Arduino Analog-Thermometer

Mit einem einfachen Temperatursensor und einem Servomotor baust du im Handumdrehen ein analoges Thermometer. Die „Nadel“ des Servos zeigt dir auf einer Schablone die Temperatur zwischen 10°C und 30°C an.

Für dieses Projekt benötigst du:

- Arduino Board (wir verwenden hier einen Uno)
- 1 oder 2 Steckplatine(n) – für 2 getrennte Stromkreise
- 1 Temperatursensor (z.B. einen TMP36)
- 1 Servomotor
- Temperatur-Schablone (z.B. diese hier von instructables.com)
- 9V Batterie mit Anschluss-Clip
- Ein paar Kabel

So sieht das fertige Projekt aus:

Intro

Eine Besonderheit dieses Projekts ist, dass du zwei voneinander getrennte Stromkreise verwendest – und der Strom

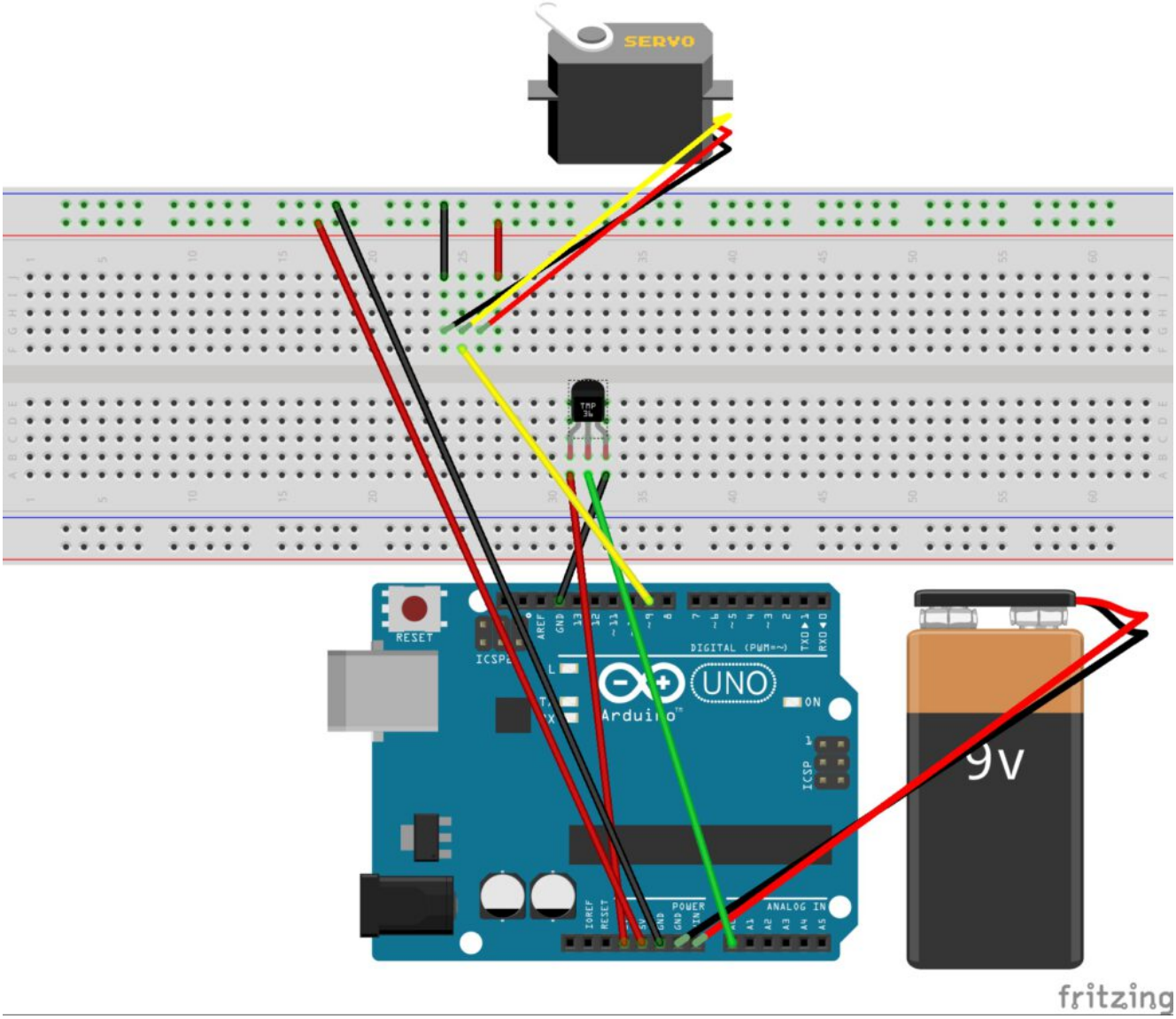
hierfür von einer 9V Batterie kommt. Der Grund dafür ist, dass ein Servomotor recht viel Strom verbraucht, wenn er mit seiner Bewegung anfängt. Das wiederum lässt die Spannung im Stromkreis abfallen, wodurch der Temperatursensor falsche Ergebnisse liefert. **Du benötigst also für dieses Arduino-Projekt eine Steckplatine, auf der du zwei getrennte Stromkreise bauen kannst – oder zwei separate Platinen.**

Den Strom selbst solltest du nicht über die USB-Buchse anlegen, sondern entweder über ein Netzteil oder mit einer 9V Batterie einspeisen. Die 5 Volt, die per USB kommen, reichen für dieses Projekt nicht aus.

Im Programm-Code wirst du den seriellen Monitor der Arduino IDE verwenden. Hierfür musst du das USB-Kabel ebenfalls anschließen – keine Angst, der Strom kommt ausschließlich von der Batterie und nicht von beiden Quellen gleichzeitig.

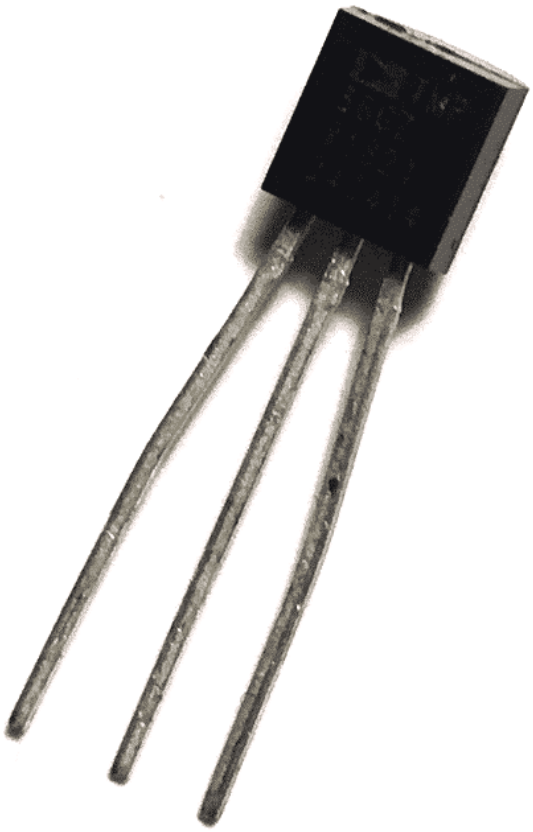
Der Aufbau

Einen möglichen Aufbau siehst du hier:



Du erkennst oben die Batterie, die an die Anschlüsse Vin und GND angeschlossen wird. Den Temperatur-Sensor versorgst du mit 3,3 Volt. Den Servomotor mit 5 Volt.

Wenn du einen TMP36 für die Temperaturmessung verwendest, verbinde den mittleren Pin mit dem Analog-Eingang A0 auf deinem Arduino-Board. Hierüber liest du dann die Temperatur ab.

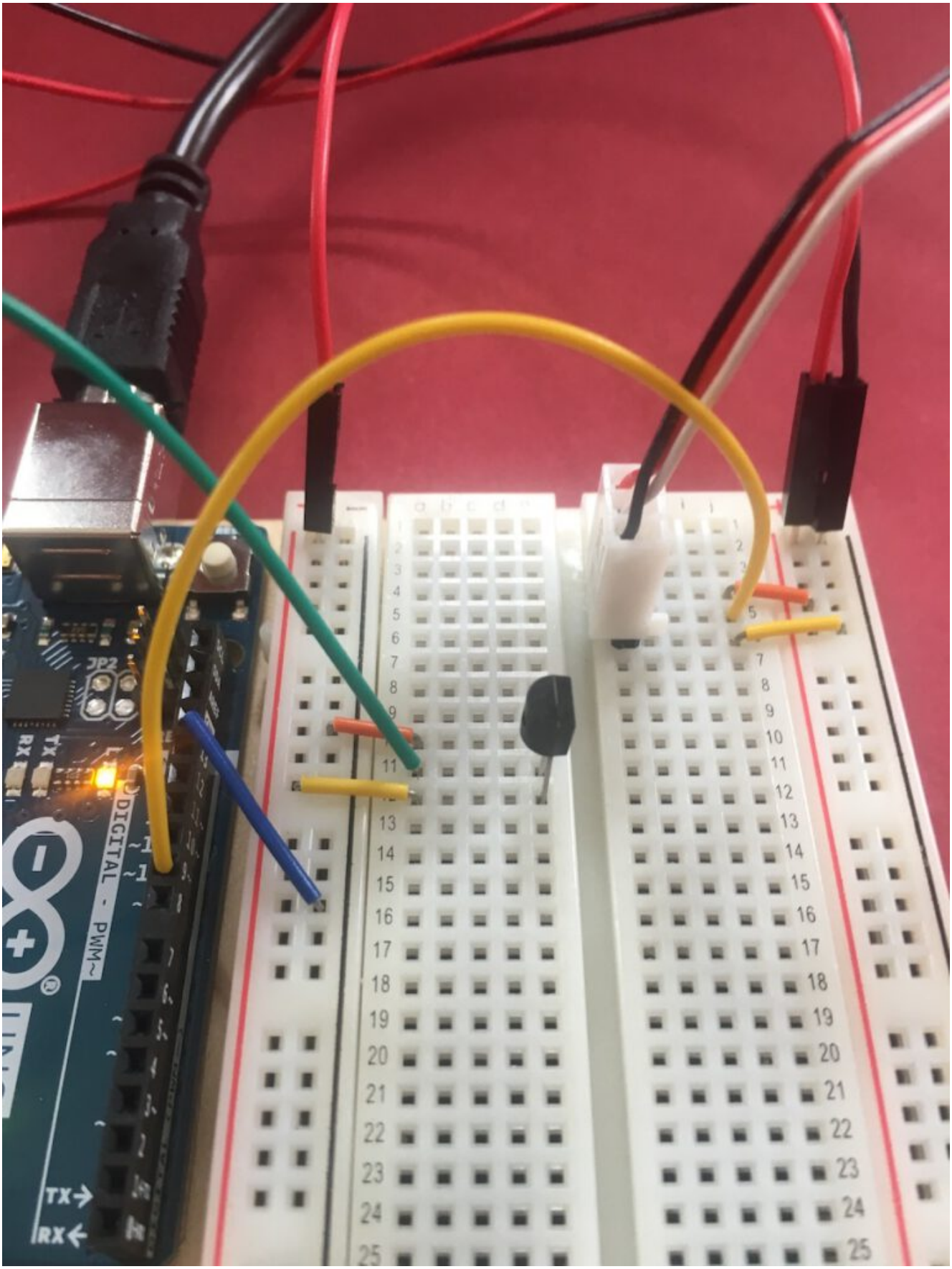


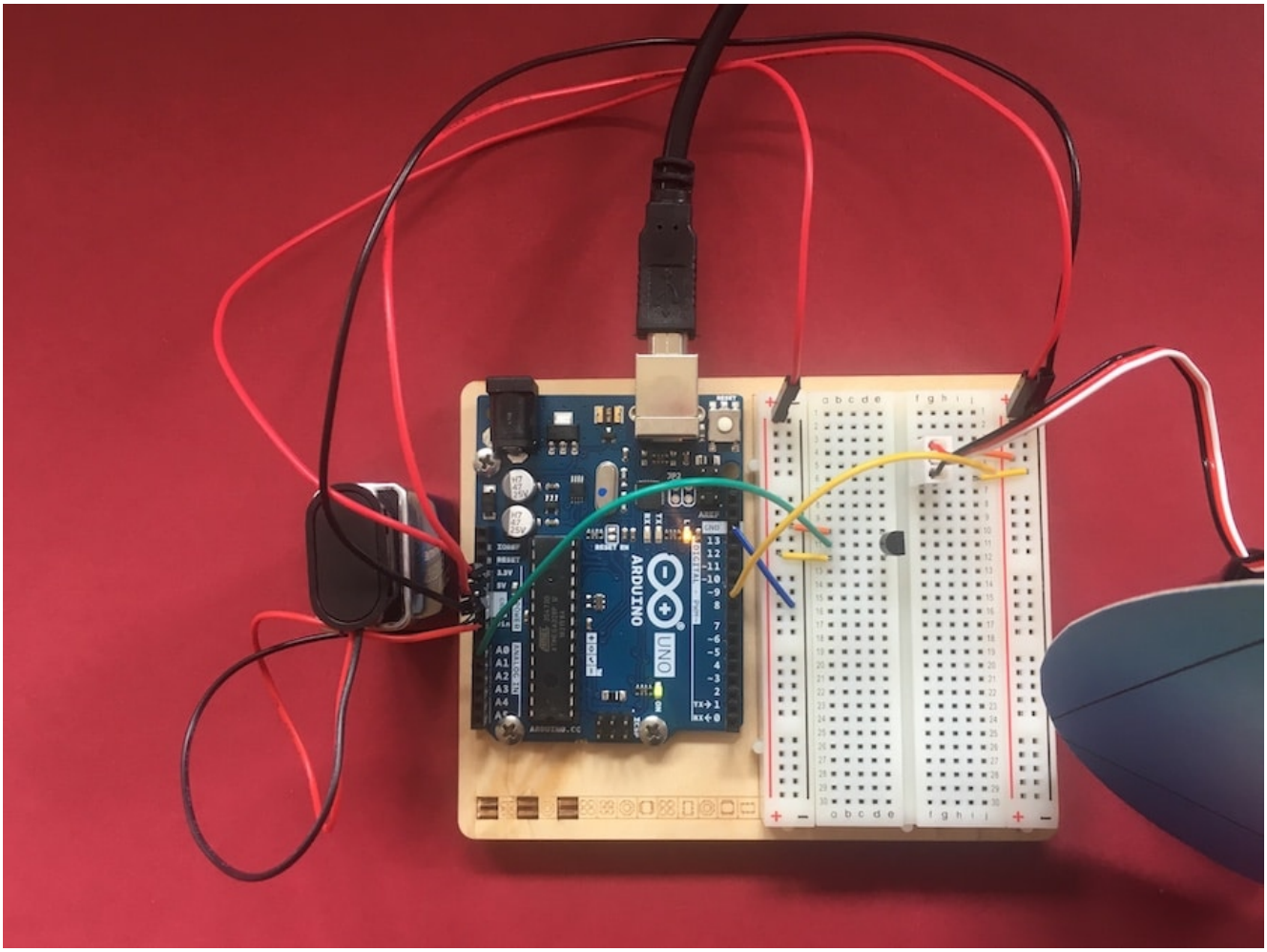
Erfahre alles Wichtige über den Temperatursensor TMP36: Messbereich, Genauigkeit und wie du ihn richtig anschließt.

TMP36 kennenlernen

Die Steuerung des Servomotors (oben das gelbe Kabel) verbindest du mit dem Digital-Ausgang 9.

Das war es im Prinzip auch schon. Bevor du den Strom anlegst, vergewissere dich noch einmal, ob alles richtig verkabelt und eingesteckt ist.


DIGITAL ~ PWM~
TX→
RX←



Die Schablone für die Anzeige

In diesem Projekt zeigt der Servomotor Temperaturen im Bereich von +10°C bis 30°C an – und zwar in einem Halbkreis, also 180°. Eine entsprechende Anzeige kannst du dir selbst gestalten oder einfach diese hier von instuctables.com herunterladen.

Druck sie dir aus, klebe sie auf einen stabilen Karton und schneide den Kreis aus. In der Mitte findest du den Ausschnitt, in den der Motor eingepasst wird.

Das Programm

Hinweis: Den vollständigen Code findest du weiter unten. Zunächst ein paar Zeilen, die noch vor dem Setup stehen.

```
#include <Servo.h>
```

```
Servo myServo;

const int temperaturePin = 0;

int angle;
```

Für die Steuerung des Servomotors verwendest Du die entsprechende Bibliothek, die Dir eine Menge Arbeit abnimmt. Diese Bibliothek sollte bereits bei Dir installiert sein.

Anschließend erstellt du ein Servo-Objekt, legst den Eingangs-Pin für den Termperatur-Sensor fest und erstellst eine Variable für den Winkel des Servomotors. Dieser Winkel entspricht später der Temperatur, auf die der Zeiger des Motors ausgerichtet wird.

Das Setup-Funktion

```
void setup() {
  myServo.attach(9);
  Serial.begin(9600);
  myServo.write(0); //
  delay(2000); //
}
```

Im Setup legst du den Pin 9 fest, um den Servo zu steuern. Anschließend richtest du die Verbindung zum seriellen Monitor in deiner IDE ein. Den Motor fährst du dann erst mal auf den Winkel 0° und gibst ihm hierfür 2000 Millisekunden Zeit.

Übrigens: Alles Wissenswerte zu Servos erfährst du in diesem Tutorial.

Der Loop und eine Extra-Funktion

Hier spielt die eigentliche Musik: Du liest den Temperatursensor aus, rechnest ein paar Werte um und bewegst den Servomotor zur richtigen Gradzahl.

Zunächst eine Funktion, die Du benötigst, um den Ausgabewert

des Temperatursensors in Volt umzurechnen:

```
float getVoltage(int pin)
{
  return (analogRead(pin) * 0.004882814);
}
```

Diese Funktion arbeitet mit einem Argument, „int pin“. Hierfür wirst du im Folgenden den Pin des Sensors nehmen. Über „return“ liest du mit der Funktion „analogRead“ den entsprechenden Wert ein und rechnest ihn in Volt um.

Jetzt zum Loop:

```
void loop() {

  float voltage, degreesC;

  voltage = getVoltage(temperaturePin);
  degreesC = (voltage - 0.5) * 100.0;

Serial.print("  voltage: ");
  Serial.print(voltage);
  Serial.print("  deg C: ");
  Serial.println(degreesC);

  angle = map(degreesC, 10, 30, 0, 180);

  if(degreesC < 10){
    delay(50);
    }else if(degreesC > 30){
      delay(50);
      }else{
         myServo.write(angle);
        }
 delay(1000);
}
```

Zunächst erstellst du zwei Variablen für Volt und Grad Celsius. Die Spannung berechnest du, in dem du die oben genannte Funktion mit dem Argument „temperaturePin“ aufrufst – die Konstante, die du ganz am Anfang als Ausgang A0 deklariert

hast. Die °C berechnest du dann wiederum aus der gerade berechneten Spannung.

Über „Serial.print“ gibst du die errechneten Werte nun an den seriellen Monitor weiter, um dort zu überprüfen, ob der Sensor auch seine Arbeit macht und die Daten plausibel sind.

Eine wichtige Funktion ist die folgende: Hier „mappst“ du die Temperaturwerte auf die möglichen Winkel des Servomotors:

```
  angle = map(degreesC, 10, 30, 0, 180);
```

Die Funktion „map“ benötigt zunächst einen Wert, den sie „mappen“ soll – in unserem Fall „degreesC“, also die Temperatur. Der zulässige Bereich liegt zwischen 10 und 30 °C – das sind also die ersten beiden Zahlen. Der Servomotor soll sich auf unserer Skala in einem Bereich von 180° bewegen – bei 0° liegen die 10°C und bei 180° die 30°C. Trage also noch die beiden Zahlen 0 und 180 ein. Fertig.

Nun zu dem Teil, der den Servomotor dazu bringt, die Temperatur auf deiner Skala anzuzeigen:

```
if(degreesC < 10){
    delay(50);
    }else if(degreesC > 30){
      delay(50);
      }else{
         myServo.write(angle);
        }
```

Da deine Skala bei +10°C anfängt und bei +30°C aufhört, musst du dem Motor mitteilen, dass er nichts tun soll, wenn die Temperatur darunter oder darüber liegt. Sollte die Temperatur aber in diesem Bereich liegen, dann gibst du den Befehl, dass er sich an den Winkelgrad bewegen soll, den du über die „map“-Funktion oben errechnet hast.

Ganz zuletzt gibst dem Motor per „delay“ 1000ms Zeit, seinen Befehl auszuführen. Das war’s! Lade den Sketch auf deinen

Arduino, setze den Servomotor in die Anzeige und probiere es gleich aus. Ein letzter Hinweis: Zu Beginn des Programms fährt dein Motor auf 10°C, also ganz nach links unten und wartet dort 2 Sekunden. Überprüfe, ob auch der Zeiger auch tatsächlich auf die 10°C zeigt. Falls nicht, passe ihn einfach an.

Hier der gesamte Sketch:

```
#include <Servo.h>

Servo myServo;

const int temperaturePin = 0;

int angle;

void setup() {
  myServo.attach(9);
  Serial.begin(9600);
  myServo.write(0);
  delay(2000);

void loop() {

  float voltage, degreesC;

  voltage = getVoltage(temperaturePin);
  degreesC = (voltage - 0.5) * 100.0;

  Serial.print("  voltage: ");
  Serial.print(voltage);
  Serial.print("  deg C: ");
  Serial.println(degreesC);

  angle = map(degreesC, 10, 30, 0, 180);

  if(degreesC < 10){
    delay(50);
    }else if(degreesC > 30){
```

```
      delay(50);
      }else{
         myServo.write(angle);
        }
 delay(1000);

float getVoltage(int pin)
{
  return (analogRead(pin) * 0.004882814);
}
```

Wie geht es weiter?

Mit dem TMP36 kannst du „nur“ die Temperatur der Umgebung messen. Möchtest du jedoch aus sicherer Entfernung die Temperatur eines Objekts messen, dann benötigst du einen Sensor, der per Infrarot misst – z.B. den GY-906. Lerne hier, wie du den Temperatursensor GY-906 anschließt und verwendest.

Wenn du überhaupt keinen Temperatursensor verwenden willst, ist vielleicht eine Abfrage bei einem Wetterdienst das Richtige: Hier erfährst du, wie du aktuelle Wetterdaten mit einem ESP8266 aus dem Internet abrufst.