Detaillierte Anleitung: Bodenfeuchtigkeit messen (1 LED)

- Installation Thonny
- Das Raspberry Pi Pico mit Thonny programmieren
- Zusammenbau
- Der Code

Installation Thonny

Wir programmieren das Raspberry Pi Pico mit der Skriptsprache MicroPython in der kostenlosen Entwicklungsumgebung (IDE) Thonny. Dafür verwenden wir einen Computer oder Laptop.

Thonny	https://thonny.org/
Micropython	https://micropython.org/

Was ist eine IDE?

Eine IDE (Integrated Development Environment) ist eine Software, die euch beim Schreiben, Testen und Ausführen von Code (Programmen) unterstützt. Sie vereint viele hilfreiche Werkzeuge an einem Ort, darunter:

- **Texteditor:** Zum Schreiben und Bearbeiten von Code.
- **Debugger:** Zum Finden und Beheben von Fehlern im Code.
- **Terminal:** Zum Ausführen des Codes und Anzeigen von Ergebnissen.

Was ist Thonny?

Thonny ist eine einfache und benutzerfreundliche IDE, die speziell für Python entwickelt wurde. Sie eignet sich besonders gut für Einsteiger*innen, die das Programmieren gerade erst lernen. Thonny bietet eine übersichtliche Benutzeroberfläche und viele hilfreiche Funktionen, die den Einstieg ins Programmieren erleichtern. Es ist ein großartiges Tool, um erste Schritte mit Python und MicroPython zu machen.

drawing drawing

Installation von Thonny auf verschiedenen Betriebssystemen

Installation auf Windows

- **Gehe zur offiziellen Thonny-Website.**
- Klicke auf den Download für Windows.
- Lade die Installationsdatei herunter und öffne sie, wenn der Download abgeschlossen ist.

- Folge den Anweisungen des Installationsassistenten, um Thonny zu installieren.
- Sobald die Installation abgeschlossen ist, kannst du Thonny über das Startmenü öffnen.

Installation auf Linux (Ubuntu)

- **Öffne das Terminal** auf deinem Ubuntu-System.
- Gib den folgenden Befehl ein, um das Paket zu aktualisieren:

```
sudo apt update
```

- Installiere Thonny, indem du den folgenden Befehl eingibst:

```
sudo apt install thonny
```

- Warte, bis die Installation abgeschlossen ist. Danach kannst du Thonny im Anwendungsmenü finden und starten.

Installation auf macOS

- **Gehe zur offiziellen Thonny-Website.**
- Klicke auf den Download für macOS.
- Lade die .dmg-Datei herunter und öffne sie, wenn der Download abgeschlossen ist.
- Ziehe das Thonny-Symbol in den Programme-Ordner, um die Installation abzuschließen.
- Öffne Thonny, indem du es im Programme-Ordner findest oder über Spotlight suchst.

> **Übung**
>
> Versuche nach der Installation, siehe unten, über die IDE Thonny eine Bibliothek zu installieren, zum Beispiel die Bibliothek 'NeoPixel'.
>
> - Thonny öffnen ->
> - Menü -> Tools -> Manage Plug-ins
> - Suche nach 'neopixel' -> installieren

Das Raspberry Pi Pico mit Thonny programmieren

Um ein neues Raspberry Pi Pico zu programmieren, müssen wir es zunächst vorbereiten. Es mag anfangs nach vielen Schritten klingen, aber sobald ihr es einmal gemacht habt, geht der Rest richtig schnell! (□ ‘‿ ’□) Wir teilen den Prozess in drei Schritte auf:

1. **Installation von MicroPython auf dem Raspberry Pi Pico**
2. **Raspberry Pi Pico mit Thonny öffnen**
3. **Ein Programm auf dem Raspberry Pi Pico speichern/laden**

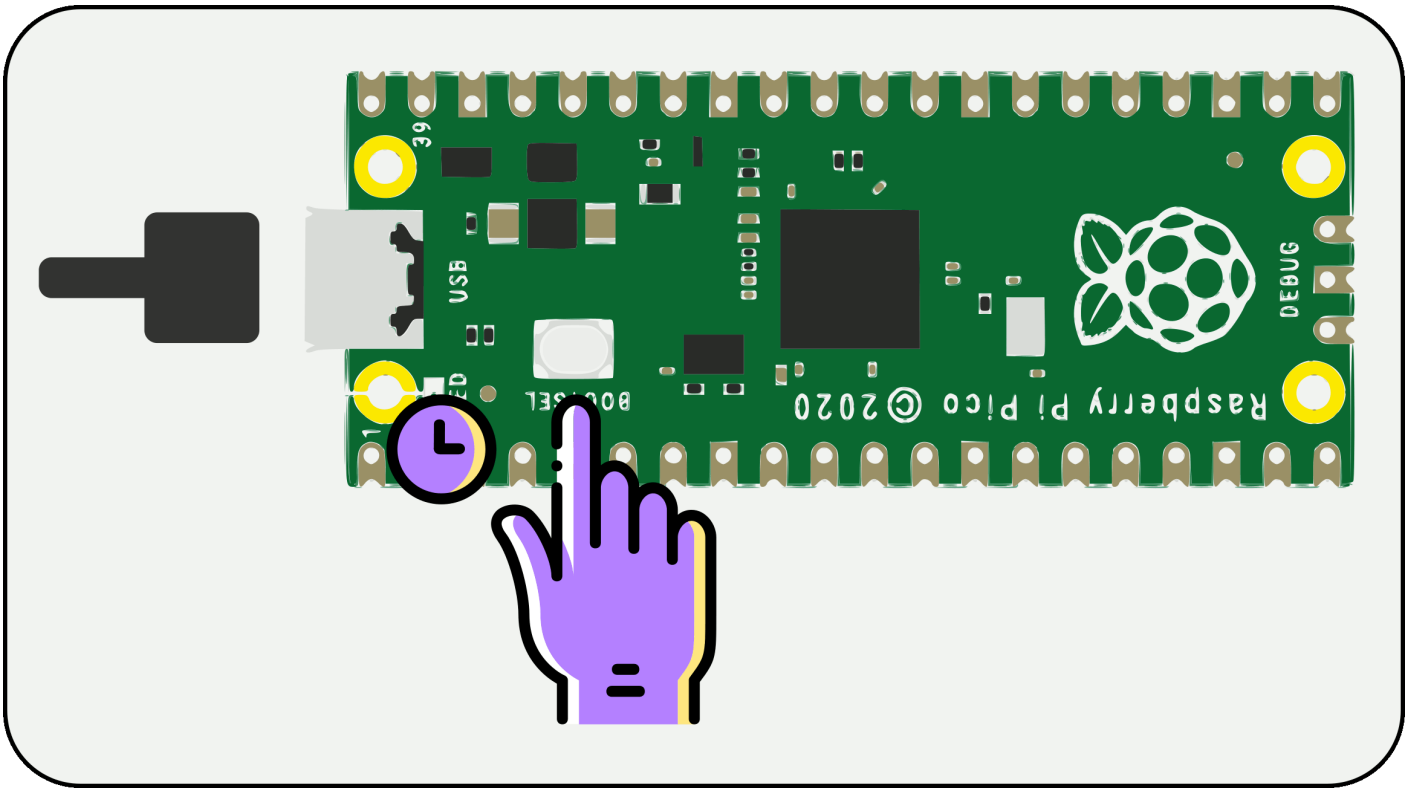
Installation von Micropython auf dem Raspberry Pi Pico

- **Ladet euch Micropython herunter:**

  Besucht die MicroPython-Website und ladet die passende UF2-Datei für das Raspberry Pi Pico herunter.

  (Zum Beispiel v1.24.0 (2024-10-25) .uf2)
- **Schließt das Raspberry Pi Pico an den Computer an:**
  Verwendet ein Micro-USB-zu-USB-A-Kabel. Haltet dabei den BOOTSEL-Knopf beim Raspberry Pi Pico gedrückt, während ihr das USB-Kabel anschließt.
  (siehe Animation)
- **Kopiert die UF2-Datei auf das neue Laufwerk:**
  Sobald das Pico als neues Laufwerk auf eurem Computer erscheint, zieht die heruntergeladene UF2-Datei per Drag-and-Drop in dieses Laufwerk. Dadurch wird MicroPython auf dem Raspberry Pi Pico installiert.

> Während der Installation oder dem Laden von MicroPython auf das Raspberry Pi Pico trennt sich das Laufwerk automatisch vom Computer. Dies zeigt an, dass die Installation abgeschlossen ist. Zieht auf keinen Fall das Kabel während dieses Prozesses vom Pico ab, da dies die Installation unterbrechen und zu Fehlern führen könnte.
>
> Sicherheit geht vor – lasst das Pico in Ruhe arbeiten! (□ ^ ᵕ ^□)



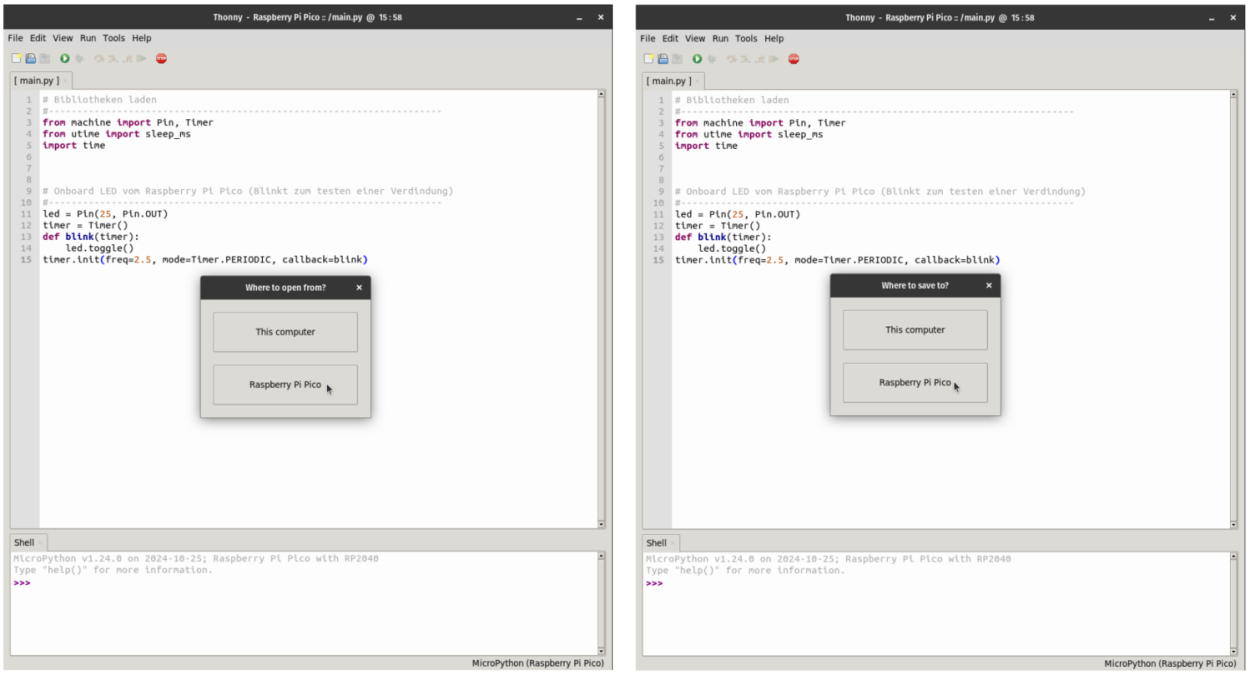
Mit Thonny ein Programm auf dem Raspberry Pi Pico speichern/laden

Nachdem ihr Thonny geöffnet habt, könnt ihr über das Menü die Option **Dateien/Files** auswählen, um Dateien zu verwalten.

- Mit der Option **Öffnen/Open** öffnet sich ein Dialog, in dem ihr auswählen könnt, ob ihr eine Datei von eurem Computer oder direkt vom Raspberry Pi Pico laden möchtet.
- Mit der Option **Speichern/Save** öffnet sich ein Dialog, in dem ihr auswählen könnt, wo ihr euer Programm speichern möchtet – entweder auf dem Computer oder auf dem Raspberry Pi Pico.

So könnt ihr eure Programme einfach verwalten und sicherstellen, dass sie immer an der richtigen Stelle gespeichert sind!

> Achtet beim Speichern auf dem Raspberry Pi Pico darauf, dass die Datei den Namen **main.py** hat. Nur mit diesem Dateinamen erkennt das Pico euer Programm automatisch und führt es nach dem Starten aus. ᓚ(^. .^)



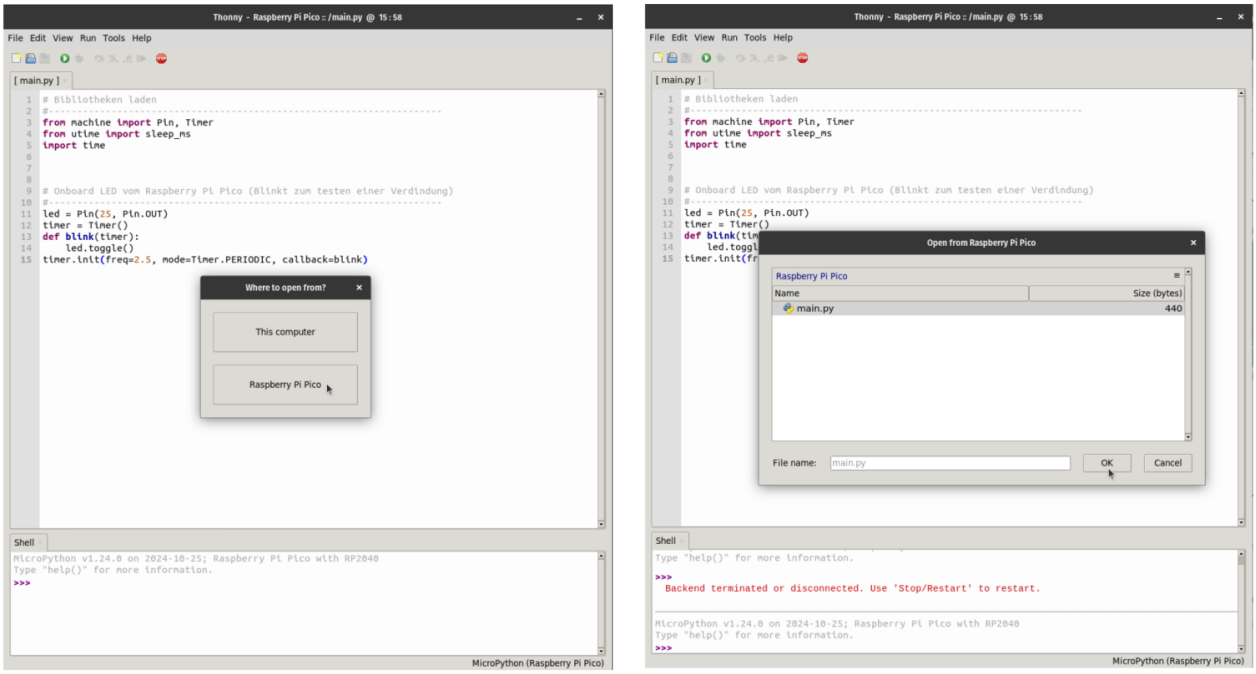
Raspberry Pi Pico mit Thonny öffnen

Normalerweise kommuniziert Thonny automatisch mit dem Raspberry Pi Pico, sobald es angeschlossen ist. Wenn ihr Thonny geöffnet habt, geht wie folgt vor:

1. Wählt im Menü die Option **Dateien/Files** aus.
2. Im neuen Dialog könnt ihr das **Raspberry Pi Pico** als Speicherort auswählen.
3. Anschließend könnt ihr eure **main.py**-Datei auf dem Pico finden, auswählen und öffnen.

So könnt ihr sicherstellen, dass euer Programm korrekt geladen und ausgeführt wird!

> Nachdem ihr eure **main.py** zum Öffnen ausgewählt habt, könnt ihr das Programm in Thonny bearbeiten. Wenn ihr Änderungen vornehmt und auf **Speichern** klickt, wird das Programm automatisch auf dem Raspberry Pi Pico aktualisiert und gespeichert. So bleiben eure Änderungen direkt auf dem Pico erhalten! ¯_(□)_/¯



Programm starten/stoppen

Nachdem ihr ein Programm erfolgreich auf das Raspberry Pi Pico gespeichert oder übertragen habt, könnt ihr es ausführen und bei Bedarf stoppen:

- **Starten:** Klickt in der Symbolleiste auf den runden **grünen Button**, um das Programm auf dem Raspberry Pi Pico zu starten. Alternativ könnt ihr die **Taste F5** drücken.
- **Stoppen:** Klickt in der Symbolleiste auf den **roten Stop-Button**, um das Programm zu stoppen. Alternativ könnt ihr die Tastenkombination **Strg + F2** verwenden.

Buttons in Thonny zur Steuerung des Programmes auf dem Raspberry Pi Pico

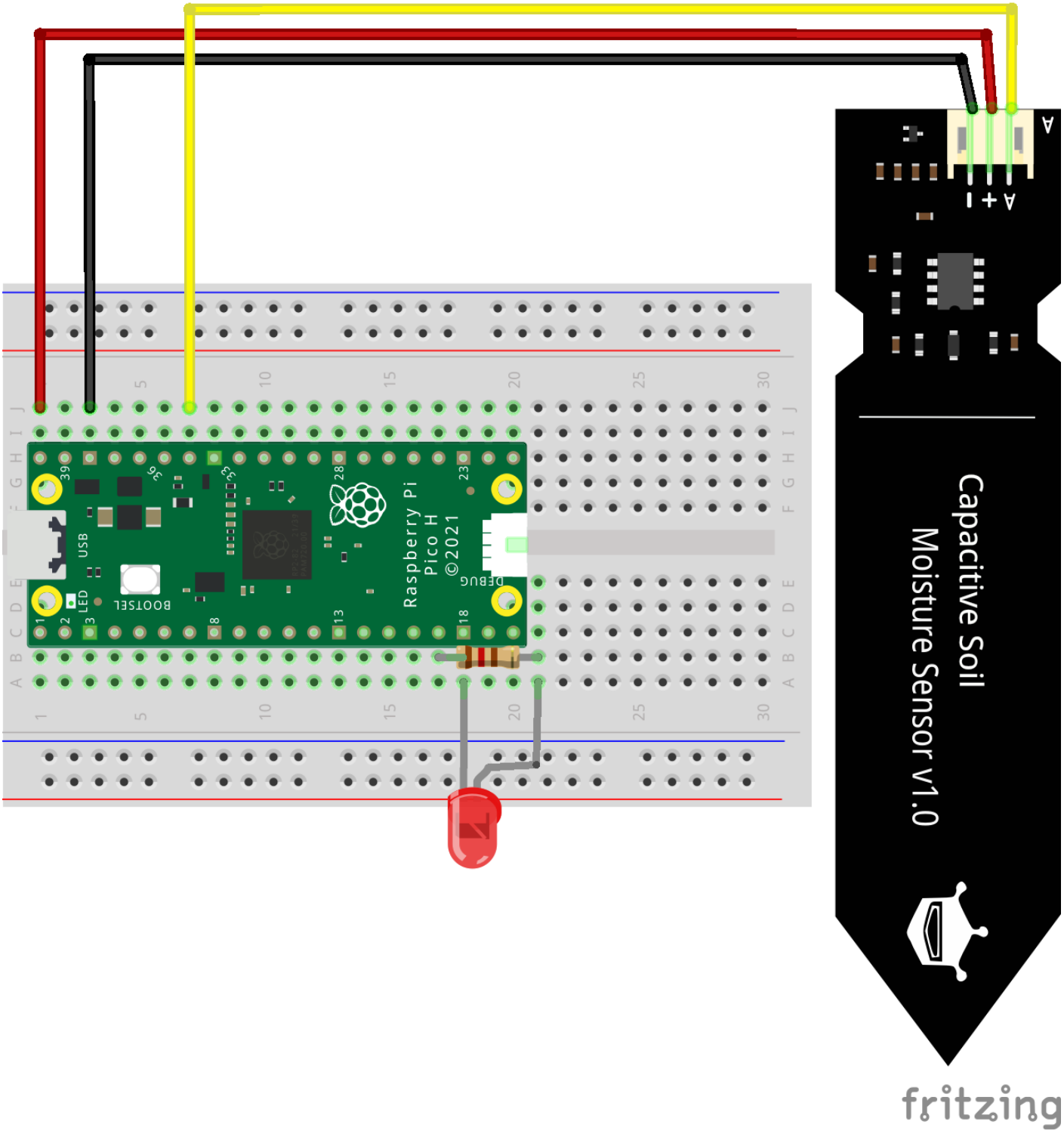


Programm **starten**



Programm **stoppen**

Zusammenbau



Beispiel-Verkabelung (siehe Abbildung oben)**:**
In der Abbildung werden rote, schwarze und grüne Kabel (Jumper) verwendet, die jeweils eine spezifische Funktion haben:

> - Rot (Power/PWR): Liefert Energie an das Bauteil, damit es funktioniert.
> - Schwarz (Ground/GRD): Schließt den Stromkreis und leitet überschüssige Energie ab.
> - Grün (Data): Überträgt die Daten zwischen dem Raspberry Pi Pico und dem Bauteil.

1) Stromversorgung einrichten:

- Verbinde den **5V-Pin** des Raspberry Pi Pico mit der **positiven Leiste** (rote Linie) des Breadboards.
- Verbinde einen **GND-Pin** des Raspberry Pi Pico mit der **negativen Leiste** (blaue Linie) des Breadboards.
  Jetzt können alle Bauteile auf dem Breadboard mit Strom versorgt werden.

2) Bodenfeuchtigkeitssensor:

- Stecke ein Jumper-Kabel vom **VCC-Pin** des Relaismoduls in die **positive Leiste** des Breadboards.
- Verbinde den **GND-Pin** des Relaismoduls mit der **negativen Leiste** des Breadboards.
- **Analogausgang (A2):** Verbinde den **A2-Pin** des Sensors mit **GPIO-Pin 28 (ADC2)** des Pico.

3) LED Dioden (rot, gelb, grün):

> Wir verbinden die Anode (+) jeder LED über einen Vorwiderstand mit einem GPIO-Pin des Pico und die Kathode (-) mit der Masse (GND).

a) Rote LED:

- **Anode (+)** (längeres Beinchen) über einen **120 Ω Widerstand** mit **GPIO-Pin 13** (GP13) des Pico verbinden.
- **Kathode (-)** (kürzeres Beinchen) mit einem **GND-Pin** des Pico verbinden.

Der Code

Hier ist ein Beispielcode für eure Programmierung in Micropython in der IDE Thonny auf dem Raspberry Pi Pico.

```
# Bibliotheken laden
#------------------------------------------------------------------
from machine import Pin, ADC
import time

# Pins für die LEDs
led_red = Pin(13, Pin.OUT)
led_yellow = Pin(14, Pin.OUT)
led_green = Pin(15, Pin.OUT)

# ADC-Pin für den Bodenfeuchtigkeitssensor
#------------------------------------------------------------------
soil_sensor = ADC(28)

# Schwellenwerte für Feuchtigkeit
#------------------------------------------------------------------
DRY_THRESHOLD = 65000    # Über diesem Wert wird es als trocken betrachtet
MOIST_THRESHOLD = 43000  # Über diesem Wert wird es als feucht betrachtet
WATERED_THRESHOLD = 41000 # Über diesem Wert wird es als gegossen betrachtet

def read_soil_moisture():
    # Bodenfeuchtigkeit auslesen
    return soil_sensor.read_u16()

while True:
    moisture_level = read_soil_moisture()

# Boden ist trocken, rote LED leuchtet
#------------------------------------------------------------------
    if moisture_level > DRY_THRESHOLD:
        led_red.value(1)
        led_yellow.value(0)
        led_green.value(0)
```

```
#print("Status: Der Boden ist trocken")

# Boden hat wenig Feuchtigkeit, gelbe LED leuchtet
#---------------------------------------------------------------------
    elif moisture_level > MOIST_THRESHOLD:
        led_red.value(0)
        led_yellow.value(1)
        led_green.value(0)
#print("Status: Der Boden ist mäßig feucht")

# Boden ist feucht genug, grüne LED leuchtet
#---------------------------------------------------------------------
    else:
        led_red.value(0)
        led_yellow.value(0)
        led_green.value(1)
#print("Status: Der Boden ist feucht")

# Trennlinie für bessere Lesbarkeit
#---------------------------------------------------------------------
#print("-----------------------------")

# Warte eine Sekunde bevor der nächste Wert gemessen wird
#---------------------------------------------------------------------
    time.sleep(1)
    # Aktuellen Feuchtigkeitswert und Schwellenwerte ausgeben
    print("Feuchtigkeitswert:", moisture_level)
```