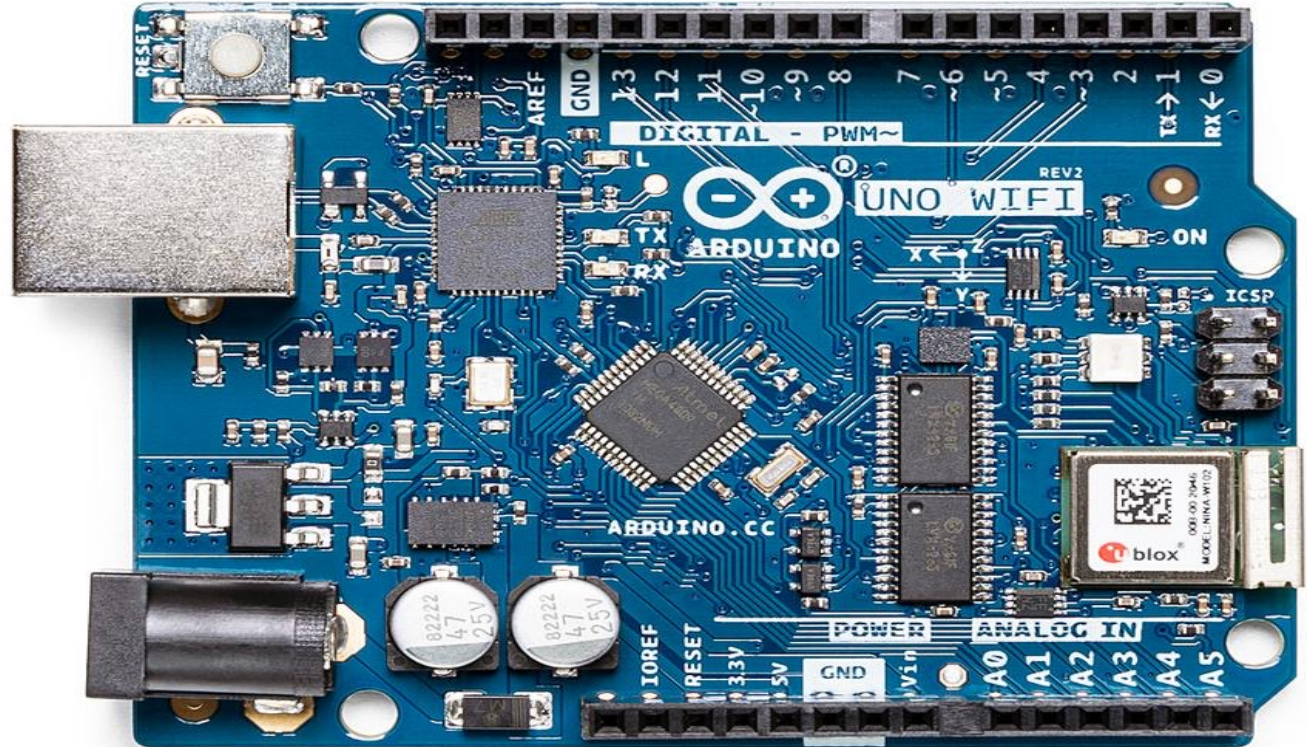
Last entry: about a minute ago

Entries: 5



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.



ThingSpeak Library

Library Manager

Type **All** Topic **All** **ThingSpeak**

ThingSpeak
by **MathWorks** Version **2.0.1** **INSTALLED**
ThingSpeak Communication Library for Arduino, ESP8266 & ESP32 ThingSpeak (<https://www.thingiverse.com/thing:1000000>)
IoT platform service that allows you to aggregate, visualize and analyze live data streams in the cloud.
[More info](#)

ThingSpeak_asukiaaa
by **Asuki Kono**
An API manager for ThingSpeak It writes field values for ThinkgSpeak.
[More info](#)

sketch_sep20a | Arduino 1.8.16

File Edit Sketch Tools Help

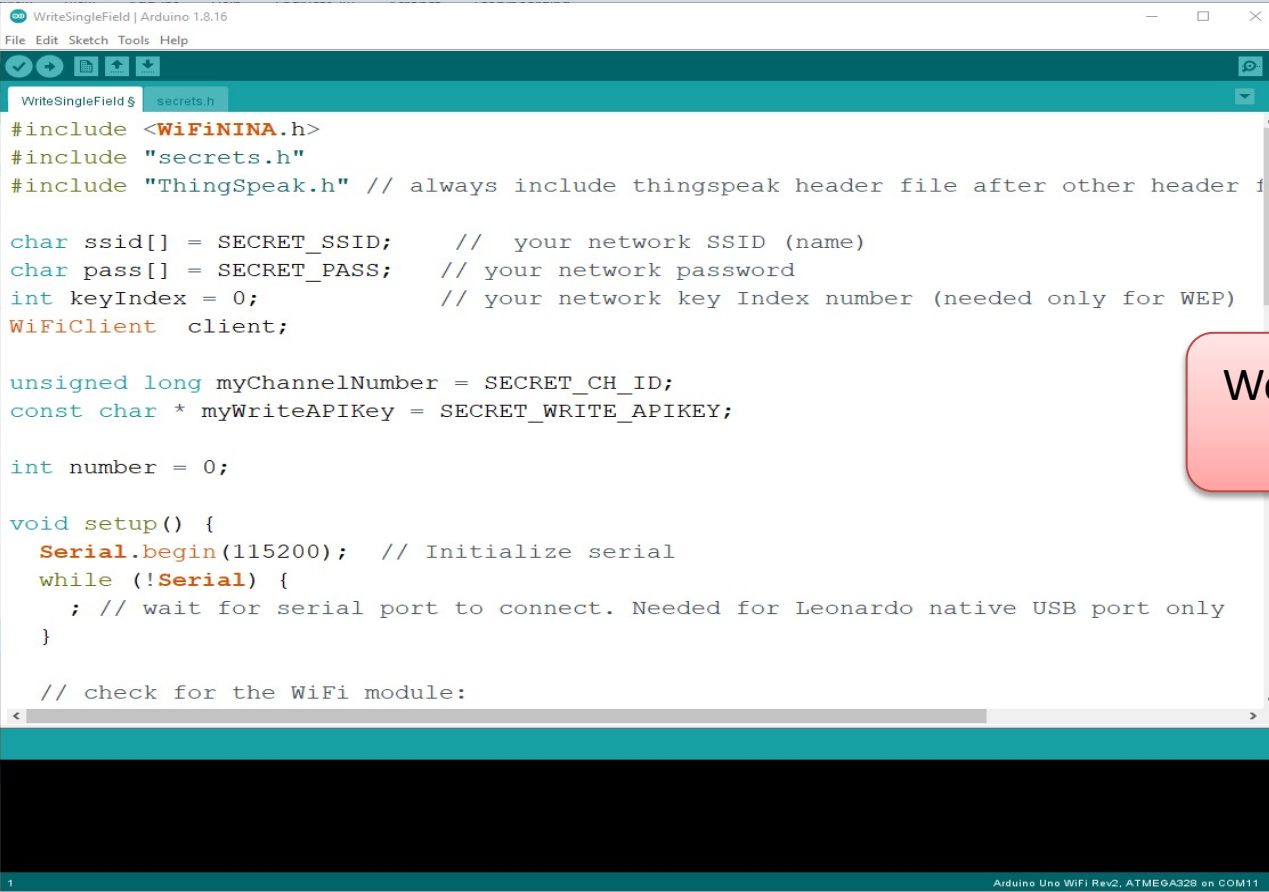
- New Ctrl+N
- Open... Ctrl+O
- Open Recent
- Sketchbook
- Examples**
- Close Ctrl+W
- Save Ctrl+S
- Save As... Ctrl+Shift+S
- Page Setup Ctrl+Shift+P
- Print Ctrl+P
- Preferences Ctrl+Comma
- Quit Ctrl+Q

- 06.Sensors
- 07.Display
- 08.Strings
- 09.USB
- 10.StarterKit_BasicKit
- 11.ArduinoISP
- Examples for any board
 - Adafruit Circuit Playground
 - Bridge
 - Ethernet
 - Firmata
 - LiquidCrystal
 - SD
 - Servo
 - Stepper
 - Temboo
 - WIFININA
 - RETIRED
- Examples for Arduino Uno WiFi Rev2
 - EEPROM
 - SoftwareSerial
 - SPI
 - Wire
- Examples from Custom Libraries
 - Adafruit Unified Sensor
 - Control
 - DAC_MCP49xx
 - DallasTemperature
 - DHT sensor library
 - Fahrenheit
 - MCP_DAC
 - OneWire
 - ThingSpeak**

```
run once:  
  
run repeatedly:
```

- ArduinoEthernet
- ArduinoMKR1000
- ArduinoMKRETHShield
- ArduinoMKRGSM1400
- ArduinoMKRVIDOR4000
- ArduinoMKRWiFi1010
- ArduinoUnoWiFi Rev2
- ArduinoWiFiShield**
 - ReadField
 - WriteMultipleFields
 - WriteSingleField
- ArduinoWiFiShield101
- ArduinoYun
- ESP32
- ESP8266
- extras

Arduino Example



```
WriteSingleField | Arduino 1.8.16
File Edit Sketch Tools Help
WriteSingleField $ secrets.h
#include <WiFiNINA.h>
#include "secrets.h"
#include "ThingSpeak.h" // always include thingspeak header file after other header f

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 number = 0;

void setup() {
  Serial.begin(115200); // Initialize serial
  while (!Serial) {
    ; // wait for serial port to connect. Needed for Leonardo native USB port only
  }

  // check for the WiFi module:
```

We use the built-in Example as a starting point

Arduino Code

Here you see the main code structure:

We have created separate Functions for:

- **CheckWiFi()**
- **ConnectWiFi()**
- **ReadSensorData()**
- **ThingSpeakWrite()**

The Functions are presented on the next pages.

```
#include <OneWire.h>
#include <DallasTemperature.h>
#include "ThingSpeak.h"
#include <WiFiNINA.h>
#include "secrets.h"

#define ONE_WIRE_BUS 2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature
sensors(&oneWire);

float tempCelcius=0;
WiFiClient client;
int wait = 20000;

void setup(void)
{
  Serial.begin(9600);
  sensors.begin();
  CheckWiFi () ;
  ThingSpeak.begin(client);
}

void loop(void)
{
  ConnectWiFi () ;
  ReadSensorData () ;
  ThingSpeakWrite () ;
  delay(wait);
}
```

Arduino Code

Secrets.h

```
#define SECRET_SSID "xxxxxxx"  
#define SECRET_PASS "xxxxxxx"  
  
#define SECRET_CH_ID xxxxxxx  
#define SECRET_WRITE_APIKEY "xxxxxxx"
```

```
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");  
  }  
}  
  
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.");  
  }  
}
```

Arduino Code

```
void ReadSensorData ()
{
    sensors.requestTemperatures ();
    tempCelcius = sensors.getTempCByIndex (0) ;
    Serial.print ("T = ");
    Serial.print (tempCelcius, 1) ;
    Serial.println ("°C");
}
```

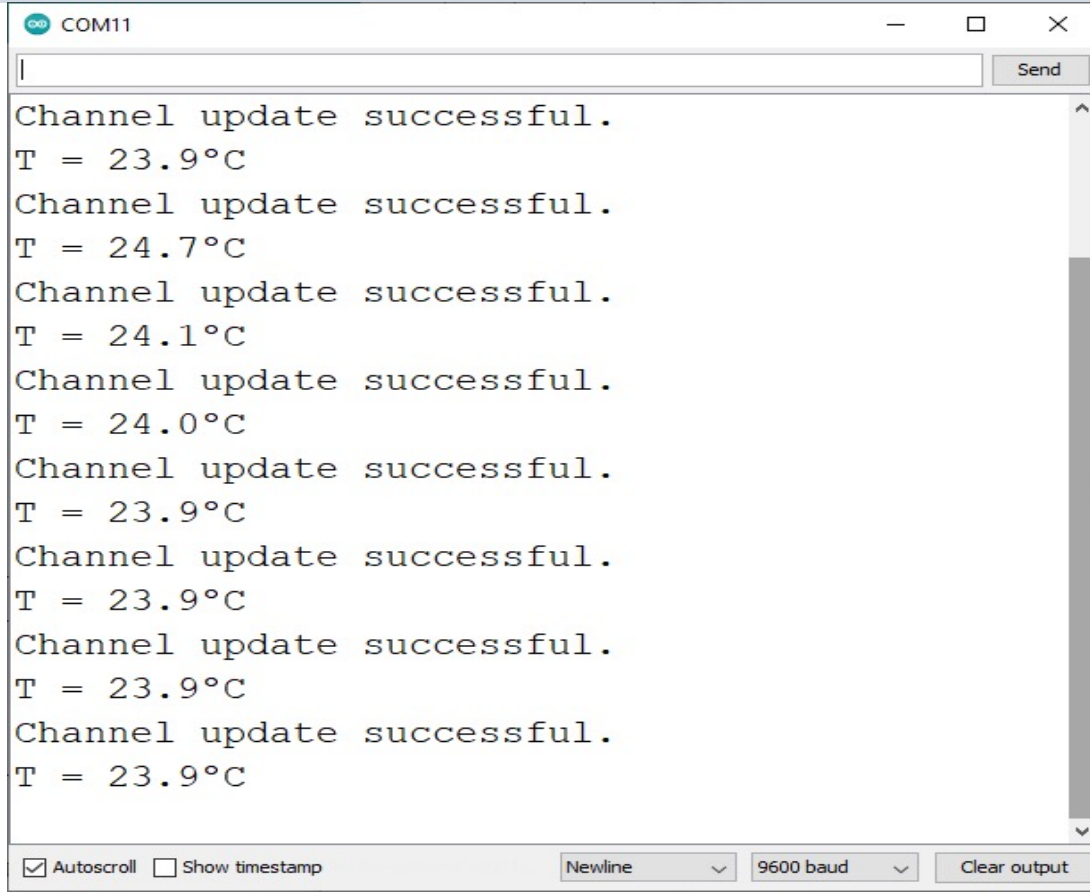

Arduino Code

Secrets.h

```
#define SECRET_SSID "xxxxxxx"  
#define SECRET_PASS "xxxxxxx"  
  
#define SECRET_CH_ID xxxxxxx  
#define SECRET_WRITE_APIKEY "xxxxxxx"
```

```
void ThingSpeakWrite ()  
{  
  unsigned long myChannelNumber = SECRET_CH_ID;  
  const char * myWriteAPIKey = SECRET_WRITE_APIKEY;  
  
  int channelField = 1;  
  
  int x = ThingSpeak.writeField(myChannelNumber, channelField, tempCelcius, myWriteAPIKey);  
  if(x == 200){  
    Serial.println("Channel update successful.");  
  }  
  else{  
    Serial.println("Problem updating channel. HTTP error code " + String(x));  
  }  
}
```

Serial Monitor



The screenshot shows a serial monitor window titled "COM11". At the top, there is a text input field and a "Send" button. The main area contains a log of messages: "Channel update successful." followed by a temperature reading "T = [value]°C". The temperature values are 23.9, 24.7, 24.1, 24.0, 23.9, 23.9, 23.9, and 23.9. At the bottom, there are control options: "Autoscroll" (checked), "Show timestamp" (unchecked), a "Newline" dropdown menu, "9600 baud" (selected), and a "Clear output" button.

```
COM11  
|  
| Send  
Channel update successful.  
T = 23.9°C  
Channel update successful.  
T = 24.7°C  
Channel update successful.  
T = 24.1°C  
Channel update successful.  
T = 24.0°C  
Channel update successful.  
T = 23.9°C  
Channel update successful.  
T = 23.9°C  
Channel update successful.  
T = 23.9°C  
Channel update successful.  
T = 23.9°C  
 Autoscroll  Show timestamp  
Newline 9600 baud Clear output
```

ThingSpeak

ThingSpeak™



Private View

Public View

Channel Settings

Sharing

API Keys

Data Import / Export

+ Add Visualizations

+ Add Widgets

MATLAB Analysis

Export recent data

MATLAB Visualization

More Information

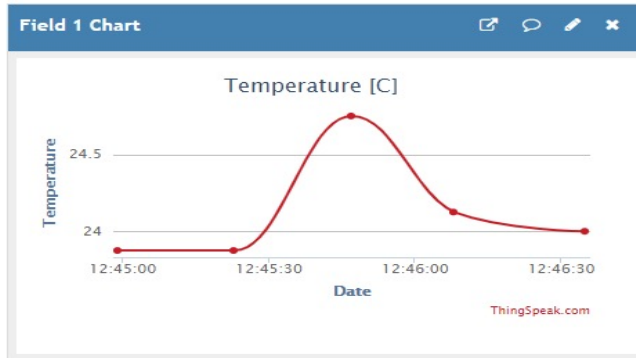
Channel 1 of 3 < >

Channel Stats

Created: [4 years ago](#)

Last entry: [about a minute ago](#)

Entries: 5



References

- <https://create.arduino.cc/projecthub/TheGadgetBoy/ds18b20-digital-temperature-sensor-and-arduino-9cc806>
- <https://create.arduino.cc/projecthub/iotboys/how-to-use-ds18b20-water-proof-temperature-sensor-2adecc>
- <https://lastminuteengineers.com/ds18b20-arduino-tutorial/>

Hans-Petter Halvorsen

University of South-Eastern Norway

www.usn.no

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

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

