LED-Streifen mit Bewegungssensor

Lass es leuchten, wenn du dich bewegst!

- Methodenkarte: LED-Streifen mit Bewegungssensor
- LED-Streifen löten
- Installation Thonny
- Das Raspberry Pi Pico mit Thonny programmieren
- Zusammenbau
- Der Code
- Online-Ressourcen

Methodenkarte: LED-Streifen mit Bewegungssensor

Lass es leuchten, wenn du dich bewegst!

Zielgruppe	Dauer	Level	Gruppengröße
ab 8 Jahre	5 bis 10 Stunden	3	3 TN's

Kurzbeschreibung

In diesem Physical-Programming-Projekt steuert ein Raspberry Pi Pico einen LED-Streifen mithilfe eines Bewegungssensors. Bei Bewegung leuchtet der LED-Streifen in verschiedenen Farben. Das Projekt verbindet technisches Wissen mit kreativer Teamarbeit und zeigt, wie digitale Technologien im Alltag genutzt werden können.

> **Ziele**
> *- Verständnis von Stromkreisen*
> *- Kreativer Umgang mit Code und digitalen Technologien*
> *- Fördern von Teamfähigkeit*

Material	Werkzeug
• 1 Raspberry Pi Pico • 1 Breadboard • 1 Jumper, Dupont Kabel Set • 1 LED-Streifen mit 15 LEDs • 1 Schrumpfschlauch • 1 Bewegungssensor (PRI-Sensor)	• Lötset/-station • Zangensatz • Abisolierzange • Heißluftpistole • Sortier-/Projektbox

Ablauf

1. **Vorbereitung (Projektleitung):**
   Das Material wird vorbereitet, auf Funktionalität getestet und ein Demomodell wird erstellt

(für weitere Hinweise und Ressourcen siehe QR-Code). Dazu gehören das richtige Einstecken von LEDs und Widerständen sowie die Installation der Programmierumgebung Thonny. Dieser Schritt dient der reibungslosen Durchführung und ersten Orientierung. Je nach Alter, Vorwissen und Beziehungsarbeit kann die Workshopzeit reduziert werden (z.B. bei den Phasen Vorbereitung und Projektstart).

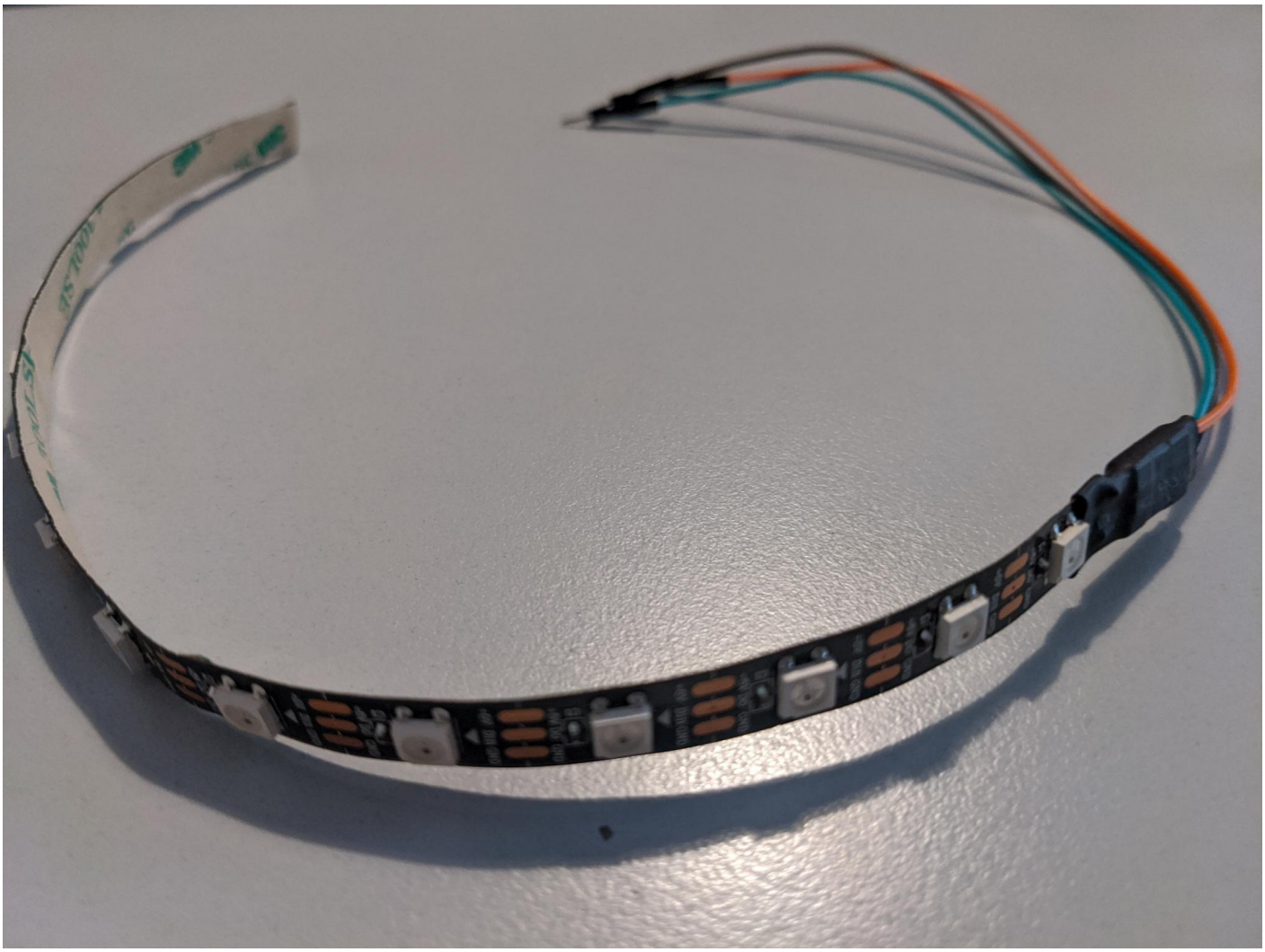
2. **Projektstart:**
   Nach einer Begrüßungsrunde teilt die Workshopleitung die Teilnehmenden in kleine Gruppen (2 bis 3 Personen) ein. Die Gruppen können sich Namen und Rollen geben, um die Zusammenarbeit zu fördern. Anschließend verteilt die Workshopleitung Projektboxen mit allen benötigten Materialien.
3. **Praktische Arbeit, Aufbau und Test**
   Die Gruppen montieren eigenständig das Bewegungsmelder-Set. Dabei testen sie die Funktionalität, indem simulierte Feuchtigkeitswerte durch LEDs angezeigt werden. Unterstützung der Workshopleitung erfolgt bei Bedarf.
4. **Programmierung:**
   Die Gruppen schließen den Raspberry Pi Pico mit Thonny an und passen den Code an. Sie verändern Schwellenwerte für den Sensor und machen die Daten in der Konsole sichtbar, um ein besseres Verständnis für die Programmierlogik zu entwickeln.
5. **Reflexion:**
   Zum Abschluss reflektieren die Gruppen ihre Zusammenarbeit und Ergebnisse. Anregungen und Ideen für Verbesserungen oder Folgeprojekte werden diskutiert, um die Lernerfahrung abzurunden.

Autor*in: Shelly Pröhl (*Büro Berlin des JFF*)

LED-Streifen löten

Vorbereitung:

Lege alles bereit, was du brauchst: den WS2812B-LED-Streifen (15 LEDs), drei Jumper-Kabel (Out/Out, männlich/männlich), einen Lötkolben, Lötzinn und eine Halterung, um den LED-Streifen beim Löten zu fixieren. Für den Schutz der Lötstellen benötigst du **Schrumpfschlauch** oder **Isolierklebeband**. Wenn du Schrumpfschlauch verwenden möchtest, brauchst du zusätzlich einen **Fön** oder eine **Heißluftpistole**.



1. **Kabel zuordnen:**
   Schau dir die drei Kontaktstellen auf dem LED-Streifen an. Sie sind mit **+5V** (Power), **GND** (Ground) und **DIN** (Data In) beschriftet. Diese entsprechen den Farben der Jumper-Kabel:
   - **Rotes Kabel:** für **+5V** (Power).
   - **Schwarzes Kabel:** für **GND** (Ground).
   - **Grünes oder gelbes Kabel:** für **DIN** (Data In).
2. **Löten:**

- Halte den LED-Streifen sicher in der Halterung oder fixiere ihn mit einer dritten Hand.
- Erwärme mit dem Lötkolben vorsichtig die Kontaktstelle **+5V** auf dem LED-Streifen und füge etwas Lötzinn hinzu.
- Löte das **rote Kabel** an diese Kontaktstelle. Achte darauf, dass die Verbindung fest ist.
- Wiederhole den Vorgang für **GND** (schwarzes Kabel) und **DIN** (grünes oder gelbes Kabel).

> **Überprüfen:**
>
> Schau dir die Lötstellen genau an. Sie sollten glänzen und fest verbunden sein, ohne dass die Lötstellen einander berühren. Falls etwas nicht richtig sitzt, kannst du die Lötstelle vorsichtig erneut erhitzen und verbessern.

Schutz und Stabilisierung:

- **Mit Schrumpfschlauch:**
  - Schneide kleine Stücke Schrumpfschlauch (ca. 1–2 cm) zurecht.
  - Schiebe die Schrumpfschläuche über die Lötstellen, bevor du das Kabel verlötest.
  - Nach dem Löten schiebst du den Schrumpfschlauch über die Lötstelle.
  - Erhitze den Schrumpfschlauch mit einem Fön oder einer Heißluftpistole, bis er sich fest um die Lötstelle zieht.
- **Mit Isolierklebeband:**
  - Wenn du keinen Schrumpfschlauch hast, wickle Isolierklebeband fest um die Lötstellen, sodass sie komplett abgedeckt und geschützt sind.

Prüft ob ihr alle Werkzeuge bereits habt! Wenn nicht, findet ihr hier eine Tabelle mit vorschlägen:

Material-/Werkzeugliste	Link	Einzelpreis	URL-Datum
Heißluftpistole	Shop Obi	39,99 EUR	27.11.2024
Schrumpfschlauch-Set	Shop Berrybase	2,50 EUR	27.11.2024
Universalschere	Shop Böttcher	0,99 EUR	27.11.2024
Isolierband (schwarz)	Shop Böttcher	9,99 EUR	27.11.2024

> **Jetzt sind deine Lötstellen nicht nur stabil, sondern auch geschützt vor Kurzschlüssen und Beschädigungen. Dein LED-Streifen ist bereit für den Einsatz!** □□□□ ˙□ ˙)✧

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**

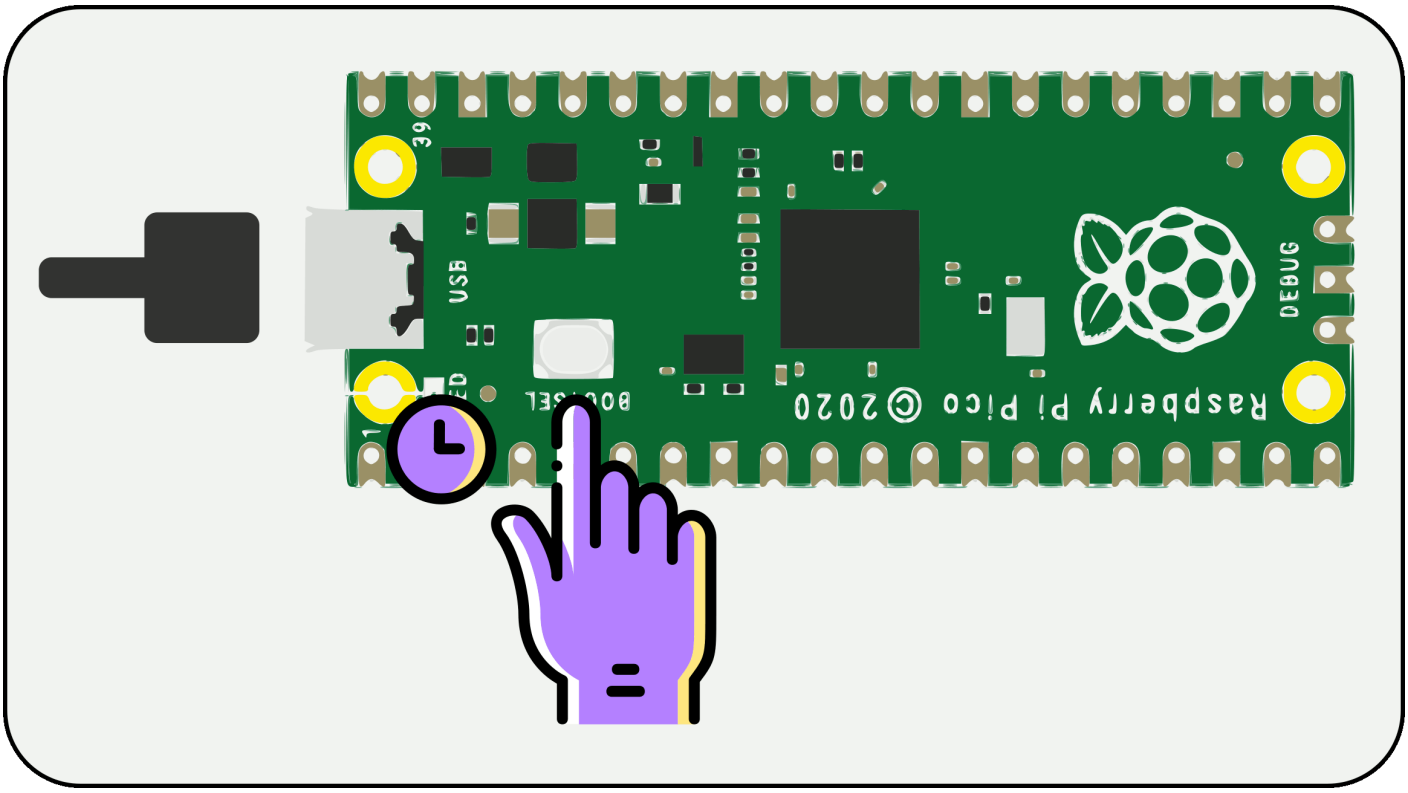
Installation von Micropython auf dem Raspberry Pi Pico

- **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 ^□)



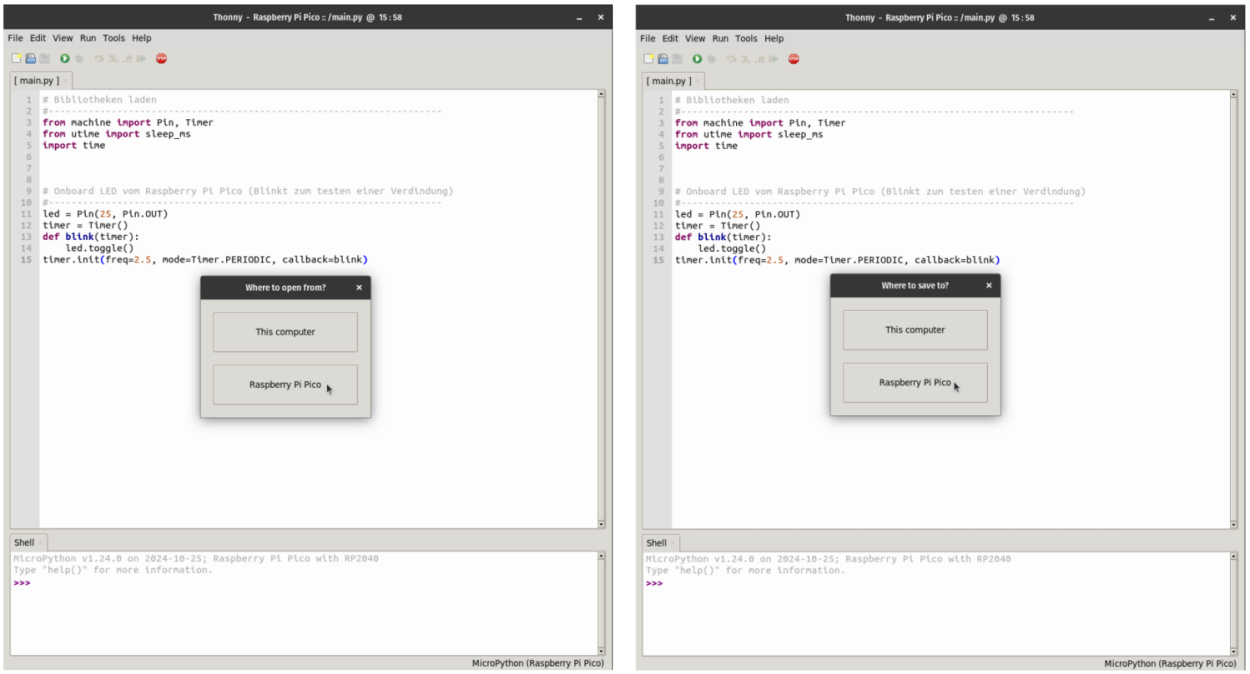
Mit Thonny ein Programm auf dem Raspberry Pi Pico speichern/laden

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(^. .^)



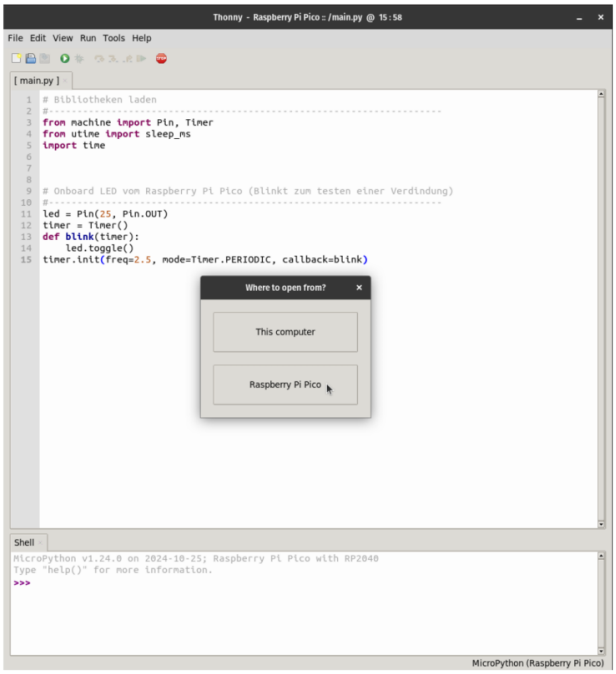
Raspberry Pi Pico mit Thonny öffnen

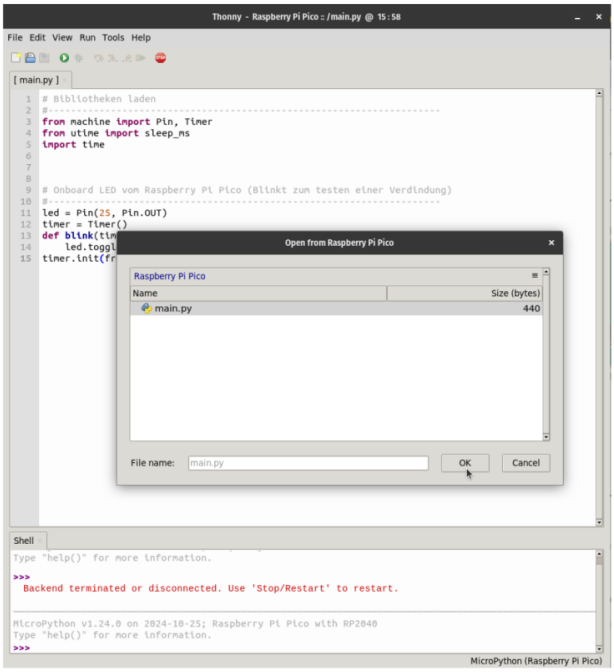
Normalerweise kommuniziert Thonny automatisch mit dem Raspberry Pi Pico, sobald es angeschlossen ist. Wenn ihr Thonny geöffnet habt, geht wie folgt vor:

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

drawing

> **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.**

a) 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.

b) PIR-Sensor anschließen:

- Stecke ein Jumper-Kabel vom **VCC-Pin** des PIR-Sensors in die **positive Leiste** des Breadboards.
- Verbinde den **GND-Pin** des PIR-Sensors mit der **negativen Leiste** des Breadboards.
- Schließe den **OUT-Pin** des PIR-Sensors mit einem Jumper-Kabel an den **GPIO-Pin GP28** des Raspberry Pi Pico an.

c) WS2812B-LED-Streifen anschließen:

- Verbinde das **rote Kabel** des LED-Streifens mit der **positiven Leiste** des Breadboards.
- Verbinde das **schwarze Kabel** des LED-Streifens mit der **negativen Leiste** des Breadboards.
- Schließe das **grüne Kabel** des LED-Streifens (Data-In) mit einem Jumper-Kabel an den **GPIO-Pin GP22** des Raspberry Pi Pico an.

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, Timer
from neopixel import NeoPixel
from utime import sleep_ms
import time
from random import randint



# Onboard LED vom Raspberry Pi Pico (Blinkt zum testen einer Verdindung)
#------------------------------------------------------------------------
led = Pin(25, Pin.OUT)
timer = Timer()
def blink(timer):
    led.toggle()
timer.init(freq=2.5, mode=Timer.PERIODIC, callback=blink)



# Bewegungssensor
#------------------------------------------------------------------------
# Definiert eine Variable für den PRI-Sensor
# PRI ist eine Abkürzung und bedeutet Pulse Repetition Interval (Deut.: Intervall der Impulsfolge) und bezieht sich auf die Frequenz, mit der ein Bewegungsmelder ein Signal auslöst, wenn dieser eine bestimmte Bewegung erkennt.
prisensorpin = machine.Pin(28, machine.Pin.IN)



# LEDs
#------------------------------------------------------------------------
# Festlegung der Farben im RGB-Code
# RGB-Farben könnt ihr unter >> https://html-color.codes/ << erstellen
colors = [
```

```
    (24, 0, 0),
    (0, 24, 0),
    (0, 0, 24),
    (12, 12, 0),
    (0, 12, 12),
    (12, 0, 12),
]
# Farben aus colors in eine Liste (Array) schreiben
array = len(colors)
# Pin für WS2812B LED-Streifen festlegen
ledpin = 22
# Anzahl der LEDs des LED-Streifens angeben
ledanzahl = 15
# Geschwindigkeit definieren! Je höher der Wert, des so langsamer die Animation
speed = 120
# Initialisierung des WS2812B LED-Streifens durch die Variable 'ledstreifen'
ledstreifen = NeoPixel(Pin(ledpin, Pin.OUT), ledanzahl)



# Eine eigene Funktion
#------------------------------------------------------------------------
# Wir bauen (definieren) unsere eigene Funktion, um eine Animation zu simulieren. die neue Funktion heißt bei uns rainbow (Deut.: Regenbogen)
def rainbow():
# In der Schleife wird eine bestimmte Anzahlt Anzahl an wiederholungen durchlaufen. Dabei ist die Anzahl der Durchläufe gleich die Anzahl der LED's. Die Variable i erhöht sich mit jeden Durchlauf, bis die Anzahl der LED's (bei uns 15) erreicht ist.
# Hinweis: Eine Iteration ist ein widerholtes Durchlaufen eines bestimmten Prozesses.
    for i in range (ledanzahl):
# Es wird eine zufällige Farbe aus dem array 'colors' für jedes LED-Element (an der Stelle i) des LED-Streifens erzeugt.
        ledstreifen[i] = colors[randint(0, array-1)]
# sendet die angegebenen LED-Streifen-Daten zu dem LED-Streifen. Dies schaltet die LEDs auf den angegebenen Einstellungen auf dem LED-Streifen ein.
    ledstreifen.write()
# Angabe der Millisekunden, wo der Code eine Pause machen soll. In Micropython gibt die sleep_ms()-Funktion dem Benutzer die Möglichkeit, Wartezeiten von einigen Millisekunden bis zu 255 Sekunden einzustellen.
    sleep_ms(speed)

# Wiederholung (Endlos-Schleife)
```

```
#------------------------------------------------------------------------
while True:
# Wenn der Bewegungssensor einen Wert erhält, wie bspw. durch eine Bewegung, soll die
    if prisensorpin.value():
        print("motion detected")
        rainbow()
    else:
      ledstreifen.fill((0,0,0))
      ledstreifen.write()
```

Online-Ressourcen

/