Manuel SCRAPY

Co-funded by
the European Union

Le soutien de la Commission européenne à la production de cette publication ne constitue pas une approbation du contenu, qui reflète uniquement les points de vue des auteurs, et la Commission ne peut être tenue responsable de toute utilisation qui pourrait être faite des informations contenues dans ce document.

AVERTISSEMENT

Lors de l'utilisation de ce produit, veuillez noter les points d'avertissement suivants :

1. Ce produit contient de nombreuses petites pièces. L'ingestion ou une mauvaise utilisation peut provoquer de graves infections et la mort. Consultez immédiatement un médecin lorsque l'accident se produit.
2. Interdire strictement l'utilisation de ce produit et de ses pièces à proximité d'une prise électrique CA ou d'autres circuits en cas de risque potentiel de choc électrique.
3. Interdire strictement l'utilisation de ce produit à proximité de tout liquide ou feu.
4. Gardez les matériaux conducteurs à l'écart de ce produit.
5. Ne laissez pas les enfants de moins de 3 ans utiliser ce produit sans la surveillance d'un adulte. Veuillez placer ce produit dans une position hors de portée des enfants de moins de 3 ans.
6. Ne stockez pas et n'utilisez pas ce produit dans des environnements extrêmes tels qu'un chaud ou un froid extrême, une humidité élevée, sous la lumière directe du soleil, etc.
7. N'oubliez pas de couper le circuit lorsque vous n'en avez pas besoin.
8. Certaines parties de ce produit peuvent devenir chaudes au toucher lorsqu'elles sont utilisées dans certaines conceptions de circuits, ce qui est normal.
9. Une mauvaise utilisation peut provoquer une surchauffe.
10. L'utilisation de composants non conformes aux spécifications peut endommager le produit.

Introduction

Le KIT SCRAPY est construit sur la base de l'utilisation du microcontrôleur Raspberry Pi Pico. Le « SCRAPY KIT » est créé dans le cadre d'un projet cofinancé par Erasmus+.

La nouvelle tendance en matière d'enseignement à distance, provoquée par la pandémie de COVID-19, crée une lacune dans l'enseignement et la pratique d'activités utilisant l'informatique physique pour atteindre les étudiants présentant un risque de faibles performances dans ces matières STEM.

L'objectif du KIT est de soutenir et de promouvoir l'apprentissage pédagogique pratique en informatique physique et en programmation, dans les cas d'enseignement à distance et en classe. Le KIT offre une combinaison de principes informatiques physiques et de programmation avec du matériel et des logiciels, conduisant à une expérience d'apprentissage innovante.

La portée de ce manuel est de :

- Vous informer des composants du Kit et de l'utilisation des principaux éléments électroniques.
- Vous guider étape par étape pour assembler efficacement le KIT, en tenant compte des précautions associées.
- Fournir des tutoriels pour l'utilisation et la connexion des composants avec le microcontrôleur Pico.
- Fournir des tutoriels avec les fonctionnalités et la portée de chaque composant électronique.

Bonne lecture et amusez-vous

pratique pratique et expérimentation avec le

KIT DE GRATTAGE

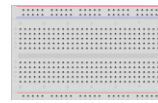
Table des matières

Inclus dans le KIT SCRAPY

Carte de dérivation GPIO Raspberry Pi Pico (2 pièces)

Module d'alimentation MB-102 (1 pièce)

Planche à pain blanche 830 pièces (1 pièce) + 400 pièces (1 pièce)



Bouton poussoir (1 pièce) et capuchon de bouton (1 pièce)

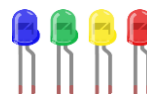
Buzzer (1 pièce)

Résistances 220 Ohm (5 pièces) + 1k Ohm (5 pièces)

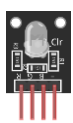
Photorésistance LDR (1 pièce)

Condensateur 100uF (1 pièce)

LED 3mm Bleu (1 pièce), Vert (1 pièce), Rouge (1 pièce) Jaune (1 pièce)

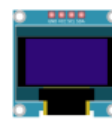


LED RVB 5 mm (1 pièce)

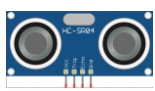


Servomoteur SG90 (1 pièce)

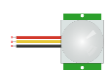
OLED I2C ICC (1 pièce)



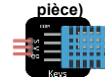
Capteur à ultrasons HC-SR04 (1 pièce)



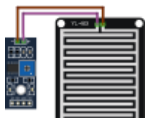
Capteur de détecteur de mouvement PIR HC-SR501 (1 pièce)



Capteur numérique de température et d'humidité DHT11 (1 pièce)

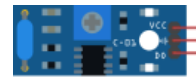


Capteur de gouttes de pluie (1 pièce)

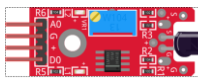


Potentiomètre rotatif linéaire B1k Ohm (1pc)

Capteur de vibrations SW-420 (1 pièce)

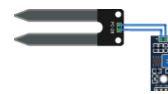


Capteur de flamme (1 pièce)

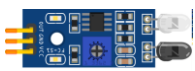


Capteur de détection sonore haute sensibilité (1 pièce)

Hygromètre de sol/capteur de détection d'humidité (1 pièce)



Module de capteur infrarouge IR KY-032 (1 pièce)



Module de manette (1 pièce)

Câble USB vers micro-USB 1m (1pc)

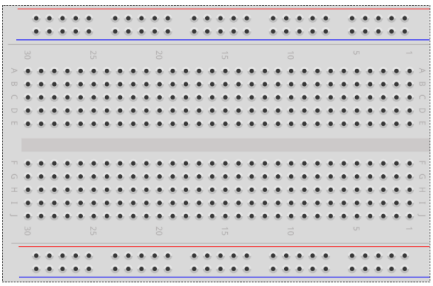
6 piles AA 1,5 V (1 pièce) + support (1 pièce)

Câbles de démarrage (6 pièces)

Vis en plastique (10 pièces), écrous en plastique (6 pièces), piliers en plastique (10 pièces)

Explication des composants

1. Qu'est-ce qu'une planche à pain ?



Une maquette est une carte en plastique percée de minuscules trous qui permettent d'insérer facilement des composants électroniques (transistors, résistances, puces, etc.) pour prototyper (construire et tester) un circuit électronique. L'intérieur est constitué de rangées de minuscules clips métalliques pour maintenir les câbles à connecter.

La plupart des planches à pain comportent des rangées de chiffres, de lettres et de signes plus et moins. Le but des étiquettes est de vous aider à localiser certains trous sur la planche à pain afin que vous puissiez suivre les instructions lors de la construction d'un circuit.

Les longues bandes sur les 2 côtés de la planche à pain sont généralement marquées respectivement de rouge et de bleu ou de rouge et de noir et de signes plus (+) et moins (-). Ces rangées sont appelées bus ou rails et sont généralement utilisées pour fournir de l'énergie électrique au circuit lorsqu'il est connecté à une alimentation (batterie ou alimentation externe).

Le « bus » positif est marqué en rouge, porte le signe plus (+) et fournit l'alimentation.

Le « bus » négatif est marqué en bleu ou noir, porte le signe moins (-) et fournit la masse.

Avantages de l'utilisation d'une maquette :

- Facilite la vérification rapide des circuits simples et complexes et facilite la vérification des circuits à leur stade initial.
- Facile à régler.
- Flexible.
- Pas de trous de perçage.
- Aucune soudure requise.
- Débogage facile des circuits et des programmes.

2. Qu'est-ce qu'une résistance ?

Une résistance est un petit paquet de résistance. Son utilisation dans un circuit réduit le courant d'une quantité précise. Pour déterminer la résistance d'une résistance, il existe un motif de bandes colorées.

Color	1st Band	2nd Band	3rd Band (5-Band Only)	Multiplier (3rd or 4th Band)	Tolerance (Last Band)
Black	0	0	0	1	
Brown	1	1	1	10	± 1%
Red	2	2	2	100	± 2%
Orange	3	3	3	1000	
Yellow	4	4	4	10000	
Green	5	5	5	100000	± 0.5%
Blue	6	6	6	1000000	± 0.25%
Violet	7	7	7	10000000	± 0.1%
Grey	8	8	8		± 0.05%
White	9	9	9		
Gold				0.1	± 5%
Silver				0.01	± 10%
None					± 1%

(Crédit image : Future Owns) disponible surhttps://www.tomshardware.com/how-to/resistor-color-codes

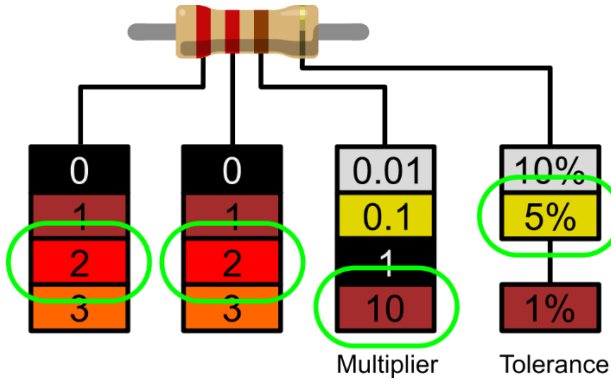
Codes de couleur courants des résistances et leurs utilisations :

Type de résistance	Code couleur à 4 bandes	Code couleur à 5 bandes	Utilisations courantes
220 ohms	Rouge-Rouge-Marron-Or	Rouge-Rouge-Noir-Noir-Or	Protection contre la lumière LED
1K Ohm (1Kiloohm)	Marron-Noir-Rouge-Or	Marron-Noir-Noir-Marron-Or	Protection LED, diviseur de tension

Les résistances n'ont pas de polarité, elles peuvent donc être utilisées dans n'importe quelle orientation dans un circuit. Mais pour identifier les valeurs correctes du code couleur de la résistance, nous devons comprendre les bandes colorées sur la résistance.

Sur une résistance typique de niveau hobby à quatre bandes, il y a trois couleurs dans un groupe. Ce sont les premier, deuxième chiffres significatifs et le multiplicateur. La bande finale est la tolérance de la résistance, une marge d'erreur si vous voulez. Pour la majorité des amateurs, une tolérance de 5% (Gold) est parfaite et courante.

Résistance de 220 ohms (4 bandes)

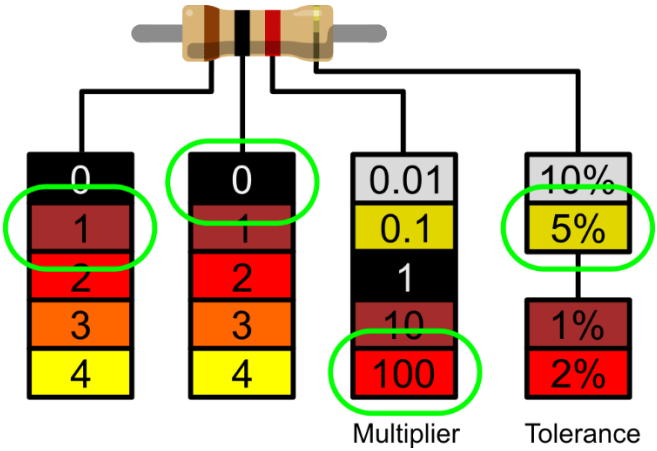


(Crédit image : Future) disponible surhttps://www.tomshardware.com/how-to/resistor-color-codes

1. Le premier chiffre significatif est le rouge et grâce au décodeur on peut voir que le rouge vaut 2.
2. Le deuxième chiffre significatif est également rouge, ce qui nous donne donc 22.
3. Le multiplicateur est marron, et cela décode à 10. Si nous multiplions 22 par 10, nous obtenons 220.
4. Le dernier groupe, la tolérance, est en or. L'or est à 5%, ce qui signifie qu'on peut accepter une résistance avec une marge d'erreur de 5%.

Pour les fabricants exigeant une plus grande précision, il existe également cinq résistances de bande comportant un troisième chiffre significatif. Le chiffre supplémentaire apporte une clarté qui peut être essentielle dans les circuits sensibles à la résistance, par exemple les instruments scientifiques et techniques.

Résistance 1K Ohm (4 bandes)



(Crédit image : Tom's Hardware) disponible surhttps://www.tomshardware.com/how-to/resistor-color-codes

1. La 1ère ligne est marron, et en utilisant le décodeur, on voit que la valeur est 1.
2. La 2ème ligne est noire, cela nous donne donc 10.
3. Le multiplicateur est rouge et décode à 100. Si nous multiplions 10 par 100, nous obtenons 1000.
4. Le dernier groupe, la tolérance, est en or. L'or est à 5%, ce qui signifie qu'on peut accepter une résistance avec une marge d'erreur de 5%.

3. Qu'est-ce qu'un condensateur ?

Un condensateur est un appareil qui stocke de l'énergie électrique dans un champ électrique. C'est un composant électronique passif à deux bornes. Il se compose de deux conducteurs électriques séparés par une distance. L'espace entre les conducteurs peut être rempli sous vide ou avec un matériau isolant appelé diélectrique. (Wikipédia)

Le condensateur 100uF est un condensateur de découplage électrolytique. Ces condensateurs sont d'excellents suppresseurs de transitoires/surtensions et leur utilisation entre l'alimentation et la masse du circuit garantit une alimentation fluide.

4. Qu'est-ce qu'une diode ?

Une diode est un composant électronique à deux bornes qui conduit le courant principalement dans un sens (conductance asymétrique) ; il a une résistance faible (idéalement nulle) dans un sens et une résistance élevée (idéalement infinie) dans l'autre.

La fonction la plus courante d'une diode est de permettre à un courant électrique de passer dans un sens (appelé sens direct de la diode), tout en le bloquant dans le sens opposé (sens inverse). En tant que telle, la diode peut être considérée comme une version électronique d'un clapet anti-retour. Ce comportement unidirectionnel est appelé rectification et est

utilisé pour convertir le courant alternatif (ac) en courant continu (dc). En tant que redresseurs, les diodes peuvent être utilisées pour des tâches telles que l'extraction de la modulation des signaux radio dans les récepteurs radio. (Wikipédiahttps://en.wikipedia.org/wiki/Diode)

5. Qu'est-ce qu'un câble volant ?

Les câbles/fils de liaison sont simplement des fils dotés de broches de connecteur à chaque extrémité, ce qui leur permet d'être utilisés pour connecter deux points entre eux sans soudure. Les fils de liaison sont généralement utilisés avec des planches à pain et d'autres outils de prototypage afin de faciliter la modification d'un circuit selon les besoins. La variation de couleur des fils peut être utilisée comme avantage afin de différencier les types de connexions, telles que la terre ou l'alimentation.

Les câbles de liaison sont généralement disponibles en trois versions : mâle à mâle, mâle à femelle et femelle à femelle. La différence entre chacun réside dans le point final du fil. Les extrémités mâles ont une broche en saillie et peuvent se brancher sur des objets, tandis que les extrémités femelles n'en ont pas et sont utilisées pour brancher des objets. Les câbles de liaison mâle-mâle sont les plus courants. Lors de la connexion de deux ports sur une maquette, un fil mâle à mâle est généralement requis. (https://blog.sparkfuneducation.com/what-is-jumper-wire)

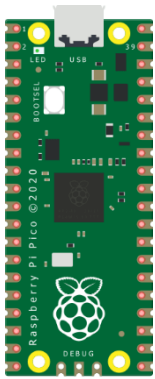
Préparation du projet

1. Lisez-moi avant utilisation

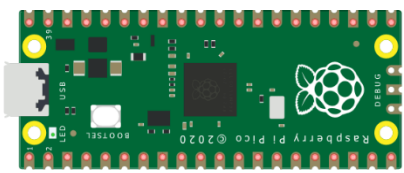
NOTE: Étant donné que les expériences impliquées sont toutes des expériences de circuit, une mauvaise connexion ou un court-circuit peut endommager votre carte de développement Pico. Veuillez toujours vérifier à nouveau le circuit avant de connecter l'alimentation.

2. Framboise Pi Pico

Voici le Raspberry Pi Pico :

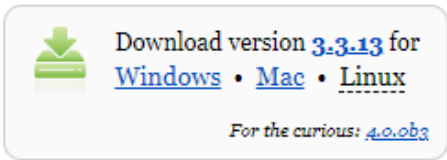


Branchez le câble micro-USB dans le port situé sur le côté gauche de la carte.



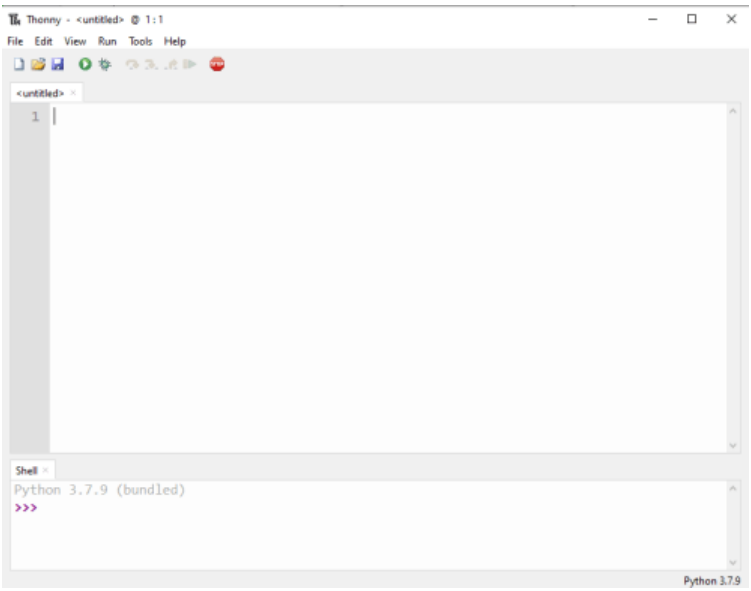
3. Installer l'IDE Thonny

Visitehttps://thonny.orget choisissez le système d'exploitation approprié. Suivez le instructions pour compléter l'installation.



Dans ce manuel, tous les tutoriels sont programmés sous Windows 10, à l'aide d'un Raspberry Pi Pico et du firmware approprié.

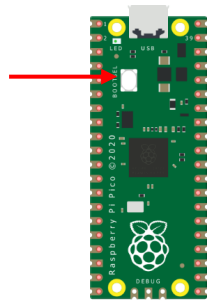
Une fois l'installation terminée, ouvrez Thonny depuis votre ordinateur.



4. Installation du micrologiciel

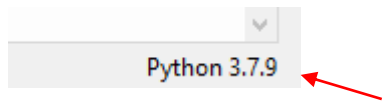
Le Raspberry Pi Pico peut être programmé à l'aide d'une variante Python, appelée MicroPython. Pour utiliser MicroPython sur le Pico, vous devez d'abord installer son firmware.

Étape 1 : Recherchez le bouton BOOTSEL sur votre Raspberry Pi Pico.

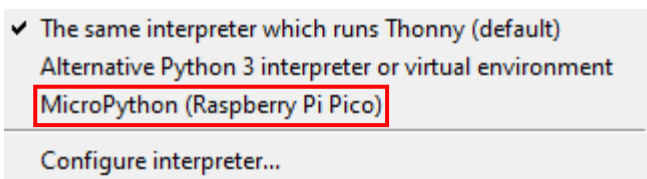


Étape 2 : Appuyez sur le bouton BOOTSEL et maintenez-le enfoncé pendant que vous connectez l'autre extrémité du câble micro-USB à votre ordinateur.

Étape 3 : Dans le coin inférieur droit de Thonny, vous verrez la version de Python que vous utilisez actuellement.

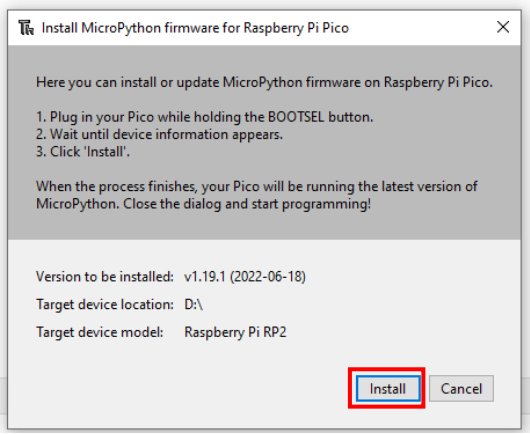


Cliquez sur la version Python et choisissez le MicroPython (Raspberry Pi Pico)

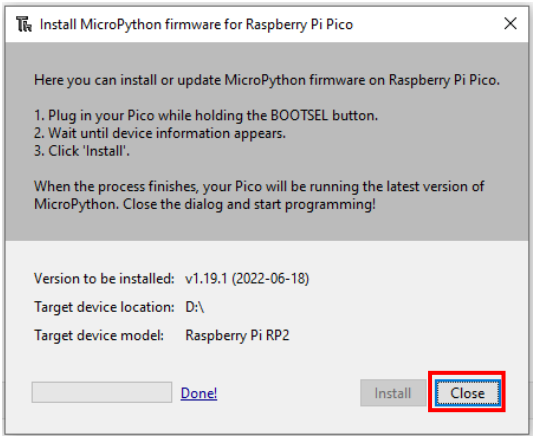


Si vous ne voyez pas cette option, assurez-vous que le câble est correctement connecté au Pico et/ou à votre ordinateur.

Étape 4 : Une boîte de dialogue apparaîtra, vous demandant d'installer la dernière version du micrologiciel sur votre Pico. Cliquez sur le bouton Installer pour copier le micrologiciel sur votre Pico.



Étape 5 : Attendez la fin de l'installation et cliquez sur Fermer.



Vous n'avez pas besoin de répéter le processus à chaque fois que vous connectez le Raspberry Pi Pico à votre ordinateur,

alors la prochaine fois, branchez-le simplement et vous êtes prêt à partir.

5. Introduction à la programmation MicroPython

Vous allez maintenant utiliser Thonny IDE pour exécuter du code Python simple afin de vous familiariser avec Thonny's Shell et MicroPython.

Assurez-vous d'abord que votre Raspberry Pi Pico est connecté à votre ordinateur et que vous avez sélectionné l'interpréteur MicroPython comme expliqué dans la section précédente.

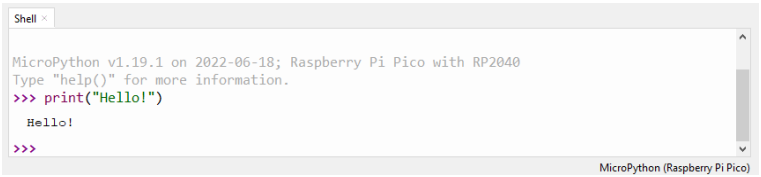
Le panneau Shell en bas de l'éditeur Thonny devrait ressembler à ceci :

Thonny est prêt à communiquer avec votre Pico en utilisant le framework REPL (read-eval-print loop), qui vous permet d'écrire du code directement sur le Shell et d'obtenir une sortie.

Tapez la commande suivante :

```
print("Bonjour!")
```

Appuyez ensuite sur la touche Entrée et voyez le résultat suivant :

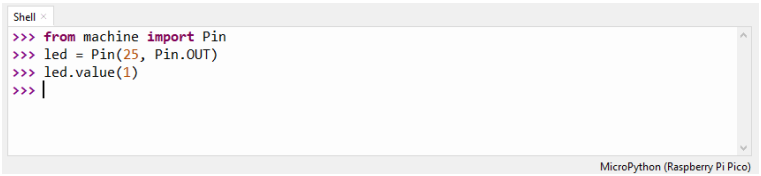


MicroPython vous permet d'ajouter des modules spécifiques au matériel, tels que`machine`, que vous pouvez utiliser pour

programmer votre Pico. Dans l'exemple suivant, vous utiliserez le`machine`module pour allumer la LED intégrée de Pico.

Écrivez le code suivant dans Thonny's Shell :

```
à partir de l'importation de la machine Badge
LED = broche (25, broche.OUT)
led.valeur(1)
```



Appuyez sur la touche Entrée et immédiatement la LED intégrée du Pico s'allumera.

Pour éteindre la LED, écrivez le code suivant :

```
led.valeur(0)
```

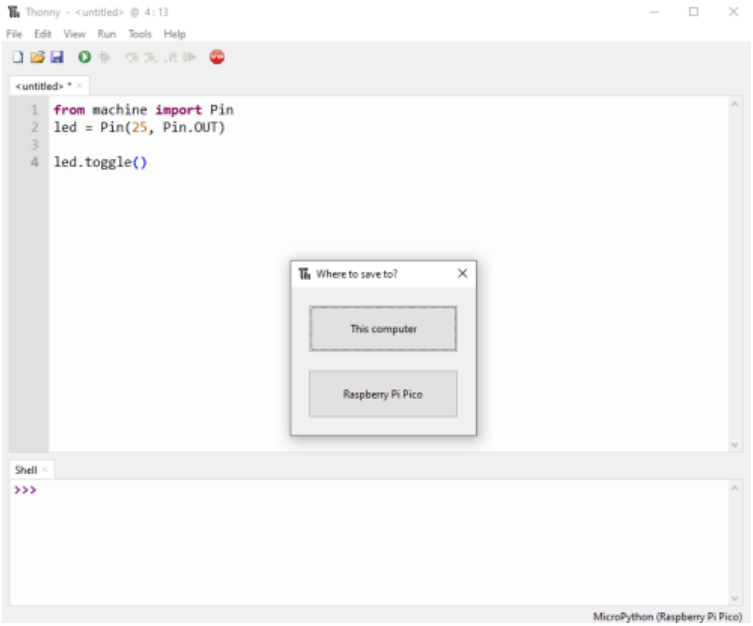
Pour le reste de cette section, vous écrirez votre premier « vrai » programme qui fera clignoter la LED intégrée à chaque fois que vous exécuterez votre programme.

Thonny Shell est utile pour essayer des commandes rapides et s'assurer que tout fonctionne correctement. Cependant, les programmes plus longs doivent être enregistrés dans un fichier .py. En utilisant Thonny, vous pouvez enregistrer des programmes directement sur le Raspberry Pi Pico, puis les exécuter.

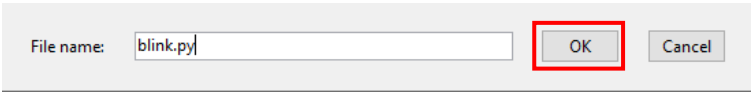
Ouvrez Thonny Python et dans le volet principal de l'éditeur, écrivez le code suivant :

```
à partir de l'importation de la machine Badge
LED = broche (25, broche.OUT)
led.toggle()
```

Maintenant, enregistrez votre programme en cliquant sur l'icône Enregistrer en haut à gauche ou en appuyant sur Ctrl+S sur votre clavier.



Thonny vous demandera où vous souhaitez que votre programme soit sauvegardé. Choisissez le Raspberry Pi Pico. Enregistrez le fichier sous Blink.py et cliquez sur OK. Vous devez toujours ajouter l'extension .py pour que Thonny reconnaisse le fichier comme un fichier Python.



Désormais, chaque fois que vous cliquez sur l'icône Lecture, vous devriez voir la LED intégrée s'allumer et s'éteindre.

En poussant votre code un peu plus loin, vous pouvez faire clignoter la LED intégrée à un certain rythme.

Écrivez le code suivant et enregistrez votre programme en utilisant le même nom que ci-dessus.

```
à partir de l'importation de la machine Badge
à partir du temps importer le sommeil
LED = broche (25, broche.OUT)
tandis que Vrai :
      led.toggle()
      dormir(1)
```

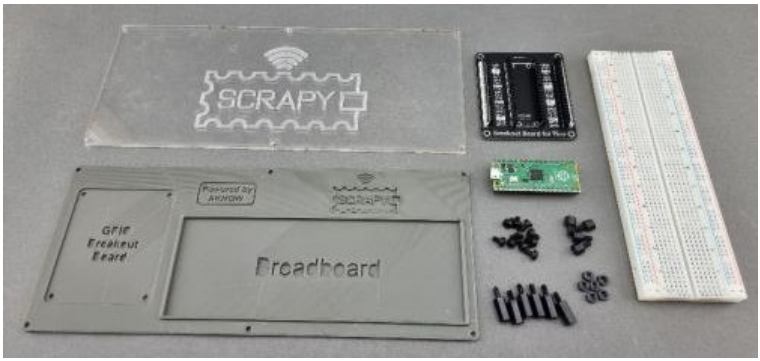
Désormais, lorsque vous exécutez le programme, la LED intégrée clignote toutes les secondes jusqu'à ce que nous arrêtions le programme. Pour arrêter le programme, vous pouvez cliquer sur l'icône STOP ou appuyer sur Ctrl+C sur votre clavier.

Dans les prochains didacticiels, nous apprendrons comment ajouter et contrôler d'autres appareils électroniques et capteurs et créer des programmes pouvant communiquer entre eux.

Assemblage du kit

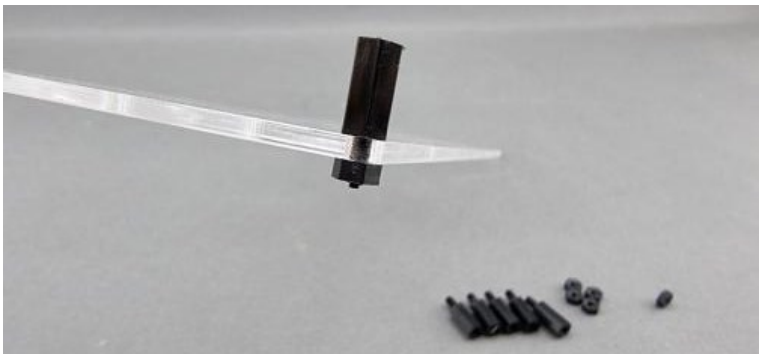
Le Kit SCRAPY comprend les éléments suivants :

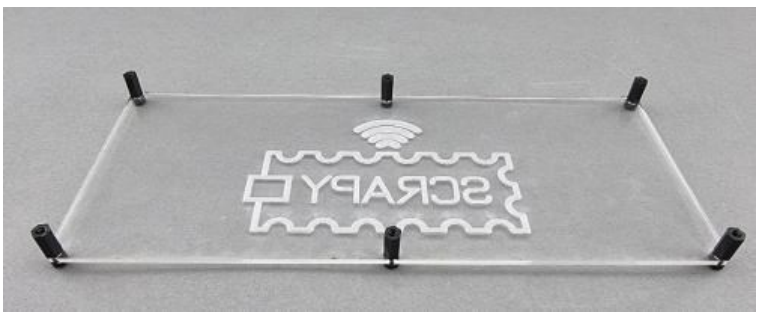
- 1x pièce imprimée en 3D
- 1 x pièce en plexiglas
- 1 x Raspberry Pi Pico
- 1 x carte de dérivation GPIO
- 1 x planche à pain (830 pièces)
- 10 x vis en plastique
- Piliers en plastique 6x12mm
- Piliers en plastique 4x6mm
- 6 x écrous en plastique



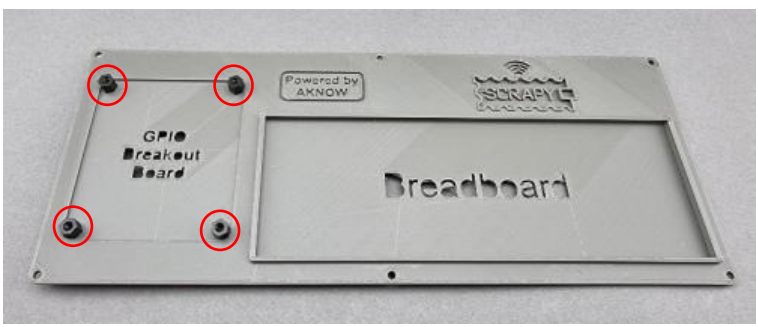
La procédure d'assemblage est simple et peut être réalisée en 6 étapes :

Étape 1 : Montez les piliers en plastique de 6 x 12 mm avec les 6 x écrous en plastique sur la pièce en plexiglas.

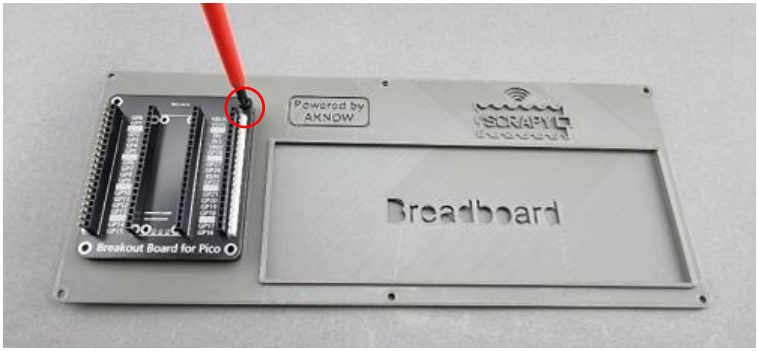




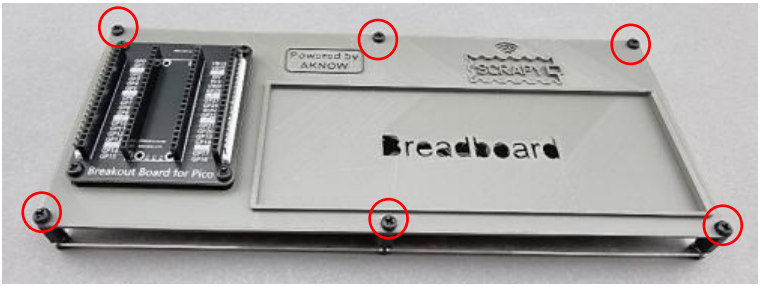
Étape 2 : Sur la pièce imprimée en 3D, placez des piliers en plastique de 4 x 6 mm dans la section « GPIO Breakout Board ».



Étape 3 : Montez la carte de dérivation GPIO sur les 4 piliers à l'aide de 4 vis en plastique.

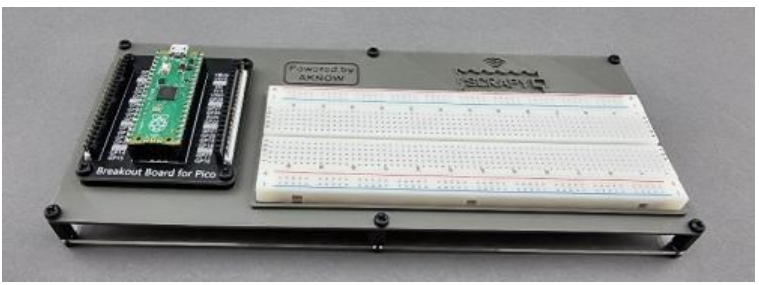


Étape 4 : Montez l'objet imprimé en 3D avec l'objet en plexiglas à l'aide de 6 vis en plastique sur les 6 piliers en plastique que vous avez montés à l'étape 1.



Étape 5 : Retirez le ruban protecteur sous la planche à pain de 830 pièces et placez-le sur la section « Planche à pain » du kit. Veuillez vous assurer que le côté positif (+) de la planche à pain est en haut, comme l'indique l'image suivante.

Étape 6 : Placez le Raspberry Pi Pico sur la carte de dérivation GPIO et appuyez dessus jusqu'à ce que toutes les broches GPIO soient correctement insérées. Assurez-vous que le port micro-USB est situé sur le côté supérieur, comme l'indique l'image.



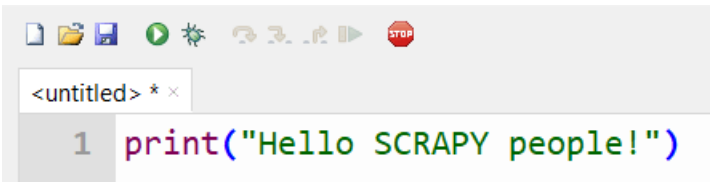
Tutoriels de base

0. « Bonjour les gens de SCRAPY ! »

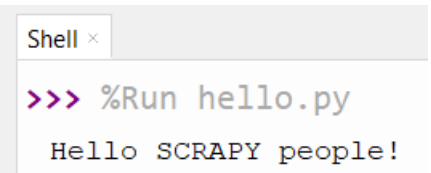
Dans ce tutoriel de base, nous allons apprendre à imprimer un message simple dans Thonny en utilisant la programmation Python.

La fonction print()

1. Allez dans l'éditeur Thonny, écrivez le code suivant et appuyez sur l'icône de lecture. Thonny vous demandera d'abord de sauvegarder votre programme. Enregistrez-le sous le nom hello.py.

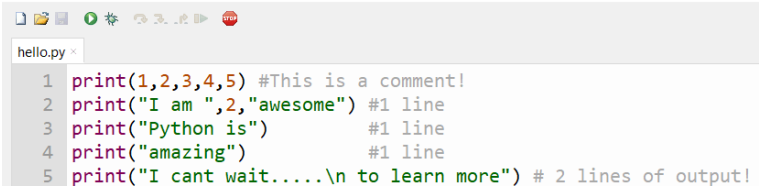


2. Vérifiez la fenêtre du shell.

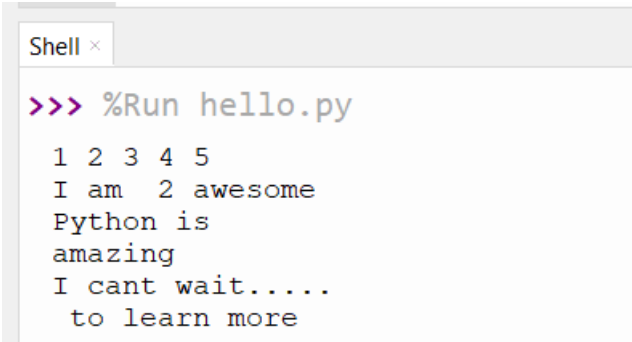


3. Bon travail! Vous venez de créer votre premier programme Python.

La fonction d'impression est une fonction Python intégrée qui nous permet d'imprimer du texte dans le shell. Il peut également prendre des paramètres. Créez un nouveau programme et copiez le code suivant, puis appuyez sur play et voyez comment le texte apparaît dans le shell.



Comme vous pouvez le voir dans la fenêtre shell, chaque fonction d'impression imprime le texte sur une ligne distincte. Cependant, si vous utilisez le « \n » (caractère de nouvelle ligne), vous pouvez changer de ligne dans la même instruction d'impression.



Exercice

Utilisez la fonction d'impression pour imprimer sur 3 lignes distinctes « Au revoir les gens de SCRAPY ! » en utilisant une seule instruction d'impression.

4. Contrôler une LED

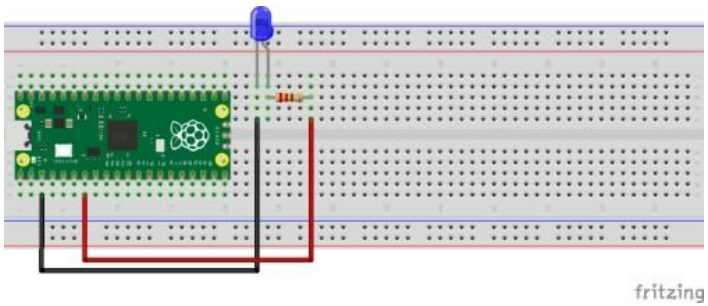
Description

Dans ce tutoriel, vous apprendrez à connecter et contrôler une lumière LED. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom `led.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 2 x câbles de raccordement mâle-mâle
- 1 x câble micro-USB
- Résistance 1x220 ohms
- 1 x LED (n'importe quelle couleur)

Schéma de câblage



– connectez l'extrémité la plus longue (+) de la LED à une résistance de 220Ohm

– connecter la résistance au GPIO5 (câble rouge)

– connectez l'extrémité la plus courte (-) de la LED à une broche GND

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /led.py @ 9 : 1

File Edit View Run Tools Help

[led.py]

```
1  from machine import Pin #import  Pin from library machine
2  from time import sleep  #import sleep from library time
3
4  led = Pin(5, Pin.OUT)
5
6  while True:
7      led.toggle()
8      sleep(1)
9
```

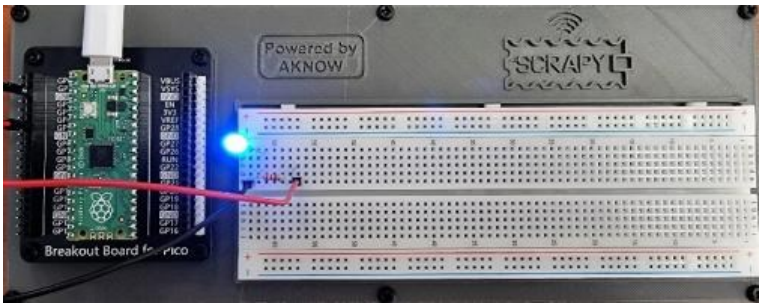
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du circuit avec le matériel fourni :



5. Bouton poussoir

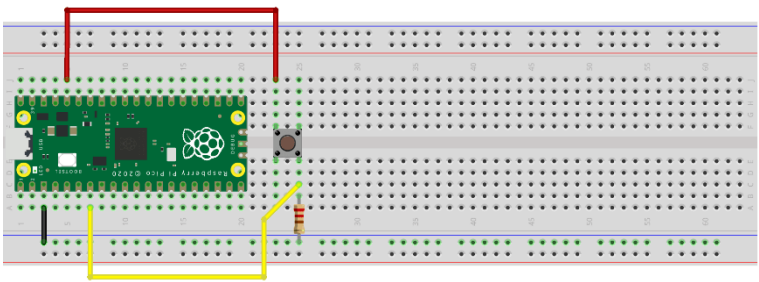
Description

Dans ce tutoriel vous apprendrez à connecter et contrôler un bouton poussoir. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`bouton.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- Résistance 1x220 ohms
- 1 x bouton poussoir (n'importe quelle couleur)
- 1 x capuchon de bouton

Schéma de câblage



– connectez le côté bouton supérieur gauche à la broche 3v3 (câble rouge)

– connectez le côté bouton inférieur droit au GPIO5 (câble jaune)

– connecter une broche GND au rail (-) (câble noir)

– connectez la résistance de 220 Ohm au rail (-) et au côté bouton en bas à droite

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /button.py @ 10 : 1

File Edit View Run Tools Help

[button.py]

```
1  from machine import Pin #import Pin and ADC from library machine
2  from time import sleep  #import sleep from library time
3
4  button = Pin(5, Pin.IN, Pin.PULL_DOWN)
5
6  while True:
7      if button.value():
8          print("Button is pressed!")
9          sleep(1)
10
```

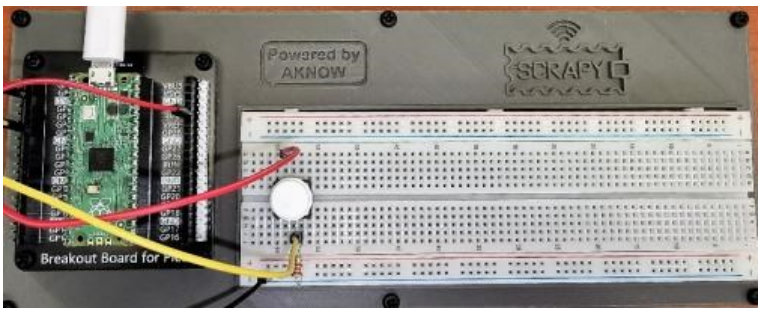
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du circuit avec le matériel fourni :



6. Avertisseur sonore

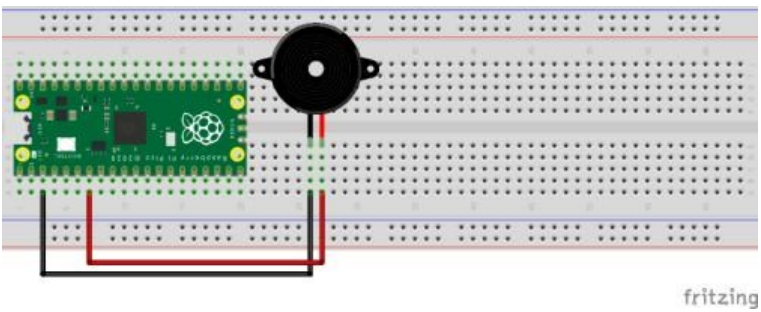
Description

Dans ce tutoriel, vous apprendrez à connecter et contrôler un buzzer. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`buzzer.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 2 x câbles de raccordement mâle-mâle
- 1 x câble micro-USB
- 1 x sonnerie

Schéma de câblage



– connectez l'extrémité la plus longue (+) du buzzer à la broche GPIO5

– connectez l'extrémité la plus courte (-) du buzzer à une broche GND

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /buzzer.py @ 16 : 1

File Edit View Run Tools Help

[buzzer.py]

```
1  from machine import Pin #import Pin from library machine
2  from time import sleep  #import sleep from library time
3
4  buzzer = Pin(5, Pin.OUT)
5
6  buzzer.value(0)
7  sleep(0.5)
8  buzzer.value(1)
9  sleep(0.5)
10 buzzer.value(0)
11 sleep(0.5)
12 buzzer.value(1)
13 sleep(0.5)
14 buzzer.value(0)
15 sleep(0.5)
16
```

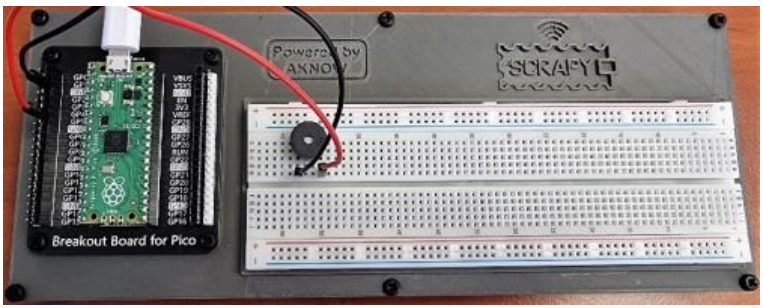
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du circuit avec le matériel fourni :



7. Potentiomètre

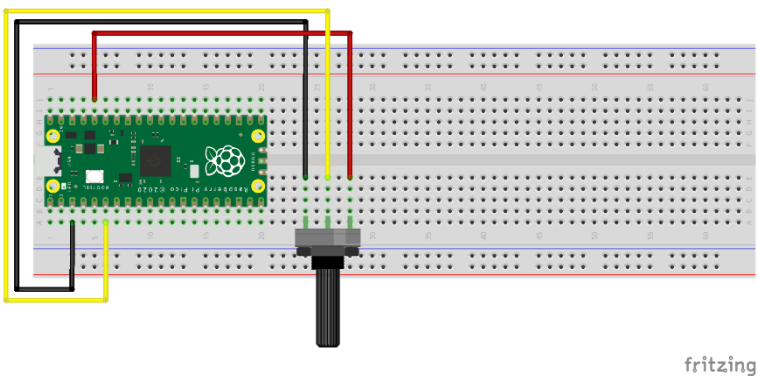
Description

Dans ce tutoriel, vous apprendrez à connecter et contrôler le potentiomètre. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`pot.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x potentiomètre

Schéma de câblage



– le câble noir doit être connecté à la broche GND (broche 3)

– le câble jaune doit être connecté à la broche GPIO5

– le câble rouge doit être connecté à la broche d'alimentation 3V3

– tournez le potentiomètre vers la gauche pour qu'il soit éteint

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /pot.py @ 9 : 1

File Edit View Run Tools Help

[pot.py]

```
1  from machine import Pin, ADC #import Pin from library machine
2  from time import sleep  #import sleep from library time
3
4  pot = ADC(Pin(26))
5
6  while True:
7      print(pot.read_u16())
8      sleep(0.5)
9
```

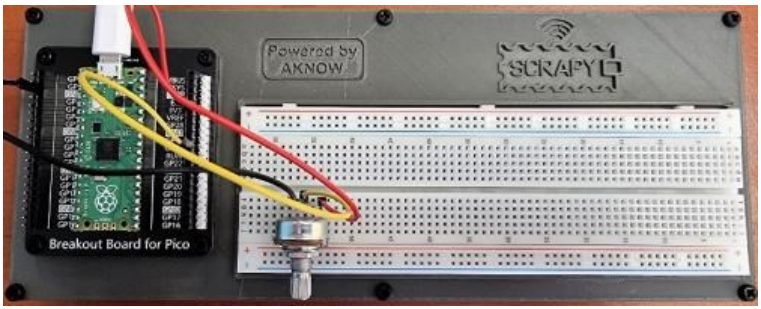
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du circuit avec le matériel fourni :



8. LED RVB

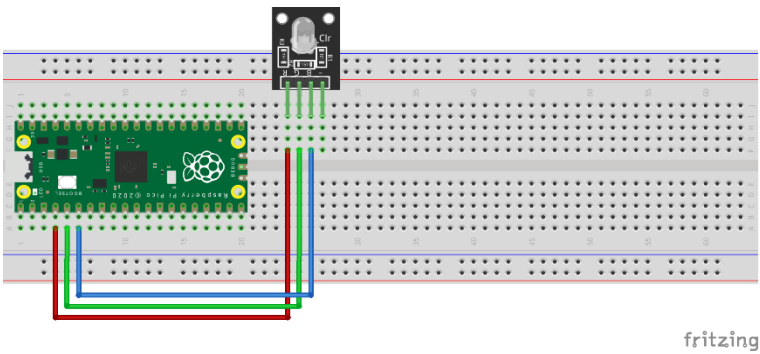
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler un module lumineux LED RVB. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`rgb.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 4 x câbles de raccordement mâle-mâle
- 1 x câble micro-USB
- 1 module LED RVB.

Schéma de câblage



– le câble rouge doit être connecté au GPIO2

– le câble vert doit être connecté au GPIO3

– le câble bleu doit être connecté au GPIO4

– aucune connexion GND n'est requise

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /rgb.py @ 14 : 1

File Edit View Run Tools Help

[rgb.py]

```
from machine import Pin, PWM #import Pin from library machine
from time import sleep  #import sleep from library time
import random
Led_R = PWM(Pin(2))
Led_G = PWM(Pin(3))
Led_B = PWM(Pin(4))
# Define the frequency
Led_R.freq(2000)
Led_G.freq(2000)
Led_B.freq(2000)
while True:
    # range of random numbers
    R=random.randint(0,65535)
    G=random.randint(0,65535)
    B=random.randint(0,65535)
    print(R,G,B)
    Led_R.duty_u16(R)
    Led_G.duty_u16(G)
    Led_B.duty_u16(B)
    sleep(1)
```

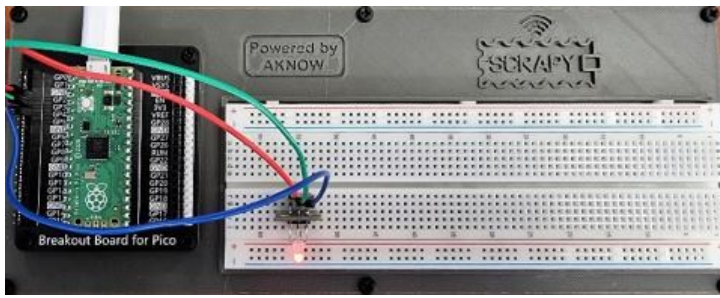
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du circuit avec le matériel fourni :



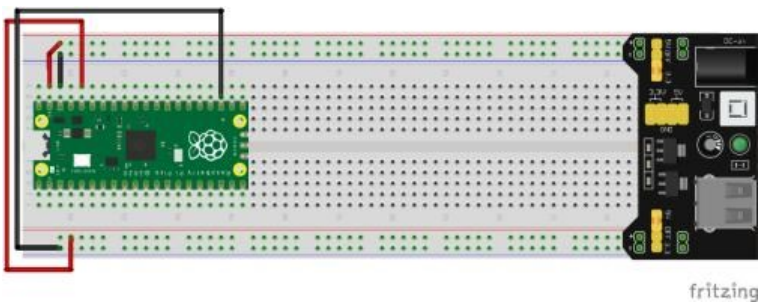
Tutoriels avancés

Avant de passer à cette section et à la suivante, nous devons apporter quelques modifications à notre kit SCRAPY. Cela implique l'ajout de quelques composants supplémentaires et leur connectivité.

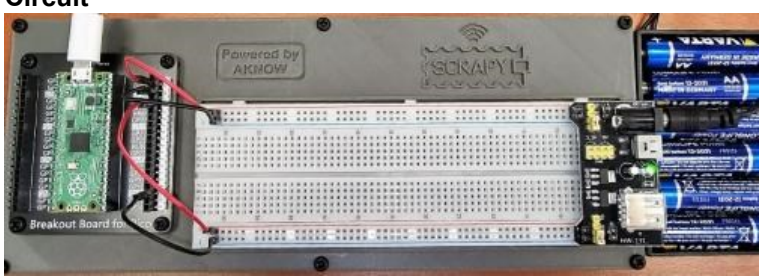
Composants

- 1 x 6 piles AA
- 1 module d'alimentation MB-102
- 4 x câbles de raccordement mâle-mâle

Veuillez suivre le schéma pour connecter les nouveaux composants :



Circuit



– cette configuration sera utilisée pour les tutoriels restants

– connexions latérales supérieures : VSYS 5V (+) rouge) et GND ((-) noir)

– connexions latérales inférieures : 3V3 (+) rouge) et GND ((-) noir)

9. Photorésistance LDR

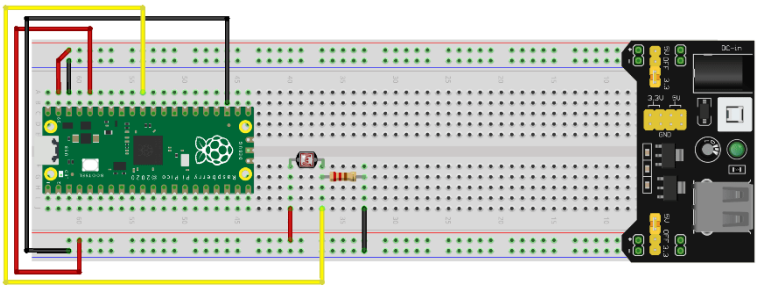
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler une photorésistance LDR. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`ldr.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x photorésistance LDR
- 1 résistance de 220 ohms

Schéma de câblage



– le côté gauche du LDR est connecté au rail 3V ((+) rouge)

– le côté droit du LDR est connecté à une résistance de 220 Ohm et à la broche GPIO26 ADC (câble jaune)

– le côté droit de la résistance est connecté au rail GND ((-) noir)

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /ldr.py @ 10 : 1

File Edit View Run Tools Help

[ldr.py]

```
1  from machine import Pin, ADC #import Pin and ADC from library machine
2  from time import sleep  #import sleep from library time
3
4  ldr = ADC(Pin(26))
5
6
7  while True:
8      print(ldr.read_u16())
9      sleep(1)
10
```

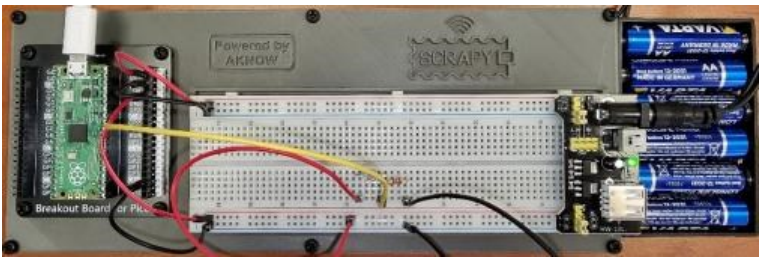
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  8642
  7329
  4032
  3504
  3840
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



10. Servomoteur

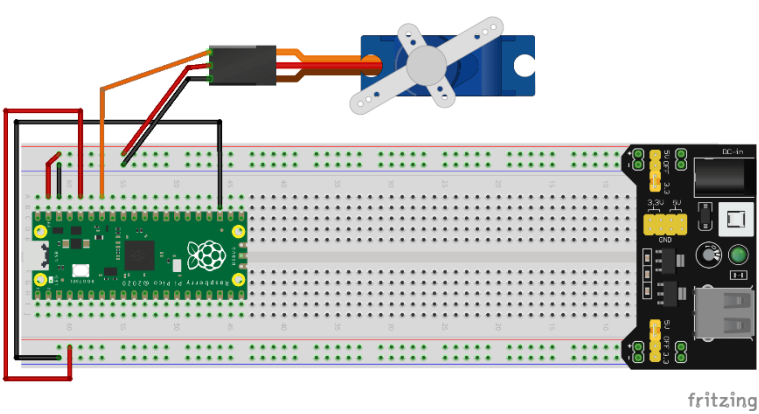
Description

Dans ce tutoriel, vous apprendrez à connecter et contrôler un servomoteur. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`servo.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x servomoteur SG-90

Schéma de câblage



– le câble rouge est connecté au rail 5V (+)

– le câble noir/marron est connecté au rail GND (-)

– le câble orange est connecté à la broche GPIO28 ADC

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /servo.py @ 13 : 20

File Edit View Run Tools Help

servo.py

```
1  from machine import Pin, PWM #import Pin and PWM from library machine
2  from time import sleep  #import sleep from library time
3
4  pwm = PWM(Pin(28))
5  pwm.freq(50)
6
7  while True:
8      for position in range(1000,9000,50):
9          pwm.duty_u16(position)
10         sleep(0.01)
11     for position in range(9000,1000,-50):
12         pwm.duty_u16(position)
13         sleep(0.01)
```

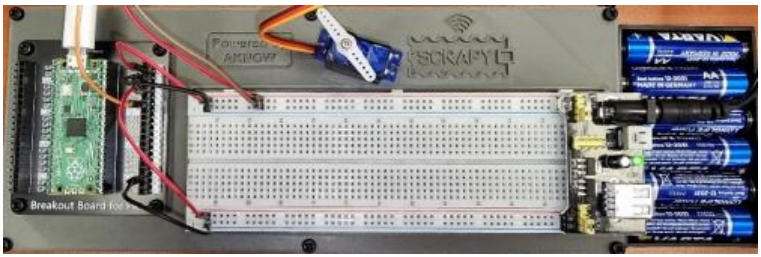
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



11. Écran OLED I2C SSD1306

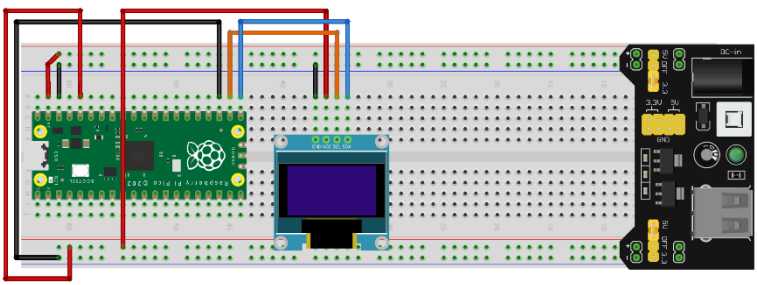
Description

Dans ce didacticiel, vous apprendrez comment connecter et contrôler l'écran OLED I2C ICC. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`oled.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 4 x câbles de raccordement mâle-mâle
- 1 x câble micro-USB
- 1 écran OLED I2C SSD1306

Schéma de câblage



– le câble rouge est connecté au rail 3v3 (+)

– le câble noir est connecté au rail GND (-)

– le câble orange est connecté à la broche GPIO17 I2C0 SCL

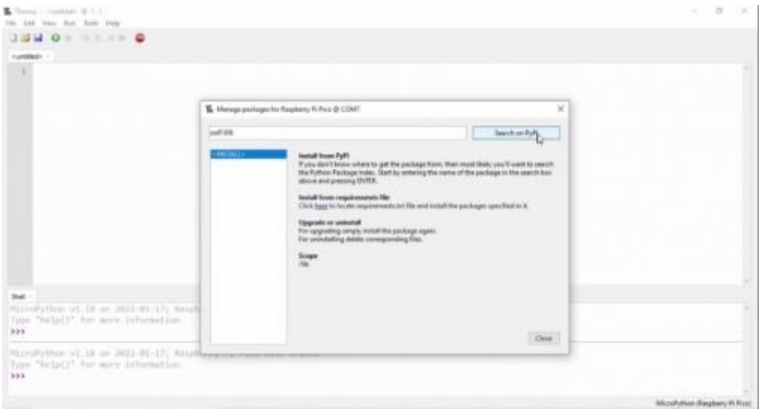
– le câble bleu est connecté à la broche GPIO26 I2C0 SDA

Code

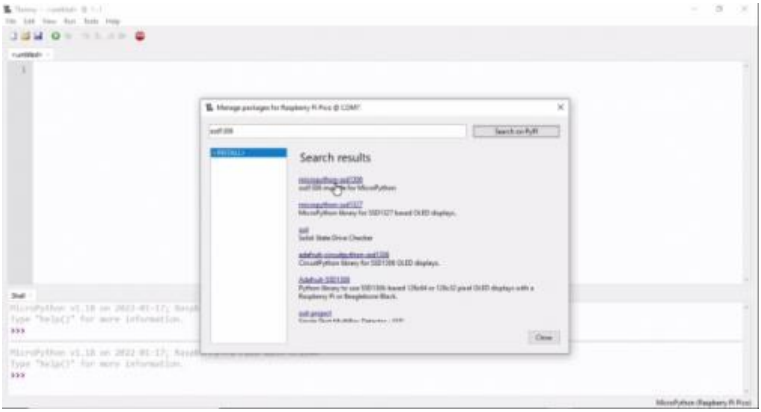
Avant de commencer la programmation de l'écran OLED, nous devons d'abord ajouter le package SSD1306 à notre RPi Pico. Pour ce faire, veuillez suivre les étapes suivantes :

1. Ouvrez Thonny et allez dans Outils→**Gérer les colis…**

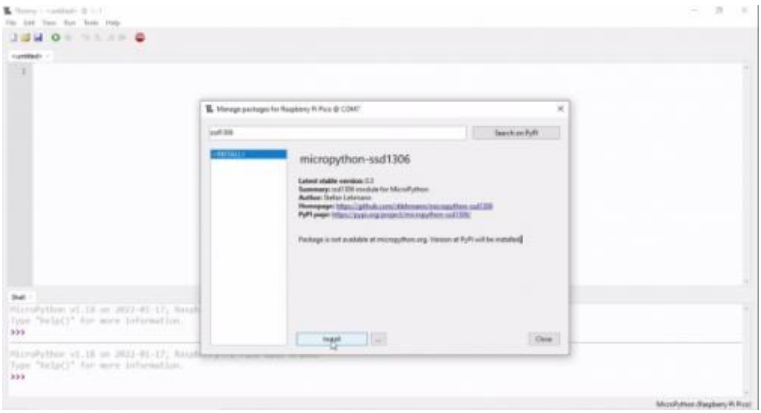
2. Dans la fenêtre de gestion des packages, tapez SSD1306 et cliquez sur Rechercher.



3. Une fois la recherche terminée, cliquez sur le micropython-ssd1306.



4. Dans la fenêtre suivante, cliquez sur Installer.



5. Attendez l'installation du package, puis cliquez sur Fermer.

Nous sommes maintenant prêts à procéder à la programmation de l'écran OLED.

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /oled.py @ 20 : 1

File Edit View Run Tools Help

[oled.py]

```
1  from machine import Pin, I2C
2  import ssd1306
3
4  WIDTH =128
5  HEIGHT = 64
6
7  PIN_SCL = 17
8  PIN_SDA = 16
9
10 i2c = I2C(0,scl=Pin(PIN_SCL),sda=Pin(PIN_SDA))
11 oled = ssd1306.SSD1306_I2C(WIDTH,HEIGHT,i2c)
12 oled.fill(0)
13
14 oled.text("OLED", 0, 0)
15 oled.text("Display", 0, 10)
16 oled.text("Hello SCRAPY", 0, 30)
17 oled.text("  People!", 0, 40)
18
19 oled.show()
20
```

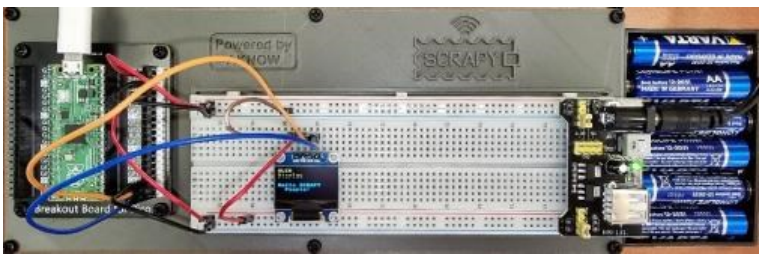
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
>>>
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



12. Module manette de jeu

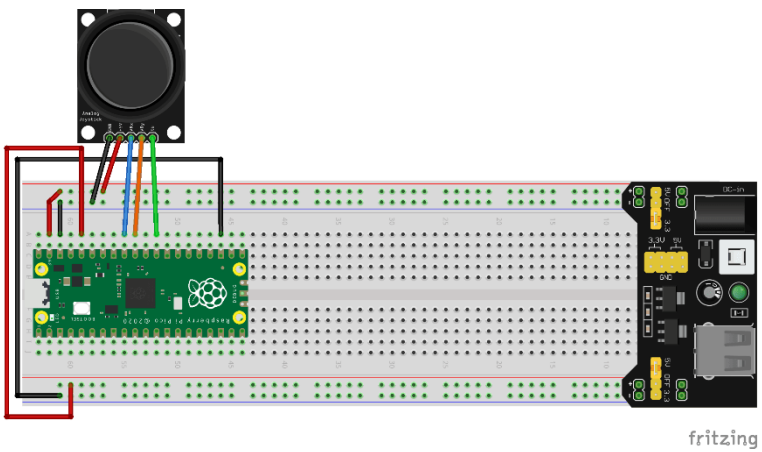
Description

Dans ce tutoriel, vous apprendrez à connecter et contrôler un module joystick. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`joystick.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 5 x câbles de raccordement mâle-mâle
- 1 x câble micro-USB
- 1 x module de manette

Schéma de câblage



– Le +5V (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– VRx (câble bleu) est connecté à la broche GPIO27 ADC1

– VRy (câble orange) est connecté à la broche GPIO26 ADC0

– Le câble SW (vert) est connecté à la broche GPIO22

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /joystick.py @ 18 : 5

File Edit View Run Tools Help

[joystick.py]

```
from machine import Pin, ADC
from time import sleep

VRX = ADC(Pin(27))
VRY = ADC(Pin(26))
SW = Pin(22,Pin.IN, Pin.PULL_UP)

while True:
    xAxis = VRX.read_u16()
    yAxis = VRY.read_u16()
    switch = SW.value()

    print("X-axis: " + str(xAxis) + ", Y-axis: " + str(yAxis) + ", Switch " + str(switch))
    if switch == 0:
        print("Push button pressed!")
    print(" ")
    sleep(1)
```

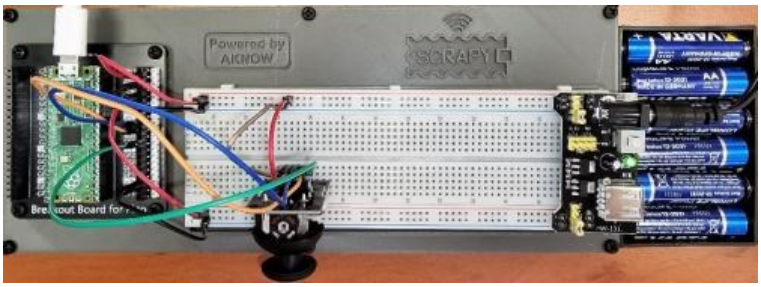
Shell

```
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
  X-axis: 32792, Y-axis: 34616, Switch 1

  X-axis: 32471, Y-axis: 34520, Switch 1

  X-axis: 32455, Y-axis: 34472, Switch 1
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



Tutoriels avec des capteurs

13. Capteur de gouttes de pluie

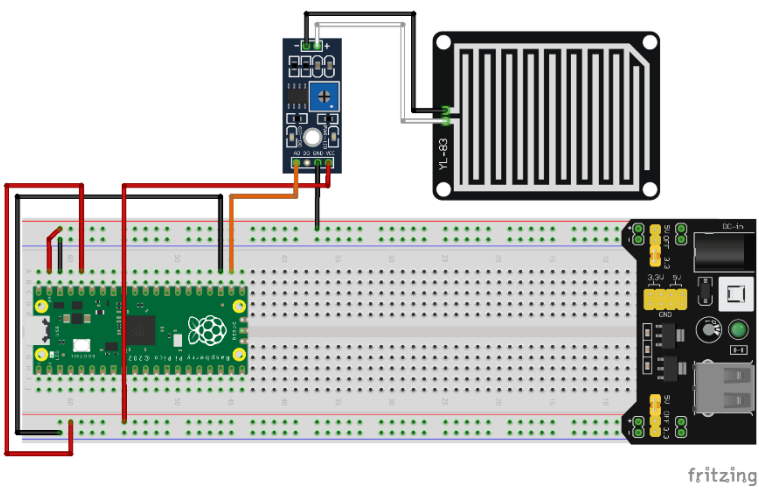
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur de gouttes de pluie. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`goutte de pluie.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur de goutte de pluie

Schéma de câblage



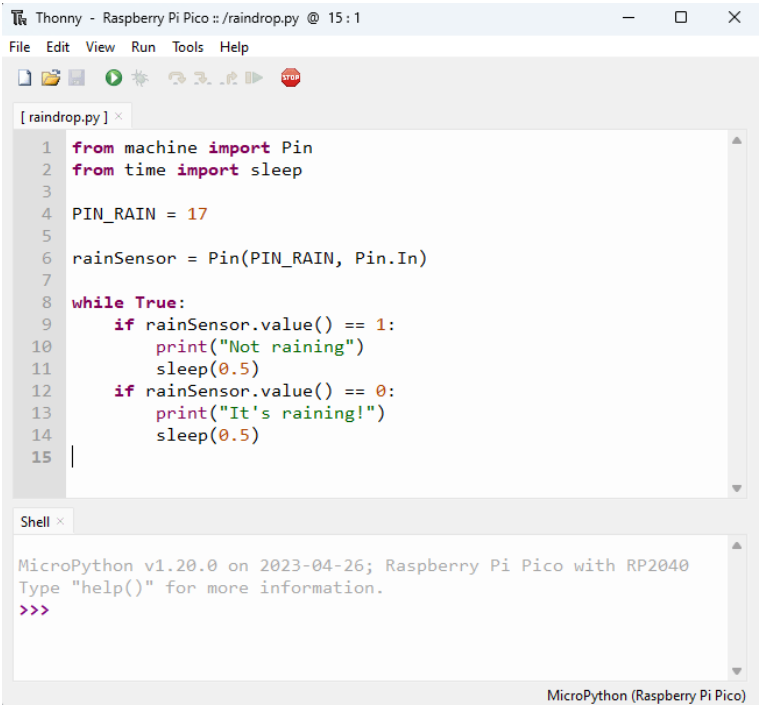
– 3v3V (câble rouge) est connecté au rail 3v3V (+)

– GND (câble noir) est connecté au rail GND (-)

– DO (câble orange) est connecté à la broche GPIO17

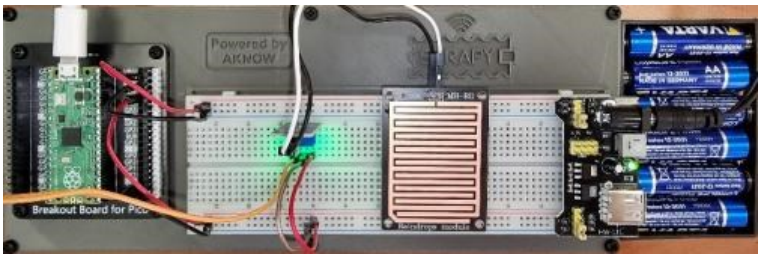
Code

Code MicroPython pour le tutoriel :



Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



14. Capteur à ultrasons HC-SR04

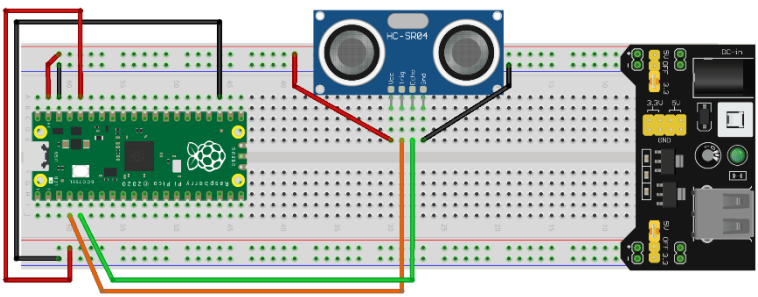
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur à ultrasons HC-SR04. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`ultrasons.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur à ultrasons HC-SR04

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

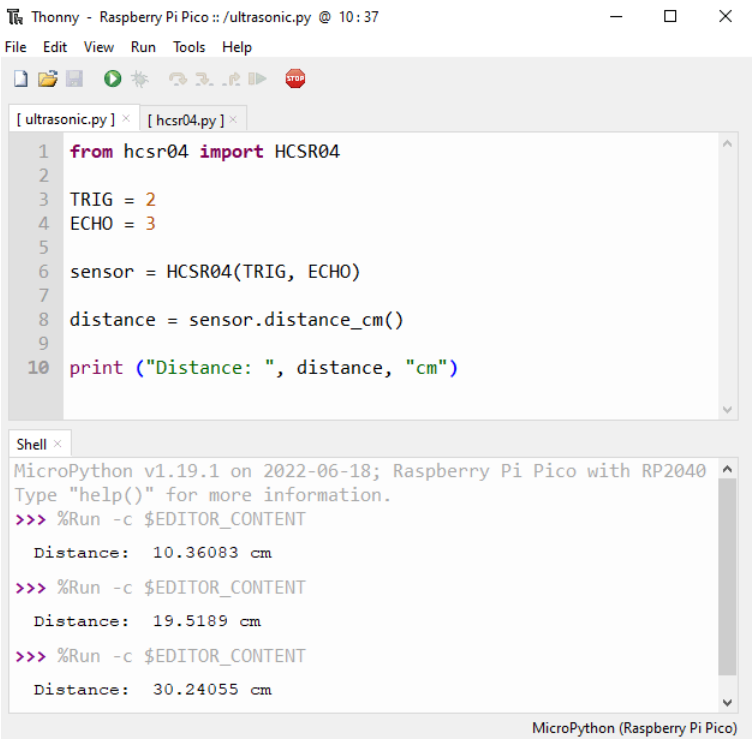
– TRIG (câble orange) est connecté à la broche GPIO2

– ECHO (câble vert) est connecté à la broche GPIO3

Code

Pour utiliser le capteur à ultrasons HC-SR04, nous pouvons développer notre propre programme ou utiliser l'une des bibliothèques disponibles en ligne, par exemple surGitHub. Si vous choisissez de télécharger la bibliothèque hcsr04.py, vous devez l'enregistrer dans votre Pico sous ce même nom.

Code MicroPython utilisant une bibliothèque existante :



Code MicroPython sans bibliothèque existante :

Thonny - Raspberry Pi Pico :: /ultrasonic.py @ 21 : 18

File Edit View Run Tools Help

[ultrasonic.py]

```
from machine import Pin
import utime

TRIG = Pin(2, Pin.OUT)
ECHO = Pin(3, Pin.IN)
def ultra():
    TRIG.low()
    utime.sleep_us(2)
    TRIG.high()
    utime.sleep_us(5)
    TRIG.low()
    while ECHO.value() == 0:
        signaloff = utime.ticks_us()
    while ECHO.value() == 1:
        signalon = utime.ticks_us()
    timepassed = signalon - signaloff
    distance = (timepassed * 0.0343) / 2
    print("The distance from object is ",distance,"cm")
while True:
    ultra()
    utime.sleep(1)
```

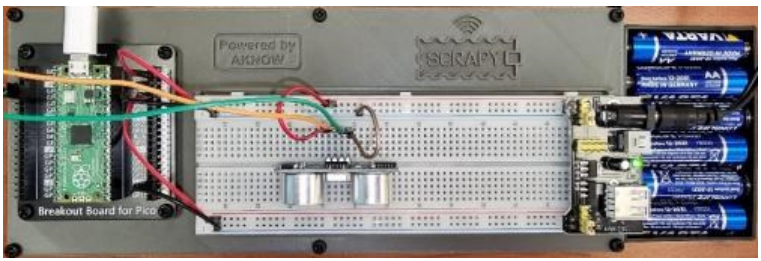
Shell

```
The distance from object is  153.9212 cm
The distance from object is  12.91395 cm
The distance from object is  22.8095 cm
The distance from object is  155.0874 cm
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



15. Capteur de mouvement PIR

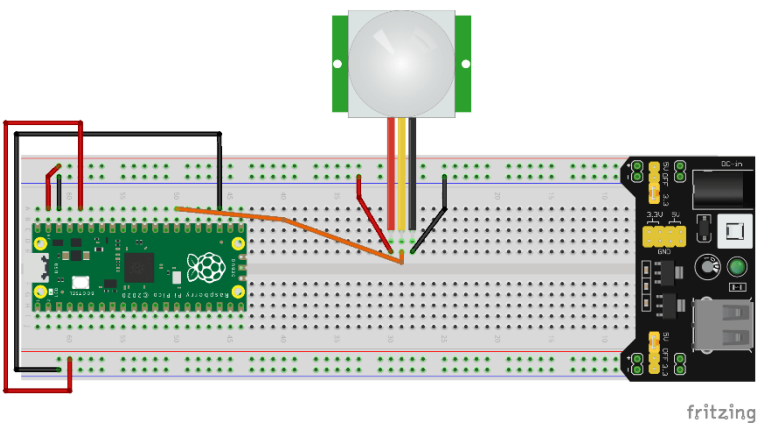
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur de mouvement PIR. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`mouvement.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-femelle
- 1 x câble micro-USB
- 1 x capteur de mouvement PIR

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– OUT (câble orange) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /motion.py @ 6 : 12

File Edit View Run Tools Help

[motion.py]

```
1  from machine import Pin
2  from time import sleep
3
4  PIR = Pin(21, Pin.IN, Pin.PULL_UP)
5
6  while True:
7      if PIR.value() == 1:
8          print("Motion detected!")
9          sleep(1)
10     if PIR.value() == 0:
11         print("No motion detected!")
12         sleep(1)
```

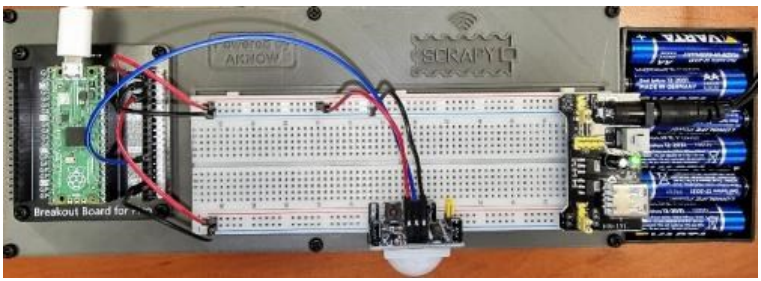
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  No motion detected!
  No motion detected!
  Motion detected!
  Motion detected!
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



16. Capteur DHT11

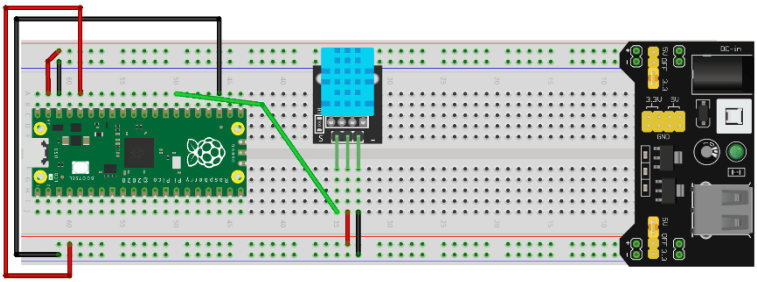
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur de température et d'humidité DHT11. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`dht11.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur de température DHT11

Schéma de câblage



– VCC (câble rouge) est connecté au rail 3v3 (+)

– GND (câble noir) est connecté au rail GND (-)

– S (câble vert) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /dht11.py @ 12 : 38

File Edit View Run Tools Help

[dht11.py]

```
1  from machine import Pin
2  from time import sleep
3  import dht
4
5  sensor = dht.DHT11(Pin(21))
6
7  while True:
8      sensor.measure()
9      temp = sensor.temperature()
10     hum = sensor.humidity()
11     print("Room temperature: ",temp, "Celcius")
12     print("Room humidity: ",hum, "%")
13     sleep(5)
```

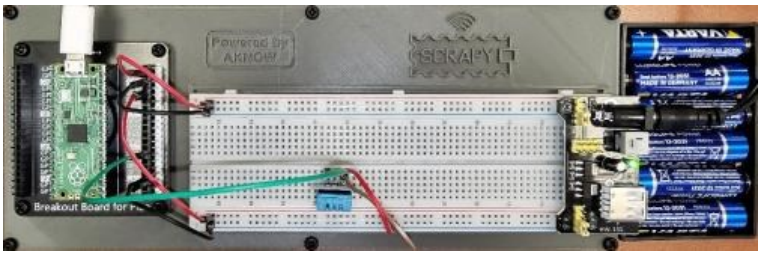
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  Room temperature:  27 Celcius
  Room humidity:  34 %
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



17. Capteur de vibrations SW-420

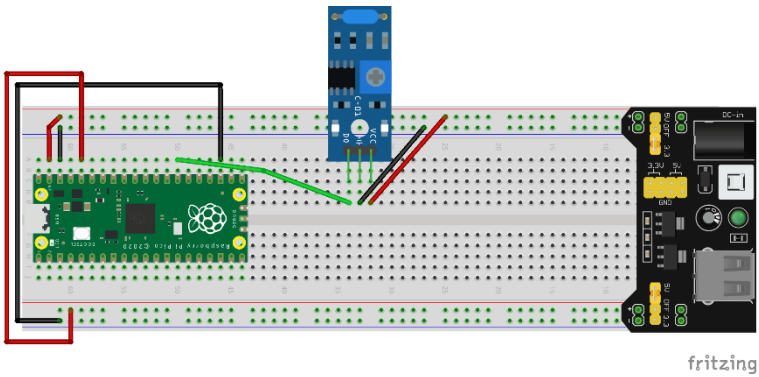
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur de vibrations SW-420. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`vibration.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur de vibrations SW-420

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– DO (câble vert) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /vibration.py @ 8 : 37

File Edit View Run Tools Help

[vibration.py]

```
1  from machine import Pin
2  from time import sleep
3
4  vibration = Pin(21, Pin.IN, Pin.PULL_UP)
5
6  while True:
7      if vibration.value() == 0:
8          print("Vibration detected!")
```

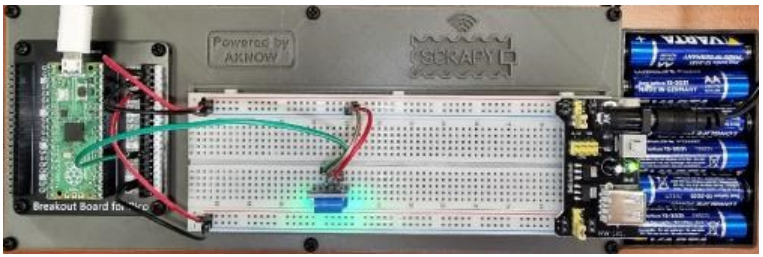
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  Vibration detected!
  Vibration detected!
  Vibration detected!
  Vibration detected!
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



18. Capteur de flamme

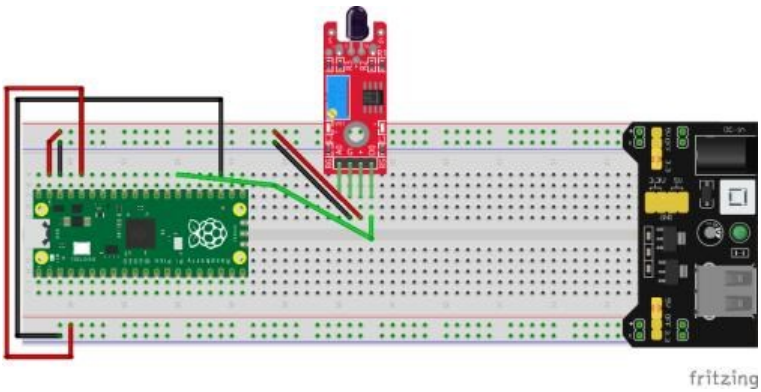
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur de flamme KY-026. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`flamme.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur de flamme KY-026

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– DO (câble vert) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /flame.py @ 10 : 16

File Edit View Run Tools Help

[flame.py]

```
1  from machine import Pin
2  from time import sleep
3
4  flame = Pin(21, Pin.IN)
5  sleep(2)
6
7  while True:
8     if flame.value() == 0:
9          print("Flame Detected")
10         sleep(3)
```

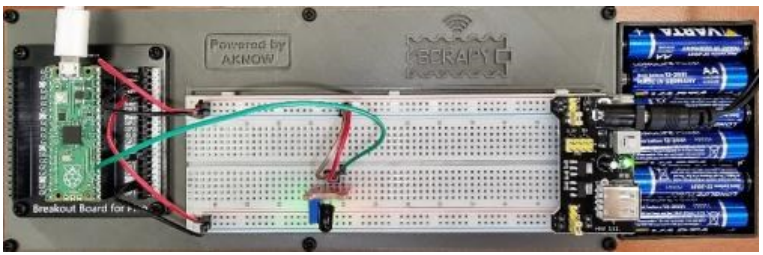
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



19. Capteur de détection sonore

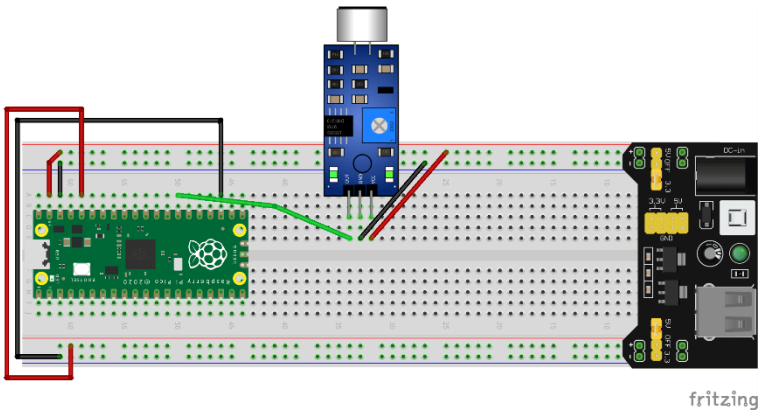
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur de détection sonore KY-037. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`son.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur sonore KY-037

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– DO/OUT (câble vert) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /sound.py @ 10 : 16

File Edit View Run Tools Help

[sound.py]

```
1  from machine import Pin
2  from time import sleep
3
4  sound = Pin(21, Pin.IN)
5  sleep(2)
6
7  while True:
8     if sound.value() == 0:
9          print("Sound Detected")
10         sleep(3)
```

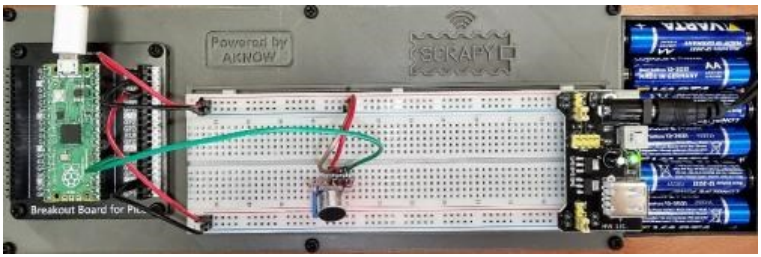
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  Sound Detected
  Sound Detected
  Sound Detected
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



20. Capteur d'humidité du sol

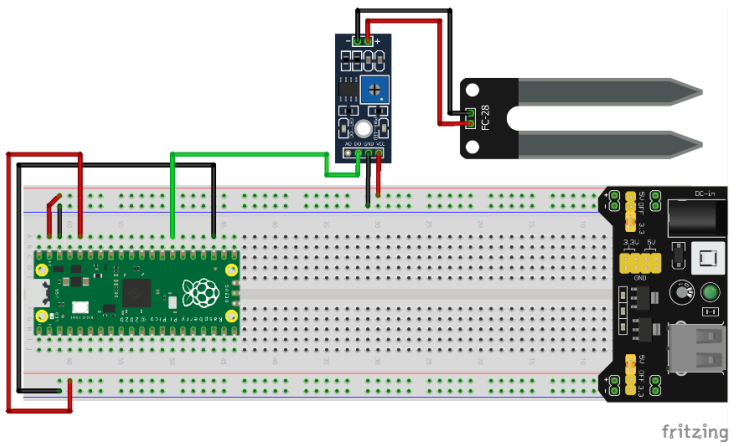
Description

Dans ce tutoriel, vous apprendrez comment connecter et contrôler le capteur d'humidité du sol. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous..., choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`sol.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur sonore KY-037

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– DO/OUT (câble vert) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /soil.py @ 10 : 16

File Edit View Run Tools Help

[soil.py]

```
1  from machine import Pin
2  from time import sleep
3
4  soil = Pin(21, Pin.IN)
5  sleep(2)
6
7  while True:
8     if soil.value() == 0:
9          print("Moisture Detected")
10         sleep(3)
```

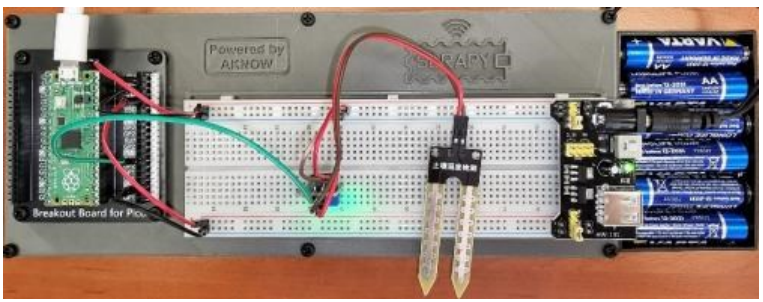
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  Moisture Detected
  Moisture Detected
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



21. Capteur infrarouge infrarouge

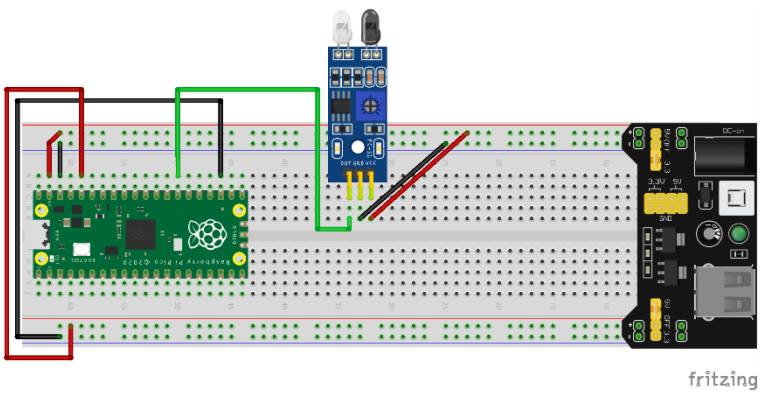
Description

Dans ce didacticiel, vous apprendrez comment connecter et contrôler le capteur infrarouge IR. Ouvrez Thonny Python, puis allez dans Fichier→Enregistrer sous…, choisissez Raspberry Pi Pico et enregistrez votre fichier sous le nom`ir.py`. Il est ensuite temps de connecter l'électronique et d'écrire votre programme. Veuillez suivre les instructions ci-dessous.

Matériel requis

- 1 x Raspberry Pi Pico
- 1 x kit de planche à pain Pico
- 1 x planche à pain pleine grandeur
- 3 x câbles de liaison mâle-mâle
- 1 x câble micro-USB
- 1 x capteur infrarouge IR

Schéma de câblage



– VCC (câble rouge) est connecté au rail 5V (+)

– GND (câble noir) est connecté au rail GND (-)

– OUT (câble vert) est connecté à la broche GPIO21

Code

Code MicroPython pour le tutoriel :

Thonny - Raspberry Pi Pico :: /ir.py @ 13 : 17

File Edit View Run Tools Help

[ir.py]

```
1  from machine import Pin
2  from time import sleep
3
4  ir = Pin(21, Pin.IN)
5  sleep(2)
6
7  while True:
8     if ir.value() == 1:
9          print("No obstacles")
10         sleep(3)
11    else:
12         print("Obstacle detected!")
13         sleep(3)
14
```

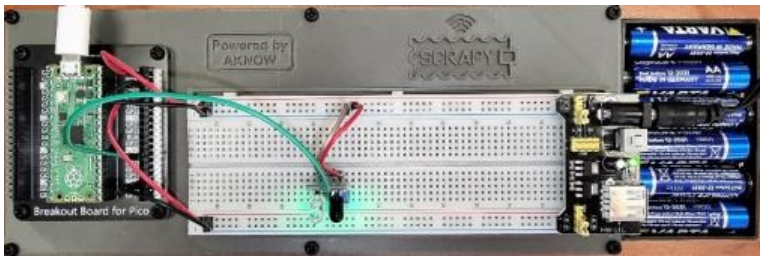
Shell

```
MicroPython v1.19.1 on 2022-06-18; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  Obstacle detected!
  Obstacle detected!
```

MicroPython (Raspberry Pi Pico)

Exemple d'image

Image de l'apparence du didacticiel avec le matériel fourni :



ANNEXE : Tableau récapitulatif de MicroPython

Sortie numérique		
Appel de la classe Pin	`depuismachineimporterÉpingle`	
Initialisation de l'objet sortie numérique	`LED = broche (valeur_pin, Pin.OUT)`	`valeur_pin`de 0 à 40
Sortie numérique ouverte (sortie 3,3 V)	`led.valeur(1)`	SUR
Fermer la sortie numérique (sortie 0 V)	`led.valeur(0)`	DÉSACTIVÉ

Entrée numérique		
Appel de la classe Pin	`depuismachineimporterÉpingle`	
Initialisation de l'objet sortie numérique	`bouton = Épingler (valeur_pin, Pin.IN)`	`valeur_pin`de 0 à 40
	`bouton = Épingler (valeur_pin), Pin.IN, Pin.PULL_UP)`	Activation de la résistance PULL UP
	`bouton = Épingler (valeur_pin), Pin.IN, Pin.PULL_DOWN)`	Activation de la résistance PULL DOWN
Lecture d'entrée	`valeur = bouton.valeur(1)`	La valeur de retour peut être 0 si la broche est à 0 V, ou 1 si la broche est à 3,3 V

Sortie analogique (modulation de largeur d'impulsion - PWM)		
Appel de la classe PWM	`depuismachineimporterMLI`	
Initialisation de la sortie analogique	`LED = PWM (broche (valeur_pin),fréquence)`	`valeur_pin` de 0 à 40 `fréquence`en HZ, de 0 à 78125
Lecture d'entrée	`conduit.devoir(cycle de service)`	`cycle de service`de 0 à 1023 (sortie 0 V à sortie 3,3 V respectivement)

Entrée analogique		
Appel de la classe ADC	`depuismachineimporterCDA`	
Initialisation de l'entrée analogique	`pot = ADC(Broche(valeur_pin))`	`valeur_pin` peut être GPIO26, GPIO27 et GPIO28
Déclaration à quelle tension l'entrée donnera sa valeur maximale (dans l'ESP32 généralement 3,3 V)	`pot.atten(ADC.ATTN_11DB)`	**ADC.ATTN_0DB :**tension sur toute la plage : 1,2 V **ADC.ATTN_2_5DB :**tension sur toute la plage : 1,5 V **ADC.ATTN_6DB**: tension sur toute la plage : 2,0 V **ADC.ATTN_11DB**: tension sur toute la plage : 3,3 V
Déclaration de la plage de valeurs d'entrée (12 bits par défaut)	`pot.largeur(ADC.WIDTH_10BIT)`	**ADC.WIDTH_9BIT**: plage de 0 à 511 **ADC.WIDTH_10BIT**: plage de 0 à 1023 **ADC.WIDTH_11BIT**: plage de 0 à 2047 **ADC.WIDTH_12BIT**: plage de 0 à 4095
Lecture d'entrée	`valeur = pot.readl()`	`valeur`est un entier compris entre 0 et le maximum de la plage spécifiée par l'instruction ADC.WIDTH_#BIT (voir précédent)

La bibliothèque du temps		
Appeler la classe de sommeil	`depuistempsimporterdormir`	
Utilisation de la fonction sommeil	`dormir(seconde)`	`seconde` est le nombre de secondes pendant lesquelles le programme sera retardé
Appeler la classe temporelle	`importertemps`	
Utilisation de la fonction temps	`heure_actuelle = heure()`	Le`heure actuelle`La variable prendra une valeur numérique, égale au nombre de secondes

		depuis la dernière réinitialisation de la carte.

Structure de l'instruction If	
si<expression1> : <instruction1> Elif<expression2> : <instruction2> Elif<expression3> : <instruction3> (...) autre: <déclaration>	<expr#> : la condition de contrôle qui doit renvoyer True ou False <statement#> : ensemble de commandes à exécuter lorsque la condition adjacente est satisfaite <expr#> (par exemple l'ensemble <statement2> est exécuté lorsque <expr2> est satisfait) <statementn> : ensemble d'instructions exécutées lorsqu'aucune des conditions <expr#> n'est satisfaite

Structure de la boucle While	
alors que<expression> : <déclaration(s)>	<expr> : la condition de contrôle qui doit renvoyer True ou False <statement#> : ensemble de commandes à exécuter tant que la condition <expr> est satisfaite

Pour la structure en boucle	
pour<var>dans<itérable> : <déclaration(s)>	**<itérable>**: une collection d'objets, par exemple une liste contenant des nombres, des caractères alphanumériques, etc. **<var>**: une variable à laquelle est affectée la valeur de l'élément suivant de la collection <iterable>. **<état(s)>**: un ensemble d'instructions exécutées à chaque itération
pour<var>dansrange(<début>, <fin>, <étape>) : <déclaration(s)>	**range(<début>, <fin>, <étape>)**: fonction qui renvoie une séquence de nombres de <start> à <end>-1, avec une différence <step> entre deux nombres consécutifs (les paramètres <start> et <step> sont facultatifs).

		<var>: variable à laquelle est affectée la valeur de l'élément suivant de la séquence produite par range. **<déclaration(s)>**: un ensemble d'instructions exécutées à chaque itération
Divers		
DHT11	`importerdht`	Importation de la bibliothèque DHT
	`capteur = dht.DHT11(Broche(code PIN))`	Initialisation de la variable capteur avec ses valeurs associéescode PIN.
	`capteur.mesure()`	Mise à jour des valeurs du capteur
	`temp = capteur.température()`	Sauvegarde de la valeur de température actuelle
	`hum = capteur.humidité()`	Sauvegarde de la valeur d'humidité actuelle
ÉCRAN OLED	`depuis machine importerI2C`	Importer la bibliothèque I2C
	`importerssd1306`	Importation de la bibliothèque ssd1306
	`i2c = I2C(-1, scl=Broche(1), sda=Pin(0))`	Initialisation de la variable i2c sur les pins SCL & SDA du Pico

	`oled_width =128`
	`oled_hauteur =64`
	`oled = ssd1306.SSD1306_I2C (oled_width, oled_height, i2c)`
		Initialisation de l'écran

	`oled.text('Bonjour, Monde 1 !',0,0)`
	`oled.text('Bonjour, Monde 2 !',0,dix)`
	`oled.text('Bonjour, Monde 3 !',0,20)`
		Stockage des messages dans le tampon d'écran

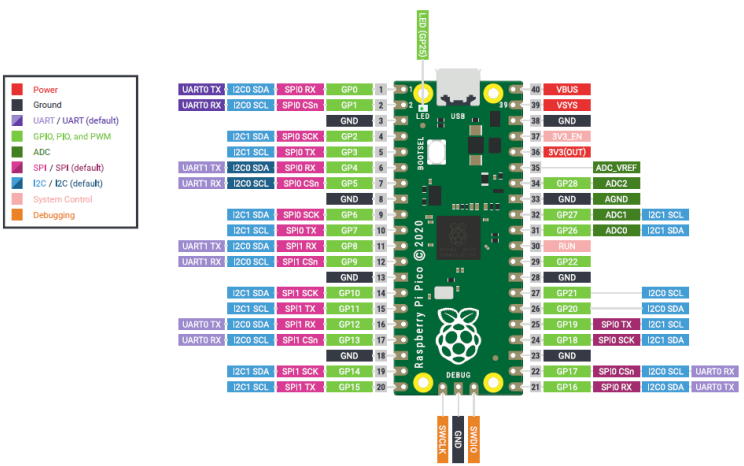
	`oled.show()`	Affichage des messages (nécessaire pour afficher les messages stockés dans le tampon d'écran)
	`display.pixel(3, 4, 1)`	Réglez le pixel situé à la position (x,y) sur l'écran, avec x=3 & y=4, à l'état 1 (c'est-à-dire affichage)

Co-funded by
the European Union

SCRAPY

2021-1-FR01-KA220-SCH-000031617

Brochage du Raspberry Pi Pico



Raspberry Pi

w :scrapykit.eu
e : info@scrapykit.eu