

---

# SunFounder 3in1 Kit

[www.sunfounder.com](http://www.sunfounder.com)

janv. 03, 2024





---

## Table des matières

---

<b>1</b>	<b>Découvrez les Composants de votre Kit</b>	<b>3</b>
1.1	Carte SunFounder R3	4
1.2	Module ESP8266	6
1.3	Plaque d'essai	9
1.4	Résistance	10
1.5	Condensateur	13
1.6	Fils de Cavalier	15
1.7	74HC595	15
1.8	LED	17
1.9	LED RVB	18
1.10	Affichage 7 segments	20
1.11	I2C LCD1602	23
1.12	Buzzer	25
1.13	Moteur TT	26
1.14	Servomoteur	27
1.15	Pompe Centrifuge	30
1.16	Module de Contrôle Moteur L9110	31
1.17	Bouton	33
1.18	Interrupteur à Lame Souple	34
1.19	Potentiomètre	35
1.20	Module de Joystick	37
1.21	Récepteur IR	38
1.22	Photorésistance	40
1.23	Thermistance	41
1.24	Capteur d'Humidité et de Température DHT11	43
1.25	Module de Suivi de Ligne	44
1.26	Module d'Humidité du Sol	45
1.27	Module d'Évitement d'Obstacle	47
1.28	Module Ultrasonique	48
<b>2</b>	<b>Découverte d'Arduino</b>	<b>51</b>
2.1	Qu'est-ce qu'Arduino ?	51
2.2	Que peut faire Arduino ?	52
2.3	Comment construire un projet Arduino	52
<b>3</b>	<b>Cours Vidéo Arduino</b>	<b>83</b>
3.1	Vidéo 1 - Présentation de ce Kit	83

3.2	Vidéo 2 - Introduction à l'IDE Arduino . . . . .	84
3.3	Vidéo 3 : Bases de Programmation et Projets LED . . . . .	84
3.4	Vidéo 4 : Systèmes Numériques et Moniteur Série . . . . .	84
3.5	Vidéo 5 : Types de Données et Variables . . . . .	85
3.6	Vidéo 6 : Contrôle de Buzzer, Moteur et Pompe à Eau . . . . .	85
3.7	Vidéo 7 : Boutons-Poussoirs et Interrupteurs à Lame Souple . . . . .	85
3.8	Vidéo 8 : MLI (PWM) et Structures de Boucles . . . . .	86
3.9	Vidéo 9 : Mesure de Tension . . . . .	86
3.10	Vidéo 10 : Instructions Conditionnelles et Tableaux . . . . .	87
3.11	Vidéo 11 : Capteur DHT11 . . . . .	87
3.12	Vidéo 12 : Fonctions et Instruction Switch-Case . . . . .	87
3.13	Vidéo 13 : Manipuler le Joystick . . . . .	88
3.14	Vidéo 14 : millis() et map() . . . . .	88
3.15	Vidéo 15 : Automatisation de l'Arrosage des Plantes . . . . .	89
3.16	Vidéo 16 : Évitement d'Obstacles par Infrarouge . . . . .	89
3.17	Vidéo 17 : Interruption . . . . .	89
3.18	Vidéo 18 : Servomoteur . . . . .	90
3.19	Vidéo 19 : Capteur Ultrasonique . . . . .	90
3.20	Vidéo 20 : LCD 1602 . . . . .	91
3.21	Vidéo 21 : Télécommande Infrarouge . . . . .	91
3.22	Vidéo 22 : Afficheur à 7 Segments . . . . .	91
3.23	Vidéo 23 : Assemblage d'une Voiture Intelligente . . . . .	92
3.24	Vidéo 24 : Contrôle d'une Voiture Intelligente . . . . .	92
3.25	Vidéo 25 : Évitement d'Obstacles pour Voiture Intelligente . . . . .	93
3.26	Vidéo 26 : Capteur Ultrasonique pour Voiture Intelligente . . . . .	93
3.27	Vidéo 27 : Suivi de la Main par Voiture Intelligente . . . . .	93
3.28	Vidéo 28 : Voiture Intelligente Autonome . . . . .	94
3.29	Vidéo 29 : Télécommande de Voiture Intelligente . . . . .	94
3.30	Vidéo 30 : Suivi de Ligne par Voiture Intelligente . . . . .	95
3.31	Vidéo 31 : Surveillance de la Température par Wi-Fi . . . . .	95
<b>4</b>	<b>Téléchargez le Code . . . . .</b>	<b>97</b>
<b>5</b>	<b>Projets de Base . . . . .</b>	<b>99</b>
5.1	1. Écriture Numérique . . . . .	99
5.2	2. Écriture Analogique . . . . .	109
5.3	3. Lecture Numérique . . . . .	116
5.4	4. Lecture Analogique . . . . .	129
5.5	5. Syntaxe Avancée . . . . .	143
5.6	6. Projet Amusant . . . . .	205
<b>6</b>	<b>Projets de Voiture . . . . .</b>	<b>229</b>
6.1	Assembler la Voiture . . . . .	229
6.2	1. Mouvement . . . . .	241
6.3	2. Mouvement par Code . . . . .	246
6.4	3. Accélération . . . . .	250
6.5	4. Suivre la ligne . . . . .	251
6.6	5. Jouer avec le module d'évitement d'obstacles . . . . .	255
6.7	6. Jouer avec le module ultrasonique . . . . .	259
6.8	7. Suivez Votre Main . . . . .	262
6.9	8. Voiture Autonome . . . . .	264
6.10	9. Télécommande . . . . .	267
6.11	10. Démarrage en Un Clic . . . . .	271
6.12	11. Calibration de la Vitesse . . . . .	272

<b>7 Projets IoT</b>	<b>275</b>
7.1 1. Démarrer avec Blynk	276
7.2 2. Obtenir des Données depuis Blynk	290
7.3 3. Envoyer des Données à Blynk	299
7.4 4. Lecteur de Musique Cloud	304
7.5 5. Surveillance de l'Environnement Domestique	310
7.6 6. Moniteur de Plantes	318
7.7 7. Portail à Limite de Courant	325
7.8 8. Voiture IoT	332
<b>8 Jouez avec Scratch</b>	<b>341</b>
8.1 1.1 Installer PictoBlox	342
8.2 1.2 Présentation de l'interface	343
8.3 1.3 Guide rapide sur PictoBlox	344
8.4 2.1 Lampe de Table	361
8.5 2.2 LED Respiration	368
8.6 2.3 Balles Colorées	375
8.7 2.4 LCD1602	383
8.8 2.5 Souris Mobile	392
8.9 2.6 Sonnette	399
8.10 2.7 Alarme de Température Basse	406
8.11 2.8 Réveil Lumineux	413
8.12 2.9 Lecture de la Température et de l'Humidité	420
8.13 2.10 Pendule	426
8.14 2.11 Ventilateur Rotatif	435
8.15 2.12 Balle Sensible à la Lumière	442
8.16 2.13 JEU - Tir	455
8.17 2.14 JEU - Gonfler le Ballon	468
8.18 2.15 JEU - Étoiles Croisées	478
8.19 2.16 JEU - Manger la Pomme	487
8.20 2.17 JEU - Perroquet Volant	498
8.21 2.18 JEU - Clone de Breakout	507
8.22 2.19 JEU - Pêche	518
8.23 2.20 JEU - Ne Touchez Pas la Tuile Blanche	527
8.24 2.21 JEU - Protège ton Cœur	548
8.25 2.22 JEU - Tueur de Dragon	565
8.26 3.1 Tester la Voiture	589
8.27 3.2 Mouvement	594
8.28 3.3 Suivre la ligne	601
8.29 3.4 Suivez Votre Main	605
8.30 3.5 Évitement d'obstacles	609
8.31 3.6 Suivez Votre Main 2	617
8.32 3.7 Évitement d'obstacles 2	621
<b>9 Cours Vidéo</b>	<b>627</b>
<b>10 FAQ</b>	<b>631</b>
10.1 Comment utiliser Blynk sur un appareil mobile ?	631
10.2 Comment re-graver le firmware pour le module ESP8266 ?	633
<b>11 Remerciements</b>	<b>645</b>
<b>12 Avis de droit d'auteur</b>	<b>647</b>

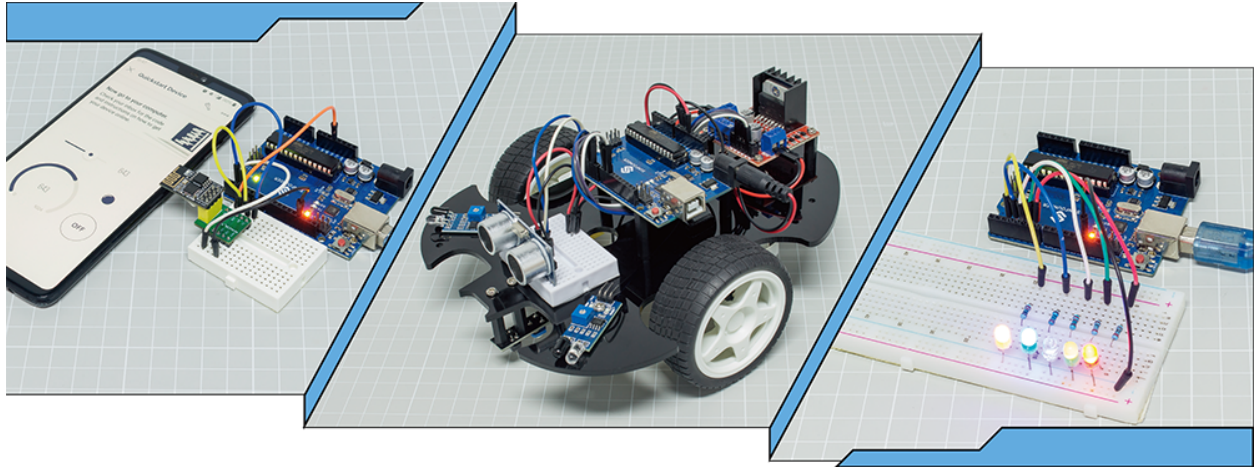


Merci d'avoir choisi notre kit de démarrage 3 en 1.

**Note :** Ce document est disponible dans les langues suivantes :

- 
- 
- 
- 
- 

Veuillez cliquer sur les liens respectifs pour accéder au document dans votre langue préférée.



Lorsque vous avez acheté un kit d'apprentissage en ligne, est-ce qu'il était accompagné d'un simple PDF ou livret décrivant uniquement les étapes de construction du projet ?

Ou peut-être souhaitez-vous construire votre propre voiture intelligente, mais celles que vous trouvez en ligne sont chères et compliquées ?

Ou avez-vous déjà vu des projets IoT utiles et intéressants réalisés par d'autres, mais vous ne savez pas par où commencer ?

Tous ces problèmes peuvent être résolus avec notre kit de démarrage 3 en 1.

Dans le kit de démarrage 3-en-1, vous trouverez un cours complet sur Arduino pour aider les débutants à apprendre Arduino, ainsi qu'une variété de projets intéressants que d'autres kits d'apprentissage n'offrent pas, comme des projets de voitures intelligentes et des projets IoT. Vous maîtriserez Arduino en suivant pas à pas le cours du kit, au lieu de simplement copier et coller du code, vous écrirez votre propre code et implémenterez votre projet Arduino comme bon vous semble.

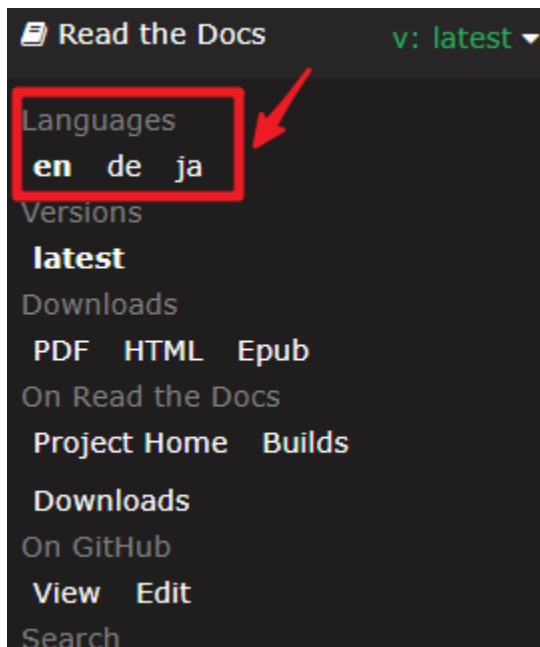
De plus, le kit fournit également plus de 30 projets de programmation Scratch pour les jeunes étudiants et les amateurs, donc les débutants n'ont pas besoin d'expérience en programmation pour écrire et créer leurs propres œuvres !

Allez-y ! Commencez à programmer Arduino en toute confiance, de zéro à héros !

Si vous avez des questions, veuillez envoyer un courriel à [service@sunfounder.com](mailto:service@sunfounder.com) et nous vous répondrons dès que possible.

### À propos de la langue d'affichage

Ce document est également disponible dans d'autres langues. Pour changer la langue d'affichage, veuillez cliquer sur l'icône **Read the Docs** située dans le coin inférieur gauche de la page.



Contenu

# CHAPITRE 1

## Découvrez les Composants de votre Kit

Après avoir ouvert l'emballage, veuillez vérifier si le nombre de composants correspond à la description du produit et si tous les composants sont en bon état.

### Packing List

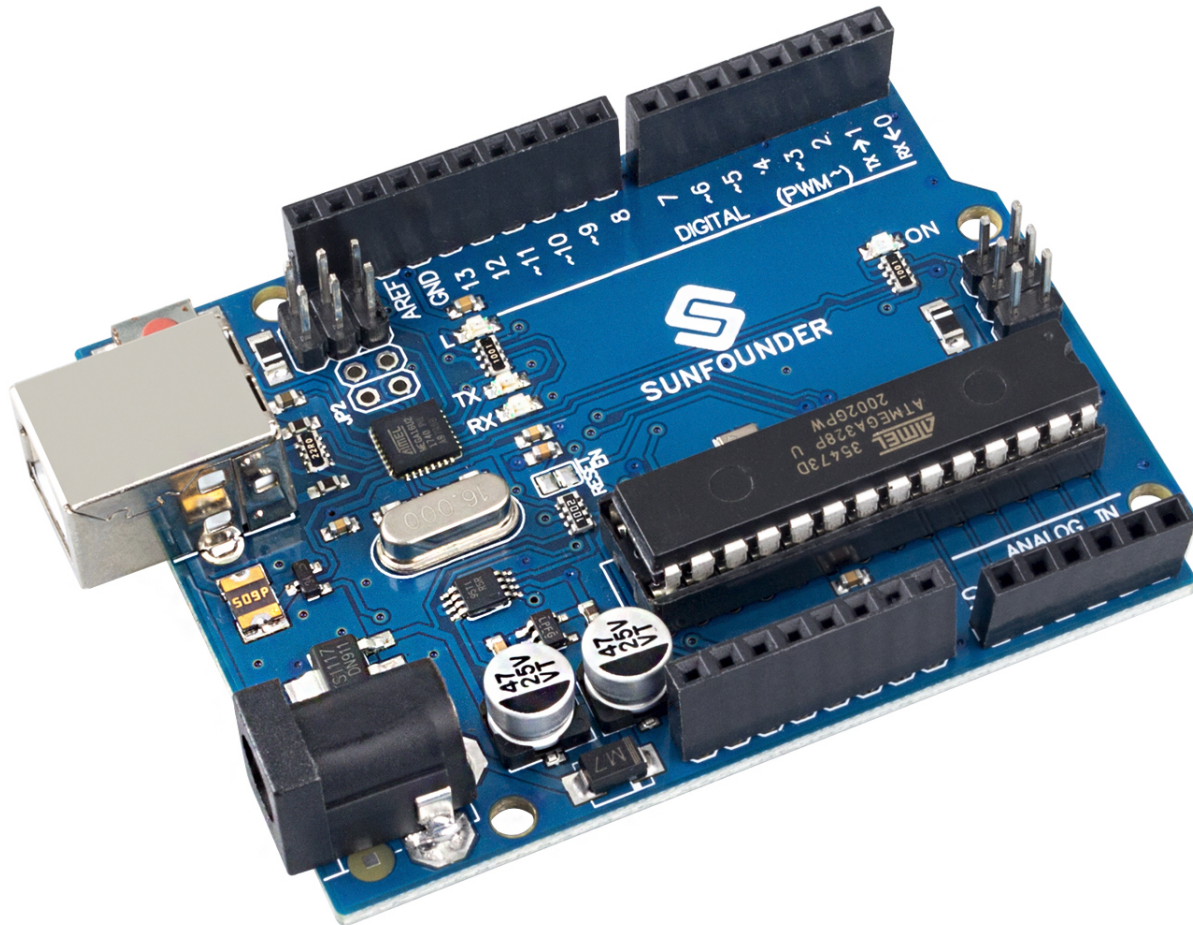
 Resistor(10Ω) 10 pcs	 Resistor(100Ω) 10 pcs	 Resistor(220Ω) 30 pcs	 Resistor(330Ω) 10 pcs	 Resistor(1KΩ) 10 pcs	 Resistor(2KΩ) 10 pcs	 Resistor(5.1KΩ) 10 pcs	 Resistor(10KΩ) 10 pcs	 Resistor(100KΩ) 10 pcs
 Resistor(1MΩ) 10 pcs	 Reed Switch 2 pcs	 Button 5 pcs	 Active/Passive Buzzer 1+1 pcs	 Capacitor 104 pF 5 pcs	 Potentiometer 1 pc	 74HC595 1 pc	 7-segment Display 1 pc	 IR Receiver 1 pc
 Green LED 5 pcs	 Red LED 5 pcs	 White LED 5 pcs	 Blue LED 5 pcs	 Yellow LED 5 pcs	 RGB LED 1 pc	 Photoresistor/ Thermistor 1+1 pcs	 M2.5x6 Screw 10 pcs	 M3 Nut 8 pcs
 M3x10 Screw 4 pcs	 M3x6 Screw 30 pcs	 M3x30 Screw 6 pcs	 M2.5x11 Standoff 6 pcs	 M3x24 Standoff 7 pcs	 M3x12 Standoff 10 pcs	 Screwdriver 1 pc	 DHT11 1 pc	 ESP8266 Module 1 pc
 Soil Moisture Module 1 pc	 Joystick Module 1 pc	 Ultrasonic Module 1 pc	 IR Obstacle Avoidance Module 2 pcs	 I2C LCD 1602 1 pc	 R3 Board 1 pc	 Line Tracking Module 1 pc	 L9110 Module 1 pc	 ESP8266 Adapter Module 1 pc
 USB Cable 1 pc	 TT Wheel 2 pcs	 TT Motor 2 pcs	 1" Universal Wheel 1 pc	 Remote Control 1 pc	 Pump 1 pc	 Acrylic Plate 1 pc	 9V Servo 1 pc	 Mini Breadboard 1 pc
 Breadboard 1 pc	 Velcro 4 pcs	 Tube 1 pc	 Jump Wire F/M 20 pcs	 F/F DuPont Wire 20 pcs	 9V Battery Cable 1 pc	 Jump Wire M/M 65 pcs	 9V Battery 1 pc	Online Tutorials: <a href="https://3in1-kit-v2.rtfid.io">https://3in1-kit-v2.rtfid.io</a>

Voici une introduction à chaque composant, incluant le principe de fonctionnement du composant et les projets correspondants.

### Carte de Contrôle



## 1.1 Carte SunFounder R3



---

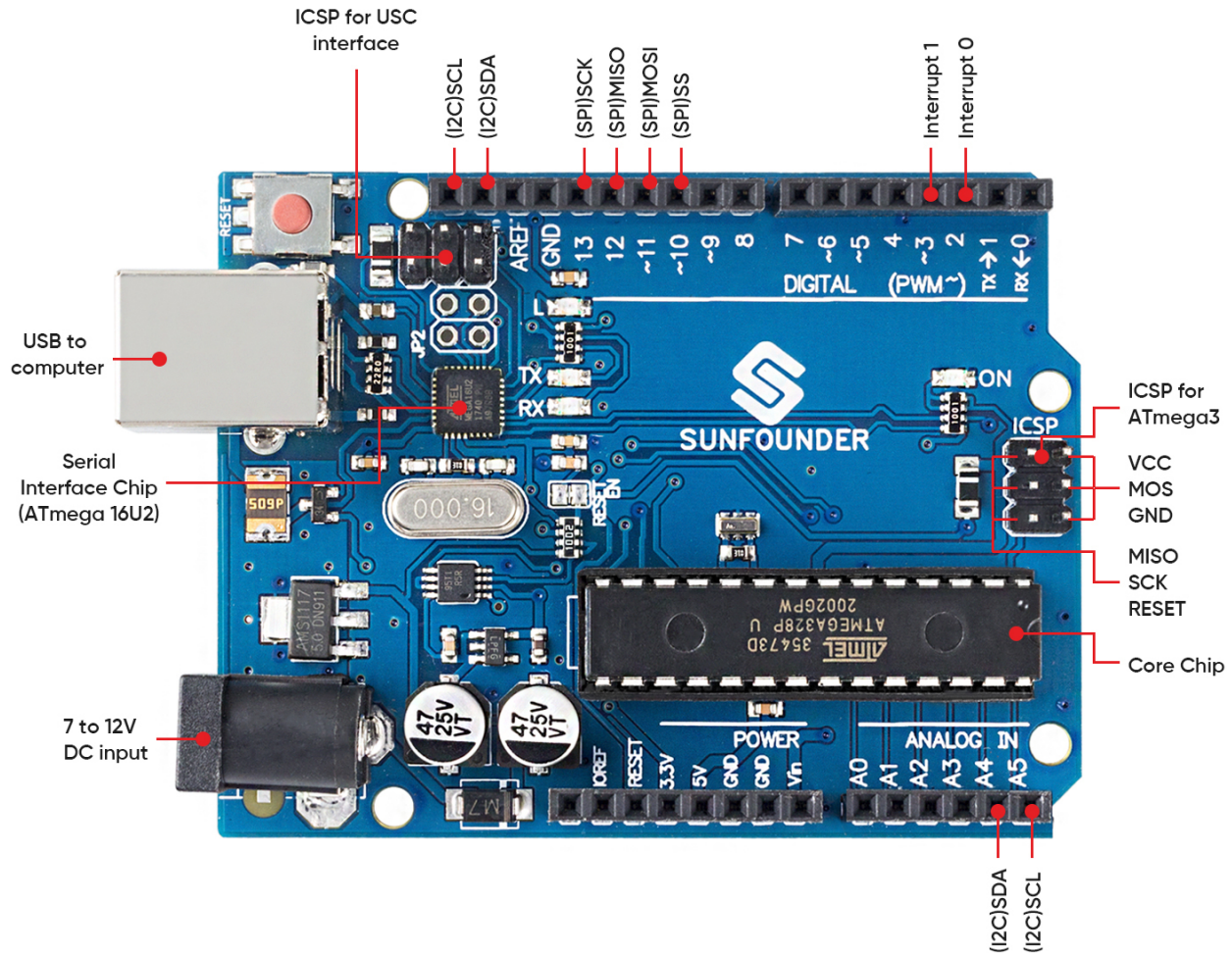
**Note :** La carte SunFounder R3 est une carte principale ayant pratiquement les mêmes fonctions que l' [Arduino Uno](#). Les deux cartes peuvent être utilisées de manière interchangeable.

---

La carte SunFounder R3 est une carte microcontrôleur basée sur l'ATmega328P ([fiche technique](#)). Elle dispose de 14 broches d'entrée/sortie numériques (dont 6 peuvent être utilisées comme sorties PWM), 6 entrées analogiques, un résonateur céramique de 16 MHz (CSTCE16M0V53-R0), une connexion USB, un jack d'alimentation, un en-tête ICSP et un bouton de réinitialisation. Elle contient tout le nécessaire pour soutenir le microcontrôleur ; il suffit de la connecter à un ordinateur avec un câble USB ou de l'alimenter avec un adaptateur AC-DC ou une batterie pour commencer.

### Paramètres Techniques



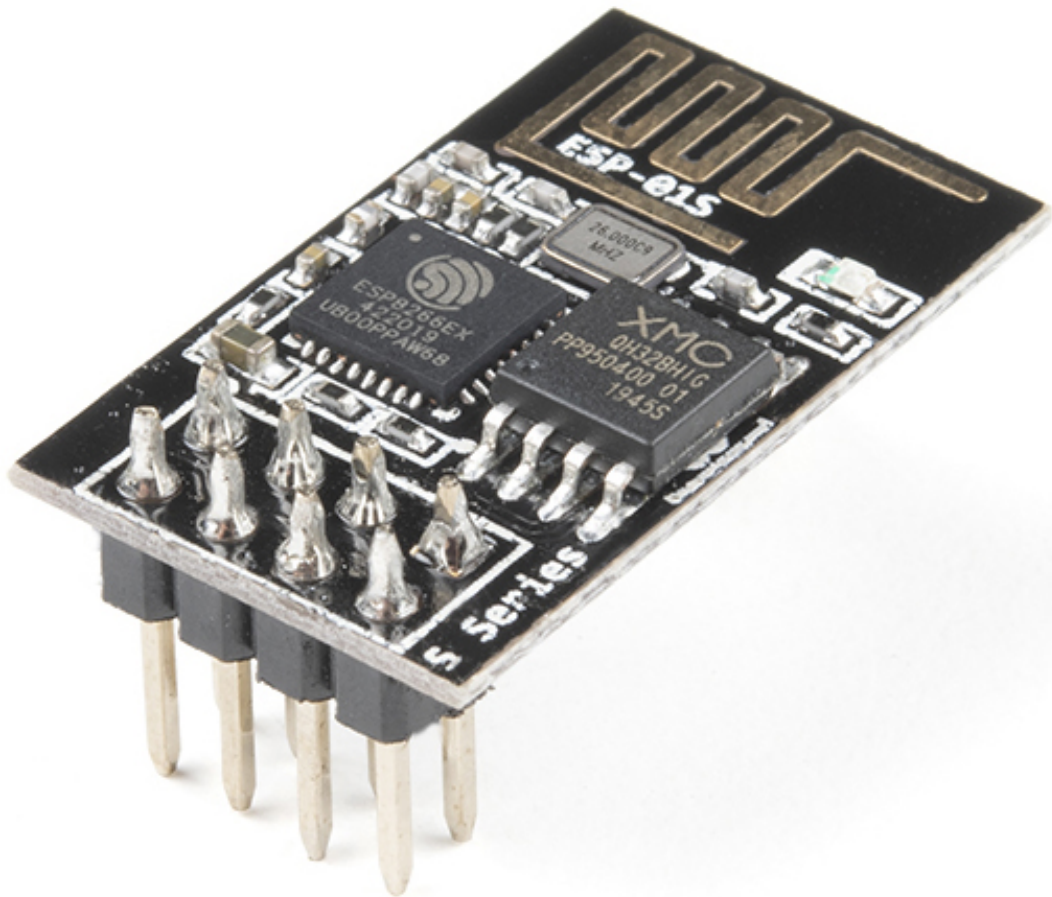


- MICROCONTRÔLEUR : ATmega328P
- TENSION DE FONCTIONNEMENT : 5V
- TENSION D'ENTRÉE (RECOMMANDÉE) : 7-12V
- TENSION D'ENTRÉE (LIMITE) : 6-20V
- BROCHES D'ENTRÉE/SORTIE NUMÉRIQUES : 14 (0-13, dont 6 fournissent une sortie PWM (3, 5, 6, 9-11))
- BROCHES D'ENTRÉE/SORTIE NUMÉRIQUES PWM : 6 (3, 5, 6, 9-11)
- BROCHES D'ENTRÉE ANALOGIQUES : 6 (A0-A5)
- COURANT CC PAR BROCHE D'E/S : 20 mA
- COURANT CC POUR LA BROCHE 3.3V : 50 mA
- MÉMOIRE FLASH : 32 KB (ATmega328P) dont 0.5 KB utilisés par le bootloader
- SRAM : 2 KB (ATmega328P)
- EEPROM : 1 KB (ATmega328P)
- VITESSE D'HORLOGE : 16 MHz
- LED\_INTÉGRÉE : 13
- LONGUEUR : 68.6 mm
- LARGEUR : 53.4 mm
- POIDS : 25 g
- Port I2C : A4(SDA), A5(SCL)

#### De Plus

- [IDE Arduino](#)
- [Arduino Programming Language Reference](#)
- [Télécharger et Installer Arduino IDE 2.0](#)
- [Fiche Technique ATmega328P](#)

## 1.2 Module ESP8266



L'ESP8266 est une puce Wi-Fi à bas coût, dotée d'un logiciel de réseau TCP/IP intégré, et d'une capacité de micro-contrôleur, produite par Espressif Systems à Shanghai, Chine.

La puce a attiré l'attention des créateurs occidentaux pour la première fois en août 2014 avec le module ESP-01, fabriqué par un fabricant tiers Ai-Thinker. Ce petit module permet aux microcontrôleurs de se connecter à un réseau Wi-Fi et d'établir des connexions TCP/IP simples en utilisant des commandes de style Hayes. Cependant, au début, il y avait presque aucune documentation en anglais sur la puce et les commandes qu'elle acceptait. Le prix très bas et le fait qu'il y avait très peu de composants externes sur le module, ce qui suggérait qu'il pourrait éventuellement être très bon marché en volume, ont attiré de nombreux hackers à explorer le module, la puce, et le logiciel qui y est installé, ainsi qu'à traduire la documentation chinoise.

Broches de l'ESP8266 et leurs fonctions :

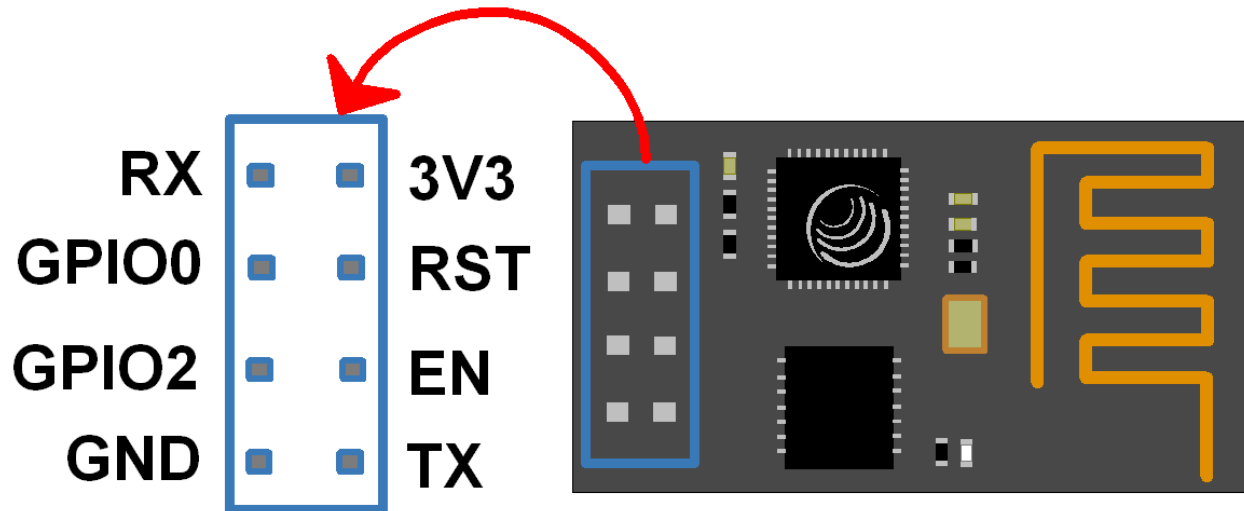
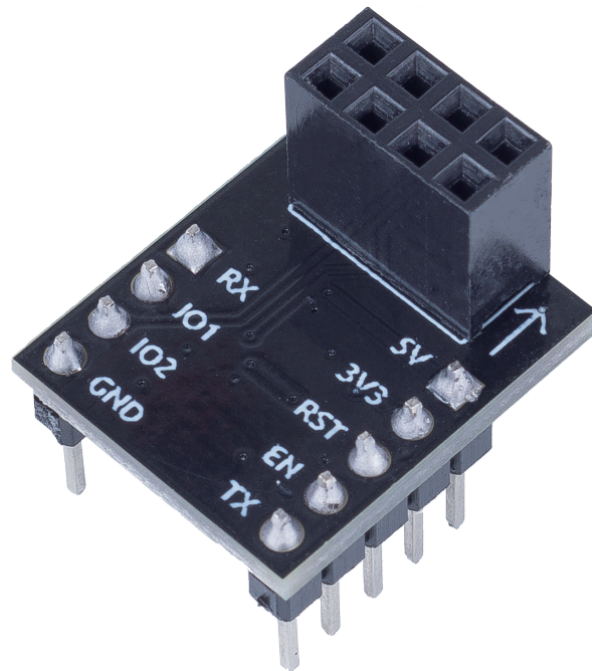


TABLEAU 1 – Broches ESP8266-01

Broche	Nom	Description
1	TXD	UART_TXD, émission ; Entrée/Sortie Générale : GPIO1 ; Il est interdit de tirer vers le bas lors du démarrage.
2	GND	GND
3	CU_PD	Fonctionne à un niveau élevé ; S'éteint lorsque un niveau bas est fourni.
4	GPIO2	Doit être à un niveau élevé lors de la mise sous tension, le tirage vers le bas matériel est interdit ; Tiré vers le haut par défaut ;
5	RST	Signal de réinitialisation externe, réinitialisation lorsque un niveau bas est fourni ; fonctionne lorsqu'un niveau élevé est fourni (niveau élevé par défaut) ;
6	GPIO0	Indicateur d'état WiFi ; Sélection du mode de fonctionnement : Tiré vers le haut : Démarrage Flash, mode de fonctionnement ; Tiré vers le bas : Téléchargement UART, mode de téléchargement
7	VCC	Alimentation électrique (3.3V)
8	RXD	UART_RXD, réception ; Entrée/Sortie Générale : GPIO3 ;

- ESP8266 - Espressif
- Ensemble d'instructions AT ESP8266

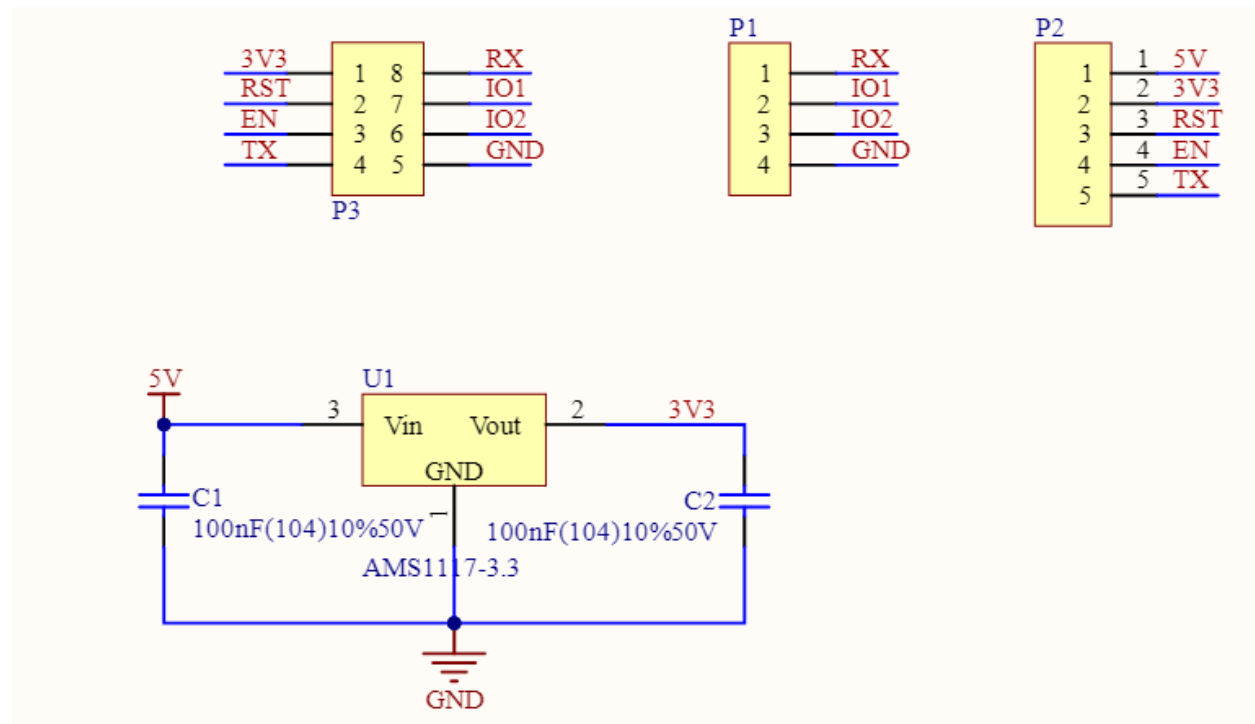
### 1.2.1 Adaptateur ESP8266



L'adaptateur ESP8266 est une carte d'extension qui permet d'utiliser le module ESP8266 sur une breadboard.

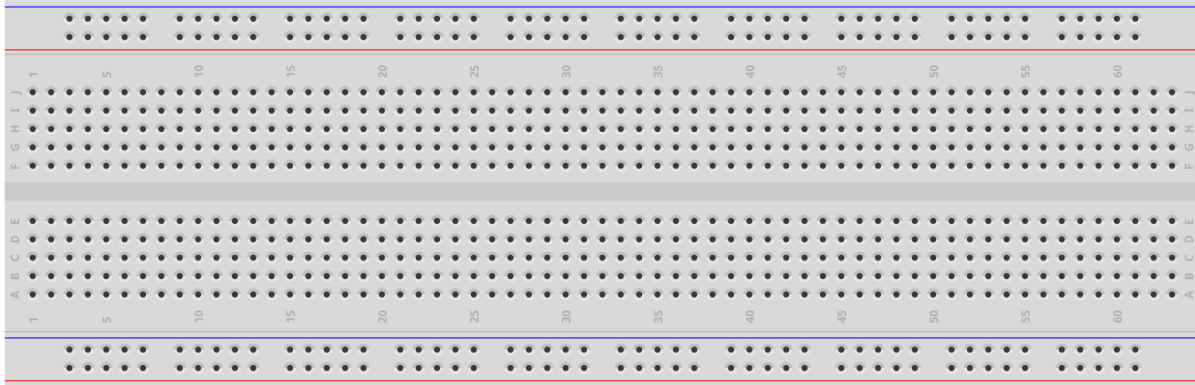
Il correspond parfaitement aux broches de l'ESP8266 lui-même, et ajoute également une broche 5V pour recevoir la tension de la carte Arduino. Le chip AMS1117 intégré est utilisé pour alimenter le module ESP8266 après avoir réduit la tension à 3.3V.

Le schéma est le suivant :



**Exemple**— *Projets IoT* (Projet IoT)**Basique**

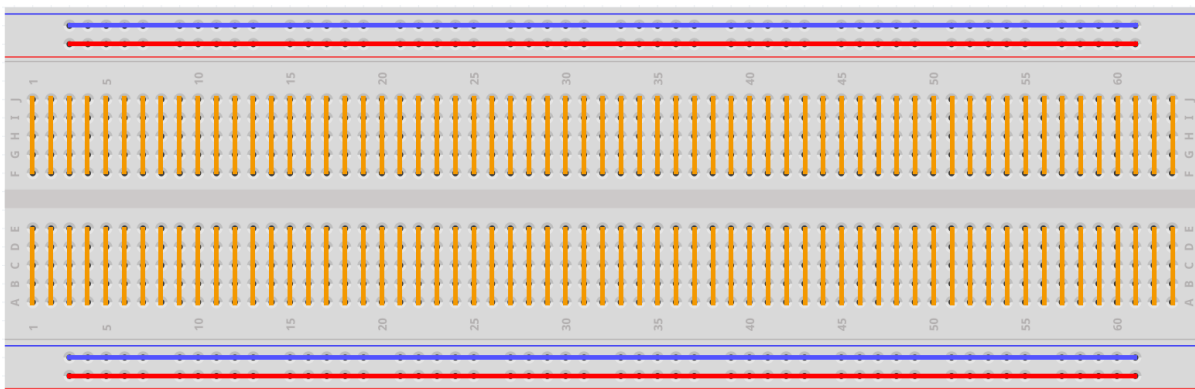
## 1.3 Plaque d'essai



Une plaque d'essai est une base de construction pour le prototypage électronique. À l'origine, le terme désignait littéralement une planche à pain, un morceau de bois poli utilisé pour trancher le pain.[1] Dans les années 1970, la plaque d'essai sans soudure (aussi connue sous le nom de plugboard ou de tableau de connexion à bornes) est devenue courante et aujourd'hui, le terme « plaque d'essai » est couramment utilisé pour désigner ces dispositifs.

Elle est utilisée pour construire et tester rapidement des circuits avant de finaliser une conception de circuit. Elle possède de nombreux trous dans lesquels les composants mentionnés ci-dessus peuvent être insérés, comme des circuits intégrés, des résistances ainsi que des fils cavaliers. La plaque d'essai vous permet de brancher et de retirer facilement les composants.

L'image montre la structure interne d'une plaque d'essai. Bien que ces trous sur la plaque d'essai semblent être indépendants les uns des autres, ils sont en réalité connectés entre eux par des bandes métalliques à l'intérieur.



Si vous voulez en savoir plus sur la plaque d'essai, consultez : [Comment utiliser une plaque d'essai - Science Buddies](#)

## 1.4 Résistance



Une résistance est un élément électronique qui peut limiter le courant d'une branche. Une résistance fixe est un type de résistance dont la valeur ne peut pas être changée, contrairement à un potentiomètre ou une résistance variable qui peuvent être ajustés.

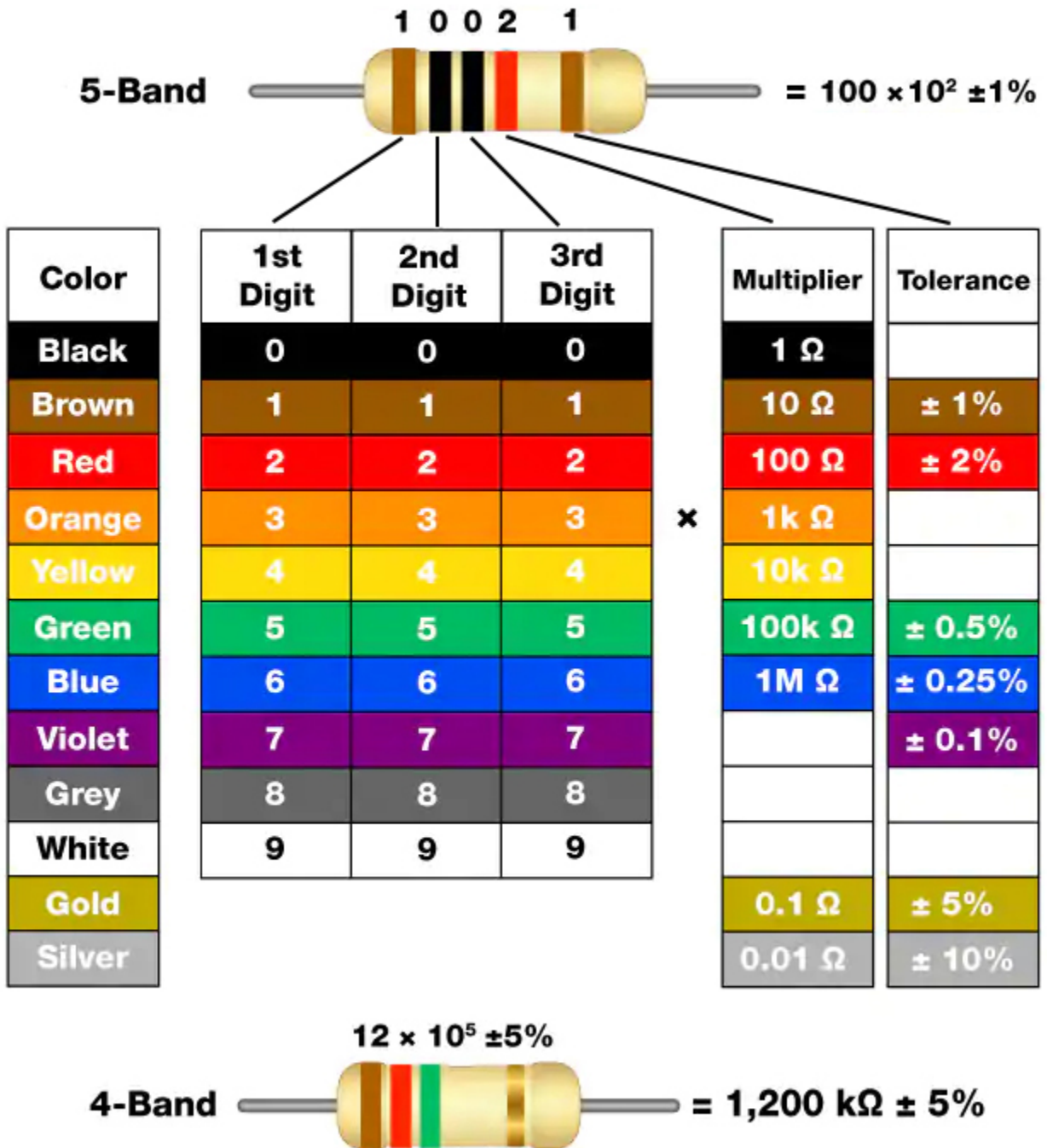
Deux symboles de circuit généralement utilisés pour une résistance. Normalement, la valeur de la résistance y est marquée. Donc, si vous voyez ces symboles dans un circuit, cela représente une résistance.



est l'unité de résistance et les unités plus grandes incluent K, M, etc. Leur relation peut être présentée comme suit :  $1\text{ M} = 1000\text{ K}$ ,  $1\text{ K} = 1000$ . Normalement, la valeur de la résistance est marquée dessus.

Lors de l'utilisation d'une résistance, nous devons d'abord connaître sa valeur. Voici deux méthodes : vous pouvez observer les bandes sur la résistance, ou utiliser un multimètre pour mesurer la résistance. La première méthode est recommandée car elle est plus pratique et plus rapide.





Comme le montre la carte, chaque couleur représente un nombre.

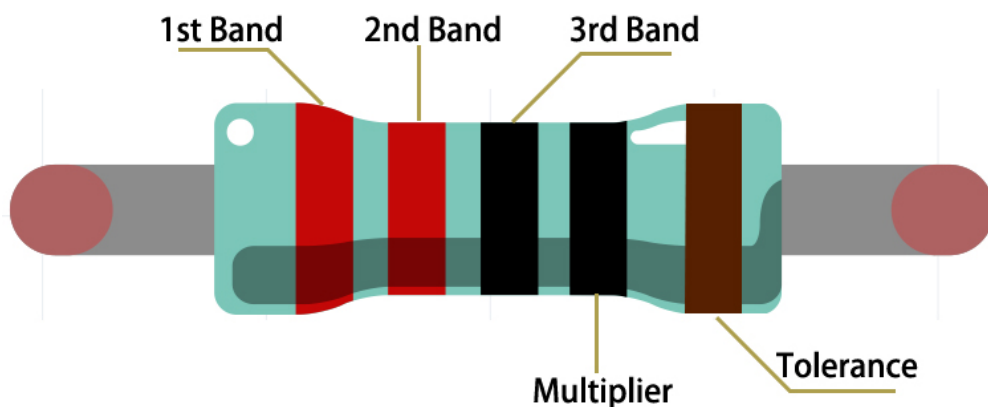
Noir	Marron	Rouge	Orange	Jaune	Vert	Bleu	Violet	Gris	Blanc	Or	Argent
0	1	2	3	4	5	6	7	8	9	0.1	0.01

Les résistances à 4 et 5 bandes sont fréquemment utilisées, sur lesquelles il y a 4 et 5 bandes chromatiques.

Normalement, lorsque vous obtenez une résistance, il peut être difficile de décider par quelle extrémité commencer pour lire la couleur. L'astuce est que l'écart entre la 4e et la 5e bande sera comparativement plus grand.

Par conséquent, vous pouvez observer l'écart entre les deux bandes chromatiques à une extrémité de la résistance ; si c'est plus grand que tout autre écart de bande, alors vous pouvez lire à partir du côté opposé.

Voyons comment lire la valeur de la résistance d'une résistance à 5 bandes comme indiqué ci-dessous.



Pour cette résistance, la valeur doit être lue de gauche à droite. La valeur doit être dans ce format : 1re Bande 2e Bande 3e Bande  $\times 10^{\text{Multiplieur}}$  () et l'erreur admissible est de  $\pm \text{Tolérance}\%$ . Ainsi, la valeur de la résistance de cette résistance est 2 (rouge) 2 (rouge) 0 (noir)  $\times 10^0$  (noir) = 220 , et l'erreur admissible est de  $\pm 1\%$  (marron).

TABLEAU 2 – Bandes de couleur de résistance courantes

Résistance	Bandes de couleur
10	marron noir noir argent marron
100	marron noir noir noir marron
220	rouge rouge noir noir marron
330	orange orange noir noir marron
1k	marron noir noir marron marron
2k	rouge noir noir marron marron
5.1k	vert marron noir marron marron
10k	marron noir noir rouge marron
100k	marron noir noir orange marron
1M	marron noir noir vert marron

Vous pouvez en apprendre plus sur les résistances sur Wiki : [Résistance - Wikipédia](#).



## 1.5 Condensateur





Le condensateur, se rapporte à la quantité de charge stockée sous une différence de potentiel donnée, notée  $C$ , et son unité internationale est le farad (F). Généralement, les charges électriques se déplacent sous l'effet d'une force dans un champ électrique. Lorsqu'il y a un milieu entre les conducteurs, le mouvement des charges électriques est entravé et les charges s'accumulent sur les conducteurs, entraînant une accumulation de charges électriques.

La quantité de charges électriques stockées est appelée capacité. Les condensateurs étant l'un des composants électroniques les plus utilisés dans les équipements électroniques, ils sont largement employés dans l'isolation en courant continu, le couplage, le contournement, le filtrage, les boucles de réglage, la conversion d'énergie et les circuits de contrôle. Les condensateurs sont divisés en condensateurs électrolytiques, condensateurs solides, etc.

Selon les caractéristiques des matériaux, les condensateurs peuvent être divisés en : condensateurs électrolytiques en aluminium, condensateurs à film, condensateurs au tantale, condensateurs céramiques, supercondensateurs, etc.

Dans ce kit, des condensateurs céramiques et électrolytiques sont utilisés.

- [Condensateur Céramique - Wikipedia](#)
- [Condensateur Électrolytique - Wikipedia](#)

Il y a des étiquettes 103 ou 104 sur les condensateurs céramiques, qui représentent la valeur de capacité, 103 =  $10 \times 10^3 \text{ pF}$ , 104 =  $10 \times 10^4 \text{ pF}$

#### Conversion d'Unité

$$1\text{F}=10^3\text{mF}=10^6\mu\text{F}=10^9\text{nF}=10^{12}\text{pF}$$

#### Exemple

- [2.6 Sonnette](#) (Projet Scratch)
- [2.16 JEU - Manger la Pomme](#) (Projet Scratch)
- [2.19 JEU - Pêche](#) (Projet Scratch)

## 1.6 Fils de Cavalier

Les fils qui connectent deux bornes sont appelés fils de cavalier. Il existe divers types de fils de cavalier. Ici, nous nous concentrons sur ceux utilisés dans les plaques d'essai. Entre autres, ils servent à transférer des signaux électriques de n'importe quel endroit sur la plaque d'essai vers les broches d'entrée/sortie d'un microcontrôleur.

Les fils de cavalier sont fixés en insérant leurs « connecteurs d'extrémité » dans les emplacements prévus dans la plaque d'essai, sous la surface de laquelle il y a quelques ensembles de plaques parallèles qui connectent les emplacements en groupes de rangées ou de colonnes selon la zone. Les « connecteurs d'extrémité » sont insérés dans la plaque d'essai, sans soudure, dans les emplacements spécifiques qui doivent être connectés dans le prototype spécifique.

Il existe trois types de fil de cavalier : Femelle-Femelle, Mâle-Mâle, et Mâle-Femelle. On l'appelle Mâle-Femelle car il a une pointe saillante à une extrémité ainsi qu'une extrémité femelle enfoncée. Mâle-Mâle signifie que les deux côtés sont mâles et Femelle-Femelle signifie que les deux extrémités sont femelles.

Male-to-Female



Male-to-Male



Female-to-Female



Plus d'un type de ces fils peut être utilisé dans un projet. La couleur des fils de cavalier est différente mais cela ne signifie pas que leur fonction est différente en conséquence ; c'est juste conçu ainsi pour mieux identifier la connexion entre chaque circuit.

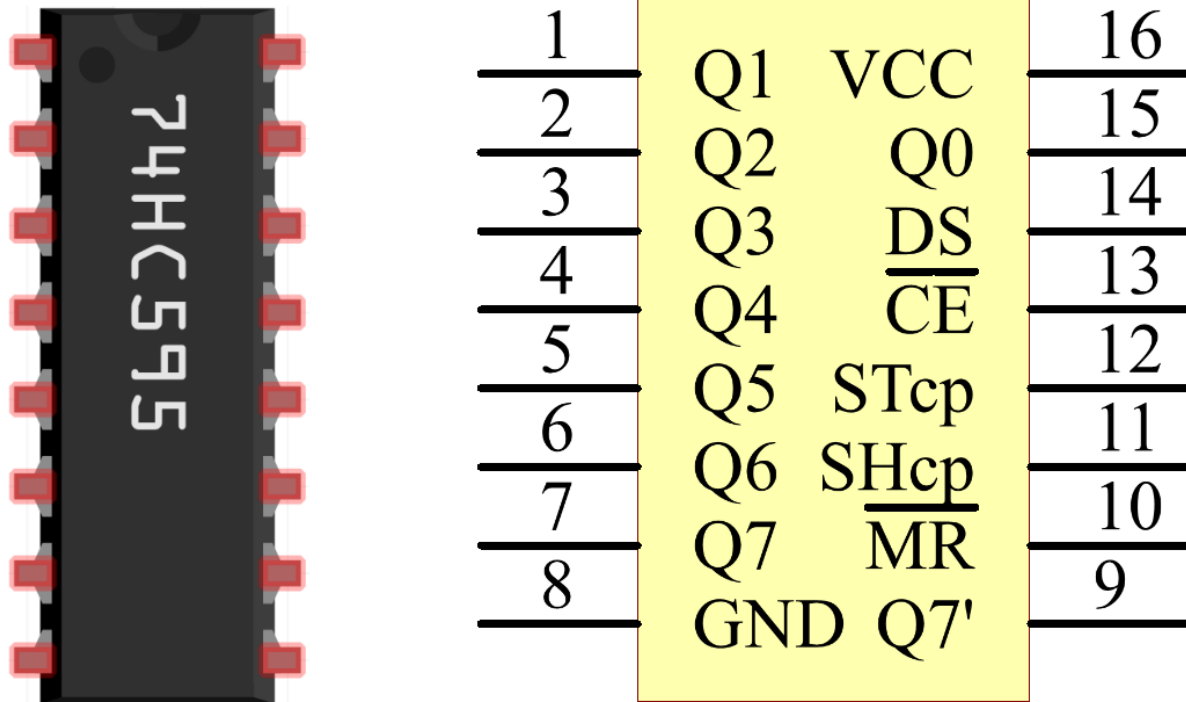
### Puce

## 1.7 74HC595



Le 74HC595 est composé d'un registre à décalage de 8 bits et d'un registre de stockage avec des sorties parallèles à trois états. Il convertit l'entrée série en sortie parallèle, permettant ainsi d'économiser les ports IO d'un MCU. Lorsque MR (broche 10) est à un niveau haut et OE (broche 13) à un niveau bas, les données sont entrées sur le front montant de SHcp et passent au registre de mémoire sur le front montant de SHcp. Si les deux horloges sont connectées ensemble, le registre à décalage est toujours un pulse en avance sur le registre de mémoire. Il y a une broche d'entrée de décalage série (Ds), une broche de sortie série (Q) et un bouton de réinitialisation asynchrone (niveau bas) dans le registre de mémoire. Le registre de mémoire sort un bus avec un 8-bit parallèle et en trois états. Lorsque OE est activé (niveau bas), les données dans le registre de mémoire sont sorties vers le bus.

— [Fiche technique du 74HC595](#)



Broches du 74HC595 et leurs fonctions :

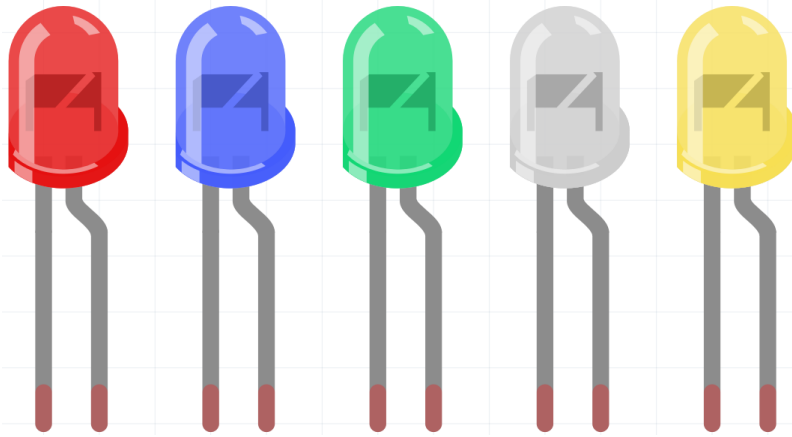
- **Q0-Q7** : broches de sortie de données parallèles 8 bits, capables de contrôler directement 8 LEDs ou 8 broches d’affichage à 7 segments.
- **Q7'** : Broche de sortie série, connectée à DS d’un autre 74HC595 pour connecter plusieurs 74HC595 en série
- **MR** : Broche de réinitialisation, active à un niveau bas ;
- **SHcp** : Entrée de séquence temporelle du registre à décalage. Sur le front montant, les données dans le registre à décalage se déplacent successivement d’un bit, c’est-à-dire que les données en Q1 passent à Q2, et ainsi de suite. Alors que sur le front descendant, les données dans le registre à décalage restent inchangées.
- **STcp** : Entrée de séquence temporelle du registre de stockage. Sur le front montant, les données dans le registre à décalage passent dans le registre de mémoire.
- **CE** : Broche d’activation de sortie, active à un niveau bas.
- **DS** : Broche d’entrée de données série
- **VCC** : Tension d’alimentation positive.
- **GND** : Masse.

#### Exemple

- [5.9 ShiftOut\(LED\)](#) (Projet de base)
- [5.10 ShiftOut\(Affichage à segments\)](#) (Projet de base)
- [7. Portail à Limite de Courant](#) (Projet IoT)

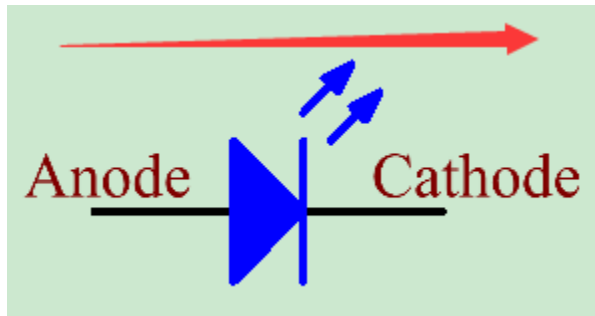
#### Affichage

## 1.8 LED



La diode électroluminescente (LED) est un type de composant capable de transformer l'énergie électrique en énergie lumineuse via des jonctions PN. Selon la longueur d'onde, elle peut être catégorisée en diode laser, diode électroluminescente infrarouge et diode électroluminescente visible, communément appelée LED.

La diode a une conductivité unidirectionnelle, donc le courant s'écoulera comme l'indique la flèche dans le symbole de circuit de la figure. Vous ne pouvez fournir que l'anode avec une puissance positive et la cathode avec une négative. Ainsi, la LED s'allumera.



Une LED a deux broches. La plus longue est l'anode et la plus courte, la cathode. Faites attention à ne pas les connecter à l'envers. Il y a une chute de tension directe fixe dans la LED, donc elle ne peut pas être connectée directement au circuit car la tension d'alimentation peut dépasser cette chute et causer la combustion de la LED. La tension directe de la LED rouge, jaune et verte est de 1,8 V et celle de la blanche est de 2,6 V. La plupart des LED peuvent supporter un courant maximal de 20 mA, donc nous devons connecter une résistance limitant le courant en série.

La formule de la valeur de résistance est la suivante :

$$R = (V_{\text{supply}} - V_D) / I$$

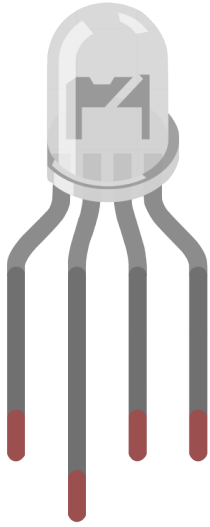
**R** représente la valeur de résistance de la résistance limitant le courant, **V<sub>supply</sub>** pour la tension d'alimentation, **V<sub>D</sub>** pour la chute de tension et **I** pour le courant de travail de la LED.

Voici une introduction détaillée pour la LED : [LED - Wikipédia](#).

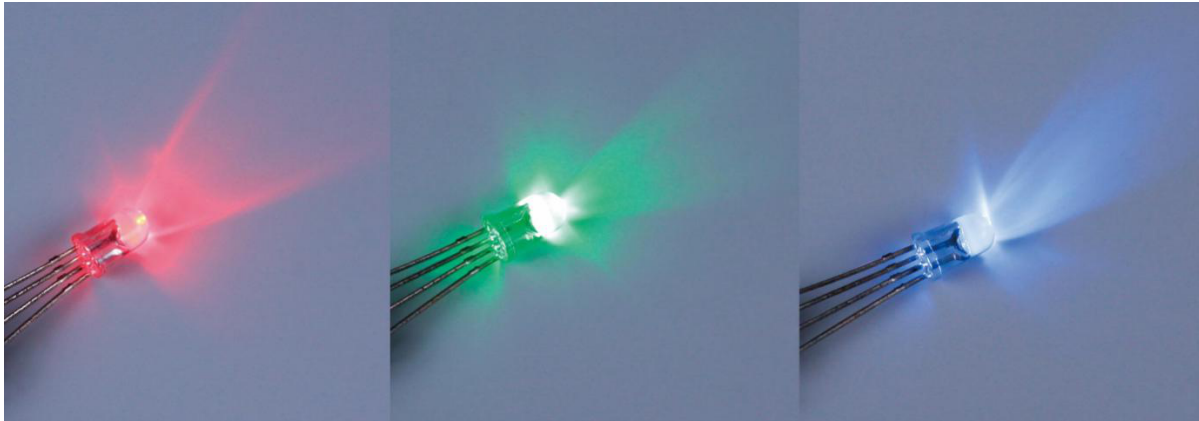
### Exemple

- [1.1 Bonjour, LED!](#) (Projet de base)
- [2.1 Estompage](#) (Projet de base)
- [2. Obtenir des Données depuis Blynk](#) (Projet IoT)
- [2.2 LED Respiration](#) (Projet Scratch)
- [2.1 Lampe de Table](#) (Projet Scratch)

## 1.9 LED RVB

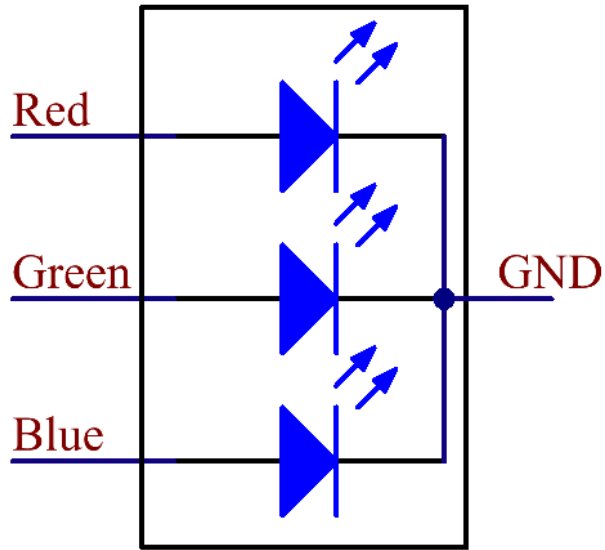


Les LED RVB émettent de la lumière de différentes couleurs. Une LED RVB intègre trois LED de couleur rouge, verte et bleue dans une coque en plastique transparent ou semi-transparent. Elle peut afficher diverses couleurs en changeant la tension d'entrée des trois broches et en les superposant, ce qui, selon les statistiques, peut créer 16 777 216 couleurs différentes.

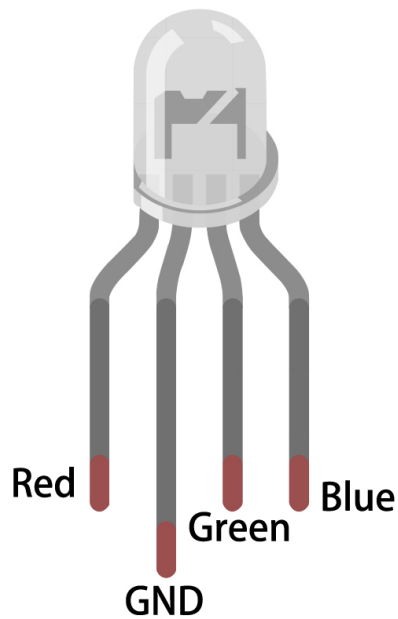


Les LED RVB peuvent être catégorisées en anode commune et cathode commune. Dans ce kit, c'est la dernière qui est utilisée. La **cathode commune**, ou CC, signifie connecter les cathodes des trois LED. Après l'avoir connectée à GND et branché les trois broches, la LED affichera la couleur correspondante.

Son symbole de circuit est montré comme la figure.



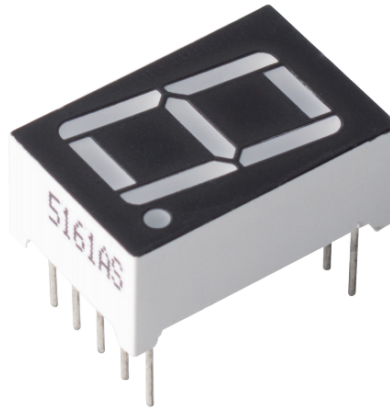
Une LED RVB a 4 broches : la plus longue est GND ; les autres sont Rouge, Vert et Bleu. Touchez sa coque en plastique et vous trouverez une entaille. La broche la plus proche de l'entaille est la première broche, marquée comme Rouge, puis GND, Vert et Bleu à leur tour.



#### Exemple

- [2.2 Lumière Colorée](#) (Projet de base)
- [5.2 Seuil](#) (Projet de base)
- [2.3 Balles Colorées](#) (Projet Scratch)

## 1.10 Affichage 7 segments

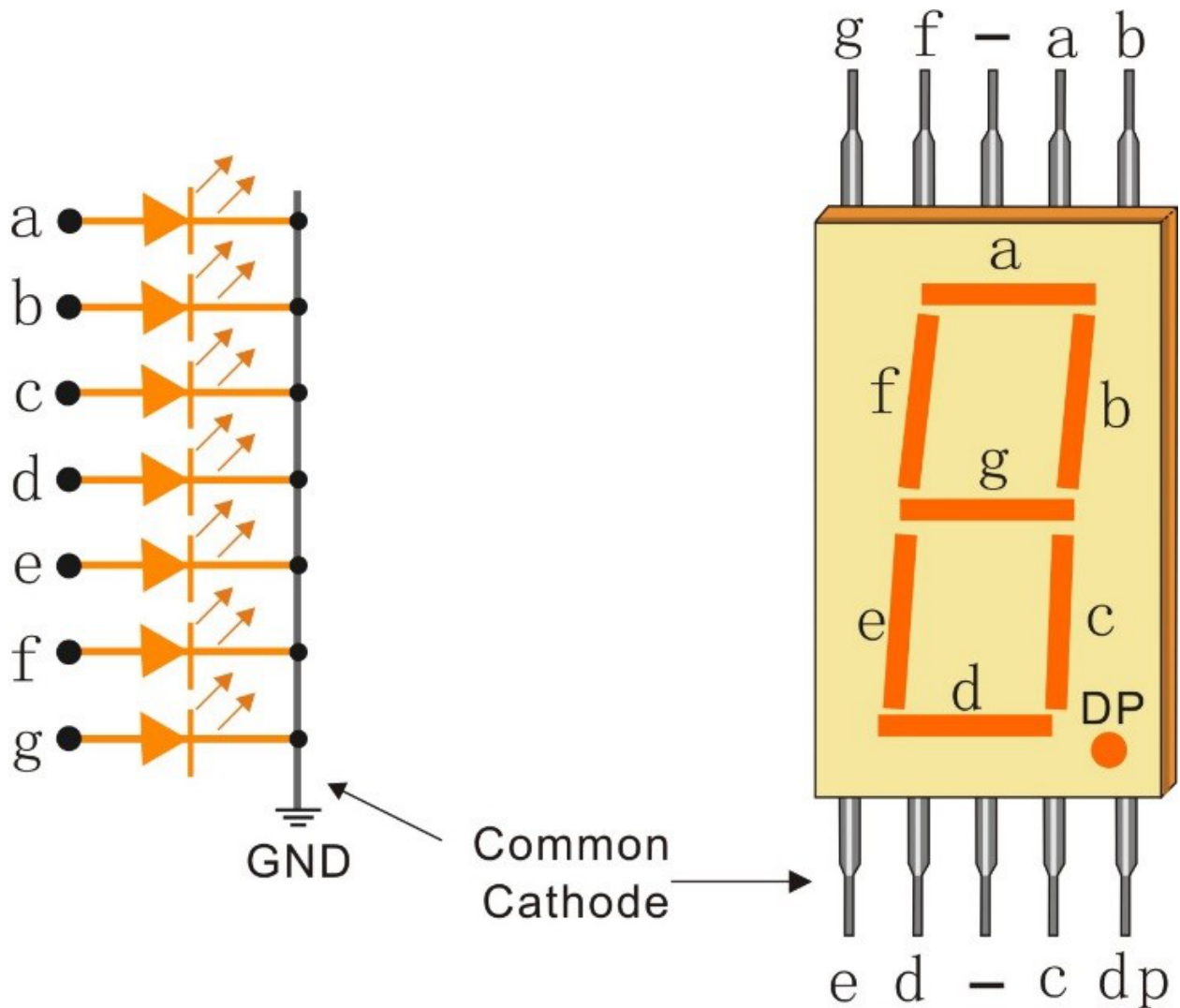


Un affichage à 7 segments est un composant en forme de 8 qui intègre 7 LED. Chaque LED est appelée un segment - lorsqu'elle est alimentée, un segment forme une partie d'un chiffre à afficher.

Il existe deux types de connexion de broches : Cathode Commune (CC) et Anode Commune (CA). Comme leur nom l'indique, un affichage CC connecte toutes les cathodes des 7 LED, tandis qu'un affichage CA connecte toutes les anodes des 7 segments.

Dans ce kit, nous utilisons l'affichage à 7 segments à Cathode Commune, voici le symbole électronique correspondant.





Chacune des LED de l’affichage est dotée d’un segment positionnel avec l’une de ses broches de connexion sortant du boîtier en plastique rectangulaire. Ces broches de LED sont étiquetées de « a » à « g », représentant chaque LED individuelle. Les autres broches de LED sont connectées ensemble, formant une broche commune. Ainsi, en polarisant directement les broches appropriées des segments de LED dans un ordre particulier, certains segments s’illuminent tandis que d’autres restent sombres, affichant ainsi le caractère correspondant sur l’affichage.

#### Codes d’affichage

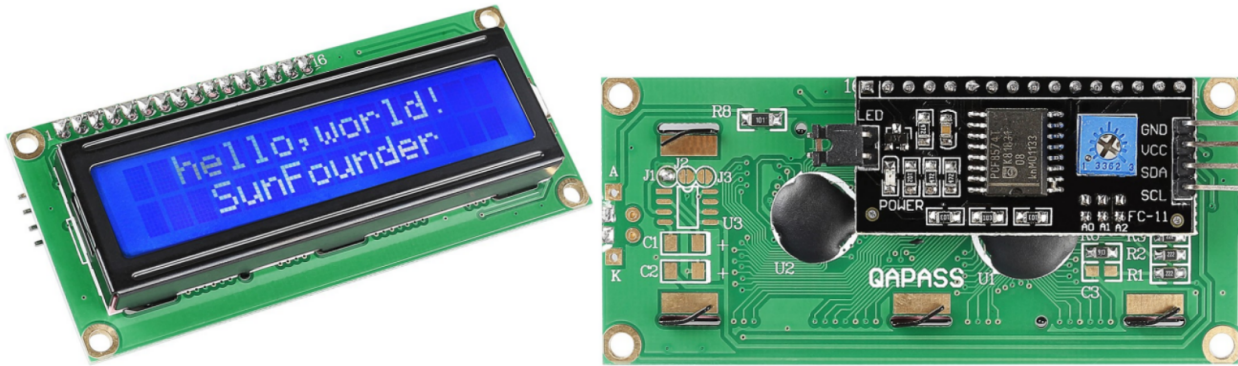
Pour vous aider à comprendre comment les affichages à 7 segments (Cathode Commune) affichent les nombres, nous avons dressé le tableau suivant. Les nombres sont les chiffres de 0 à F affichés sur l’affichage à 7 segments; (DP) GFEDCBA se réfère à l’ensemble de LED correspondant réglé sur 0 ou 1, par exemple, 00111111 signifie que DP et G sont réglés sur 0, tandis que les autres sont réglés sur 1. Par conséquent, le nombre 0 est affiché sur l’affichage à 7 segments, tandis que le Code HEX correspond au nombre hexadécimal.

Numbers	Common Cathode		Numbers	Common Cathode	
	(DP)GFEDCBA	Hex Code		(DP)GFEDCBA A	Hex Code
0	00111111	0x3f	A	01110111	0x77
1	00000110	0x06	B	01111100	0x7c
2	01011011	0x5b	C	00111001	0x39
3	01001111	0x4f	D	01011110	0x5e
4	01100110	0x66	E	01111001	0x79
5	01101101	0x6d	F	01110001	0x71
6	01111101	0x7d			
7	00000111	0x07			
8	01111111	0x7f			
9	01101111	0x6f			

**Exemple**

- 5.10 ShiftOut(Affichage à segments) (Projet de base)
- 7. Portail à Limite de Courant (Projet IoT)

## 1.11 I2C LCD1602



- **GND** : Masse
- **VCC** : Alimentation en tension, 5V.
- **SDA** : Ligne de données série. Connecter à VCC via une résistance de tirage.
- **SCL** : Ligne d'horloge série. Connecter à VCC via une résistance de tirage.

Comme nous le savons tous, bien que les écrans LCD et certains autres affichages enrichissent grandement l'interaction homme-machine, ils partagent une faiblesse commune. Lorsqu'ils sont connectés à un contrôleur, de multiples IOs sont occupés sur le contrôleur qui ne dispose pas de tant de ports externes. Cela limite également d'autres fonctions du contrôleur.

Par conséquent, un LCD1602 avec un module I2C a été développé pour résoudre ce problème. Le module I2C intègre une puce PCF8574 I2C qui convertit les données série I2C en données parallèles pour l'affichage LCD.

- [Fiche technique PCF8574](#)

### Adresse I2C

L'adresse par défaut est généralement 0x27, dans quelques cas, elle peut être 0x3F.

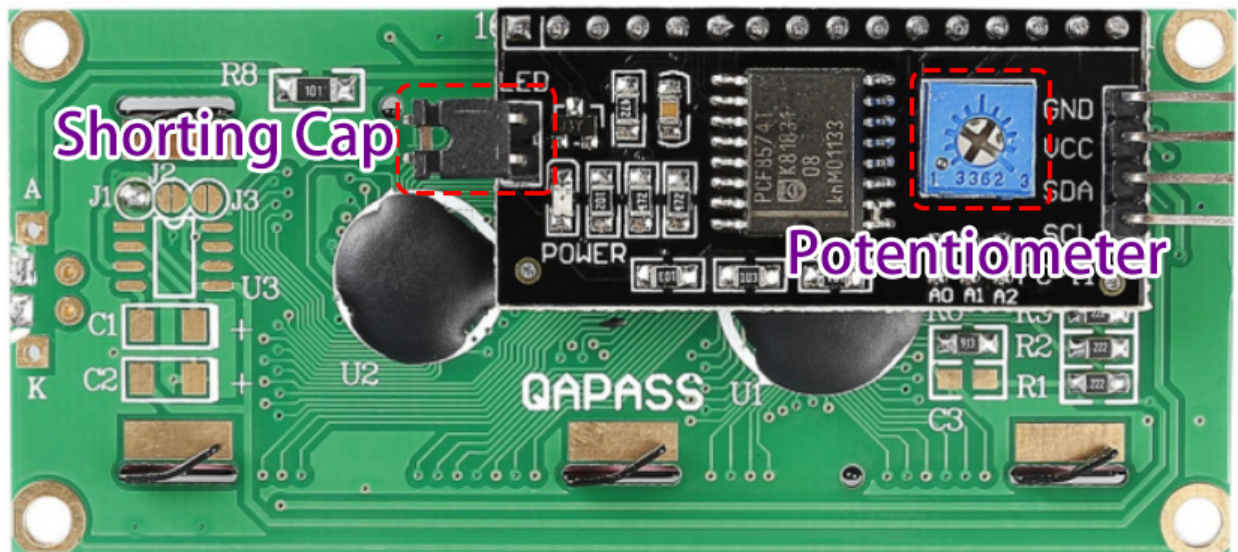
Prenant l'adresse par défaut de 0x27 comme exemple, l'adresse de l'appareil peut être modifiée en court-circuitant les pads A0/A1/A2; dans l'état par défaut, A0/A1/A2 est 1, et si le pad est court-circuité, A0/A1/A2 est 0.

## Slave Address

0	0	1	0	0	A2	A1	A0	
0	0	1	0	0	1	1	1	0x27
0	0	1	0	0	1	1	0	0x26
0	0	1	0	0	1	0	1	0x25
0	0	1	0	0	0	1	1	0x23
.....								
0	0	1	0	0	0	0	0	0x20

### Rétroéclairage/Contraste

Le rétroéclairage peut être activé par un capuchon de cavalier, retirez le capuchon pour désactiver le rétroéclairage. Le potentiomètre bleu à l'arrière sert à ajuster le contraste (le rapport de luminosité entre le blanc le plus brillant et le noir le plus sombre).



- **Capuchon de Court-Circuit** : Le rétroéclairage peut être activé par ce capuchon, retirez ce capuchon pour désactiver le rétroéclairage.
- **Potentiomètre** : Il sert à ajuster le contraste (la clarté du texte affiché), qui est augmenté dans le sens des aiguilles

d'une montre et diminué dans le sens inverse.

### Exemple

- 5.11.1 *Affichage à Cristaux Liquides* (Projet de base)
- 5.12 *Lecture Série* (Projet de base)

### Son

## 1.12 Buzzer



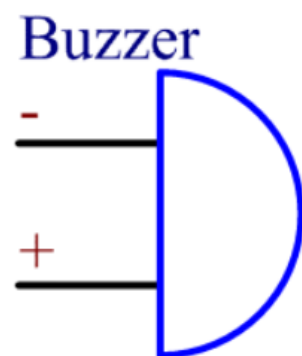
En tant que type de buzzer électronique à structure intégrée, les buzzers, alimentés par une tension continue (DC), sont largement utilisés dans les ordinateurs, imprimantes, photocopieuses, alarmes, jouets électroniques, dispositifs électroniques automobiles, téléphones, minuteries et autres produits électroniques ou dispositifs sonores.

Les buzzers peuvent être catégorisés en actifs et passifs (voir l'image ci-dessous). Retournez le buzzer pour que ses broches soient orientées vers le haut : le buzzer avec un circuit imprimé vert est un buzzer passif, tandis que celui entouré d'un ruban noir est un buzzer actif.

La différence entre un buzzer actif et un buzzer passif :

Un buzzer actif possède une source d'oscillation intégrée, donc il émet un son lorsqu'il est électrifié. Cependant, un buzzer passif n'a pas une telle source, donc il ne bipera pas avec des signaux DC ; à la place, vous devez utiliser des ondes carrées dont la fréquence est entre 2K et 5K pour le piloter. Le buzzer actif est souvent plus cher que le passif en raison de multiples circuits d'oscillation intégrés.

Voici le symbole électrique d'un buzzer. Il a deux broches avec des pôles positif et négatif. Un + sur la surface représente l'anode et l'autre est la cathode.



Vous pouvez vérifier les broches du buzzer, la plus longue est l'anode et la plus courte est la cathode. Veuillez ne pas les mélanger lors de la connexion, sinon le buzzer ne produira pas de son.

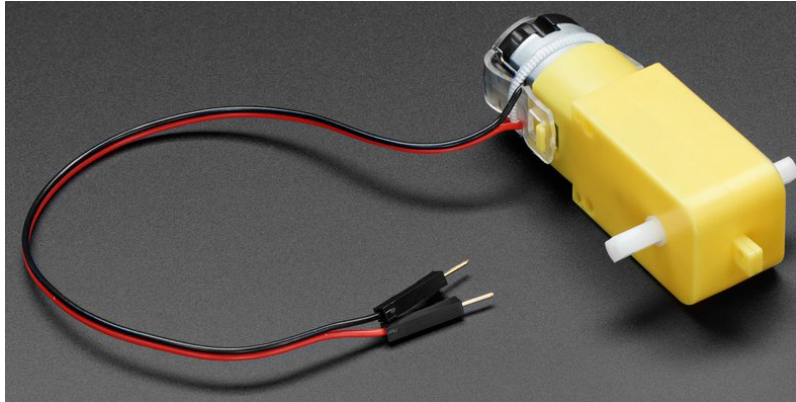
Buzzer - Wikipédia

### Exemple

- *1.2 Bip* (Projet de base)
- *5.7 Tone() ou noTone()* (Projet de base)
- *4. Lecteur de Musique Cloud* (Projet IoT)

### Pilote

## 1.13 Moteur TT



Il s'agit d'un moteur à engrenages CC TT avec un rapport de réduction de 1 :48. Il est équipé de 2 fils de 200mm avec des connecteurs mâles de 0,1 » qui s'adaptent à une plaque de montage. Parfait pour être branché sur une plaque d'essai ou un bloc de jonction.

Vous pouvez alimenter ces moteurs avec une tension de 3 ~ 6VDC, mais bien sûr, ils tourneront un peu plus vite à des tensions plus élevées.

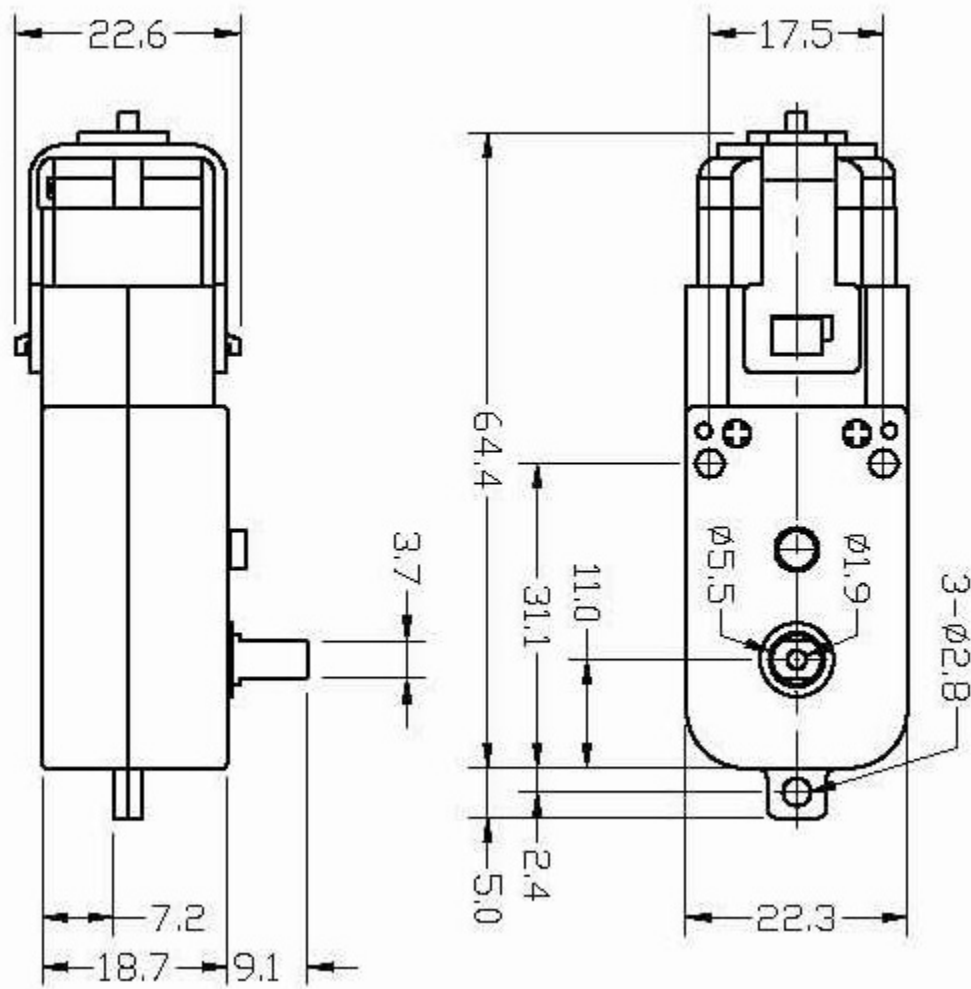
Notez que ce sont des moteurs très basiques sans encodeur intégré, contrôle de vitesse ou retour de position. La tension entre et la rotation sort. Il y aura des variations d'un moteur à l'autre, donc si vous avez besoin d'un mouvement précis, vous aurez besoin d'un système de retour séparé.

### Détails Techniques

- Tension Nominale : 3~6V
- Courant à Vide Continu : 150mA +/- 10%
- Vitesse de Fonctionnement Minimale (3V) : 90+/- 10% RPM
- Vitesse de Fonctionnement Minimale (6V) : 200+/- 10% RPM
- Couple de Calage (3V) : 0.4kg.cm
- Couple de Calage (6V) : 0.8kg.cm
- Rapport de Réduction : 1 :48
- Dimensions du Corps : 70 x 22 x 18mm
- Longueur des Fils : 200mm & 28 AWG
- Poids : 30.6g

### Dessin Dimensionnel





### Example

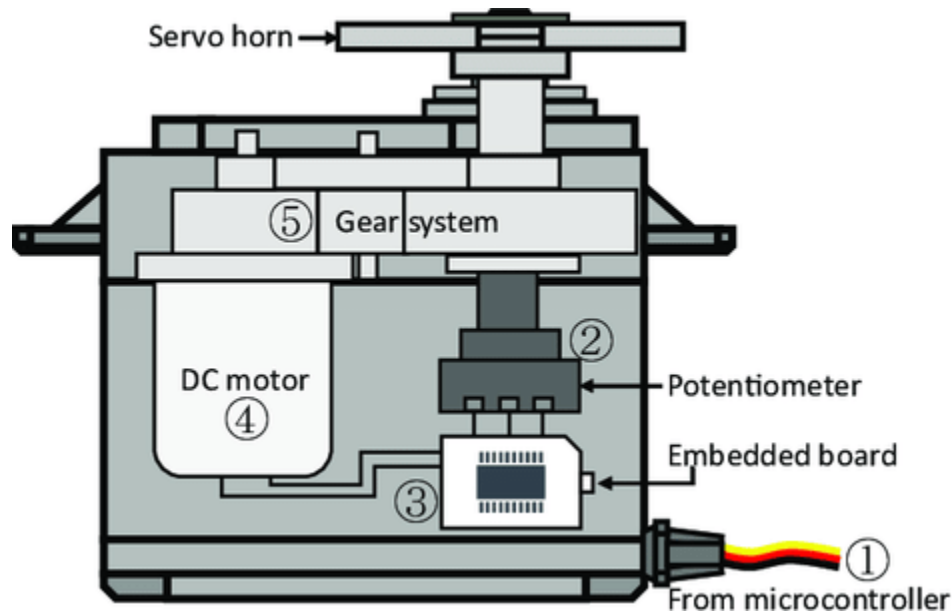
- 1.3 Moteur (Projet de base)
- 1. Mouvement (Projet Voiture)
- 3. Accélération (Projet Voiture)
- 8. Voiture IoT (Projet IoT)
- 3.1 Tester la Voiture (Projet Scratch)

## 1.14 Servomoteur



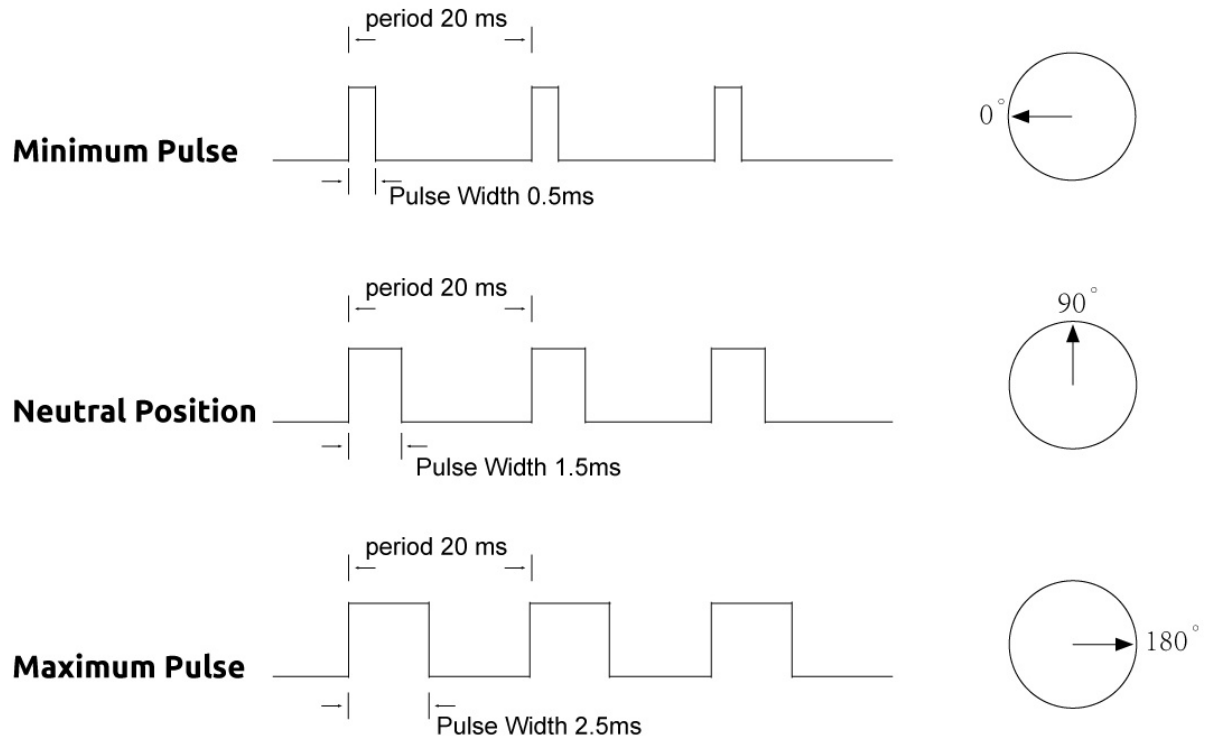
Un servomoteur est généralement composé des éléments suivants : boîtier, axe, système d'engrenages, potentiomètre, moteur à courant continu (DC) et carte intégrée.

Son fonctionnement est le suivant : le microcontrôleur envoie des signaux PWM au servomoteur, puis la carte intégrée dans le servo reçoit les signaux via la broche de signal et contrôle le moteur interne pour le faire tourner. En conséquence, le moteur entraîne le système d'engrenages, puis motive l'axe après décélération. L'axe et le potentiomètre du servo sont connectés ensemble. Lorsque l'axe tourne, il entraîne le potentiomètre, de sorte que le potentiomètre émet un signal de tension à la carte intégrée. La carte détermine ensuite la direction et la vitesse de rotation en fonction de la position actuelle, de sorte qu'elle puisse s'arrêter exactement à la position définie et y rester.



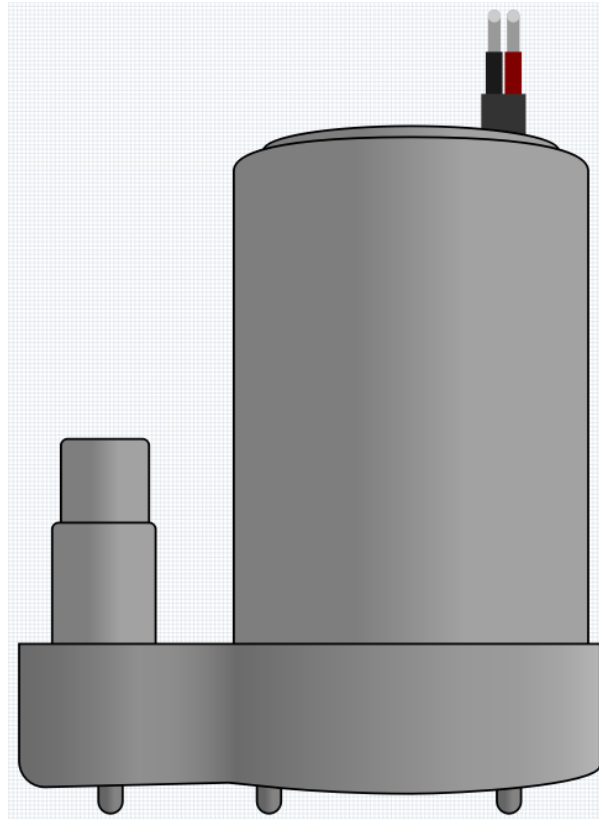
L'angle est déterminé par la durée d'une impulsion appliquée au fil de commande. Cela s'appelle la modulation de largeur d'impulsion (PWM). Le servomoteur s'attend à voir une impulsion toutes les 20 ms. La longueur de l'impulsion déterminera la rotation du moteur. Par exemple, une impulsion de 1,5 ms fera tourner le moteur à la position de 90 degrés (position neutre). Lorsqu'une impulsion inférieure à 1,5 ms est envoyée à un servo, le servo tourne vers une position et maintient son axe de sortie à un certain nombre de degrés dans le sens antihoraire à partir du point neutre. Lorsque l'impulsion est plus large que 1,5 ms, l'inverse se produit. La largeur minimale et maximale de l'impulsion qui commande le servo pour tourner vers une position valide dépendent de chaque servo. Généralement, l'impulsion minimale sera d'environ 0,5 ms de large et l'impulsion maximale sera de 2,5 ms de large.



**Exemple**

- 5.5 Utiliser une Bibliothèque Interne (Projet de base)
- 7. Portail à Limite de Courant (Projet IoT)
- 2.10 Pendule (Projet Scratch)

## 1.15 Pompe Centrifuge



La pompe centrifuge transforme l'énergie cinétique rotationnelle en énergie hydrodynamique pour transporter le fluide. L'énergie de rotation provient du moteur électrique. Le fluide entre dans la roue de la pompe le long ou près de l'arbre rotatif, est accéléré par la roue, s'écoule radialement vers l'extérieur dans la chambre de diffusion ou de volute, puis s'écoule de là.

Les utilisations courantes des pompes centrifuges comprennent le pompage de l'eau, des eaux usées, dans l'agriculture, ainsi que dans les industries pétrolière et pétrochimique.

— [Pompe Centrifuge - Wikipédia](#)

### Caractéristiques

- **Plage de Tension** : DC 3 ~ 4,5V
- **Courant de Fonctionnement** : 120 ~ 180mA
- **Puissance** : 0,36 ~ 0,91W
- **Hauteur Maximale de Refoulement** : 0,35 ~ 0,55M
- **Débit Maximum** : 80 ~ 100 L/H
- **Durée de Vie Continue** : 100 heures
- **Indice de Protection de l'Eau** : IP68
- **Mode de Conduite** : DC, Entraînement Magnétique
- **Matériau** : Plastique Technique
- **Diamètre Extérieur de Sortie** : 7,8 mm
- **Diamètre Intérieur de Sortie** : 6,5 mm
- Il s'agit d'une pompe submersible et doit être utilisée comme telle. Elle a tendance à surchauffer; il y a un risque de surchauffe si elle est utilisée hors de l'eau.

### Exemple

— [1.4 Pompage](#) (Projet de base)

## 1.16 Module de Contrôle Moteur L9110

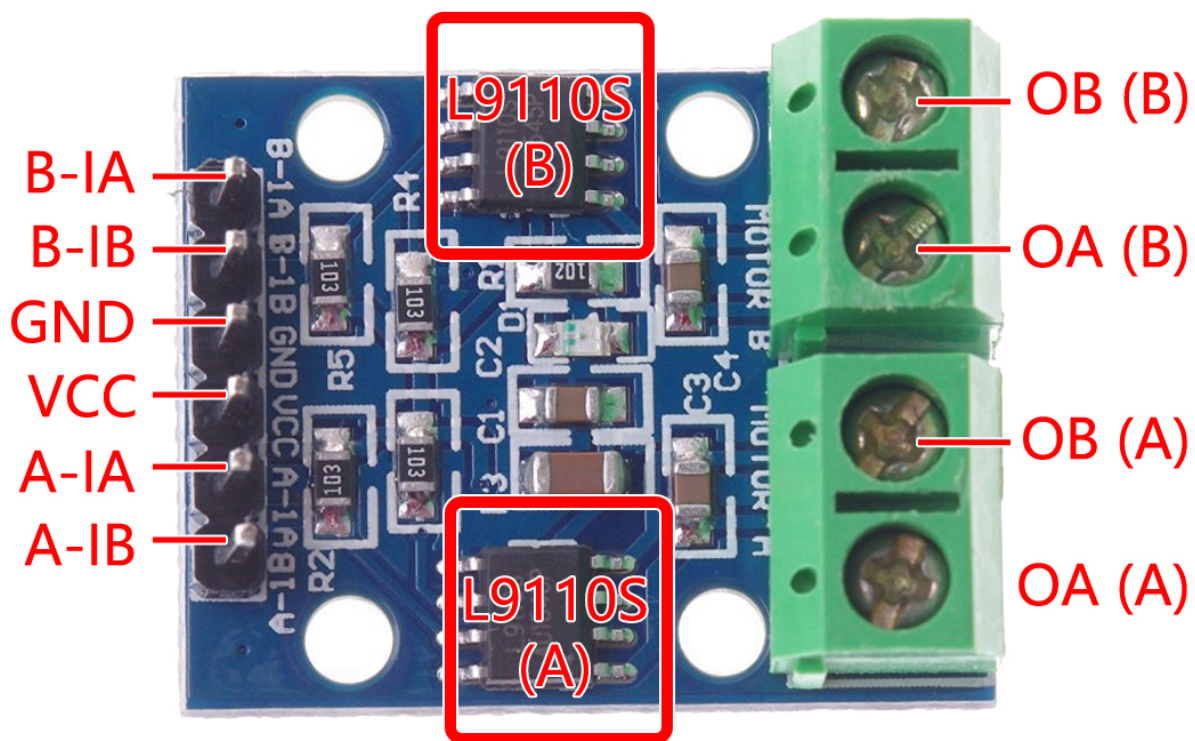
Le module de contrôle moteur L9110 est conçu pour piloter simultanément deux moteurs. Il intègre une paire de puces de pilotage indépendantes L9110S, chaque canal offrant une sortie de courant stable allant jusqu'à 800mA.

Couvrant une plage de tension de 2,5V à 12V, le module s'accorde aisément avec des microcontrôleurs de 3,3V et 5V.

Offrant une solution simplifiée, le module de contrôle moteur L9110 facilite la gestion des moteurs dans un large éventail d'applications. Grâce à son architecture à deux canaux, il permet l'orchestration indépendante de deux moteurs, idéal pour des projets nécessitant des opérations de moteurs doubles.

Avec sa sortie de courant continu puissante, ce module alimente avec assurance des moteurs de petite à moyenne taille, ouvrant la voie à diverses entreprises robotiques, d'automatisation et centrées sur le moteur. Sa large plage de tension ajoute une adaptabilité, s'alignant sur diverses configurations d'alimentation électrique.

Conçu dans un souci de facilité d'utilisation, le module offre des bornes d'entrée et de sortie intuitives, simplifiant les connexions aux microcontrôleurs ou appareils de contrôle similaires. De plus, il ne lésine pas sur la sécurité – des protections intégrées contre les surintensités et les surchauffes renforcent la fiabilité et la sécurité des opérations des moteurs.



- **B-1A & B-1B(B-2A)** : Broches d'entrée pour contrôler la direction de rotation du moteur B.
- **A-1A & A-1B** : Broches d'entrée pour contrôler la direction de rotation du moteur A.
- **OA & OB(A)** : Broches de sortie du moteur A.
- **OA & OB(B)** : Broches de sortie du moteur B.
- **VCC** : Broche d'entrée d'alimentation (2,5V-12V).
- **GND** : Broche de masse.

### Caractéristiques

- 2 puces de contrôle moteur L9110S intégrées
- Contrôle moteur à double canal.

- Contrôle indépendant de la direction de rotation des moteurs.
- Sortie de courant élevée (800mA par canal).
- Large plage de tension (2,5V-12V).
- Conception compacte.
- Bornes d'entrée et de sortie pratiques.
- Caractéristiques de protection intégrées.
- Applications polyvalentes.
- Taille du PCB : 29.2mm x 23mm
- Température de fonctionnement : -20°C ~ 80°C
- Indicateur LED de mise sous tension

### Principe de Fonctionnement

Voici la table de vérité du moteur B :

Cette table de vérité montre les différents états du moteur B en fonction des valeurs des broches d'entrée B-1A et B-1B(B-2A). Elle indique la direction de rotation (dans le sens des aiguilles d'une montre ou dans le sens inverse), le freinage ou l'arrêt du moteur B.

B-1A	B-1B(B-2A)	État du moteur B
1	0	Rotation dans le sens des aiguilles d'une montre
0	1	Rotation dans le sens inverse des aiguilles d'une montre
0	0	Freinage
1	1	Arrêt

Voici la table de vérité du moteur A :

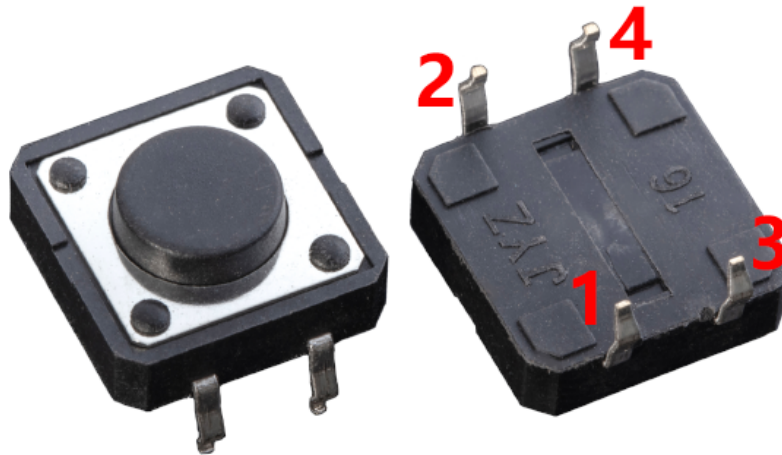
Cette table de vérité montre les différents états du moteur A en fonction des valeurs des broches d'entrée A-1A et A-1B. Elle indique la direction de rotation (dans le sens des aiguilles d'une montre ou dans le sens inverse), le freinage ou l'arrêt du moteur A.

A-1A	A-1B	État du moteur A
1	0	Rotation dans le sens des aiguilles d'une montre
0	1	Rotation dans le sens inverse des aiguilles d'une montre
0	0	Freinage
1	1	Arrêt

- *1.3 Moteur* (Projet de base)
- *1. Mouvement* (Projet Voiture)
- *3. Accélération* (Projet Voiture)
- *8. Voiture IoT* (Projet IoT)

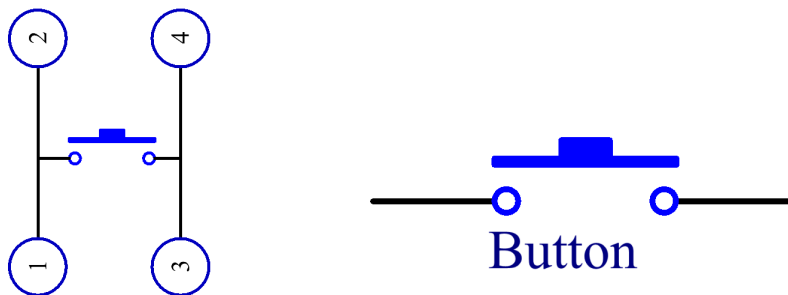
### Contrôleur

## 1.17 Bouton

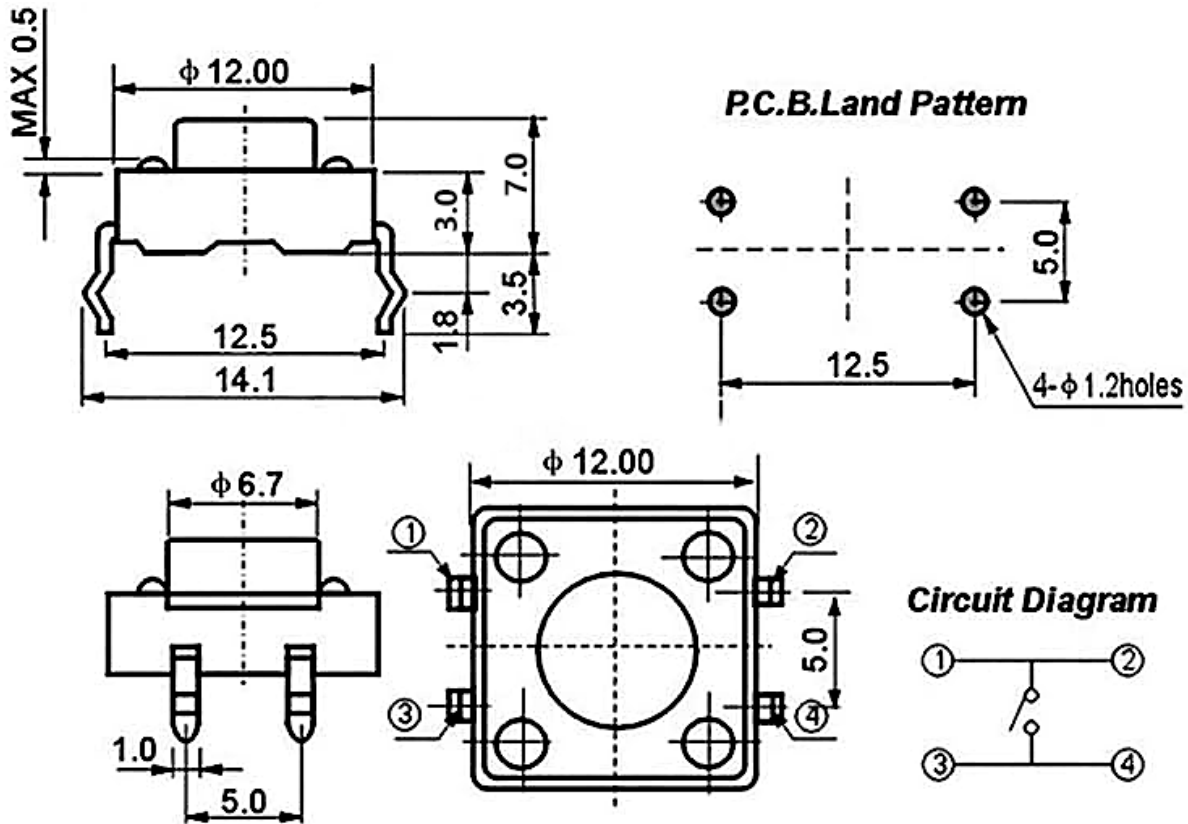


Les boutons sont un composant commun utilisé pour contrôler les dispositifs électroniques. Ils sont généralement utilisés comme interrupteurs pour connecter ou interrompre des circuits. Bien que les boutons existent dans une variété de tailles et de formes, celui utilisé ici est un mini-bouton de 6mm comme montré dans les images suivantes. La broche 1 est connectée à la broche 2 et la broche 3 à la broche 4. Ainsi, il suffit de connecter soit la broche 1 à la broche 3, soit la broche 2 à la broche 4.

La structure interne d'un bouton est la suivante. Le symbole ci-dessous à droite est généralement utilisé pour représenter un bouton dans les circuits.



Puisque la broche 1 est connectée à la broche 2, et la broche 3 à la broche 4, lorsque le bouton est pressé, les 4 broches sont connectées, fermant ainsi le circuit.



### Exemple

- 3.1 *Lecture de la Valeur du Bouton* (Projet de base)
- 2.6 *Sonnette* (Projet Scratch)
- 2.16 *JEU - Manger la Pomme* (Projet Scratch)
- 2.19 *JEU - Pêche* (Projet Scratch)

## 1.18 Interrupteur à Lame Souple



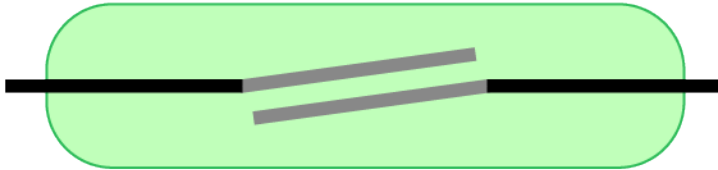
L'interrupteur à lame souple est un commutateur électrique qui fonctionne au moyen d'un champ magnétique appliqué. Il a été inventé par Walter B. Ellwood des Bell Telephone Laboratories en 1936 et breveté aux États-Unis le 27 juin 1940, sous le numéro de brevet 2264746.

Le principe de fonctionnement d'un interrupteur à lame souple est très simple. Deux lames (généralement en fer et nickel, deux métaux) qui se chevauchent aux points d'extrémité sont scellées dans un tube en verre, avec les deux lames se chevauchant et séparées par un petit espace (seulement quelques microns). Le tube en verre est rempli d'un gaz inerte de haute pureté (tel que l'azote), et certains interrupteurs à lame souple sont conçus pour avoir un vide à l'intérieur afin d'améliorer leur performance à haute tension.

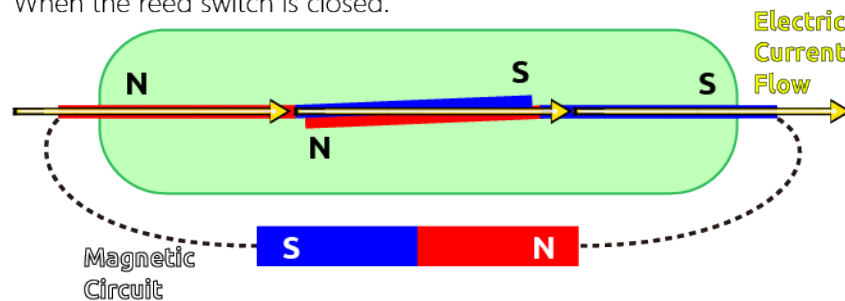
La lame agit comme un conducteur de flux magnétique. Les deux lames ne sont pas en contact lorsqu'elles ne sont pas encore en fonctionnement ; lorsqu'elles traversent un champ magnétique généré par un aimant permanent ou une bobine électromagnétique, le champ magnétique appliqué provoque des polarités différentes près de leurs points d'extrémité, et lorsque la force magnétique dépasse la force de ressort des lames elles-mêmes, les deux lames seront attirées l'une

vers l'autre pour conduire le circuit; lorsque le champ magnétique s'affaiblit ou disparaît, les lames sont relâchées en raison de leur propre élasticité, et les surfaces de contact se sépareront pour ouvrir le circuit.

When the reed switch is open.



When the reed switch is closed.

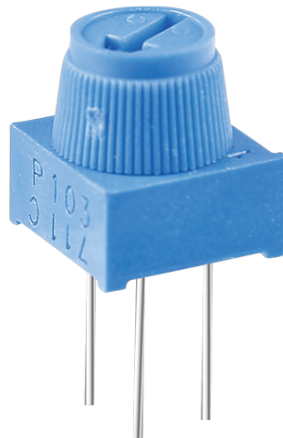


— Interrupteur à Lame Souple - Wikipédia

#### Exemple

- 3.2 *Ressentir le Magnétisme* (Projet de base)
- 7. *Portail à Limite de Courant* (Projet IoT)

## 1.19 Potentiomètre

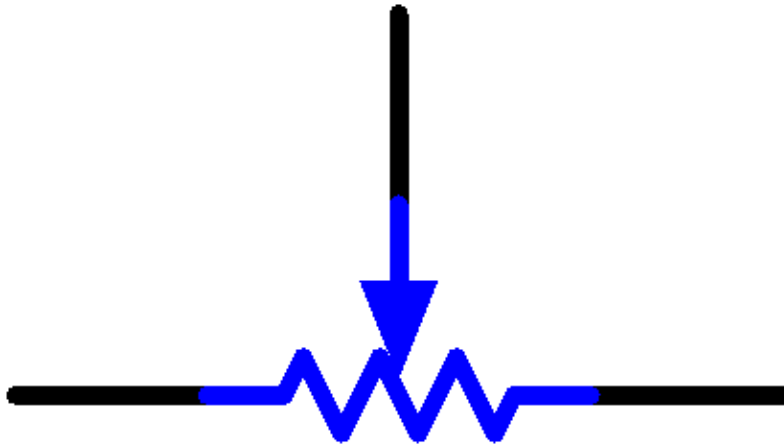


Le potentiomètre est également un composant de résistance à 3 bornes dont la valeur peut être ajustée selon une variation régulière.

Les potentiomètres se présentent sous différentes formes, tailles et valeurs, mais ils ont tous les points communs suivants :

- Ils ont trois bornes (ou points de connexion).
- Ils disposent d'un bouton, d'une vis ou d'un curseur qui peut être déplacé pour faire varier la résistance entre la borne centrale et l'une des bornes extérieures.

— La résistance entre la borne centrale et l'une des bornes extérieures varie de 0 à la résistance maximale du potentiomètre lorsque le bouton, la vis ou le curseur est déplacé.  
Voici le symbole de circuit du potentiomètre.



Les fonctions du potentiomètre dans le circuit sont les suivantes :

1. Agir comme un diviseur de tension

Le potentiomètre est une résistance réglable en continu. Lorsque vous ajustez l'axe ou la poignée coulissante du potentiomètre, le contact mobile glissera sur la résistance. À ce moment, une tension peut être délivrée en fonction de la tension appliquée sur le potentiomètre et de l'angle de rotation ou du déplacement du bras mobile.

2. Agir comme un rhéostat

Lorsque le potentiomètre est utilisé comme un rhéostat, connectez la broche centrale et l'une des 2 autres broches dans le circuit. Ainsi, vous pouvez obtenir une valeur de résistance modifiée en douceur et de manière continue dans le déplacement du contact mobile.

3. Agir comme un régulateur de courant

Lorsque le potentiomètre agit comme un régulateur de courant, la borne de contact coulissante doit être connectée comme l'une des bornes de sortie.

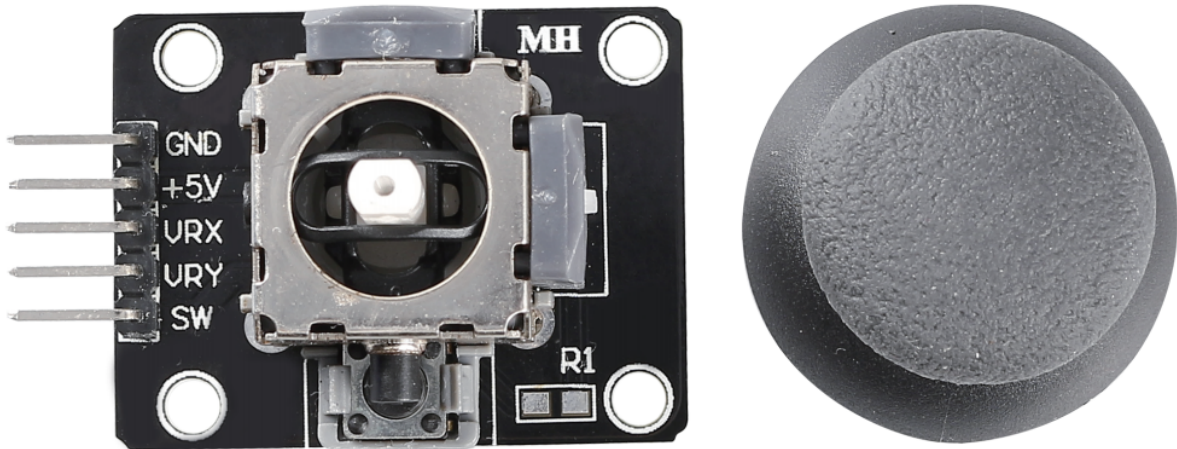
Si vous souhaitez en savoir plus sur le potentiomètre, consultez : [Potentiomètre - Wikipédia](#)

### Exemple

- *4.1 Tournez le Bouton* (Projet de base)
- *2.5 Souris Mobile* (Projet Scratch)
- *2.18 JEU - Clone de Breakout* (Projet Scratch)



## 1.20 Module de Joystick

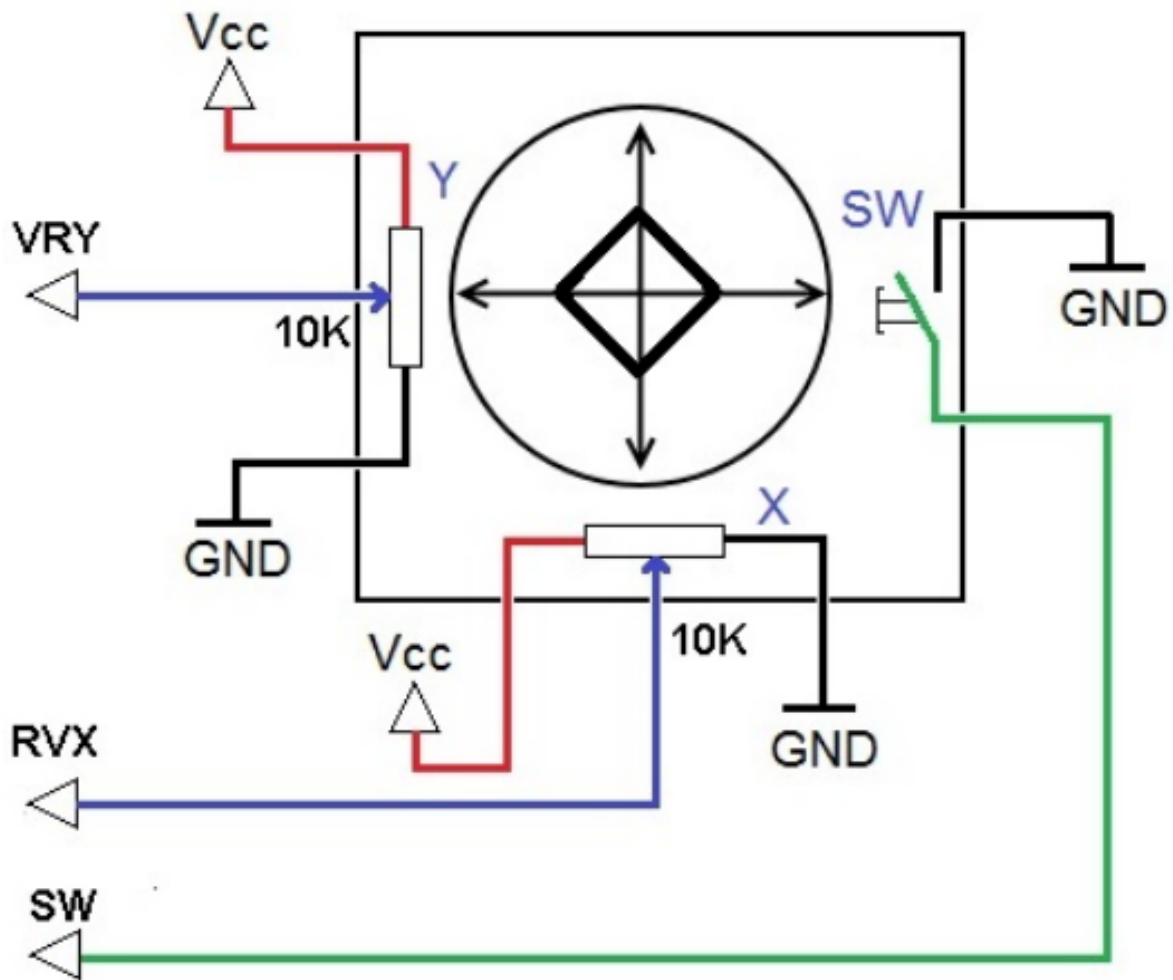


L'idée de base d'un joystick est de traduire le mouvement d'un bâton en informations électroniques que l'ordinateur peut traiter.

Pour communiquer une gamme complète de mouvements à l'ordinateur, un joystick doit mesurer la position du bâton sur deux axes - l'axe X (de gauche à droite) et l'axe Y (de haut en bas). Comme en géométrie de base, les coordonnées X-Y déterminent exactement la position du bâton.

Pour déterminer l'emplacement du bâton, le système de contrôle du joystick surveille simplement la position de chaque axe. La conception conventionnelle du joystick analogique le fait avec deux potentiomètres, ou résistances variables.

Le joystick dispose également d'une entrée numérique qui est actionnée lorsque le joystick est pressé vers le bas.

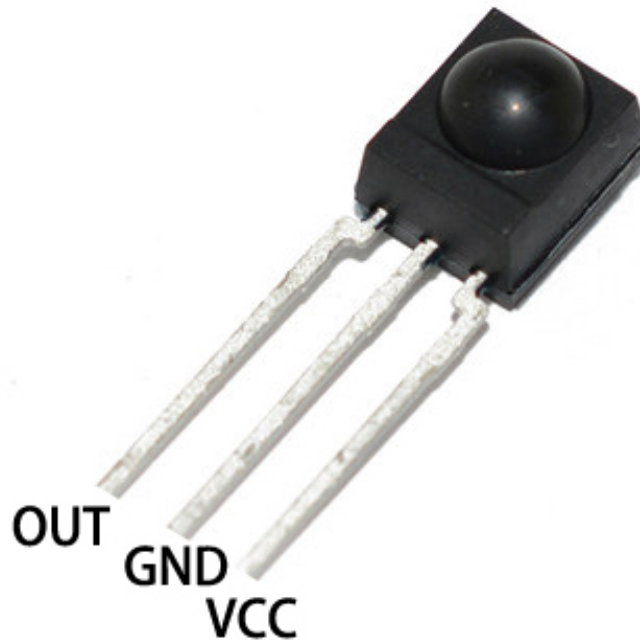


### Exemple

- 4.3 Manipuler le Joystick (Projet de base)
- 2.15 JEU - Étoiles Croisées (Projet Scratch)
- 2.22 JEU - Tueur de Dragon (Projet Scratch)

## 1.21 Récepteur IR

### Récepteur IR



- OUT : Sortie du signal
- GND : Masse
- VCC : alimentation électrique, 3.3v~5V

Un récepteur infrarouge est un composant qui reçoit des signaux infrarouges et peut recevoir de manière indépendante des rayons infrarouges et sortir des signaux compatibles avec le niveau TTL. Il est de taille similaire à un transistor classique emballé dans du plastique et convient à toutes sortes de télécommandes infrarouges et de transmissions infrarouges.

La communication infrarouge, ou IR, est une technologie de communication sans fil populaire, peu coûteuse et facile à utiliser. La lumière infrarouge a une longueur d'onde légèrement plus longue que la lumière visible, elle est donc imperceptible à l'œil humain - idéale pour la communication sans fil. Un schéma de modulation courant pour la communication infrarouge est la modulation à 38KHz.

- Capteur de Réception IR **HS0038B**, haute sensibilité
- Peut être utilisé pour la télécommande
- Alimentation Électrique : 5V
- Interface : Numérique
- Fréquence de Modulation : 38Khz
- Définitions des Broches : (1) Sortie (2) Vcc (3) Masse
- Taille : 23.5mm x 21.5mm

#### Télécommande



Il s'agit d'une télécommande sans fil infrarouge mince avec 21 boutons de fonction et une distance de transmission

allant jusqu'à 8 mètres, adaptée pour commander une large gamme d'appareils dans une chambre d'enfant.

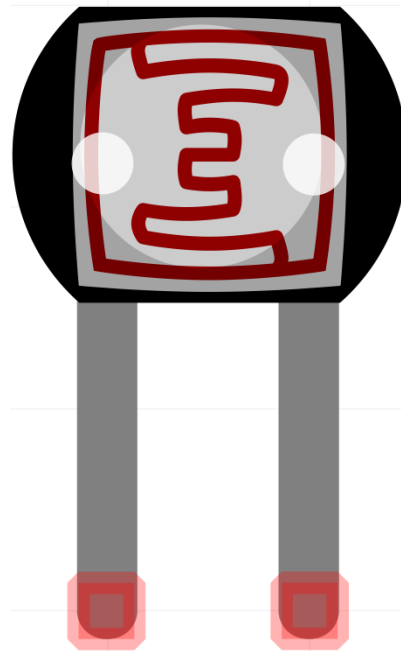
- Taille : 85x39x6mm
- Portée de la télécommande : 8-10m
- Batterie : Batterie au lithium-manganèse de type bouton 3V
- Fréquence porteuse infrarouge : 38KHz
- Matériau de collage de surface : PET de 0.125mm
- Vie effective : plus de 20 000 fois

### Exemple

- *5.11.2 Récepteur IR* (Projet de base)
- *9. Télécommande* (Projet Voiture)
- *10. Démarrage en Un Clic* (Projet Voiture)

### Capteur

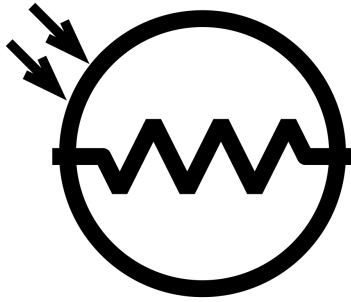
## 1.22 Photorésistance



Une photorésistance ou cellule photoélectrique est une résistance variable contrôlée par la lumière. La résistance d'une photorésistance diminue avec l'augmentation de l'intensité lumineuse incidente ; en d'autres termes, elle présente une photoconductivité.

Une photorésistance peut être utilisée dans des circuits détecteurs sensibles à la lumière et dans des circuits de commutation activés par la lumière ou l'obscurité, agissant comme un semi-conducteur résistant. Dans l'obscurité, une photorésistance peut avoir une résistance aussi élevée que plusieurs mégaohms (M), tandis qu'à la lumière, elle peut avoir une résistance aussi faible que quelques centaines d'ohms.

Voici le symbole électronique de la photorésistance.

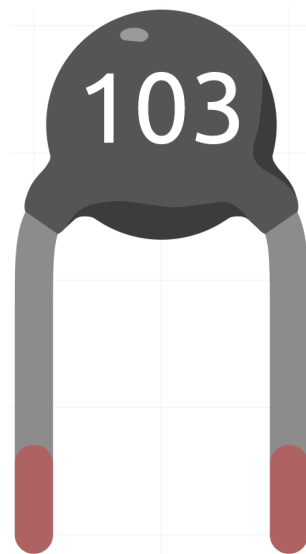


— [Photorésistance - Wikipédia](#)

#### Exemple

- [4.2 Ressentir la Lumière](#) (Projet de base)
- [5. Surveillance de l'Environnement Domestique](#) (Projet IoT)
- [6. Moniteur de Plantes](#) (Projet IoT)

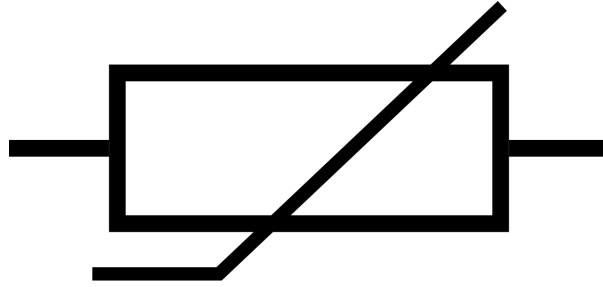
## 1.23 Thermistance



Une thermistance est un type de résistance dont la résistance dépend fortement de la température, bien plus que dans les résistances standard. Le mot est une combinaison de thermal (thermique) et resistor (résistance). Les thermistances sont largement utilisées comme limiteurs de courant d'appel, capteurs de température (type à coefficient de température négatif ou NTC généralement), protecteurs de surintensité auto-réarmables et éléments chauffants autorégulants (type à coefficient de température positif ou PTC généralement).

— [Thermistance - Wikipédia](#)

Voici le symbole électronique de la thermistance.



Il existe deux types fondamentaux de thermistances :

- Avec les thermistances NTC, la résistance diminue à mesure que la température augmente, généralement en raison d'une augmentation des électrons de conduction élevés par l'agitation thermique de la bande de valence. Un NTC est couramment utilisé comme capteur de température, ou en série avec un circuit comme limiteur de courant d'appel.
- Avec les thermistances PTC, la résistance augmente à mesure que la température augmente, généralement en raison d'une augmentation de l'agitation thermique du réseau, en particulier celles des impuretés et imperfections. Les thermistances PTC sont généralement installées en série avec un circuit et utilisées pour protéger contre les conditions de surintensité, en tant que fusibles réarmables.

Dans ce kit, nous utilisons un NTC. Chaque thermistance a une résistance normale. Ici, elle est de 10k ohms, mesurée à 25 degrés Celsius.

Voici la relation entre la résistance et la température :

$$R_T = R_N * \exp(B(1/T_K - 1/T_N))$$

- **$R_T$**  est la résistance de la thermistance NTC lorsque la température est  $T_K$ .
- **$R_N$**  est la résistance de la thermistance NTC sous la température nominale  $T_N$ . Ici, la valeur numérique de  $R_N$  est 10k.
- **$T_K$**  est une température en Kelvin et l'unité est K. Ici, la valeur numérique de  $T_K$  est 273.15 + degrés Celsius.
- **$T_N$**  est une température nominale en Kelvin; l'unité est également K. Ici, la valeur numérique de  $T_N$  est 273.15+25.
- Et  **$B(\text{beta})$** , la constante de matériau de la thermistance NTC, est également appelée indice de sensibilité thermique avec une valeur numérique 3950.
- **exp** est l'abréviation d'exponentielle, et le nombre de base e est un nombre naturel et équivaut à environ 2.7.

Convertissez cette formule  $T_K = 1 / (\ln(R_T/R_N)/B + 1/T_N)$  pour obtenir la température en Kelvin qui moins 273.15 équivaut aux degrés Celsius.

Cette relation est une formule empirique. Elle est précise uniquement lorsque la température et la résistance sont dans la plage effective.

### Exemple

- [6.3 Alarme de Haute Température](#) (Projet de base)
- [4.5 Thermomètre](#) (Projet de base)
- [2.7 Alarme de Température Basse](#) (Projet Scratch)

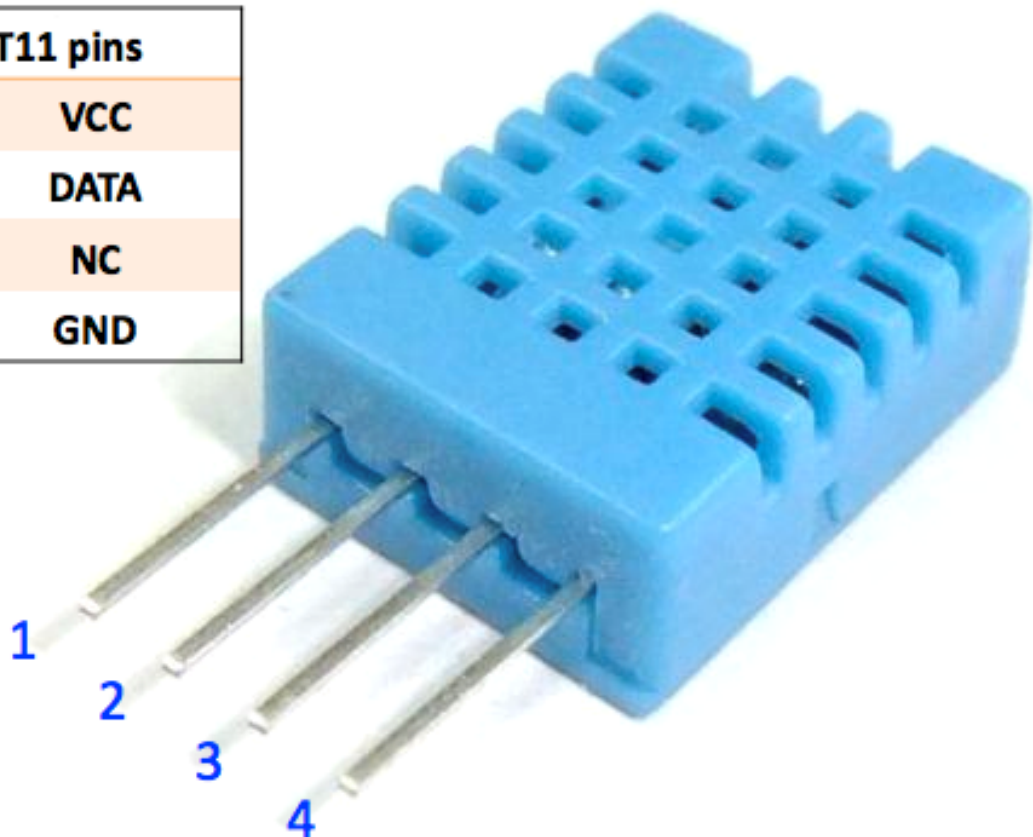
## 1.24 Capteur d'Humidité et de Température DHT11

Le capteur numérique de température et d'humidité DHT11 est un capteur composite qui contient une sortie de signal numérique calibrée de la température et de l'humidité. La technologie des modules numériques dédiés et la technologie de détection de température et d'humidité sont appliquées pour garantir que le produit a une haute fiabilité et une excellente stabilité à long terme.

Le capteur comprend un composant de détection d'humidité résistif et un dispositif de mesure de température NTC, et est connecté à un microcontrôleur 8 bits haute performance.

Seules trois broches sont disponibles pour l'utilisation : VCC, GND et DATA. Le processus de communication commence par la ligne DATA envoyant des signaux de départ au DHT11, et le DHT11 reçoit les signaux et retourne un signal de réponse. Ensuite, l'hôte reçoit le signal de réponse et commence à recevoir les données d'humidité de 40 bits (8 bits d'humidité entière + 8 bits d'humidité décimale + 8 bits de température entière + 8 bits de température décimale + 8 bits de somme de contrôle).

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



### Caractéristiques

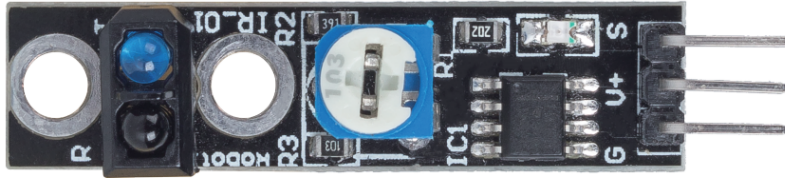
1. Plage de mesure de l'humidité : 20 - 90%RH
2. Plage de mesure de la température : 0 - 60°C
3. Sortie de signaux numériques indiquant la température et l'humidité
4. Tension de travail : DC 5V ; Taille du PCB : 2.0 x 2.0 cm
5. Précision de mesure de l'humidité :  $\pm 5\%$  RH
6. Précision de mesure de la température :  $\pm 2^\circ\text{C}$

- [Fiche technique du DHT11](#)

### Exemple

- [5.11.3 Température - Humidité](#) (Projet de base)
- [5. Surveillance de l'Environnement Domestique](#) (Projet IoT)
- [6. Moniteur de Plantes](#) (Projet IoT)

## 1.25 Module de Suivi de Ligne



- S : Habituellement niveau bas, passe à niveau haut lorsque la ligne noire est détectée.
- V+ : Alimentation électrique, 3.3v~5V
- G : Masse

Il s'agit d'un module de suivi de ligne à 1 canal qui, comme son nom l'indique, suit les lignes noires sur un fond blanc ou les lignes blanches sur un fond noir.



Le module utilise un capteur infrarouge TCRT500, qui se compose d'une LED infrarouge (bleue) et d'un phototransistor sensible à la lumière (noir).

- La LED infrarouge bleue, une fois alimentée, émet une lumière infrarouge invisible à l'œil humain.
- Le phototransistor noir, utilisé pour recevoir la lumière infrarouge, possède une résistance interne dont la résistance varie en fonction de la lumière infrarouge reçue ; plus il reçoit de lumière infrarouge, plus sa résistance diminue et vice versa.

Il y a un comparateur LM393 sur le module, qui est utilisé pour comparer la tension du phototransistor avec la tension réglée (ajustée par le potentiomètre) ; si elle est supérieure à la tension réglée, la sortie est 1 ; sinon la sortie est 0.

Par conséquent, lorsque le tube émetteur infrarouge brille sur une surface noire, parce que le noir absorbe la lumière, le transistor photosensible reçoit moins de lumière infrarouge, sa résistance augmentera (augmentation de la tension), après le comparateur LM393, la sortie est de niveau haut.

De même, lorsqu'il brille sur une surface blanche, la lumière réfléchie devient plus abondante et la résistance du phototransistor diminue (diminution de la tension) ; par conséquent, le comparateur sort un niveau bas et la LED indicatrice s'allume.

- [TCRT5000](#)

### Caractéristiques

- Utilisation du capteur d'émission infrarouge TCRT5000
- Distance de détection : 1-8mm, distance focale de 2.5mm



- Signal de sortie du comparateur propre, bonne forme d'onde, capacité de pilotage supérieure à 15mA
- Utilisation d'un potentiomètre pour ajuster la sensibilité
- Tension de fonctionnement : 3.3V-5V
- Sortie numérique : 0 (blanc) et 1 (noir)
- Utilise un comparateur LM393 à large tension.
- Taille : 42mmx10mm

#### Exemple

- *3.4 Détecter la Ligne* (Projet de base)
- *4. Suivre la ligne* (Projet Voiture)
- *2.21 JEU - Protège ton Cœur* (Projet Scratch)

## 1.26 Module d'Humidité du Sol

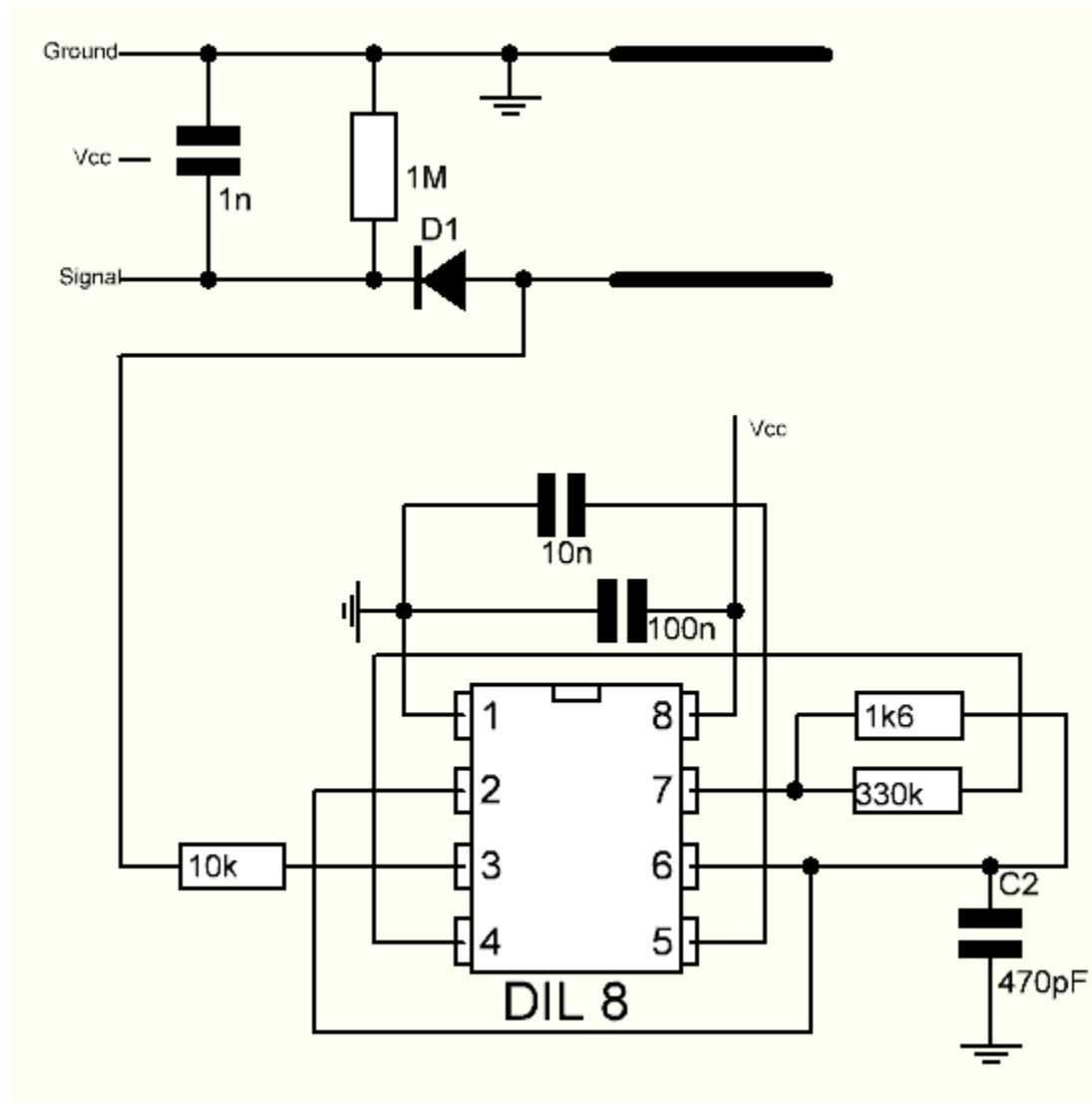


- GND : Masse
- VCC : Alimentation électrique, 3.3v~5V
- AUOT : Sort la valeur d'humidité du sol, plus le sol est humide, plus sa valeur est petite.

Ce capteur capacitif d'humidité du sol se distingue de la plupart des capteurs résistifs sur le marché, en utilisant le principe de l'induction capacitive pour détecter l'humidité du sol. Il évite le problème des capteurs résistifs qui sont très sensibles à la corrosion et prolonge considérablement sa durée de vie.

Il est fabriqué à partir de matériaux résistants à la corrosion et possède une excellente durée de vie. Insérez-le dans le sol autour des plantes et surveillez les données d'humidité du sol en temps réel. Le module comprend un régulateur de tension intégré qui lui permet de fonctionner sur une plage de tension de 3.3 ~ 5.5 V. Il est idéal pour les microcontrôleurs basse tension avec des alimentations de 3.3 V et 5 V.

Le schéma matériel du capteur capacitif d'humidité du sol est montré ci-dessous.



Il y a un oscillateur à fréquence fixe, construit avec un circuit intégré de minuterie 555. L'onde carrée générée est ensuite envoyée au capteur comme un condensateur. Cependant, pour le signal d'onde carrée, le condensateur a une certaine réactance ou, pour simplifier, une résistance avec une résistance ohmique pure (résistance de 10k sur la broche 3) pour former un diviseur de tension.

Plus l'humidité du sol est élevée, plus la capacité du capteur est grande. En conséquence, l'onde carrée a moins de réactance, ce qui réduit la tension sur la ligne de signal, et plus petite est la valeur de l'entrée analogique à travers le microcontrôleur.

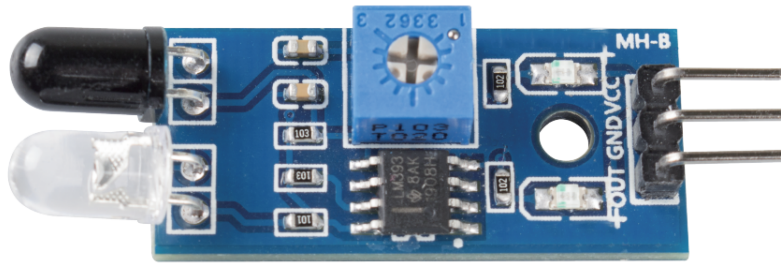
### Spécifications

- Tension de fonctionnement : 3.3 ~ 5.5 VDC
- Tension de sortie : 0 ~ 3.0VDC
- Courant de fonctionnement : 5mA
- Interface : PH2.0-3P
- Dimensions : 3.86 x 0.905 pouces (L x l)
- Poids : 15g

### Exemple

- 4.4 Mesurer l'Humidité du Sol (Projet de base)
- 6. Moniteur de Plantes (Projet IoT)

## 1.27 Module d'Évitement d'Obstacle

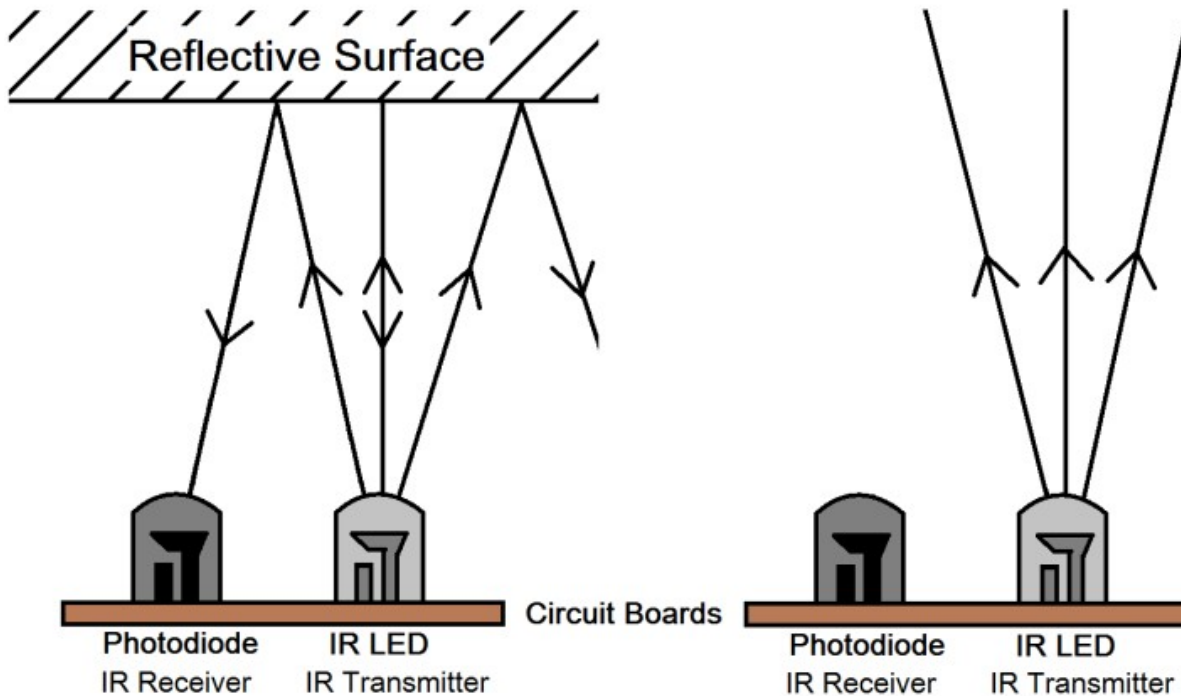


- **VCC** : Alimentation électrique, 3.3 ~ 5V DC.
- **GND** : Masse
- **OUT** : Broche de signal, habituellement au niveau haut, et passe au niveau bas lorsqu'un obstacle est détecté.

Le module d'évitement d'obstacle IR a une forte adaptabilité à la lumière ambiante, il possède une paire de tubes infrarouges émetteurs et récepteurs.

Le tube émetteur émet une fréquence infrarouge, lorsque la direction de détection rencontre un obstacle, le rayonnement infrarouge est reçu par le tube récepteur, après le traitement du circuit comparateur, l'indicateur s'allume et émet un signal de niveau bas.

La distance de détection peut être ajustée par un potentiomètre, la plage de distance effective est de 2-30cm.



### Exemple

- [3.3 Détecter l'Obstacle](#) (Projet de base)
- [5. Jouer avec le module d'évitement d'obstacles](#) (Projet Voiture)
- [8. Voiture Autonome](#) (Projet Voiture)
- [7. Portail à Limite de Courant](#) (Projet IoT)
- [2.13 JEU - Tir](#) (Projet Scratch)
- [2.20 JEU - Ne Touchez Pas la Tuile Blanche](#) (Projet Scratch)

## 1.28 Module Ultrasonique

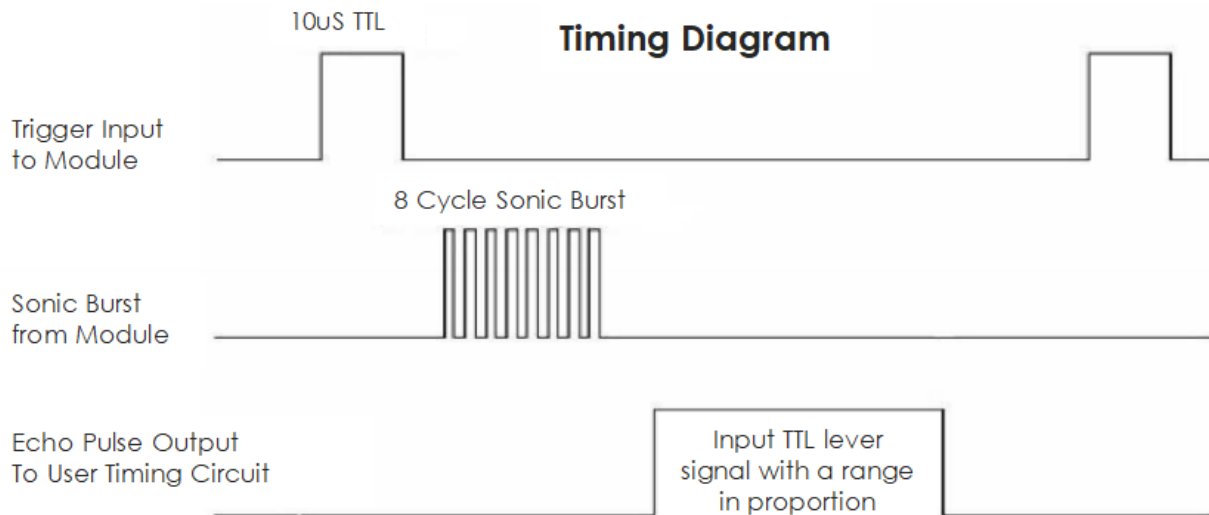


Le module de télémétrie ultrasonique offre une fonction de mesure sans contact de 2 cm à 400 cm, et la précision de mesure peut atteindre 3 mm. Il garantit que le signal reste stable jusqu'à 5 m, puis s'affaiblit progressivement après 5 m, jusqu'à disparaître à la position de 7 m.

Le module comprend des émetteurs ultrasoniques, un récepteur et un circuit de contrôle. Les principes de base sont les suivants :

1. Utiliser un basculeur IO pour traiter un signal de niveau haut d'au moins 10us.
2. Le module envoie automatiquement huit signaux de 40 kHz et détecte s'il y a un retour de signal d'impulsion.
3. Si le signal revient, en passant le niveau haut, la durée de sortie IO haute est le temps écoulé entre l'émission de l'onde ultrasonique et son retour. Ici, distance testée = (temps haut x vitesse du son (340 m/s) / 2.

Le diagramme temporel est présenté ci-dessous.



Vous avez juste besoin de fournir une impulsion courte de 10us pour l'entrée de déclenchement pour commencer la télémétrie, et ensuite le module émettra une salve de 8 cycles d'ultrasons à 40 kHz et augmentera son écho. Vous pouvez calculer la distance à travers l'intervalle de temps entre l'envoi du signal de déclenchement et la réception du signal d'écho.

Formule :  $us / 58 = \text{centimètres}$  ou  $us / 148 = \text{pouces}$  ; ou : la distance = temps de niveau haut \* vitesse (340M/S) / 2 ; il est conseillé d'utiliser un cycle de mesure supérieur à 60 ms afin d'éviter les collisions de signaux entre le signal de déclenchement et le signal d'écho.

**Exemple**

- 5.8 *Fonction Définie par l'Utilisateur* (Projet de base)
- 7. *Suivez Votre Main* (Projet Voiture)
- 6. *Jouer avec le module ultrasonique* (Projet Voiture)
- 2.17 *JEU - Perroquet Volant* (Projet Scratch)



---

## Découverte d'Arduino

---

Si vous ne connaissez rien à Arduino. Voici quelques mots que j'aimerais vous présenter : électronique, design, programmation, et même Maker. Certains d'entre vous peuvent penser que ces mots sont assez éloignés de nous, mais en réalité, ils ne le sont pas du tout. Parce qu'Arduino peut nous emmener dans le monde de la programmation et nous aider à réaliser le rêve d'être un Maker. Dans cette session, nous allons apprendre :

- Qu'est-ce qu'Arduino ?
- Que peut faire Arduino ?
- Comment construire un projet Arduino ?

### 2.1 Qu'est-ce qu'Arduino ?

Tout d'abord, je vais vous faire une brève introduction à Arduino.

Arduino est une plateforme de prototypage électronique open-source pratique, flexible et facile à utiliser, comprenant des cartes Arduino matérielles de divers modèles et le logiciel Arduino IDE. Il convient non seulement aux ingénieurs pour le prototypage rapide, mais aussi aux artistes, designers, amateurs, tout en étant presque un outil incontournable pour les Makers modernes.

Arduino constitue un système assez vaste. Il comprend un logiciel, un matériel, et une très grande communauté en ligne de personnes qui ne se sont jamais rencontrées mais sont capables de travailler ensemble grâce à un hobby commun. Chacun dans la famille Arduino utilise sa sagesse, crée avec ses mains et partage une grande invention après l'autre. Et vous pouvez également en faire partie.

## 2.2 Que peut faire Arduino ?

En parlant de cela, vous pouvez vous demander ce qu'Arduino peut réellement faire. Autant le dire, Arduino résoudra tous vos problèmes.

Techniquement parlant, Arduino est un contrôleur logique programmable. C'est une carte de développement qui peut être utilisée pour créer de nombreuses créations électroniques excitantes et créatives : telles que des voitures télécommandées, des bras robotiques, des robots bioniques, des maisons intelligentes, etc.

Les cartes Arduino sont simples, claires et puissantes, adaptées aux étudiants, aux makers et même aux programmeurs professionnels.

À ce jour, les passionnés d'électronique du monde entier continuent de développer des créations électroniques créatives basées sur les cartes de développement Arduino.

## 2.3 Comment construire un projet Arduino

Suivez ces étapes pour apprendre à utiliser Arduino à partir de zéro !

### 2.3.1 Télécharger et Installer Arduino IDE 2.0

L'IDE Arduino, connu sous le nom d'Environnement de Développement Intégré Arduino, fournit tout le support logiciel nécessaire pour compléter un projet Arduino. C'est un logiciel de programmation spécialement conçu pour Arduino, fourni par l'équipe Arduino, qui nous permet d'écrire des programmes et de les télécharger sur la carte Arduino.

L'IDE Arduino 2.0 est un projet open source. C'est un grand pas par rapport à son robuste prédécesseur, l'IDE Arduino 1.x, et il est livré avec une interface utilisateur remaniée, un gestionnaire amélioré des cartes et des bibliothèques, un débogueur, une fonction d'auto-complétion et bien plus encore.

Dans ce tutoriel, nous allons montrer comment télécharger et installer l'IDE Arduino 2.0 sur votre ordinateur Windows, Mac ou Linux.

#### Exigences

- Windows - Win 10 et plus récent, 64 bits
- Linux - 64 bits
- Mac OS X - Version 10.14 : « Mojave » ou plus récent, 64 bits

#### Télécharger l'IDE Arduino 2.0

1. Visitez .
2. Téléchargez l'IDE pour votre version de système d'exploitation.





## Arduino IDE 2.0.0

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the [Arduino IDE 2.0 documentation](#).

Nightly builds with the latest bugfixes are available through the section below.

SOURCE CODE

The Arduino IDE 2.0 is open source and its source code is hosted on [GitHub](#).

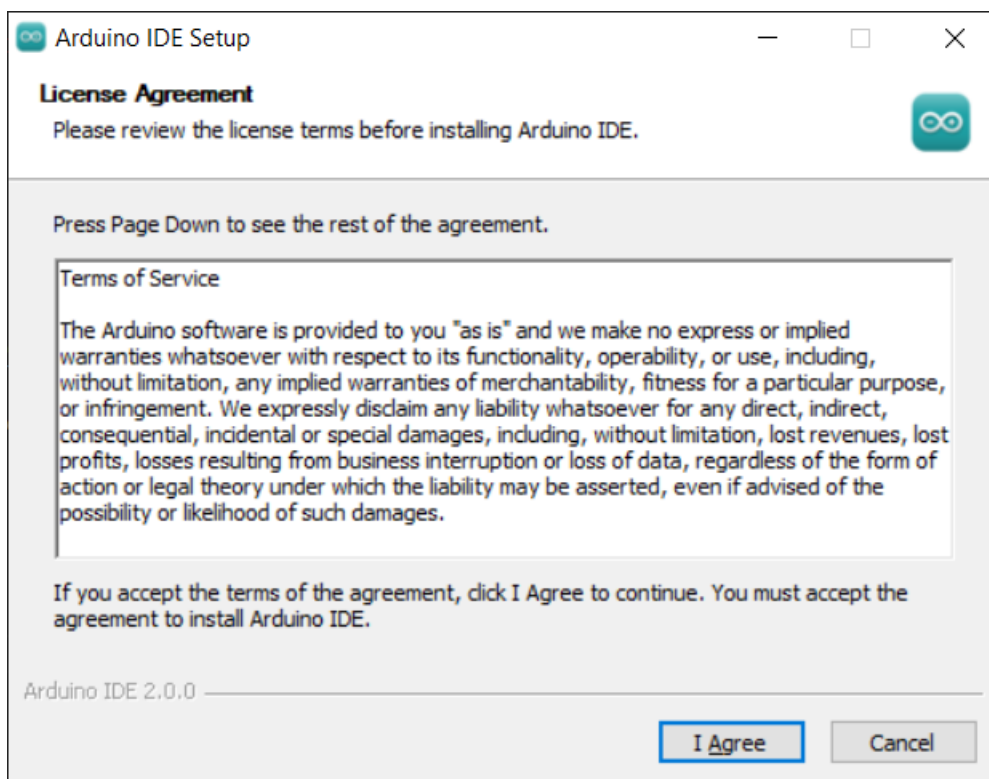
### DOWNLOAD OPTIONS

<b>Windows</b>	Win 10 and newer, 64 bits
<b>Windows</b>	MSI Installer
<b>Windows</b>	ZIP file
<b>Linux</b>	AppImage 64 bits (X86-64)
<b>Linux</b>	ZIP file 64 bits (X86-64)
<b>macOS</b>	10.14: "Mojave" or newer, 64 bits

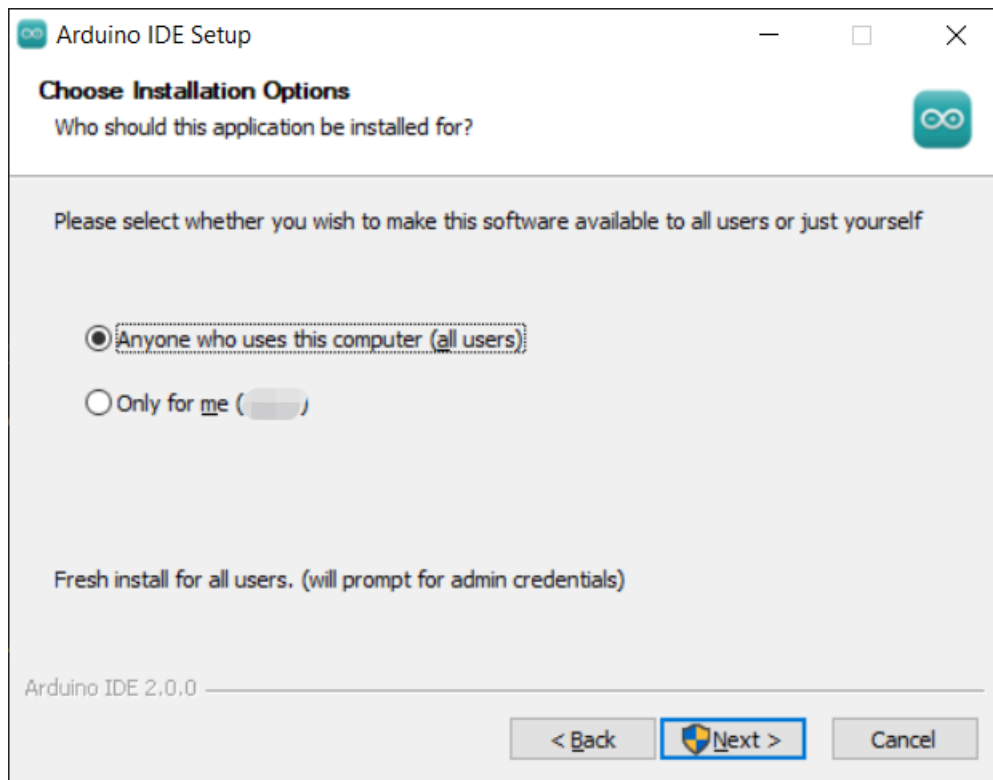
## Installation

### Windows

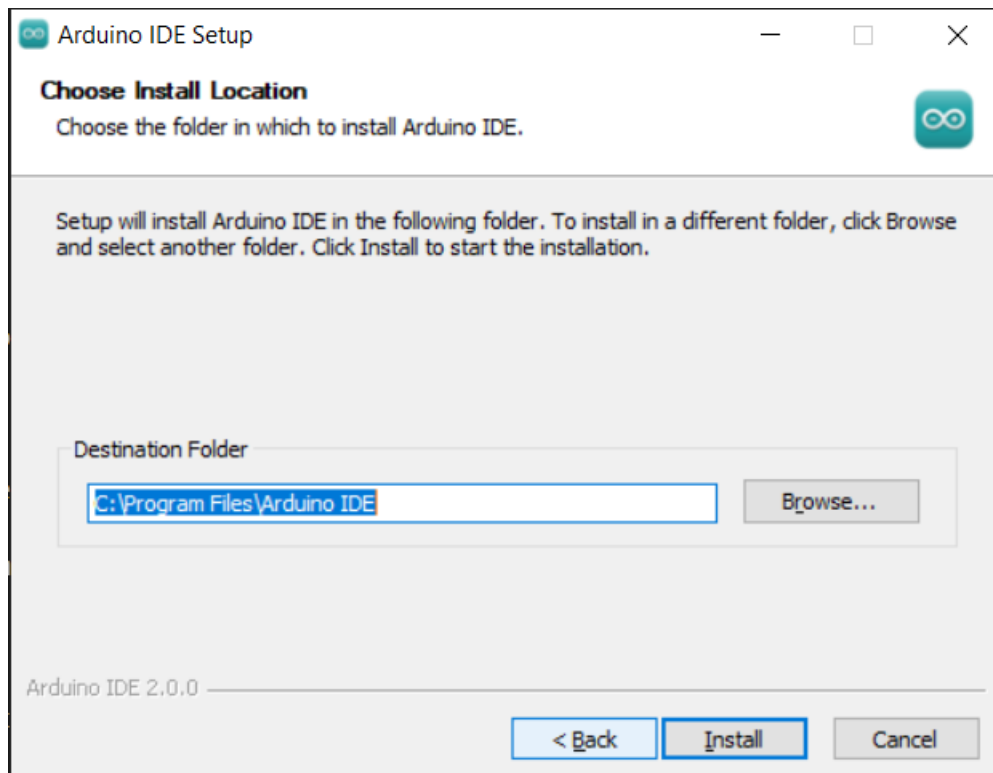
1. Double-cliquez sur le fichier `arduino-ide_xxxx.exe` pour exécuter le fichier téléchargé.
2. Lisez l'Accord de Licence et acceptez-le.



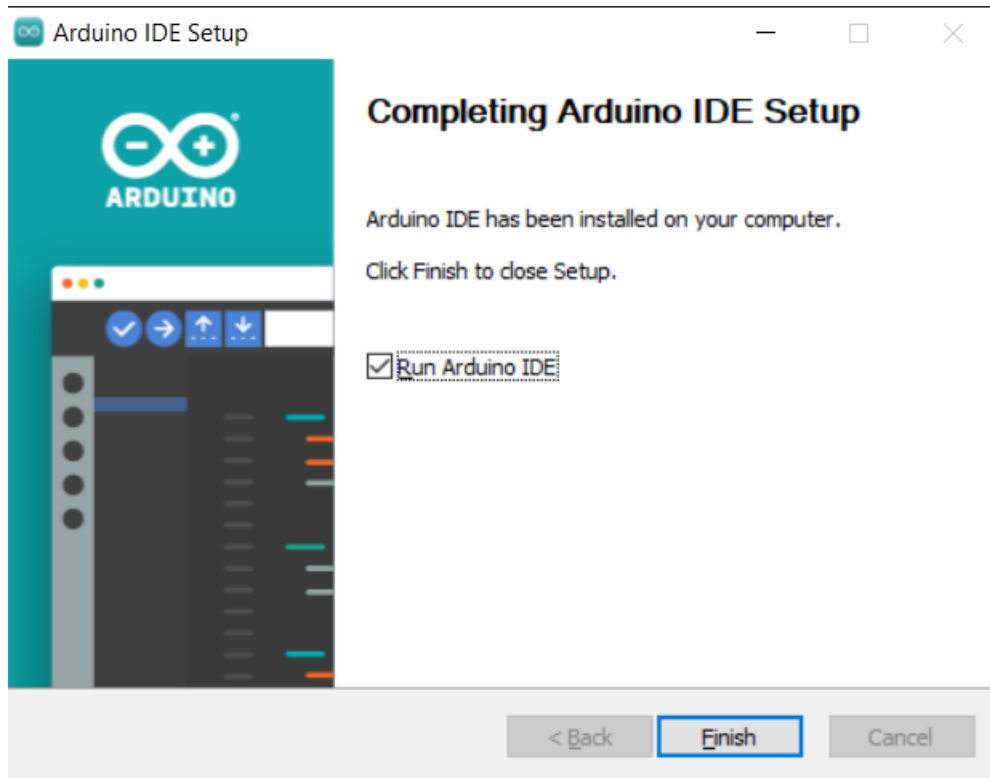
3. Choisissez les options d'installation.



4. Choisissez le lieu d'installation. Il est recommandé d'installer le logiciel sur un disque autre que le disque système.

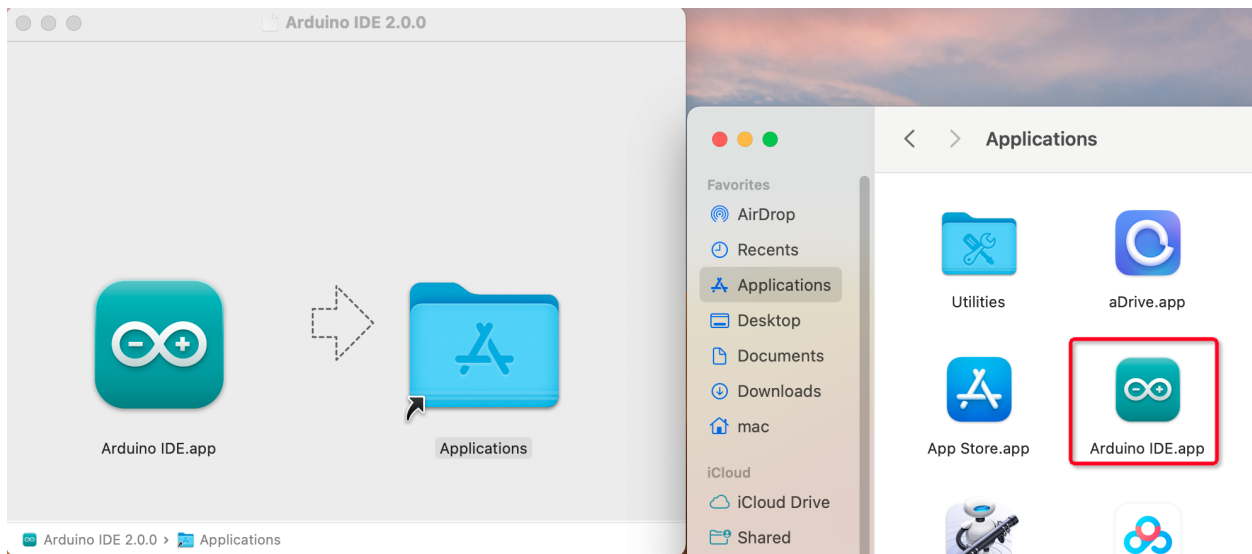


5. Puis terminez.



## macOS

Double-cliquez sur le fichier `arduino_ide_XXXX.dmg` téléchargé et suivez les instructions pour copier l' **Arduino IDE.app** dans le dossier **Applications**, vous verrez l'IDE Arduino installé avec succès après quelques secondes.

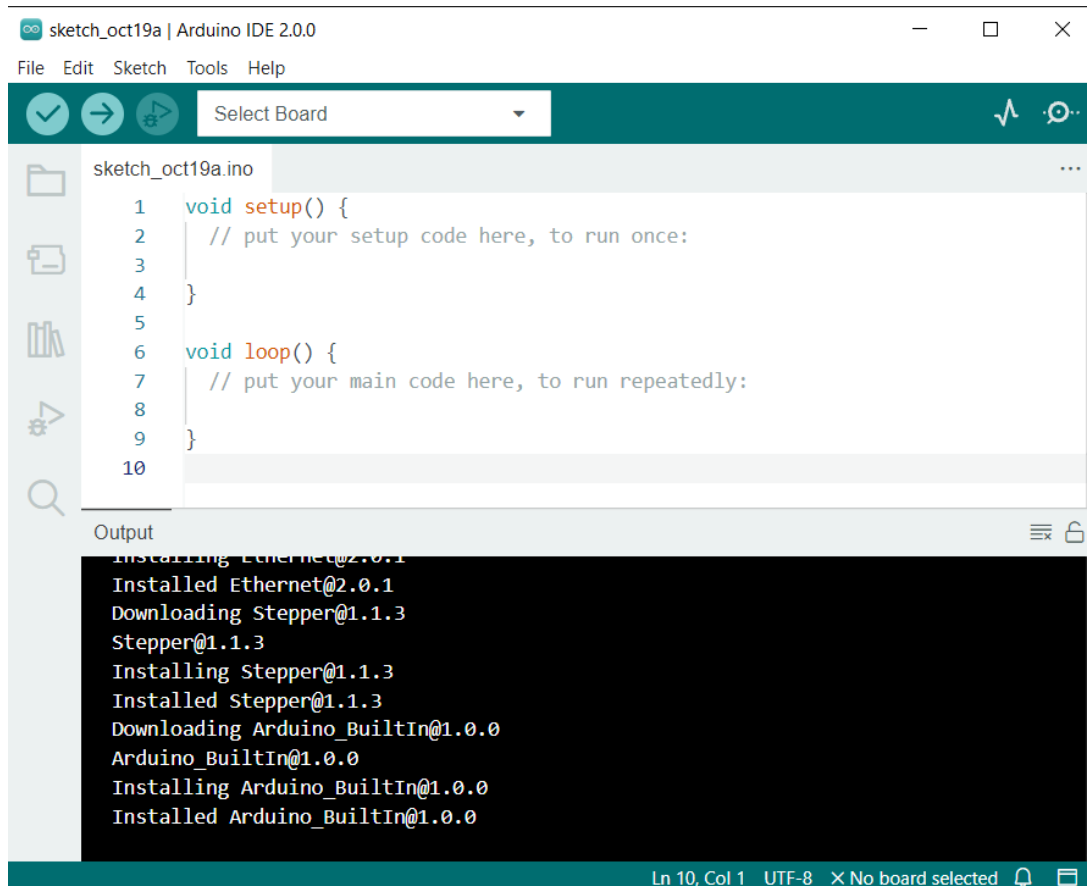


## Linux

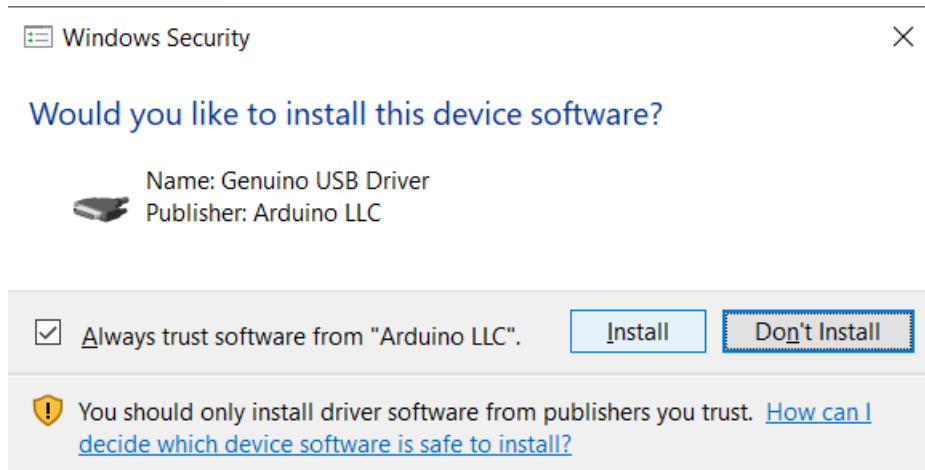
Pour le tutoriel sur l'installation de l'IDE Arduino 2.0 sur un système Linux, veuillez vous référer à : <https://docs.arduino.cc/software/ide-v2/tutorials/getting-started/ide-v2-downloading-and-installing#linux>

### Ouvrir l'IDE

1. Lorsque vous ouvrez pour la première fois l'IDE Arduino 2.0, il installe automatiquement les cartes Arduino AVR, les bibliothèques intégrées et les autres fichiers requis.



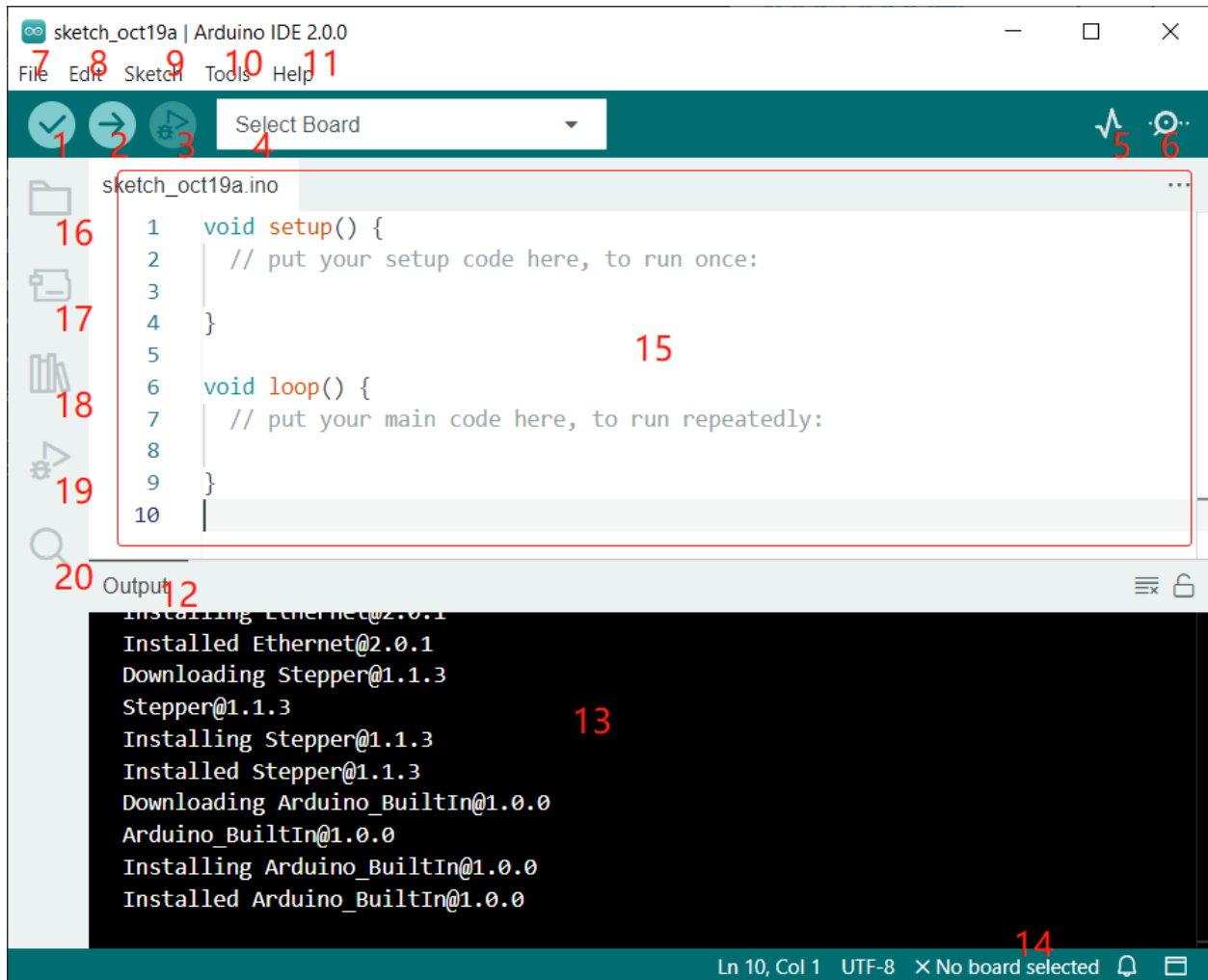
2. De plus, votre pare-feu ou centre de sécurité peut afficher quelques fois une fenêtre vous demandant si vous souhaitez installer un pilote de périphérique. Veuillez installer tous les pilotes nécessaires.



3. Maintenant, votre IDE Arduino est prêt !

**Note :** Dans le cas où certaines installations n'auraient pas fonctionné en raison de problèmes de réseau ou d'autres raisons, vous pouvez rouvrir l'IDE Arduino et il terminera le reste de l'installation. La fenêtre de sortie ne s'ouvrira pas automatiquement après que toutes les installations soient complètes, à moins que vous ne cliquiez sur Vérifier ou Télécharger.

## 2.3.2 Présentation de l'IDE Arduino

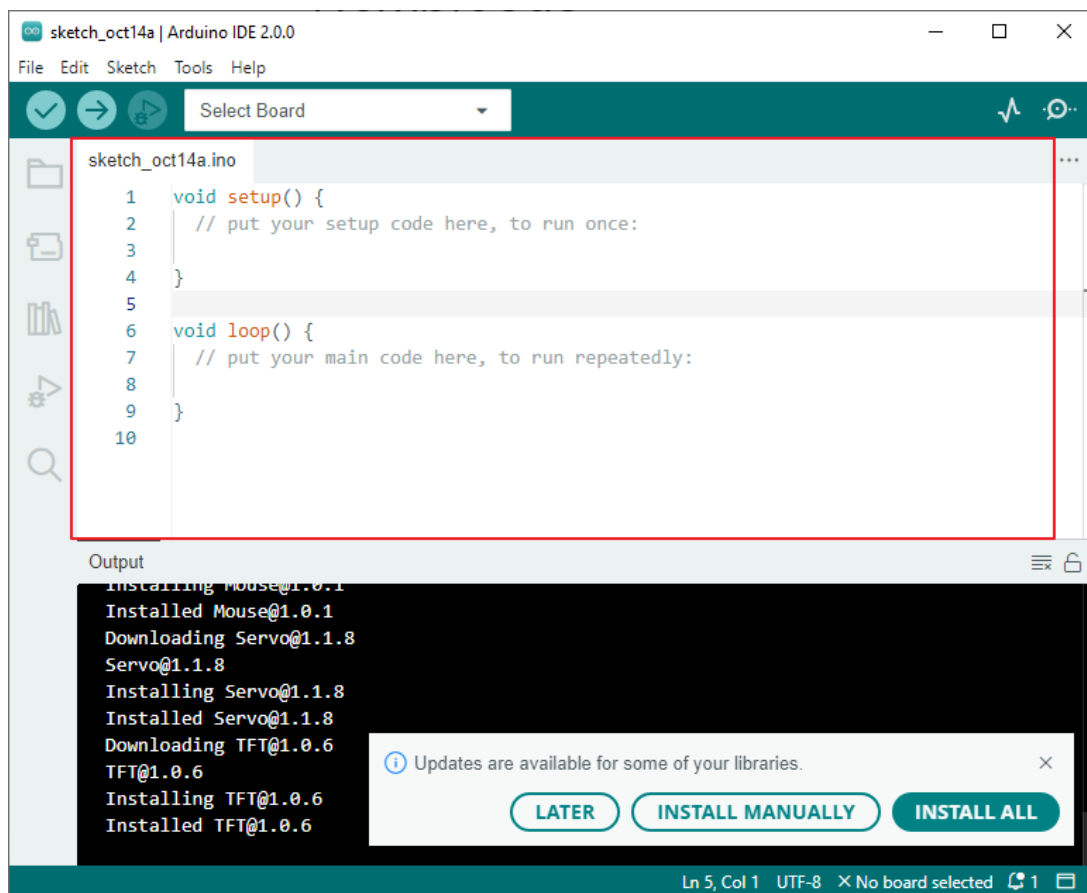


1. **Verify** : Compilez votre code. Tout problème de syntaxe sera signalé avec des erreurs.
2. **Upload** : Chargez le code sur votre carte. Lorsque vous cliquez sur le bouton, les LED RX et TX de la carte clignotent rapidement et ne s'arrêtent pas avant que le téléchargement soit terminé.
3. **Debug** : Pour une vérification des erreurs ligne par ligne.
4. **Select Board** : Configuration rapide de la carte et du port.
5. **Serial Plotter** : Vérifiez le changement de la valeur lue.
6. **Serial Monitor** : Cliquez sur le bouton et une fenêtre apparaîtra. Elle reçoit les données envoyées par votre carte de contrôle. C'est très utile pour le débogage.
7. **File** : Cliquez sur le menu et une liste déroulante apparaîtra, incluant la création, l'ouverture, l'enregistrement, la fermeture de fichiers, la configuration de certains paramètres, etc.
8. **Edit** : Cliquez sur le menu. Dans la liste déroulante, il y a des opérations d'édition comme **Cut**, **Copy**, **Paste**, **Find**, et ainsi de suite, avec leurs raccourcis correspondants.
9. **Sketch** : Comprend des opérations comme **Verify**, **Upload**, **Add** des fichiers, etc. La fonction plus importante est **Include Library** – où vous pouvez ajouter des bibliothèques.
10. **Tool** : Comprend certains outils – le plus fréquemment utilisé est Carte (la carte que vous utilisez) et Port (le port où se trouve votre carte). Chaque fois que vous souhaitez télécharger le code, vous devez les sélectionner ou les vérifier.

11. **Help** : Si vous êtes débutant, vous pouvez consulter les options sous le menu et obtenir l'aide dont vous avez besoin, y compris les opérations dans l'IDE, des informations d'introduction, le dépannage, l'explication du code, etc.
12. **Output Bar** : Changez l'onglet de sortie ici.
13. **Output Window** : Affiche les informations.
14. **Board and Port** : Ici, vous pouvez prévisualiser la carte et le port sélectionnés pour le téléchargement du code. Vous pouvez les sélectionner à nouveau par **Tools -> Board / Port** si l'un d'eux est incorrect.
15. La zone d'édition de l'IDE. Vous pouvez écrire du code ici.
16. **Sketchbook** : Pour gérer les fichiers de sketch.
17. **Board Manager** : Pour gérer le pilote de la carte.
18. **Library Manager** : Pour gérer vos fichiers de bibliothèque.
19. **Debug** : Aide au débogage du code.
20. **Search** : Recherchez des codes dans vos sketches.

### 2.3.3 Comment créer, ouvrir ou enregistrer un sketch ?

1. Lorsque vous ouvrez l'IDE Arduino pour la première fois ou créez un nouveau sketch, vous verrez une page comme celle-ci, où l'IDE Arduino crée un nouveau fichier pour vous, appelé « sketch ».



Ces fichiers de sketch ont un nom temporaire régulier, à partir duquel vous pouvez déterminer la date de création du fichier. `sketch_oct14a.ino` signifie le premier sketch du 14 octobre, `.ino` est le format de fichier de ce sketch.

2. Essayons maintenant de créer un nouveau sketch. Copiez le code suivant dans l'IDE Arduino pour remplacer le code original.

```

1 void setup() {
2     // put your setup code here, to run once:
3     pinMode(13,OUTPUT);
4 }
5
6 void loop() {
7     // put your main code here, to run repeatedly:
8     digitalWrite(13,HIGH);
9     delay(500);
10    digitalWrite(13,LOW);
11    delay(500);
12 }

```

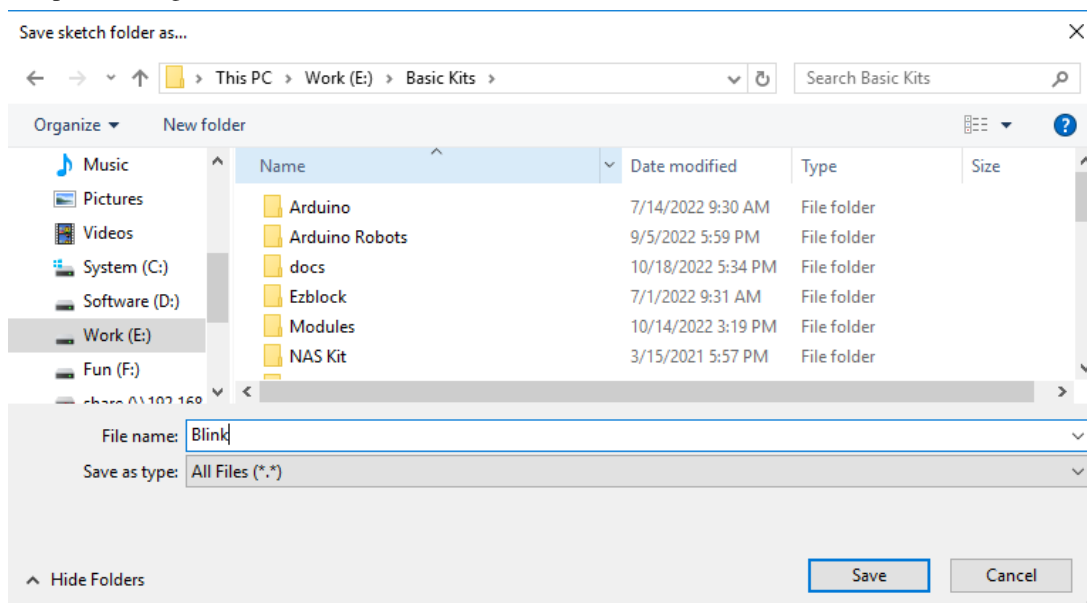
```

void setup() {
    // put your setup code here, to run once:
    pinMode(13,OUTPUT);
}

void loop() {
    // put your main code here, to run repeatedly:
    digitalWrite(13,HIGH);
    delay(500);
    digitalWrite(13,LOW);
    delay(500);
}

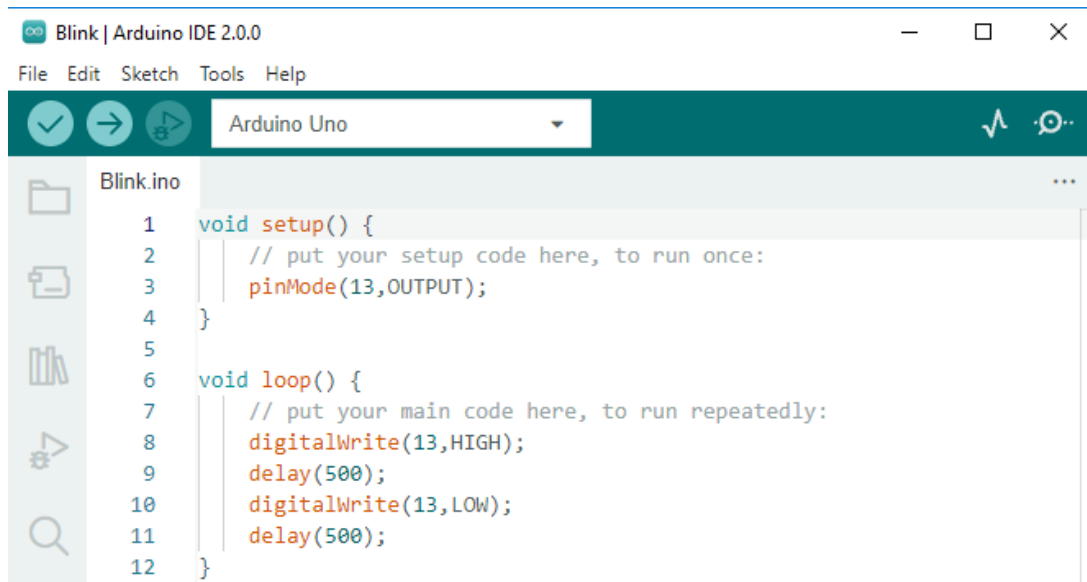
```

3. Appuyez sur Ctrl+S ou cliquez sur **File** -> **Save**. Le Sketch est enregistré dans : C:\Users\{votre\_utilisateur}\Documents\Arduino par défaut, vous pouvez le renommer ou trouver un nouveau chemin pour l'enregistrer.





4. Après l'enregistrement réussi, vous verrez que le nom dans l'IDE Arduino a été mis à jour.



Veuillez continuer avec la section suivante pour apprendre à télécharger ce sketch créé sur votre carte Arduino.

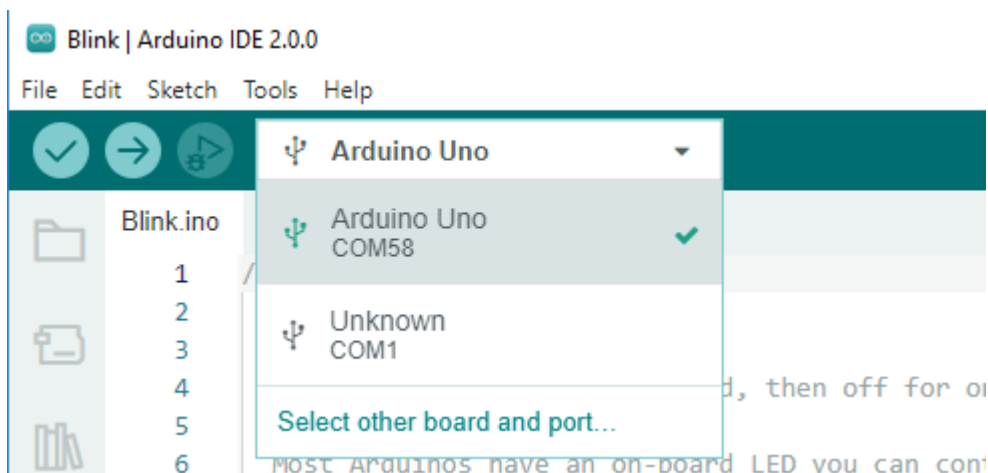
### 2.3.4 Comment télécharger un sketch sur la carte ?

Dans cette section, vous apprendrez comment télécharger le sketch créé précédemment sur la carte Arduino, ainsi que quelques considérations.

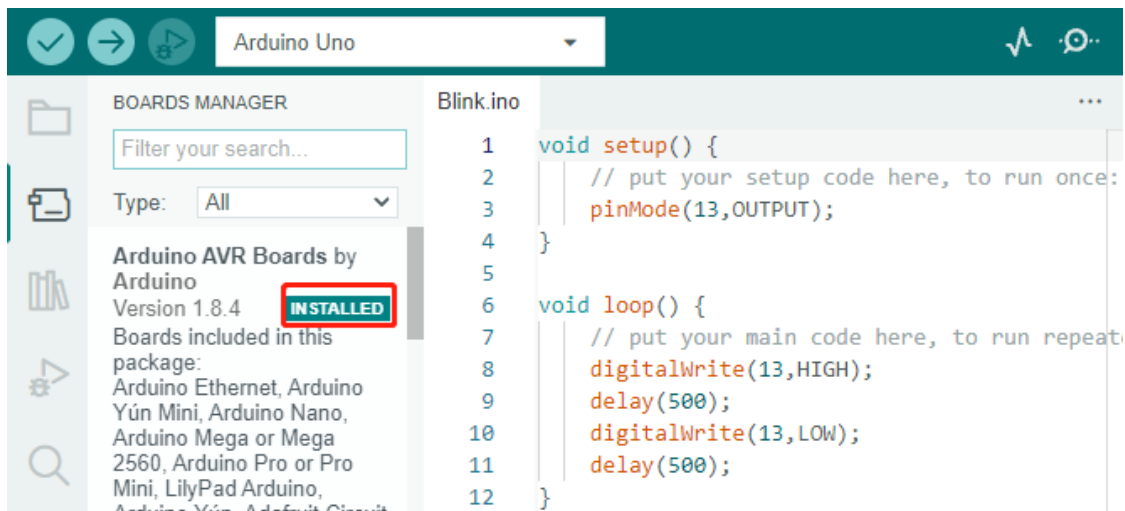
#### 1. Choisir la Carte et le port

Les cartes de développement Arduino sont généralement fournies avec un câble USB. Vous pouvez l'utiliser pour connecter la carte à votre ordinateur.

Sélectionnez la **Carte** et le **Port** corrects dans l'IDE Arduino. Normalement, les cartes Arduino sont automatiquement reconnues par l'ordinateur et se voient attribuer un port, que vous pouvez donc sélectionner ici.



Si votre carte est déjà branchée mais n'est pas reconnue, vérifiez si le logo **INSTALLED** apparaît dans la section **Arduino AVR Boards** du **Boards Manager**, si ce n'est pas le cas, faites défiler un peu vers le bas et cliquez sur **INSTALL**.



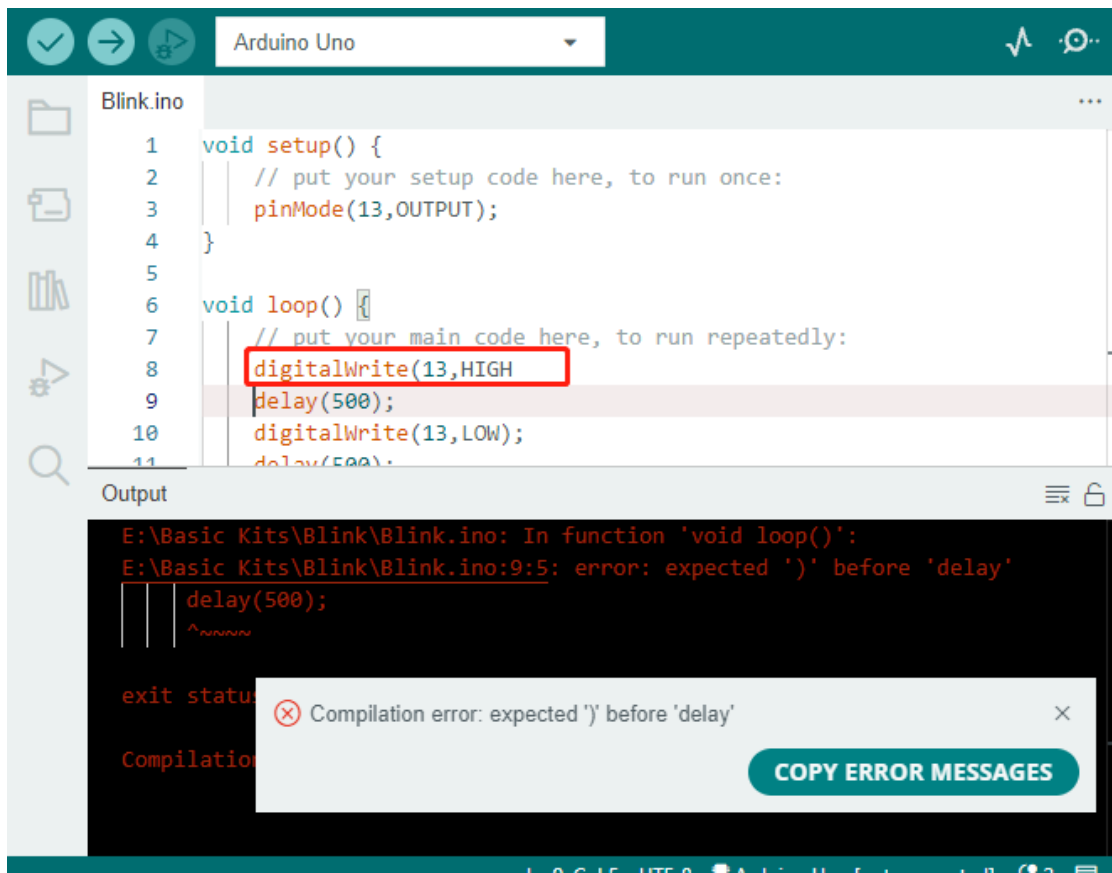
La réouverture de l'IDE Arduino et le rebranchement de la carte Arduino résoudront la plupart des problèmes. Vous pouvez également cliquer sur **Tools** -> **Board** ou **Port** pour les sélectionner.

## 2. Vérifier le Sketch

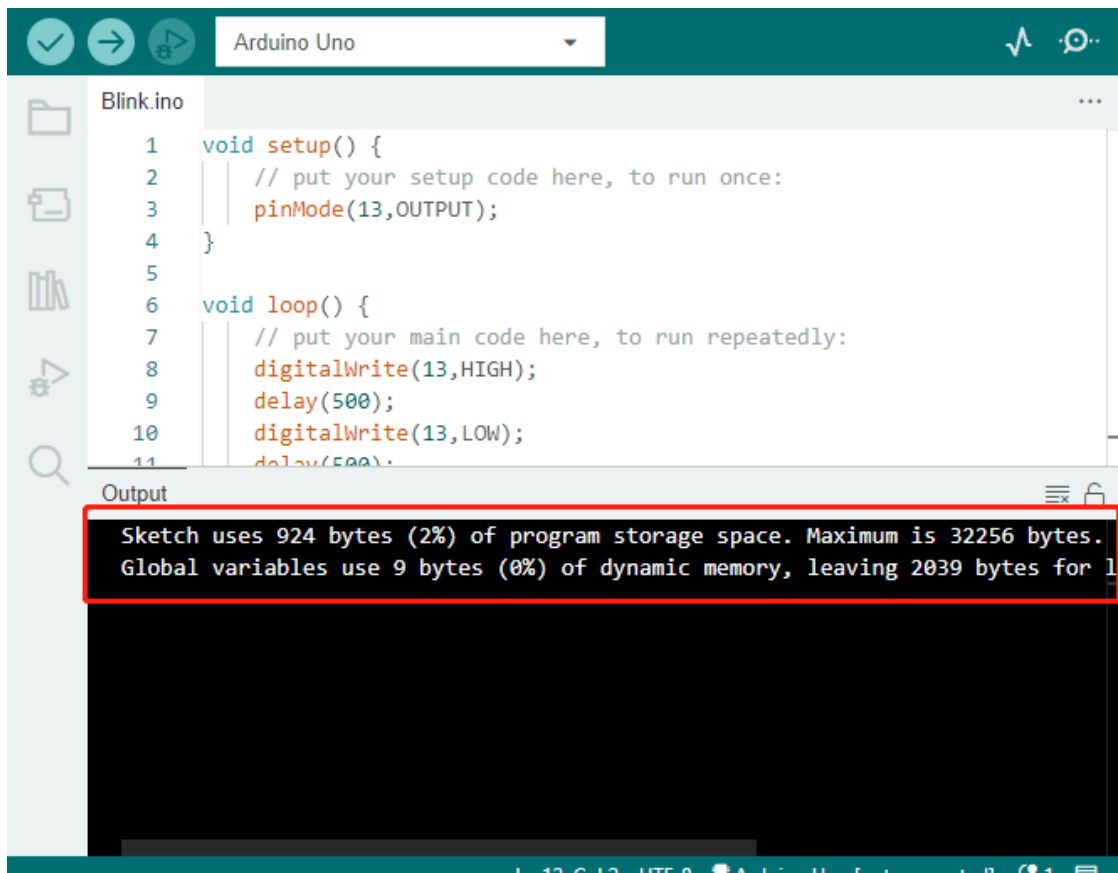
Après avoir cliqué sur le bouton Vérifier, le sketch sera compilé pour voir s'il y a des erreurs.



Vous pouvez l'utiliser pour trouver des erreurs si vous supprimez des caractères ou tapez quelques lettres par erreur. Depuis la barre de messages, vous pouvez voir où et quel type d'erreurs se sont produites.

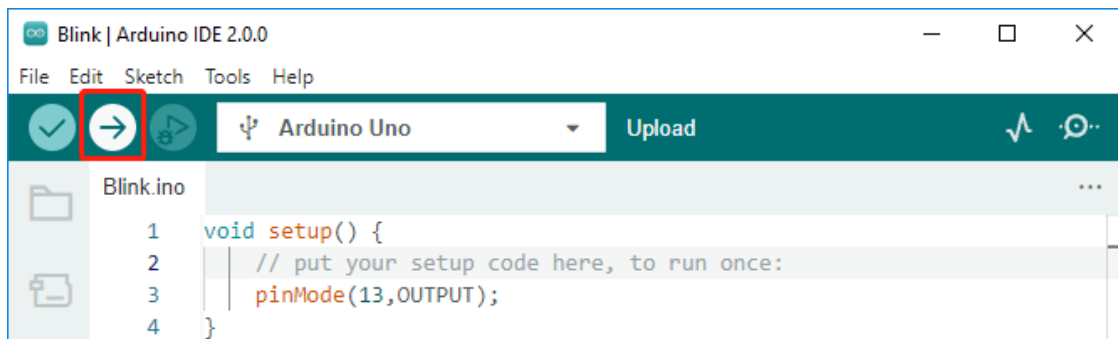


S'il n'y a pas d'erreurs, vous verrez un message comme celui ci-dessous.

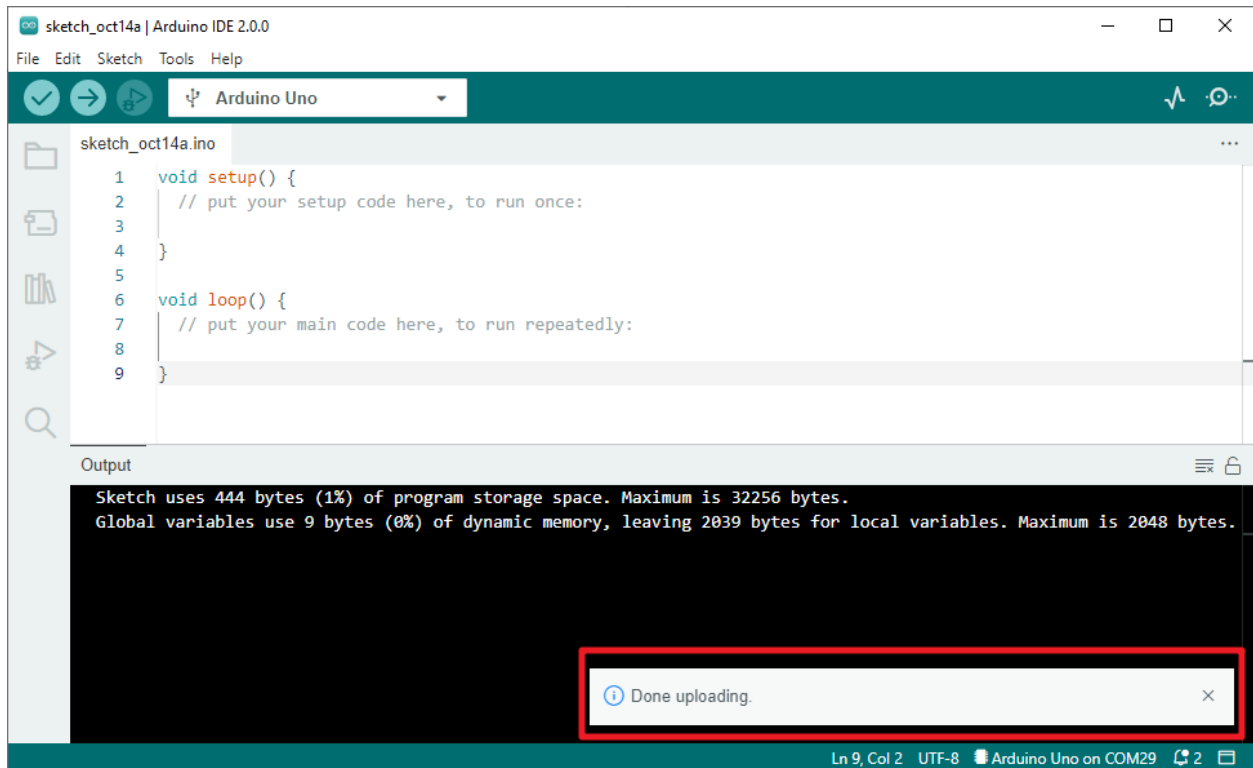


### 3. Télécharger le sketch

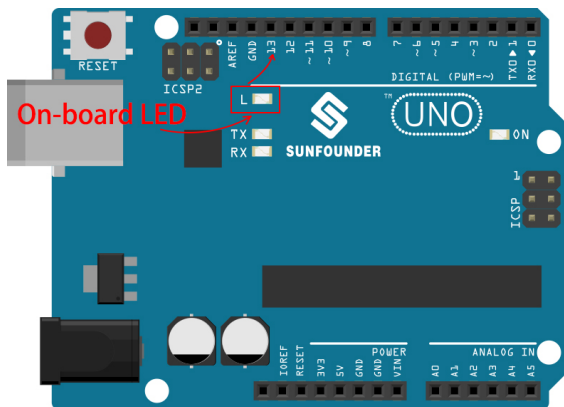
Après avoir complété les étapes ci-dessus, cliquez sur le bouton **Upload** pour charger ce sketch sur la carte.



Si le téléchargement est réussi, vous pourrez voir l'invite suivante.



En même temps, la LED intégrée à la carte clignotera.



La carte Arduino exécutera automatiquement le sketch après l'application de l'alimentation une fois le sketch téléchargé. Le programme en cours peut être écrasé en téléchargeant un nouveau sketch.

### 2.3.5 Structure du Programme Arduino

Jetons un coup d'œil au nouveau fichier de sketch. Bien qu'il ait quelques lignes de code, c'est en réalité un sketch « vide ». Le téléchargement de ce sketch sur la carte de développement ne provoquera aucun effet.

```
void setup() {
  // put your setup code here, to run once:
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
void loop() {  
  // put your main code here, to run repeatedly:  
  
}
```

Si nous supprimons `setup()` et `loop()` pour rendre le sketch vraiment `blank`, vous constaterez qu'il ne passe pas la vérification. Ils sont l'équivalent du squelette humain, et sont indispensables.

Lors de la création d'un sketch, `setup()` est exécuté en premier, et le code à l'intérieur (dans `{}`) est exécuté après que la carte soit alimentée ou réinitialisée et seulement une fois. `loop()` est utilisé pour écrire la fonction principale, et le code à l'intérieur s'exécute en boucle après l'exécution de `setup()`.

Pour mieux comprendre `setup()` et `loop()`, utilisons quatre sketches. Leur but est de faire clignoter la LED intégrée de l'Arduino. Veuillez exécuter chaque expérience à tour de rôle et enregistrer leurs effets spécifiques.

— Sketch 1 : Faire clignoter la LED intégrée en continu.

```
void setup() {  
  // put your setup code here, to run once:  
  pinMode(13,OUTPUT);  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
  digitalWrite(13,HIGH);  
  delay(500);  
  digitalWrite(13,LOW);  
  delay(500);  
}
```

— Sketch 2 : Faire clignoter la LED intégrée une seule fois.

```
void setup() {  
  // put your setup code here, to run once:  
  pinMode(13,OUTPUT);  
  digitalWrite(13,HIGH);  
  delay(500);  
  digitalWrite(13,LOW);  
  delay(500);  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

— Sketch 3 : Faire clignoter lentement la LED intégrée une fois, puis rapidement.

```
void setup() {  
  // put your setup code here, to run once:  
  pinMode(13,OUTPUT);  
  digitalWrite(13,HIGH);  
  delay(1000);  
  digitalWrite(13,LOW);  
  delay(1000);  
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
void loop() {  
  // put your main code here, to run repeatedly:  
  digitalWrite(13,HIGH);  
  delay(200);  
  digitalWrite(13,LOW);  
  delay(200);  
}
```

— Sketch 4 : Signaler une erreur.

```
void setup() {  
  // put your setup code here, to run once:  
  pinMode(13,OUTPUT);  
}  
  
digitalWrite(13,HIGH);  
delay(1000);  
digitalWrite(13,LOW);  
delay(1000);  
  
void loop() {  
  // put your main code here, to run repeatedly:  
}
```

Avec l'aide de ces sketches, nous pouvons résumer plusieurs caractéristiques de `setup-loop`.

- `loop()` s'exécute de manière répétée après que la carte soit alimentée.
- `setup()` s'exécute une seule fois après que la carte soit alimentée.
- Après que la carte soit alimentée, `setup()` s'exécute en premier, suivi par `loop()`.
- Le code doit être écrit à l'intérieur du cadre `{ }` de `setup()` ou `loop()`, en dehors du cadre, il y aura une erreur.

**Note :** Des instructions telles que `digitalWrite(13,HIGH)` sont utilisées pour contrôler la LED intégrée, et nous parlerons de leur utilisation en détail dans les chapitres suivants.

## 2.3.6 Règle de Rédaction d'un Sketch

Si vous demandez à un ami d'allumer les lumières pour vous, vous pouvez dire « Allume les lumières. », ou « Lumières allumées, mon pote. », vous pouvez utiliser n'importe quel ton de voix que vous souhaitez.

Cependant, si vous voulez que la carte Arduino fasse quelque chose pour vous, vous devez suivre les règles de rédaction des programmes Arduino pour taper les commandes.

Ce chapitre contient les règles de base du langage Arduino et vous aidera à comprendre comment traduire le langage naturel en code.

Bien sûr, c'est un processus qui prend du temps pour se familiariser, et c'est aussi la partie du processus la plus sujette aux erreurs pour les débutants, donc si vous faites souvent des erreurs, c'est normal, essayez juste quelques fois de plus.

### Point-virgule ;

Tout comme l'écriture d'une lettre, où vous mettez un point à la fin de chaque phrase comme fin, le langage Arduino vous demande d'utiliser ; pour indiquer à la carte la fin de la commande.

Prenons l'exemple familier du « clignotement de la LED intégrée ». Un sketch correct devrait ressembler à ceci.

Exemple :

```
void setup() {  
    // put your setup code here, to run once:  
    pinMode(13,OUTPUT);  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
    digitalWrite(13,HIGH);  
    delay(500);  
    digitalWrite(13,LOW);  
    delay(500);  
}
```

Ensuite, examinons les deux sketches suivants et devinons s'ils peuvent être correctement reconnus par Arduino avant de les exécuter.

Sketch A :

```
void setup() {  
    // put your setup code here, to run once:  
    pinMode(13,OUTPUT);  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
    digitalWrite(13,HIGH)  
    delay(500)  
    digitalWrite(13,LOW)  
    delay(500)  
}
```

Sketch B :

```
void setup() {  
    // put your setup code here, to run once:  
    pinMode(13,OUTPUT);  
}  
  
void loop() {  
    // put your main code here, to run repeatedly:  
    digitalWrite(13,  
HIGH); delay  
    (500  
    );  
    digitalWrite(13,
```

(suite sur la page suivante)



(suite de la page précédente)

```
LOW);
    delay(500)
;
}
```

Le résultat est que le **Sketch A** signale une erreur et le **Sketch B** fonctionne.

- Les erreurs dans le **Sketch A** sont dues à l'absence de ; et bien qu'il semble normal, l'Arduino ne peut pas le lire.
- Le **Sketch B**, bien qu'il paraisse anti-humain, est en fait compréhensible pour l'Arduino. En effet, l'indentation, les sauts de ligne et les espaces dans les instructions sont des éléments qui n'existent pas dans les programmes Arduino, donc pour le compilateur Arduino, il apparaît identique à l'exemple.

Cependant, veuillez ne pas écrire votre code comme le **Sketch B**, car ce sont généralement des personnes naturelles qui écrivent et consultent le code, donc ne vous compliquez pas la tâche.

### Accolades {}

Les {} sont un composant principal du langage de programmation Arduino, et elles doivent apparaître par paires. Une meilleure convention de programmation est d'insérer une structure nécessitant des accolades en tapant l'accolade droite juste après avoir tapé l'accolade gauche, puis en déplaçant le curseur entre les accolades pour insérer l'instruction.

### Commentaire //

Le commentaire est la partie du sketch que le compilateur ignore. Ils sont généralement utilisés pour expliquer aux autres comment fonctionne le programme.

Si nous écrivons deux barres obliques adjacentes dans une ligne de code, le compilateur ignorera tout jusqu'à la fin de la ligne.

Si nous créons un nouveau sketch, il est livré avec deux commentaires, et si nous supprimons ces deux commentaires, le sketch ne sera en aucun cas affecté.

```
void setup() {
    // put your setup code here, to run once:
}

void loop() {
    // put your main code here, to run repeatedly:
}
```

Le commentaire est très utile en programmation, et plusieurs utilisations courantes sont listées ci-dessous.

- Utilisation A : Se dire à soi-même ou à d'autres ce que cette section de code fait.

```
void setup() {
    pinMode(13,OUTPUT); //Set pin 13 to output mode, it controls the onboard LED
}

void loop() {
    digitalWrite(13,HIGH); // Activate the onboard LED by setting pin 13 high
    delay(500); // Status quo for 500 ms
    digitalWrite(13,LOW); // Turn off the onboard LED
}
```

(suite sur la page suivante)

```
    delay(500); // Status quo for 500 ms
}
```

- Utilisation B : Invalider temporairement certaines déclarations (sans les supprimer) et les décommenter lorsque vous avez besoin de les utiliser, pour ne pas avoir à les réécrire. Cela est très utile lors du débogage du code et de la tentative de localisation des erreurs du programme.

```
void setup() {
    pinMode(13,OUTPUT);
    // digitalWrite(13,HIGH);
    // delay(1000);
    // digitalWrite(13,LOW);
    // delay(1000);
}

void loop() {
    digitalWrite(13,HIGH);
    delay(200);
    digitalWrite(13,LOW);
    delay(200);
}
```

**Note :** Utilisez le raccourci Ctrl+/ pour vous aider à commenter ou décommenter rapidement votre code.

### Commentaire /\*\*/

Comme // pour les commentaires. Ce type de commentaire peut s'étendre sur plus d'une ligne, et une fois que le compilateur lit /\*, il ignore tout ce qui suit jusqu'à ce qu'il rencontre \*/.

Exemple 1 :

```
/* Blink */

void setup() {
    pinMode(13,OUTPUT);
}

void loop() {
    /*
    The following code will blink the onboard LED
    You can modify the number in delay() to change the blinking frequency
    */
    digitalWrite(13,HIGH);
    delay(500);
    digitalWrite(13,LOW);
    delay(500);
}
```

## #define

C'est un outil utile de C++.

```
#define identifieur token-string
```

Le compilateur remplace automatiquement `identifieur` par `token-string` lorsqu'il le lit, ce qui est généralement utilisé pour des définitions constantes.

Par exemple, voici un sketch qui utilise `define`, ce qui améliore la lisibilité du code.

```
#define ONBOARD_LED 13
#define DELAY_TIME 500

void setup() {
    pinMode(ONBOARD_LED, OUTPUT);
}

void loop() {
    digitalWrite(ONBOARD_LED, HIGH);
    delay(DELAY_TIME);
    digitalWrite(ONBOARD_LED, LOW);
    delay(DELAY_TIME);
}
```

Pour le compilateur, cela ressemble réellement à ceci.

```
void setup() {
    pinMode(13, OUTPUT);
}

void loop() {
    digitalWrite(13, HIGH);
    delay(500);
    digitalWrite(13, LOW);
    delay(500);
}
```

Nous pouvons voir que l'identifiant est remplacé et n'existe pas dans le programme. Par conséquent, il y a plusieurs mises en garde lors de son utilisation.

1. Une `token-string` ne peut être modifiée manuellement et ne peut pas être convertie en d'autres valeurs par arithmétique dans le programme.
2. Évitez d'utiliser des symboles tels que `;`. Par exemple.

```
#define ONBOARD_LED 13;

void setup() {
    pinMode(ONBOARD_LED, OUTPUT);
}

void loop() {
    digitalWrite(ONBOARD_LED, HIGH);
}
```

Le compilateur le reconnaîtra comme suit, ce qui sera signalé comme une erreur.

```
void setup() {  
  pinMode(13,OUTPUT);  
}  
  
void loop() {  
  digitalWrite(13,HIGH);  
}
```

---

**Note :** Une convention de nommage pour `#define` est de mettre en majuscule l'identifiant pour éviter la confusion avec les variables.

---

### 2.3.7 Variable

La variable est l'un des outils les plus puissants et cruciaux dans un programme. Elle nous aide à stocker et à appeler des données dans nos programmes.

Le fichier de sketch suivant utilise des variables. Il stocke les numéros de broche de la LED intégrée dans la variable `ledPin` et un nombre « 500 » dans la variable `delayTime`.

```
int ledPin = 13;  
int delayTime = 500;  
  
void setup() {  
  pinMode(ledPin,OUTPUT);  
}  
  
void loop() {  
  digitalWrite(ledPin,HIGH);  
  delay(delayTime);  
  digitalWrite(ledPin,LOW);  
  delay(delayTime);  
}
```

Attendez, est-ce un doublon de ce que fait `#define` ? La réponse est NON.

- Le rôle de `#define` est de remplacer simplement et directement le texte, il n'est pas considéré par le compilateur comme faisant partie du programme.
- Une variable, en revanche, existe au sein du programme et est utilisée pour stocker et appeler des valeurs. Une variable peut également modifier sa valeur dans le programme, ce qu'un `define` ne peut pas faire.

Le fichier de sketch ci-dessous ajoute lui-même à la variable et cela fera clignoter la LED intégrée plus longtemps après chaque clignotement.

```
int ledPin = 13;  
int delayTime = 500;  
  
void setup() {  
  pinMode(ledPin,OUTPUT);  
}  
  
void loop() {  
  digitalWrite(ledPin,HIGH);  
  delay(delayTime);
```

(suite sur la page suivante)

(suite de la page précédente)

```
digitalWrite(ledPin,LOW);
delay(delayTime);
delayTime = delayTime+200; //Each execution increments the value by 200
}
```

## Déclarer une variable

Déclarer une variable signifie créer une variable.

Pour déclarer une variable, vous avez besoin de deux choses : le type de données et le nom de la variable. Le type de données doit être séparé de la variable par un espace, et la déclaration de la variable doit être terminée par un ;.

Utilisons cette variable comme exemple.

```
int delayTime;
```

### Type de Donnée

Ici, `int` est un type de donnée appelé type entier, qui peut être utilisé pour stocker des entiers de -32768 à 32766. Il ne peut pas être utilisé pour stocker des décimales.

Les variables peuvent contenir différents types de données autres que des entiers. Le langage Arduino (qui, rappelons-le, est du C++) offre un support intégré pour quelques-uns d'entre eux (seuls les plus fréquemment utilisés et utiles sont listés ici) :

- `float` : Stocke un nombre décimal, par exemple 3.1415926.
- `byte` : Peut contenir des nombres de 0 à 255.
- `boolean` : Ne détient que deux valeurs possibles, `True` ou `False`, même s'il occupe un octet en mémoire.
- `char` : Contient un nombre de -127 à 127. Comme il est marqué comme un `char`, le compilateur essaiera de le faire correspondre à un caractère du .
- `string` : Peut stocker une chaîne de caractères, par exemple Halloween.

### Nom de Variable

Vous pouvez donner à la variable le nom que vous voulez, comme `i`, `apple`, `Bruce`, `R2D2`, `Sectumsempra`, mais il y a quelques règles de base à suivre.

1. Décrire à quoi elle sert. Ici, j'ai nommé la variable `delayTime`, donc vous pouvez facilement comprendre ce qu'elle fait. Ça fonctionnerait si je nommais la variable `barryAllen`, mais cela confondrait la personne qui regarde le code.
2. Utiliser une nomenclature régulière. Vous pouvez utiliser CamelCase comme je l'ai fait, avec le T initial dans `delayTime` pour qu'il soit facile de voir que la variable est composée de deux mots. Aussi, vous pouvez utiliser UnderScoreCase pour écrire la variable comme `delay_time`. Cela n'affecte pas le fonctionnement du programme, mais cela aiderait le programmeur à lire le code si vous utilisez la nomenclature que vous préférez.
3. Ne pas utiliser des mots-clés. De manière similaire à ce qui se passe lorsque nous tapons « `int` », l'IDE Arduino le coloriera pour vous rappeler que c'est un mot avec un but spécial et ne peut pas être utilisé comme nom de variable. Changez le nom de la variable si elle est colorée.
4. Les symboles spéciaux ne sont pas autorisés. Par exemple, l'espace, `#`, `$`, `/`, `+`, `%`, etc. La combinaison de lettres anglaises (sensibles à la casse), de soulignements et de nombres (mais les nombres ne peuvent pas être utilisés comme premier caractère d'un nom de variable) est assez riche.

### Attribuer une valeur à une variable

Une fois que nous avons déclaré la variable, il est temps de stocker les données. Nous utilisons l'opérateur d'affectation (c'est-à-dire `=`) pour mettre de la valeur dans la variable.

Nous pouvons attribuer des valeurs à la variable dès que nous la déclarons.

```
int delayTime = 500;
```

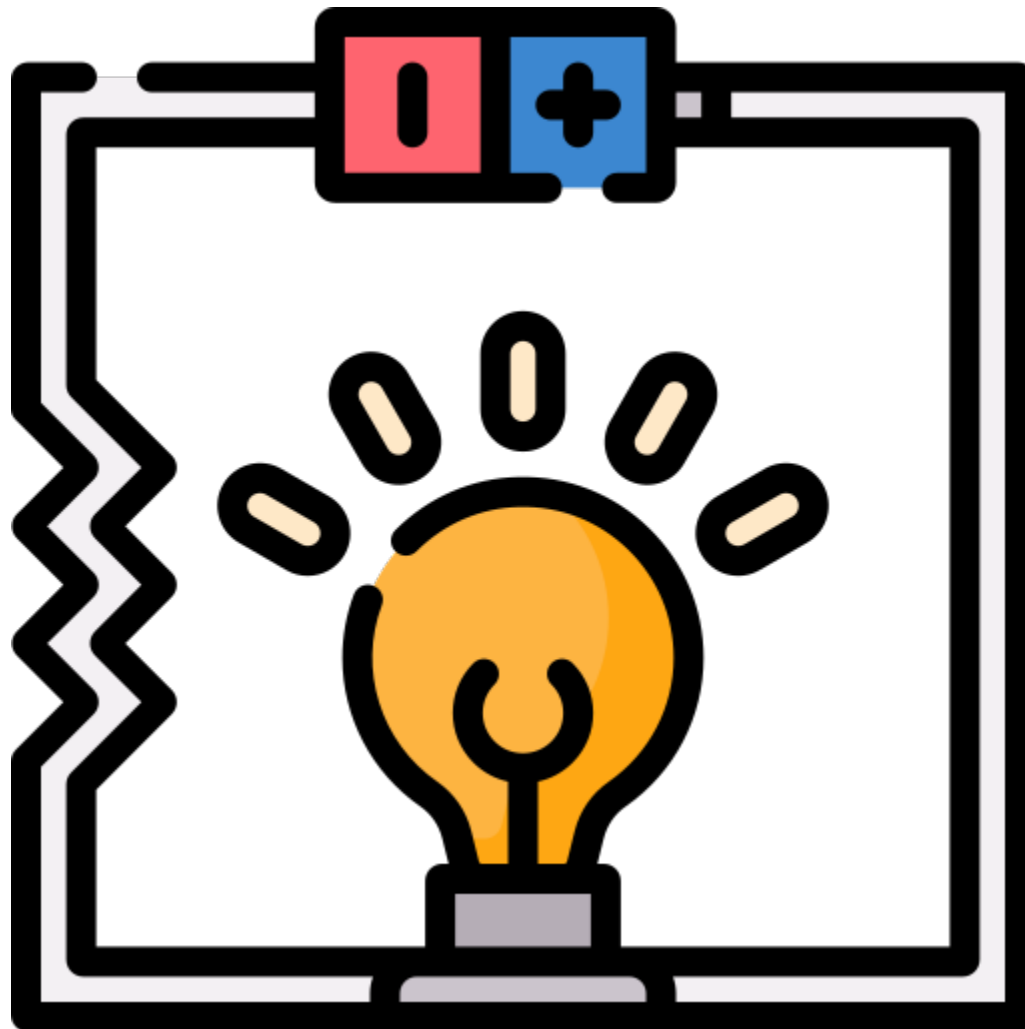
Il est également possible de lui attribuer une nouvelle valeur à un moment donné.

```
int delayTime; // no value  
delayTime = 500; // value is 500  
delayTime = delayTime +200; // value is 700
```

## 2.3.8 Comment Construire le Circuit

Beaucoup des choses que vous utilisez tous les jours sont alimentées par l'électricité, comme les lumières de votre maison et l'ordinateur sur lequel vous lisez.

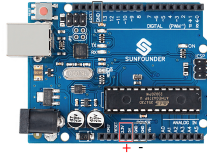
Pour utiliser l'électricité, vous devez construire un circuit électrique. Fondamentalement, un circuit est un chemin à travers lequel l'électricité circule, ou un circuit électronique, et est composé de dispositifs et de composants électriques (appareils) qui sont connectés d'une certaine manière, tels que des résistances, des condensateurs, des alimentations et des interrupteurs.



Un circuit est un chemin fermé dans lequel les électrons se déplacent pour créer un courant électrique. Pour faire circuler le courant, il doit y avoir un chemin conducteur entre la borne positive de l'alimentation et la borne négative, ce qui est

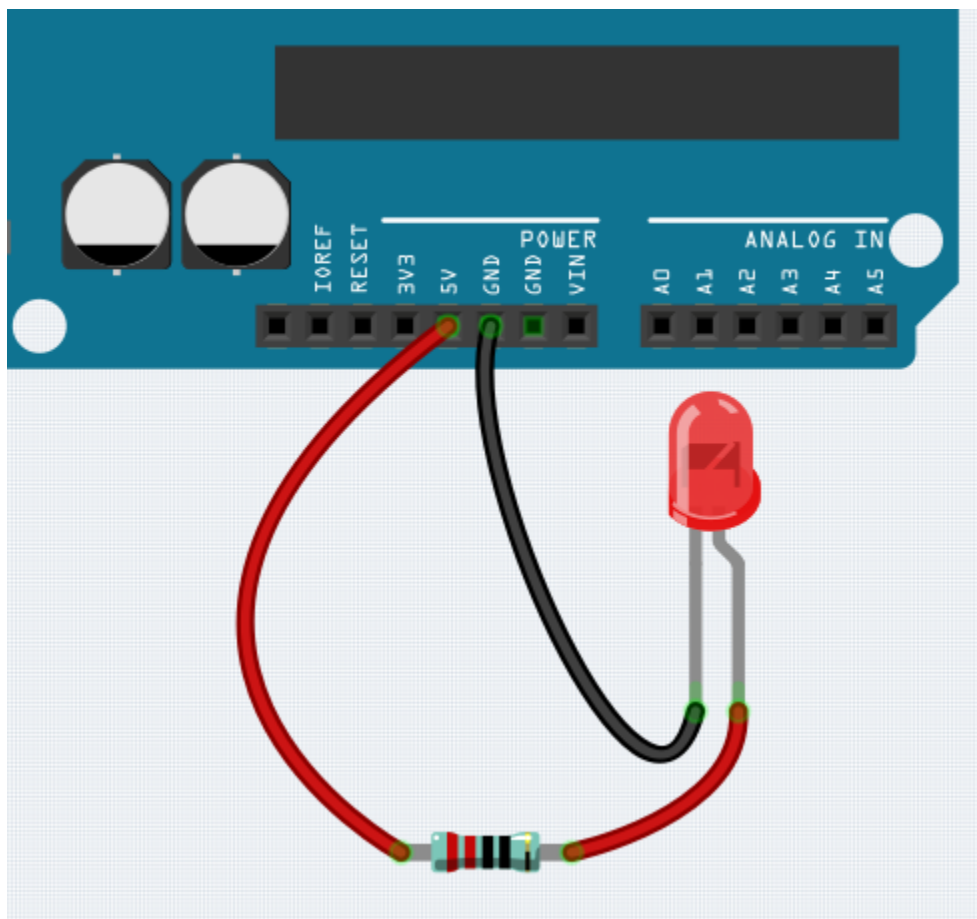
appelé un circuit fermé (s'il est interrompu, on parle de circuit ouvert).

La carte Arduino dispose de quelques broches de sortie d'alimentation (positives) et de quelques broches de masse (négatives). Vous pouvez utiliser ces broches comme les côtés positif et négatif de l'alimentation en branchant la source d'alimentation sur la carte.



Avec l'électricité, vous pouvez créer des œuvres avec de la lumière, du son et du mouvement. Vous pouvez allumer une LED en connectant la broche longue à la borne positive et la broche courte à la borne négative. La LED se détruira très rapidement si vous faites cela, donc vous devez ajouter une résistance de 220\* dans le circuit pour la protéger.

Le circuit qu'ils forment est montré ci-dessous.



Vous pouvez vous poser des questions cette fois-ci : comment construire ce circuit ? Tenir les fils à la main ou coller les broches et les fils ?

Dans cette situation, les plaques d'essai sans soudure seront vos alliées les plus fortes.

### Bonjour, Breadboard !

Une breadboard (plaque d'essai) est une plaque en plastique rectangulaire avec un tas de petits trous. Ces trous nous permettent d'insérer facilement des composants électroniques et de construire des circuits électroniques. Les breadboards ne fixent pas de manière permanente les composants électroniques, donc nous pouvons facilement réparer un circuit et recommencer si quelque chose ne va pas.

**Note :** Il n'est pas nécessaire d'avoir des outils spéciaux pour utiliser les breadboards. Cependant, de nombreux composants électroniques sont très petits, et une paire de pinces peut nous aider à mieux saisir les petites pièces.

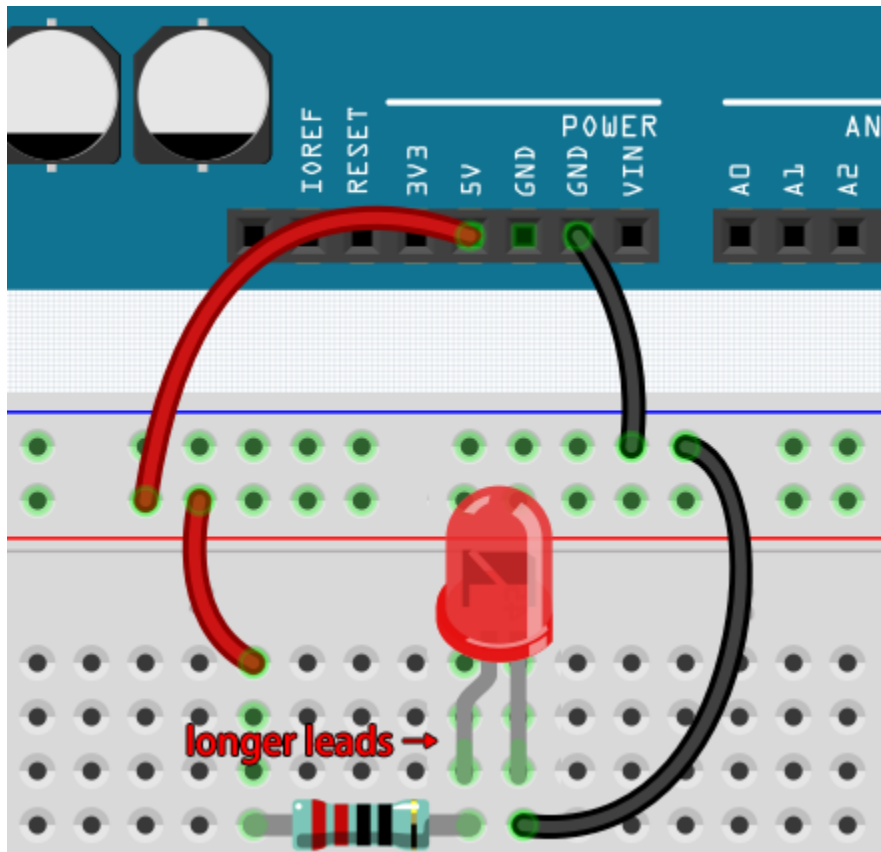
Sur Internet, nous pouvons trouver beaucoup d'informations sur les breadboards.

- [Comment Utiliser une Breadboard - Science Buddies](#)
- [Qu'est-ce qu'une BREADBOARD ? - Makezine](#)

Voici quelques choses que vous devez savoir sur les breadboards (planches à pain).

1. Chaque groupe de demi-rangées (comme la colonne A-E dans la rangée 1 ou la colonne F-J dans la rangée 3) est connecté. Par conséquent, si un signal électrique entre par A1, il peut sortir par B1, C1, D1, E1, mais pas par F1 ou A2.
2. Dans la plupart des cas, les deux côtés de la breadboard sont utilisés comme bus d'alimentation, et les trous de chaque colonne (environ 50 trous) sont connectés ensemble. En règle générale, les alimentations positives sont connectées aux trous près du fil rouge, et les alimentations négatives aux trous près du fil bleu.
3. Dans un circuit, le courant circule du pôle positif au pôle négatif après avoir traversé la charge. Dans ce cas, un court-circuit peut se produire.

**Suivons la direction du courant pour construire le circuit !**



1. Dans ce circuit, nous utilisons la broche 5V de la carte pour alimenter la LED. Utilisez un câble de liaison mâle à mâle (M2M) pour le connecter au bus d'alimentation rouge.



2. Pour protéger la LED, le courant doit passer par une résistance de 220 ohms. Connectez une extrémité (n'importe laquelle) de la résistance au bus d'alimentation rouge, et l'autre extrémité à une rangée libre de la breadboard.

---

**Note :** La bague de couleur de la résistance de 220 ohms est rouge, rouge, noire, noire et marron.

---

3. Si vous prenez la LED, vous verrez que l'une de ses broches est plus longue que l'autre. Connectez la broche la plus longue à la même rangée que la résistance, et la broche la plus courte à une autre rangée.

---

**Note :** La broche la plus longue est l'anode, qui représente le côté positif du circuit ; la broche la plus courte est la cathode, qui représente le côté négatif.

L'anode doit être connectée à la broche GPIO via une résistance ; la cathode doit être connectée à la broche GND.

---

4. Utilisez un câble de liaison mâle à mâle (M2M) pour connecter la broche courte de la LED au bus d'alimentation négatif de la breadboard.
5. Connectez la broche GND de la carte au bus d'alimentation négatif à l'aide d'un câble de liaison.

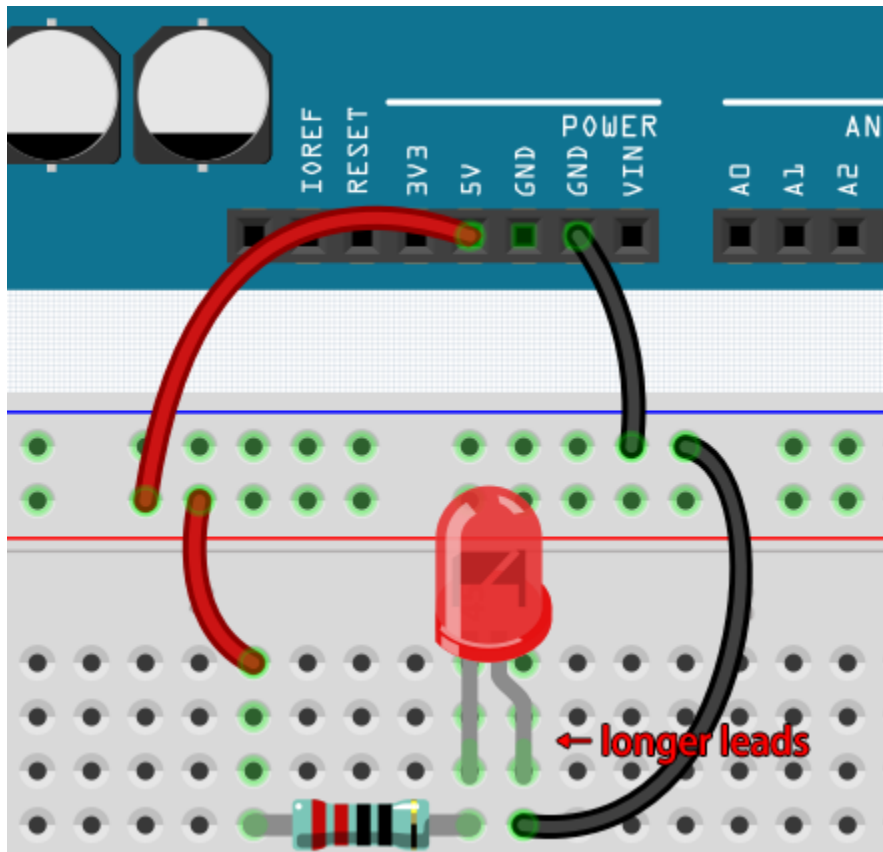
### Attention aux courts-circuits

Les courts-circuits peuvent se produire lorsque deux composants qui ne devraient pas être connectés le sont « accidentellement ». Ce kit comprend des résistances, des transistors, des condensateurs, des LED, etc. qui ont de longues broches métalliques pouvant se heurter les unes aux autres et provoquer un court-circuit. Certains circuits sont simplement empêchés de fonctionner correctement lorsqu'un court-circuit se produit. Occasionnellement, un court-circuit peut endommager de manière permanente des composants, en particulier entre l'alimentation et le bus de masse, provoquant un échauffement du circuit, la fusion du plastique sur la breadboard et même la combustion des composants !

Par conséquent, assurez-vous toujours que les broches de tous les composants électroniques sur la breadboard ne se touchent pas entre elles.

### Orientation du circuit

Il existe une orientation pour les circuits, et cette orientation joue un rôle significatif dans certains composants électroniques. Il y a des dispositifs avec polarité, ce qui signifie qu'ils doivent être connectés correctement en fonction de leurs pôles positif et négatif. Les circuits construits avec une mauvaise orientation ne fonctionneront pas correctement.



Si vous inversez la LED dans ce simple circuit que nous avons construit plus tôt, vous constaterez qu'elle ne fonctionne plus.

En revanche, certains dispositifs n'ont pas de direction, comme les résistances dans ce circuit, donc vous pouvez les inverser sans affecter le fonctionnement normal des LED.

La plupart des composants et modules avec des étiquettes telles que « + », « - », « GND », « VCC » ou ayant des broches de longueurs différentes doivent être connectés au circuit d'une manière spécifique.

### Protection du circuit

Le courant est le taux auquel les électrons passent devant un point dans un circuit électrique complet. À sa base, le courant = flux. Un ampère, ou amp, est l'unité internationale utilisée pour mesurer le courant. Il exprime la quantité d'électrons (parfois appelée « charge électrique ») passant devant un point dans un circuit sur une période donnée.

La force motrice (tension) derrière le flux de courant est appelée tension et est mesurée en volts (V).

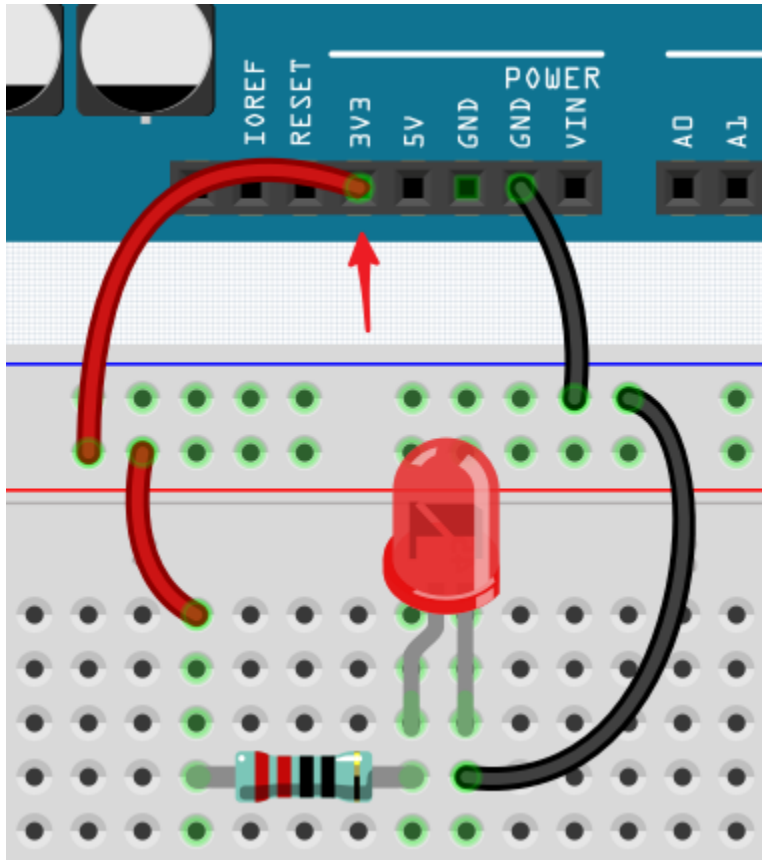
La résistance (R) est la propriété du matériau qui restreint le flux de courant, et elle est mesurée en ohms (Ω).

Selon la loi d'Ohm (tant que la température reste constante), le courant, la tension et la résistance sont proportionnels. Le courant d'un circuit est proportionnel à sa tension et inversement proportionnel à sa résistance.

Par conséquent, courant (I) = tension (V) / résistance (R).

— [Loi d'Ohm - Wikipedia](#)

À propos de la loi d'Ohm, nous pouvons faire une expérience simple.



En changeant le fil reliant 5V à 3.3V, la LED devient plus faible. Si vous changez la résistance de 220 ohms à 1000 ohms (anneau de couleur : marron, noir, noir, marron et marron), vous remarquerez que la LED devient plus faible qu'auparavant. Plus la résistance est grande, plus la LED est faible.

**Note :** Pour une introduction aux résistances et comment calculer les valeurs de résistance, voir [Résistance](#).

La plupart des modules emballés ne nécessitent qu'un accès à la tension appropriée (généralement 3,3V ou 5V), comme le module ultrasonique.

Cependant, dans vos circuits auto-construits, vous devez être conscient de la tension d'alimentation et de l'utilisation des résistances pour les dispositifs électriques.

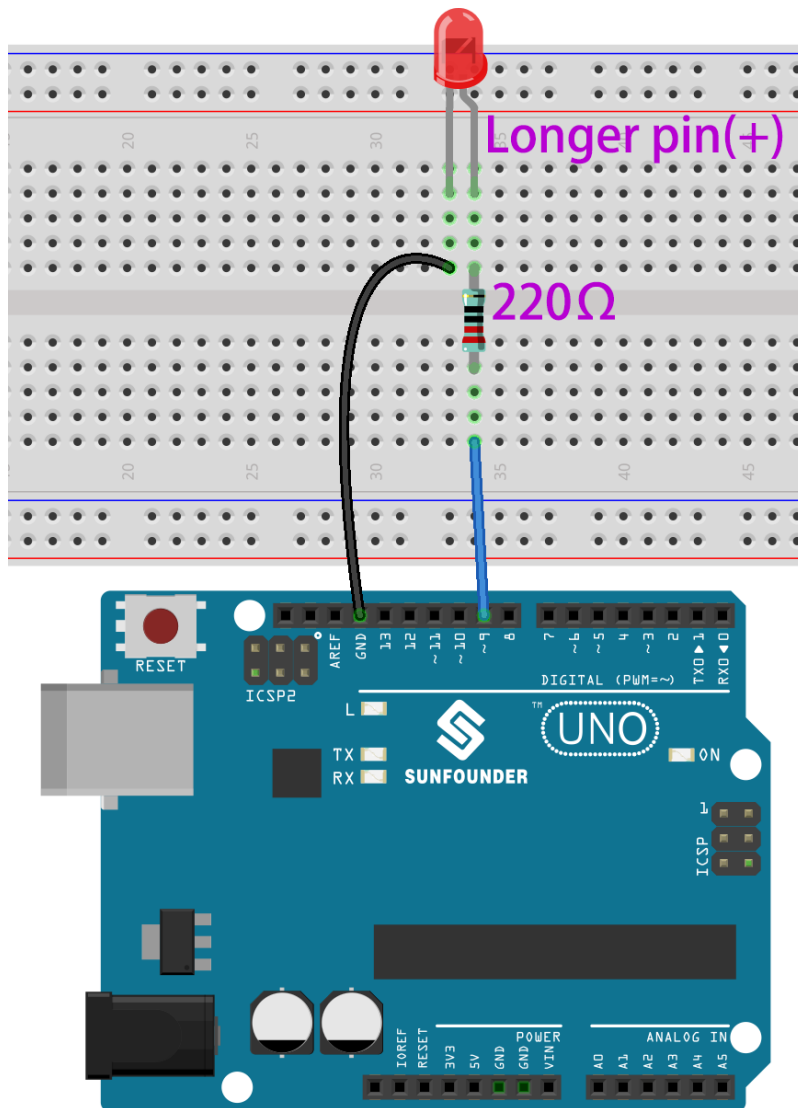
Par exemple, les LED consomment généralement 20mA de courant, et leur chute de tension est d'environ 1,8V. Selon la loi d'Ohm, si nous utilisons une alimentation de 5V, nous devons connecter une résistance d'au moins 160 ohms  $((5-1,8)/20\text{mA})$  pour ne pas brûler la LED.

### Contrôle du circuit avec Arduino

Maintenant que nous avons une compréhension de base de la programmation Arduino et des circuits électroniques, il est temps de faire face à la question la plus critique : Comment contrôler les circuits avec Arduino.

En termes simples, la manière dont Arduino contrôle un circuit est en changeant le niveau des broches sur la carte. Par exemple, lors du contrôle d'une LED embarquée, il s'agit d'écrire un signal de haut ou de bas niveau à la broche 13.

Essayons maintenant de coder la carte Arduino pour contrôler le clignotement d'une LED sur la plaque d'essai. Construisez le circuit de sorte que la LED soit connectée à la broche 9.



Ensuite, téléchargez ce sketch sur la carte de développement Arduino.

```
int ledPin = 9;
int delayTime = 500;

void setup() {
  pinMode(ledPin, OUTPUT);
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
void loop() {  
    digitalWrite(ledPin,HIGH);  
    delay(delayTime);  
    digitalWrite(ledPin,LOW);  
    delay(delayTime);  
}
```

Ce sketch est très similaire à celui que nous avons utilisé pour contrôler le clignotement de la LED embarquée, la différence est que la valeur de `ledPin` a été changée en 9. Cela est dû au fait que nous essayons de contrôler le niveau de la broche 9 cette fois-ci.

Maintenant, vous pouvez voir la LED sur la plaque d'essai clignoter.



Embarquez pour un voyage à travers le monde d'Arduino avec le Cours Vidéo Arduino complet, utilisant le kit de démarrage 3 en 1 de SunFounder. Cette série commence par une introduction à l'écosystème Arduino et aux capacités de la carte uno R3, posant ainsi les bases pour une exploration approfondie des applications pratiques et des techniques de programmation. Vous apprendrez les bases du contrôle des LEDs, de la compréhension de la communication série et de la manipulation de divers composants tels que les LEDs RVB, les boutons et les registres à décalage. Le cours propose également une variété de projets amusants comme des projets de voitures intelligentes et des projets IoT. Tant que vous suivez le cours étape par étape, vous maîtriserez Arduino, non pas en copiant et collant du code, mais en écrivant votre propre code et en implémentant vos projets Arduino à votre manière.

### Projets

## 3.1 Vidéo 1 - Présentation de ce Kit

Ce tutoriel accessible aux débutants présente Arduino comme une plateforme open-source et détaille les composants et les utilisations du kit Arduino trois-en-un de SunFounder.

- **Introduction au Cours** : Introduction au cours Arduino de Robojax, conçu pour les débutants absolus.
- **Bases d'Arduino** : Explication d'Arduino en tant que plateforme électronique accessible et open-source.
- **Exigences du Cours** : Aperçu des logiciels et du matériel nécessaires pour commencer avec Arduino.
- **Les Cartes Arduino Expliquées** : Examen détaillé des différentes cartes Arduino, y compris Uno, Mega et les variantes Wi-Fi.
- **Déballage du Kit** : Déballage complet du kit SunFounder, mettant en évidence ses composants divers.

### Vidéo

## 3.2 Vidéo 2 - Introduction à l'IDE Arduino

Ce tutoriel propose un guide pour les débutants sur la configuration du logiciel Arduino et l'exploration du kit Arduino trois-en-un de SunFounder, essentiel pour les projets d'automatisation à la maison et à l'école.

- **Configuration de l'IDE Arduino** : Instructions pour télécharger et installer l'IDE Arduino, y compris les exigences système.
- **Navigation dans l'IDE Arduino** : Parcours détaillé de l'interface et des fonctionnalités de l'IDE Arduino.
- **Guide de Connexion des Cartes** : Tutoriel sur la connexion et la configuration d'une carte Arduino avec un ordinateur.
- **Tutoriel sur la Documentation Arduino** : Comment utiliser la documentation officielle d'Arduino et les forums communautaires pour l'assistance à la programmation.
- **Exploration du Kit SunFounder** : Aperçu de la documentation du kit SunFounder et des ressources de projet.

Vidéo

## 3.3 Vidéo 3 : Bases de Programmation et Projets LED

Ce tutoriel offre une introduction complète à la programmation Arduino, couvrant la structure de base du code, les composants de la carte, et un projet de clignotement de LED adapté aux débutants.

- **Structure du Code** : Explication détaillée des fonctions `setup()` et `loop()` en programmation Arduino.
- **Carte Arduino Uno** : Examen approfondi de la carte Arduino Uno, y compris les fonctions des broches et leur utilisation.
- **Comprendre les Résistances** : Explication des résistances, y compris comment lire les codes de couleurs et mesurer la résistance.
- **Bases des LED** : Vue d'ensemble des diodes électroluminescentes (LEDs), y compris comment identifier l'anode et la cathode.
- **Projet Pratique LED** : Guide étape par étape pour mettre en place un projet de clignotement de LED basique sur une plaque d'essai.

Vidéo

Tutoriels En Ligne Connexes

- *1.1 Bonjour, LED!*

## 3.4 Vidéo 4 : Systèmes Numériques et Moniteur Série

Cette vidéo se penche sur les fondamentaux des différents systèmes numériques utilisés dans Arduino et démontre les applications pratiques du Moniteur Série pour la programmation et le débogage.

- **Explication des Systèmes Numériques** : Explication détaillée des systèmes numériques décimal, binaire, octal et hexadécimal et leur pertinence dans Arduino.
- **Techniques de Conversion de Données** : Méthodes pour convertir des données entre différents systèmes numériques.
- **Utilisation du Moniteur Série** : Guide d'utilisation du Moniteur Série Arduino pour le débogage et la visualisation de données.
- **Impression de Données dans le Moniteur Série** : Techniques pour imprimer des valeurs ASCII, des caractères et des données numériques dans le Moniteur Série.
- **Gestion des Entrées Utilisateur** : Démonstrations de la lecture et de l'interprétation des entrées utilisateur via le Moniteur Série.

Vidéo



## 3.5 Vidéo 5 : Types de Données et Variables

Cette vidéo fournit un tutoriel approfondi sur divers types de données et variables dans Arduino, mettant en évidence leur déclaration, utilisation de la mémoire et applications pratiques dans les sketches.

- **Compréhension des Types de Données** : Explication complète des types de données comme les entiers, les caractères, les flottants et les booléens dans Arduino.
- **Déclaration et Utilisation des Variables** : Guide détaillé sur la déclaration, l'attribution et l'utilisation des variables dans les sketches Arduino.
- **Représentation des Données Numériques** : Aperçu de la représentation des données numériques en valeurs binaires, hexadécimales et décimales.
- **Manipulation des Chaînes de Caractères** : Techniques pour la concaténation, la conversion et la manipulation des chaînes de caractères dans Arduino.
- **Vue d'Ensemble des Types de Données Spéciaux** : Exploration des types de données spéciaux tels que les octets, les caractères non signés et les mots, y compris leur stockage et leur plage.

Vidéo

## 3.6 Vidéo 6 : Contrôle de Buzzer, Moteur et Pompe à Eau

Cette vidéo propose des tutoriels pratiques sur le contrôle d'un buzzer actif, d'un moteur et d'une pompe à eau avec Arduino, démontrant des compétences essentielles pour la robotique et les systèmes automatisés.

- **Aperçu du Kit SunFounder** : Introduction au kit Arduino trois-en-un de SunFounder et son utilité dans les projets de robotique et d'automatisation domestique.
- **Contrôle du Buzzer Actif** : Guide pas à pas pour connecter et programmer un buzzer actif avec Arduino.
- **Tutoriel de Contrôle du Moteur** : Instructions détaillées sur l'utilisation d'un driver de moteur L298N pour le contrôle du moteur dans les applications de voiture intelligente.
- **Projet de Pompe à Eau** : Démonstration du contrôle d'une pompe à eau pour les systèmes d'arrosage automatisés.
- **Configuration de l'Environnement Arduino** : Comment configurer et modifier les sketches Arduino pour chaque projet.
- **Exécution Pratique du Projet** : Exécution pratique de chaque projet avec des conseils pour le dépannage et l'optimisation.

Vidéo

### Tutoriels En Ligne Connexes

- [1.2 Bip](#)
- [1.3 Moteur](#)
- [1.4 Pompe](#)

## 3.7 Vidéo 7 : Boutons-Poussoirs et Interrupteurs à Lime Souple

Ce tutoriel Arduino explore les fondamentaux de la lecture numérique, en se concentrant sur la détection des pressions de boutons-poussoirs et la détection des champs magnétiques avec des interrupteurs à lame souple.

- **Bases de la Lecture Numérique** : Comprendre le concept de lecture numérique dans Arduino, en distinguant les signaux hauts et bas.
- **Détection de Bouton-Poussoir** : Instructions détaillées pour connecter un bouton-poussoir et lire son état avec Arduino.
- **Détection par Interrupteur à Lime Souple** : Démonstration de l'utilisation d'un interrupteur à lame souple pour détecter les champs magnétiques, y compris le câblage et la configuration du code.
- **Configuration de l'IDE Arduino** : Guide étape par étape pour préparer l'IDE Arduino pour les projets de lecture numérique.

- **Utilisation des Résistances de Tirage Internes** : Techniques pour utiliser un bouton-poussoir sans résistance externe en exploitant les résistances de tirage internes d'Arduino.
- **Conseils de Dépannage** : Solutions aux problèmes courants rencontrés dans la lecture numérique, tels que les broches flottantes.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [3.1 Lecture de la Valeur du Bouton](#)
- [3.2 Ressentir le Magnétisme](#)

## 3.8 Vidéo 8 : MLI (PWM) et Structures de Boucles

Ce tutoriel Arduino se penche sur l'utilisation de la MLI (Modulation de Largeur d'Impulsion) pour contrôler des dispositifs tels que les moteurs et les LED, ainsi qu'un guide complet sur la programmation de boucles telles que for, while et do-while.

- **Fondamentaux de la MLI (PWM)** : Comprendre la MLI et les cycles de travail pour contrôler le comportement des dispositifs.
- **Écriture Analogique avec Arduino** : Utilisation de `analogWrite` pour moduler la force du signal pour les moteurs et les LED.
- **Démonstrations de Boucle For** : Exemples pratiques de boucles for pour l'atténuation progressive des LED.
- **Utilisation de la Boucle While** : Mise en œuvre de boucles while pour une exécution efficace du programme.
- **Mécaniques de la Boucle Do-While** : Exploration de la structure unique et des applications des boucles do-while dans les projets Arduino.
- **Contrôle Pratique des LED** : Configuration et code étape par étape pour ajuster la luminosité des LED à l'aide de la MLI.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [2.1 Estompage](#)

## 3.9 Vidéo 9 : Mesure de Tension

Apprenez à lire et mesurer la tension continue à l'aide de la fonction `analogRead` d'Arduino, comprenez le CAN (convertisseur analogique-numérique) et explorez l'utilisation de potentiomètres pour des mesures de tension précises.

- **Fonction AnalogRead** : Une introduction à l'utilisation de la fonction `analogRead` d'Arduino pour la mesure de tension continue.
- **Aperçu des Broches Analogiques** : Comprendre le rôle des broches analogiques (A0 - A5) dans la lecture de tension.
- **Types de Signaux** : Différencier les signaux numériques et analogiques et leurs niveaux de tension.
- **Bases du CAN** : Explication de la conversion analogique-numérique et de l'importance des résolutions 10 bits et 12 bits.
- **Types de Potentiomètres** : Démonstration de divers potentiomètres pour la mesure de tension.
- **Méthodes de Mesure de Tension** : Méthodes détaillées pour lire et afficher avec précision les valeurs de tension.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [4.1 Tournez le Bouton](#)

## 3.10 Vidéo 10 : Instructions Conditionnelles et Tableaux

Explorez les fondamentaux des instructions conditionnelles et des tableaux dans Arduino, y compris leurs applications pratiques dans le contrôle des LED et la gestion des données.

- **Instructions Conditionnelles** : Apprentissage de l'utilisation des instructions if, else et else-if dans Arduino pour la prise de décision.
- **Exemples Pratiques** : Application de la logique conditionnelle dans des scénarios tels que le contrôle de la température et la régulation du trafic.
- **Tableaux dans Arduino** : Démonstration de la création, de la modification et de l'utilisation de tableaux pour stocker et gérer des données.
- **Contrôle des LED avec des Tableaux** : Comment utiliser des tableaux pour un contrôle efficace de plusieurs LED.
- **Compréhension des Flottants** : Introduction au type de données float et gestion de la précision dans les opérations numériques.
- **Utilisation de Boucles avec des Tableaux** : Exploitation des boucles "for" pour itérer sur les éléments d'un tableau, simplifiant le code pour des tâches comme le séquençage des LED.

Vidéo

## 3.11 Vidéo 11 : Capteur DHT11

Cette vidéo propose un tutoriel approfondi sur les bibliothèques Arduino et l'utilisation pratique du capteur de température et d'humidité DHT11, y compris l'installation, le codage et un exemple d'application.

- **Introduction aux Bibliothèques Arduino** : Explication du concept, de l'objectif et des méthodes d'installation des bibliothèques Arduino.
- **Capacités du Capteur DHT11** : Spécifications techniques, y compris la plage de température et d'humidité, la précision et la tension de fonctionnement.
- **Guide de Câblage du Capteur** : Instructions étape par étape sur la façon de connecter correctement le capteur DHT11 à une carte Arduino.
- **Lecture des Données du Capteur** : Techniques d'utilisation du code Arduino pour lire et interpréter les données du capteur DHT11.
- **Parcours du Code** : Explication détaillée du code Arduino nécessaire pour le fonctionnement du capteur DHT11.
- **Démonstration Pratique** : Mise en œuvre d'une application réelle en déclenchant un buzzer en fonction des lectures de température du capteur DHT11.

Vidéo

Tutoriels En Ligne Connexes

- [5.11.3 Température - Humidité](#)

## 3.12 Vidéo 12 : Fonctions et Instruction Switch-Case

Explorez les fondamentaux de la définition et de l'utilisation des fonctions dans la programmation Arduino, y compris les différents types de fonctions, la gestion des paramètres, les valeurs de retour et l'utilisation des instructions switch-case.

- **Compréhension des Fonctions dans Arduino** : Introduction à la structure de base et à l'objectif des fonctions dans la programmation Arduino.
- **Types de Fonctions** : Explique la différence entre les fonctions void et celles qui retournent des types de données spécifiques comme des entiers ou des flottants.
- **Utilisation des Paramètres et des Valeurs de Retour** : Démontre comment passer des paramètres aux fonctions et utiliser les valeurs qu'elles retournent dans le programme principal.

- **Exemples Pratiques de Fonctions** : Fournit des exemples réels de fonctions pour divers calculs et opérations conditionnelles.
- **Syntaxe et Utilisation de Switch-Case** : Offre une alternative aux instructions if-else, montrant comment switch-case peut simplifier la structure du code.
- **Application des Fonctions dans les Projets** : Montre les applications pratiques des fonctions dans les projets Arduino, telles que le contrôle des dispositifs ou le traitement des données des capteurs.

Vidéo

### 3.13 Vidéo 13 : Manipuler le Joystick

Ce tutoriel se penche sur l'utilisation d'un joystick XY avec Arduino, couvrant sa structure, son câblage, la lecture de ses positions, la détection du bouton-poussoir, et la programmation d'Arduino pour répondre aux mouvements du joystick.

- **Mécanique et Structure du Joystick** : Un aperçu des composants du joystick XY, y compris ses potentiomètres pour le contrôle directionnel et le bouton-poussoir intégré.
- **Connexion du Joystick à Arduino** : Instructions détaillées pour connecter correctement le joystick à une carte Arduino pour une lecture précise du signal.
- **Interprétation des Mouvements du Joystick** : Techniques de lecture des mouvements des axes X et Y du joystick à travers la programmation Arduino.
- **Détection des Pressions sur le Bouton** : Conseils pour détecter et programmer des réponses à l'activation du bouton-poussoir du joystick.
- **Code Arduino pour l'Entrée du Joystick** : Guide étape par étape pour écrire un code Arduino interprétant et affichant les mouvements du joystick sur un moniteur série.
- **Actions Réactives à l'Entrée du Joystick** : Démonstration de la programmation d'Arduino pour effectuer des actions spécifiques, comme déclencher un buzzer, en fonction de la position du joystick.

Vidéo

Tutoriels En Ligne Connexes

- [\*4.3 Manipuler le Joystick\*](#)

### 3.14 Vidéo 14 : millis() et map()

Ce tutoriel se concentre sur l'utilisation de millis() pour le suivi du temps et de map() pour la conversion de valeurs dans Arduino, en présentant des applications pratiques telles que les réponses de boutons temporisées et le contrôle de la luminosité des LED.

- **Fonctionnalité de Millis** : millis() comme une fonction de chronométrage dans Arduino, commençant à zéro au lancement du programme et s'incrémentant chaque milliseconde.
- **Événements Temporisés avec millis()** : Comment utiliser millis() pour exécuter des événements à des intervalles de temps spécifiques sans arrêter l'ensemble du programme, contrairement à delay().
- **Timing de Pression de Bouton** : Exemple d'utilisation de millis() pour détecter une pression de bouton et exécuter une action après une durée définie.
- **Introduction à la Fonction Map** : Introduction à la fonction map(), utilisée pour convertir des valeurs numériques d'une plage à une autre.
- **Ajustement de la Luminosité des LED** : Démonstration pratique de l'utilisation de map() pour ajuster les niveaux de luminosité des LED en mappant des valeurs en pourcentage à la plage PWM.
- **Codage Efficace avec millis() et map()** : Présentation de pratiques de codage Arduino efficaces en combinant millis() pour un chronométrage non bloquant et map() pour une conversion intuitive des valeurs.

Vidéo

## 3.15 Vidéo 15 : Automatisation de l'Arrosage des Plantes

Ce tutoriel couvre l'utilisation d'un capteur d'humidité du sol avec Arduino pour surveiller l'humidité du sol et automatiser l'arrosage à travers des démonstrations de code et de matériel.

- **Bases du Capteur d'Humidité du Sol** : Introduction au capteur d'humidité du sol, à ses broches et à la manière dont il mesure l'humidité du sol.
- **Câblage Capteur-Arduino** : Instructions étape par étape pour connecter le capteur d'humidité à l'Arduino.
- **Code Arduino pour la Lecture de l'Humidité** : Explication du code Arduino pour lire et interpréter les niveaux d'humidité à partir du capteur.
- **Comprendre la Sortie du Capteur** : Analyse des valeurs de sortie du capteur pour déterminer les conditions d'humidité du sol.
- **Logique d'Arrosage Automatisé** : Développement d'un code Arduino avancé pour automatiser l'arrosage en fonction des niveaux d'humidité du sol.
- **Mise en Œuvre et Test Pratiques** : Démonstrations du capteur en action dans le sol, montrant comment le système peut déclencher un buzzer ou contrôler une pompe à eau pour un arrosage automatisé.

### Vidéo

#### Tutoriels En Ligne Connexes

- [4.4 Mesurer l'Humidité du Sol](#)

## 3.16 Vidéo 16 : Évitement d'Obstacles par Infrarouge

Apprenez à configurer et à programmer un capteur d'évitement d'obstacles infrarouge avec Arduino pour détecter les obstacles et activer un buzzer.

- **Compréhension de la Mécanique du Capteur IR** : Vue d'ensemble des composants du capteur infrarouge et de leur fonctionnement pour détecter les obstacles.
- **Connexion Capteur-Arduino** : Instructions étape par étape pour le câblage du capteur infrarouge à la carte Arduino.
- **Programmation pour la Détection d'Obstacles** : Introduction à la rédaction et à la compréhension du code Arduino pour la détection d'obstacles à l'aide du capteur.
- **Démonstration de Détection d'Obstacles en Temps Réel** : Démonstration de la capacité du capteur à détecter des objets à différentes distances et dans différentes conditions.
- **Ajustement de la Sensibilité** : Guide détaillé pour ajuster la sensibilité du capteur pour différentes plages de détection.
- **Intégration d'un Buzzer** : Modification de la configuration et du code pour inclure un buzzer qui s'active lors de la détection d'un obstacle.

### Vidéo

#### Tutoriels En Ligne Connexes

- [3.3 Détecter l'Obstacle](#)

## 3.17 Vidéo 17 : Interruption

Explorez les fonctionnalités des interrupts d'Arduino et des Résistances Dépendantes de la Lumière (LDR) pour construire des projets intelligents de détection jour/nuit.

- **Explication des Interrupts d'Arduino** : Plongez dans la mécanique des interrupts d'Arduino pour un codage plus efficace.
- **Utilisation Efficace des Broches d'Interrupt** : Apprenez quelles broches Arduino prennent en charge les interrupts et comment les utiliser.
- **Configuration Pratique des Interrupts** : Démonstration de la mise en place d'un bouton-poussoir et d'un buzzer pour fonctionner avec les interrupts d'Arduino.

- **Introduction aux LDR** : Découvrez comment les LDR changent de résistance en fonction de la lumière et leurs applications.
- **Techniques de Mesure des LDR** : Apprenez à mesurer la résistance des LDR dans différentes conditions d'éclairage à l'aide d'un multimètre.
- **Projets de Détection Jour/Nuit** : Implémentez des projets basés sur LDR pour la commutation automatique de dispositifs comme des buzzers ou des lumières en fonction de l'intensité lumineuse.

**Vidéo**

**Tutoriels En Ligne Connexes**

- [5.13 Interruption](#)

## 3.18 Vidéo 18 : Servomoteur

Ce tutoriel explore les fondamentaux de l'utilisation des servomoteurs avec Arduino, couvrant tout, du câblage et du codage aux démonstrations pratiques de projets.

- **Aperçu des Servomoteurs** : Introduction aux servomoteurs, mettant en évidence leur rôle dans diverses applications télécommandées et robotiques.
- **Types et Caractéristiques** : Exploration des différents types de servomoteurs et de leurs capacités de couple.
- **Comprendre les Composants des Servos** : Description des différentes parties d'un servomoteur, y compris les bras et les engrenages.
- **Guide de Câblage Arduino** : Instructions étape par étape pour connecter des servomoteurs aux cartes Arduino.
- **Programmation pour la Précision** : Démonstration de l'utilisation du code Arduino pour contrôler les angles et les mouvements des servos.
- **Exemples de Projets Pratiques** : Applications réelles et modifications des servomoteurs dans les projets Arduino.

**Vidéo**

**Tutoriels En Ligne Connexes**

- [5.5 Utiliser une Bibliothèque Interne](#)

## 3.19 Vidéo 19 : Capteur Ultrasonique

Ce tutoriel explore l'utilisation d'un capteur ultrasonique avec Arduino, démontrant comment câbler, programmer et mesurer précisément les distances.

- **Bases des Capteurs Ultrasoniques** : Introduction aux capteurs ultrasoniques et à leur fonctionnalité dans la mesure de distance.
- **Mécanisme Opérationnel** : Compréhension des ondes ultrasoniques et de leur principe de réflexion pour mesurer les distances.
- **Câblage du Capteur à Arduino** : Guide étape par étape pour connecter le capteur ultrasonique à Arduino.
- **Configuration de la Bibliothèque** : Comment télécharger et installer la bibliothèque nécessaire pour le capteur.
- **Programmation pour la Mesure** : Explication détaillée du code Arduino pour le fonctionnement du capteur ultrasonique.
- **Tests Pratiques de Mesure** : Démonstration de la capacité du capteur à mesurer différentes distances avec précision.

**Vidéo**

**Tutoriels En Ligne Connexes**

- [5.8 Fonction Définie par l'Utilisateur](#)

## 3.20 Vidéo 20 : LCD 1602

Ce tutoriel complet explore l'utilisation de l'écran LCD 1602 d'Arduino pour afficher diverses données de capteurs, y compris la température, l'humidité et la distance.

- **Bases du LCD 1602** : Introduction aux fonctionnalités et capacités de l'écran LCD 1602.
- **Utilisation du Module I2C** : Simplification des connexions LCD avec le module I2C et son installation.
- **Configuration de l'Adresse LCD** : Comment configurer et identifier correctement l'adresse I2C pour l'écran LCD.
- **Configuration de la Bibliothèque Arduino** : Conseils pour installer la bibliothèque nécessaire à l'intégration du LCD.
- **Affichage de Texte et de Caractères** : Démonstration de l'affichage de texte de base et de la création de caractères personnalisés sur l'écran LCD.
- **Intégration de Capteurs** : Mise en avant de l'intégration du capteur de température DHT, d'un potentiomètre et d'un capteur ultrasonique avec l'écran LCD pour l'affichage en temps réel des données.

### Vidéo

#### Tutoriels En Ligne Connexes

- [5.11.1 Affichage à Cristaux Liquides](#)
- [5.12 Lecture Série](#)

## 3.21 Vidéo 21 : Télécommande Infrarouge

Ce tutoriel démontre comment utiliser une télécommande infrarouge avec Arduino pour contrôler diverses fonctions, en se concentrant sur le décodage des signaux et des applications pratiques comme l'activation d'un buzzer.

- **Comprendre la Technologie Infrarouge** : Aperçu du fonctionnement des télécommandes et des récepteurs infrarouges avec Arduino.
- **Câblage du Récepteur IR** : Guide détaillé pour connecter correctement le récepteur infrarouge à la carte Arduino.
- **Installation des Bibliothèques Nécessaires** : Processus étape par étape pour installer la bibliothèque de télécommande IR dans l'IDE Arduino.
- **Décodeur de Signaux IR** : Explication sur la manière de décoder les signaux d'une télécommande IR à l'aide d'Arduino.
- **Action à la Pression d'une Touche** : Programmation d'Arduino pour effectuer des actions spécifiques basées sur les entrées de la télécommande.
- **Aperçu des Projets Futurs** : Prévisualisation des projets avancés à venir utilisant Arduino et la technologie infrarouge.

### Vidéo

#### Tutoriels En Ligne Connexes

- [5.11.2 Récepteur IR](#)

## 3.22 Vidéo 22 : Afficheur à 7 Segments

Ce tutoriel vous guide dans la création d'un compteur numérique et de dés électroniques en utilisant Arduino, en présentant à la fois l'assemblage et les aspects de codage.

- **Assemblage de l'Afficheur à 7 Segments** : Instructions pour assembler et comprendre un afficheur à 7 segments.
- **Aperçu de la Fiche Technique** : Explication de la fiche technique du modèle 5161AS, en se concentrant sur les spécifications et l'utilisation optimale.
- **Guide de Câblage du Compteur** : Guide détaillé sur le câblage du compteur numérique, en mettant l'accent sur la puce 74HC595 et les connexions d'affichage.



- **Décodage du Code du Compteur** : Explication approfondie du code Arduino pour le fonctionnement du compteur numérique.
- **Projet de Dés Électroniques** : Guide étape par étape pour construire et coder des dés électroniques, incluant la mise en œuvre d'une interruption par bouton-poussoir.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [5.10 ShiftOut\(Affichage à segments\)](#)

## 3.23 Vidéo 23 : Assemblage d'une Voiture Intelligente

Apprenez à assembler et à câbler une voiture intelligente en utilisant la plateforme Arduino, détaillé du début à la fin, y compris la configuration du moteur et du pilote.

- **Assemblage de la Voiture Intelligente** : Conseils pour assembler la base, les roues et les moteurs de la voiture, en assurant une configuration correcte.
- **Bases des Moteurs à Courant Continu** : Explique le principe de fonctionnement des moteurs à courant continu et leur rôle dans la fonctionnalité de la voiture intelligente.
- **Câblage du Pilote de Moteur** : Instructions détaillées sur le câblage du Pilote de Moteur Double Pont Complet L298N pour un contrôle optimal.
- **Test de Câblage du Moteur** : Réalisation d'un test basique pour vérifier le câblage correct et la fonctionnalité des moteurs.
- **Démonstration de Mouvement** : Présentation des capacités de mouvement de base de la voiture, démontrant un assemblage et un câblage réussis.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [Assembler la Voiture](#)
- [1. Mouvement](#)
- [2. Mouvement par Code](#)

## 3.24 Vidéo 24 : Contrôle d'une Voiture Intelligente

Ce tutoriel guide les spectateurs à travers le contrôle d'une voiture intelligente en utilisant Arduino, en se concentrant sur le mouvement directionnel, l'arrêt et les ajustements de vitesse.

- **Mouvements de Base de la Voiture** : Introduction à la programmation de la voiture intelligente pour des mouvements de base tels que avancer, reculer, tourner à gauche, tourner à droite et s'arrêter.
- **Configuration et Installation du Câblage** : Guide étape par étape pour configurer le câblage entre l'Arduino et la voiture intelligente.
- **Code Arduino pour le Mouvement** : Explication détaillée du code Arduino nécessaire pour le contrôle directionnel de la voiture intelligente.
- **Techniques de Contrôle de la Vitesse** : Techniques pour programmer la voiture pour des variations de vitesse, y compris l'accélération et la décélération.
- **Démonstrations de Mouvement Pratiques** : Démonstrations en direct montrant la mise en œuvre des mouvements de la voiture et du contrôle de la vitesse.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [3. Accélération](#)



## 3.25 Vidéo 25 : Évitement d'Obstacles pour Voiture Intelligente

Ce tutoriel enseigne comment programmer une voiture intelligente pour éviter les obstacles en utilisant des capteurs infrarouges, dans le cadre du cours Arduino Robojax avec le kit SunFounder.

- **Bases de l'Évitement d'Obstacles** : Aperçu de l'utilisation de capteurs infrarouges pour la détection et l'évitement d'obstacles par la voiture intelligente.
- **Connaissances Requises** : Accent sur les connaissances préalables nécessaires en matière de contrôle des mouvements de la voiture et des bases des capteurs infrarouges.
- **Installation et Alignement des Capteurs** : Détails sur l'alignement correct des capteurs pour une détection optimale des obstacles.
- **Câblage pour la Détection d'Obstacles** : Guide étape par étape pour connecter les capteurs à l'Arduino et à la voiture.
- **Explication Complète du Code pour l'Évitement d'Obstacles** : Explication complète du code Arduino pour la détection des obstacles et la réponse à ceux-ci.
- **Démonstration en Direct de l'Évitement d'Obstacles** : Démonstration en temps réel montrant la voiture intelligente naviguant à travers les obstacles.

**Vidéo**

**Tutoriels En Ligne Connexes**

- *5. Jouer avec le module d'évitement d'obstacles*

## 3.26 Vidéo 26 : Capteur Ultrasonique pour Voiture Intelligente

Dans ce tutoriel, vous apprendrez à programmer une voiture intelligente avec un Arduino et des capteurs ultrasoniques pour des fonctionnalités avancées de détection et de navigation d'obstacles.

- **Introduction au Capteur Ultrasonique** : Exploration des capacités des capteurs ultrasoniques dans la navigation des voitures intelligentes.
- **Instructions de Câblage du Capteur** : Guide étape par étape pour connecter le capteur ultrasonique à la carte Arduino.
- **Décodage du Code Arduino** : Explication approfondie du code utilisé pour traiter les données des capteurs et contrôler la voiture.
- **Contrôle du Moteur via les Données du Capteur** : Comment utiliser les lectures du capteur ultrasonique pour contrôler les moteurs de la voiture.

**Vidéo**

**Tutoriels En Ligne Connexes**

- *6. Jouer avec le module ultrasonique*

## 3.27 Vidéo 27 : Suivi de la Main par Voiture Intelligente

Dans ce tutoriel, vous apprendrez à programmer une voiture intelligente pour suivre les mouvements de la main à l'aide d'Arduino, de capteurs infrarouges et ultrasoniques.

- **Fonctionnalité de Suivi de la Main par la Voiture** : Vue d'ensemble de la capacité de la voiture à pister et suivre les mouvements de la main.
- **Intégration des Capteurs pour la Détection** : Utilisation de capteurs infrarouges et ultrasoniques pour détecter la position et la distance de la main.
- **Guide de Câblage Détaillé** : Instructions étape par étape pour câbler correctement les capteurs à la carte Arduino.
- **Explication Complète du Code** : Explication du code Arduino pour traiter les entrées des capteurs et contrôler la voiture.

- **Contrôle Réactif du Moteur** : Ajustement des réponses du moteur de la voiture en fonction des données des capteurs pour un mouvement précis.
- **Optimisation du Code pour la Précision** : Affinage du code pour assurer des lectures de capteurs fiables et cohérentes.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [7. Suivez Votre Main](#)

## 3.28 Vidéo 28 : Voiture Intelligente Autonome

Apprenez à créer une voiture autonome simple en utilisant Arduino, équipée de capteurs ultrasoniques et infrarouges pour la navigation et l'évitement des obstacles.

- **Introduction au Projet** : Vue d'ensemble de la fabrication d'une voiture autonome avec Arduino.
- **Intégration des Capteurs** : Utilisation de capteurs ultrasoniques et infrarouges pour la navigation et la détection d'obstacles.
- **Décodage Détaillé du Code** : Explication du code Arduino qui contrôle les mouvements de la voiture en réponse aux données des capteurs.
- **Configuration des Moteurs et des Capteurs** : Configuration des broches Arduino pour le contrôle du moteur et les entrées des capteurs.
- **Logique de Mouvement via le Code** : Mise en œuvre de déclarations conditionnelles dans le code pour des mouvements intelligents de la voiture.
- **Test de Fonctionnalité du Code** : Observation de la réponse de la voiture aux obstacles et de ses capacités de navigation à travers le moniteur série.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [8. Voiture Autonome](#)

## 3.29 Vidéo 29 : Télécommande de Voiture Intelligente

Ce tutoriel enseigne comment utiliser une télécommande infrarouge pour contrôler une voiture intelligente Arduino, y compris les directions de mouvement et les ajustements de vitesse.

- **Introduction à la Télécommande** : Comprendre les bases du contrôle d'une voiture Arduino avec une télécommande infrarouge.
- **Fonctions des Touches de la Télécommande** : Explication détaillée de la fonction de chaque touche de la télécommande pour le contrôle de la voiture.
- **Instructions de Câblage** : Configuration étape par étape du câblage pour le récepteur infrarouge et une LED supplémentaire.
- **Explication et Démonstration du Code** : Guide approfondi du code Arduino et de son impact sur le contrôle de la voiture.

#### Vidéo

#### Tutoriels En Ligne Connexes

- [9. Télécommande](#)

### 3.30 Vidéo 30 : Suivi de Ligne par Voiture Intelligente

Ce tutoriel explore comment programmer une voiture intelligente Arduino pour le suivi de ligne en utilisant des capteurs infrarouges.

- **Introduction au Suivi de Ligne** : Comprendre les bases du suivi de ligne dans les voitures intelligentes Arduino.
- **Détails sur le Capteur de Suivi de Ligne** : Aperçu du module émetteur et récepteur infrarouge utilisé pour le suivi de ligne.
- **Réglage du Capteur pour la Précision** : Conseils pour ajuster le capteur pour une détection précise de la ligne.
- **Explication Détaillée du Code Arduino** : Explication détaillée du code qui régit la capacité de suivi de ligne de la voiture.
- **Démonstration des Valeurs du Capteur** : Comment l'Arduino différencie les surfaces noires et blanches à l'aide du capteur.
- **Démonstration Pratique du Suivi de Ligne** : Présentation de l'application pratique du suivi de ligne dans une voiture Arduino.

#### Vidéo

#### Tutoriels En Ligne Connexes

- *4. Suivre la ligne*

### 3.31 Vidéo 31 : Surveillance de la Température par Wi-Fi

Un guide complet sur la configuration de l'ESP8266 ESP-01 avec un capteur DHT11 pour surveiller la température via Wi-Fi et l'afficher sur divers appareils.

- **Introduction à l'ESP8266 et au DHT11** : Exploration des composants utilisés pour la surveillance de la température basée sur le Wi-Fi.
- **Fondamentaux du Serveur Web** : Comprendre le fonctionnement des serveurs web dans le contexte de l'accès à distance aux données.
- **Configuration de l'IDE Arduino pour ESP8266** : Configuration de l'IDE Arduino pour programmer le module ESP8266.
- **Guide de Câblage du Capteur** : Instructions détaillées pour connecter le capteur DHT11 à l'ESP8266.
- **Données de Température sur le Moniteur Série** : Programmation de l'ESP8266 pour afficher les lectures de température sur un moniteur série.
- **Affichage des Données de Température à Distance** : Démonstration de la manière de visualiser les données de température sur des appareils mobiles et de bureau via le Wi-Fi.

#### Vidéo

#### Tutoriels En Ligne Connexes

- *5. Surveillance de l'Environnement Domestique*



## CHAPITRE 4

---

Téléchargez le Code

---

Téléchargez le code pertinent depuis le lien ci-dessous.

- [Kit SunFounder 3 en 1 pour Arduino](#)
- Ou consultez le code sur [Kit SunFounder 3 en 1 pour Arduino - GitHub](#)



Ce chapitre est utilisé pour apprendre à contrôler des circuits électroniques en utilisant Arduino.

En fonction des composants, les méthodes de contrôle de base d'Arduino peuvent être divisées en quatre types :

- *1. Écriture Numérique* : Définir la tension de sortie de la broche en haute ou basse, qui peut être utilisée pour allumer et éteindre la lumière.
- *2. Écriture Analogique* : Écrire la valeur analogique (*onde PWM*) sur la broche, qui peut être utilisée pour ajuster la luminosité de la lumière.
- *3. Lecture Numérique* : Lire le signal de niveau de la broche numérique, qui peut être utilisé pour lire l'état de fonctionnement de l'interrupteur.
- *4. Lecture Analogique* : Lire la tension de la broche analogique, qui peut être utilisée pour lire l'état de fonctionnement du bouton.

Il existe également certains composants qui nécessitent des bibliothèques supplémentaires pour être utilisés, et ceux-ci sont regroupés sous la section *5.11 Installer des bibliothèques externes*.

Enfin, le kit fournit également quelques *6. Projet Amusant*, qui comprend de nombreuses manipulations simples et utiles. Essayez cette section de code et vous comprendrez comment fonctionnent la plupart des projets simples.

## 5.1 1. Écriture Numérique

L'**Écriture Numérique** consiste à émettre ou écrire un signal numérique sur une broche numérique. Le signal numérique n'a que deux états, 0 ou 1, 0V ou 5V, ce qui permet à certains composants, tels que la LED et le buzzer, d'être allumés ou éteints.

Sur la carte Arduino R3, il y a 14 broches d'entrée/sortie numériques de 0 à 13. Utilisez maintenant les fonctions `pinMode()` et `digitalWrite()` pour écrire un niveau élevé ou bas sur ces broches numériques.

- `pinMode(pin, mode)` : Configurez la broche spécifique en tant que INPUT ou OUTPUT, ici elle doit être définie comme OUTPUT.

### Syntaxe

`pinMode(pin, mode)`

### Paramètres

- `pin` : le numéro de broche Arduino pour définir le mode.

- mode : INPUT, OUTPUT, ou INPUT\_PULLUP.
- digitalWrite(pin, value) : Écrivez un niveau élevé (5V) ou un niveau bas (0V) sur une broche numérique pour changer l'état de fonctionnement du composant. Si la broche a été configurée en tant que sortie avec pinMode(), sa tension sera réglée sur la valeur correspondante : 5V (ou 3,3V sur les cartes 3,3V) pour HIGH, 0V (masse) pour LOW.

### Syntaxe

digitalWrite(pin, value)

### Paramètres

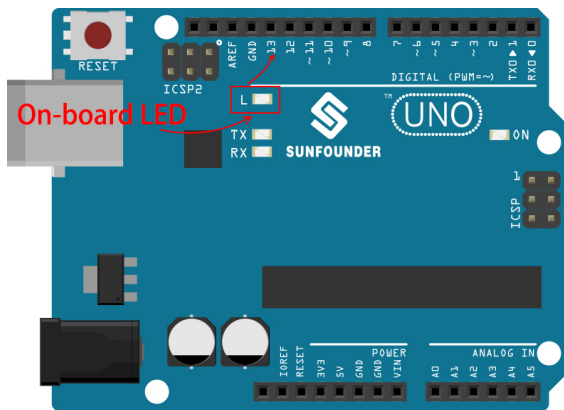
- pin : le numéro de broche Arduino.
- value : HIGH ou LOW.

### Exemple d'Écriture Numérique :

```
const int pin = 13;

void setup() {
  pinMode(pin, OUTPUT);    // sets the digital pin as output
}

void loop() {
  digitalWrite(pin, HIGH); // sets the digital pin on
  delay(1000);             // waits for a second
  digitalWrite(pin, LOW);  // sets the digital pin off
  delay(1000);             // waits for a second
}
```



### Notes et Avertissements

- Les broches 0~13 sont toutes des broches numériques.
- N'utilisez pas les broches 0 et 1, car elles sont utilisées pour communiquer avec l'ordinateur. Connecter quoi que ce soit à ces broches perturbera la communication, y compris provoquant l'échec de la mise en ligne.
- Si les broches numériques sont toutes utilisées, les broches analogiques (A0-A5) peuvent également être utilisées comme broches numériques.

### Composants Connexes

Ci-dessous se trouvent les composants connexes, vous pouvez cliquer pour apprendre comment les utiliser.



### 5.1.1 1.1 Bonjour, LED!

Tout comme écrire « Bonjour, monde ! » est la première étape pour apprendre à programmer, utiliser un programme pour contrôler une LED est l'introduction traditionnelle à l'apprentissage de la programmation physique.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

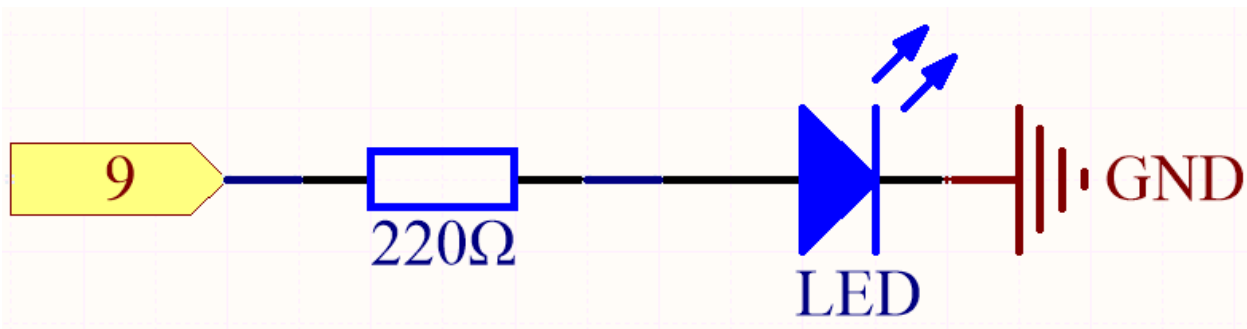
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

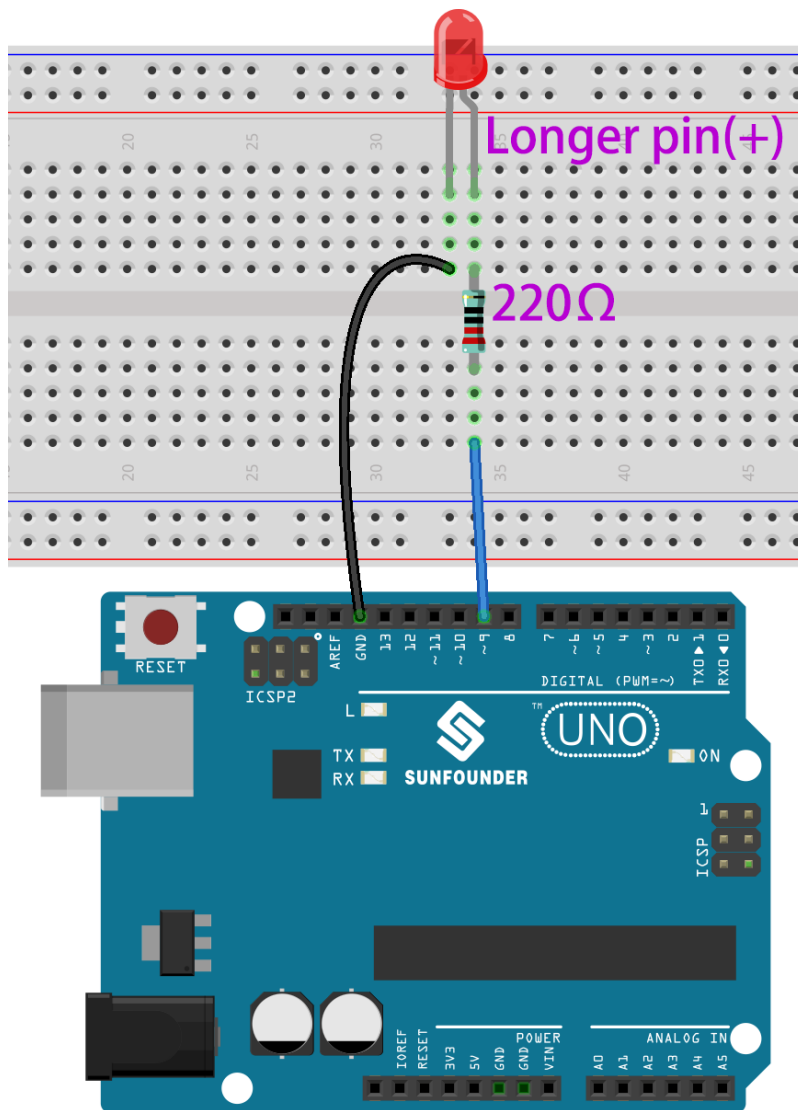
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	

#### Schéma



Le principe de ce circuit est simple et la direction du courant est indiquée sur la figure. Lorsque la broche 9 émet un niveau haut (5V), la LED s'illuminera après la résistance limitatrice de courant de 220 ohms. Lorsque la broche 9 émet un niveau bas (0v), la LED s'éteindra.

#### Câblage



## Code

### Note :

- Vous pouvez ouvrir le fichier 1.1.hello\_led.ino dans le chemin 3in1-kit\basic\_project\1.1.hello\_led.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, vous verrez la LED clignoter.

### Comment ça fonctionne ?

Ici, nous connectons la LED à la broche numérique 9, nous devons donc déclarer une variable int appelée ledpin au début du programme et lui attribuer une valeur de 9.

```
const int ledPin = 9;
```

Maintenant, initialisez la broche dans la fonction setup(), où vous devez configurer la broche en mode OUTPUT.

```
void setup() {
  pinMode(ledPin, OUTPUT);
}
```

Dans `loop()`, `digitalWrite()` est utilisé pour fournir un signal de niveau haut de 5V pour `ledPin`, ce qui provoquera une différence de tension entre les broches de la LED et allumera la LED.

```
digitalWrite(ledPin, HIGH);
```

Si le signal de niveau est changé en LOW, le signal de `ledPin` sera ramené à 0 V pour éteindre la LED.

```
digitalWrite(ledPin, LOW);
```

Un intervalle entre allumé et éteint est nécessaire pour permettre aux gens de voir le changement, donc nous utilisons un code `delay(1000)` pour laisser le contrôleur ne rien faire pendant 1000 ms.

```
delay(1000);
```

### 5.1.2 1.2 Bip

Le buzzer actif est un dispositif de sortie numérique typique, aussi facile à utiliser que d'allumer une LED !

Deux types de buzzers sont inclus dans le kit. Nous devons utiliser le buzzer actif. Retournez-les, le dos scellé (et non le PCB exposé) est celui que nous recherchons.



#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

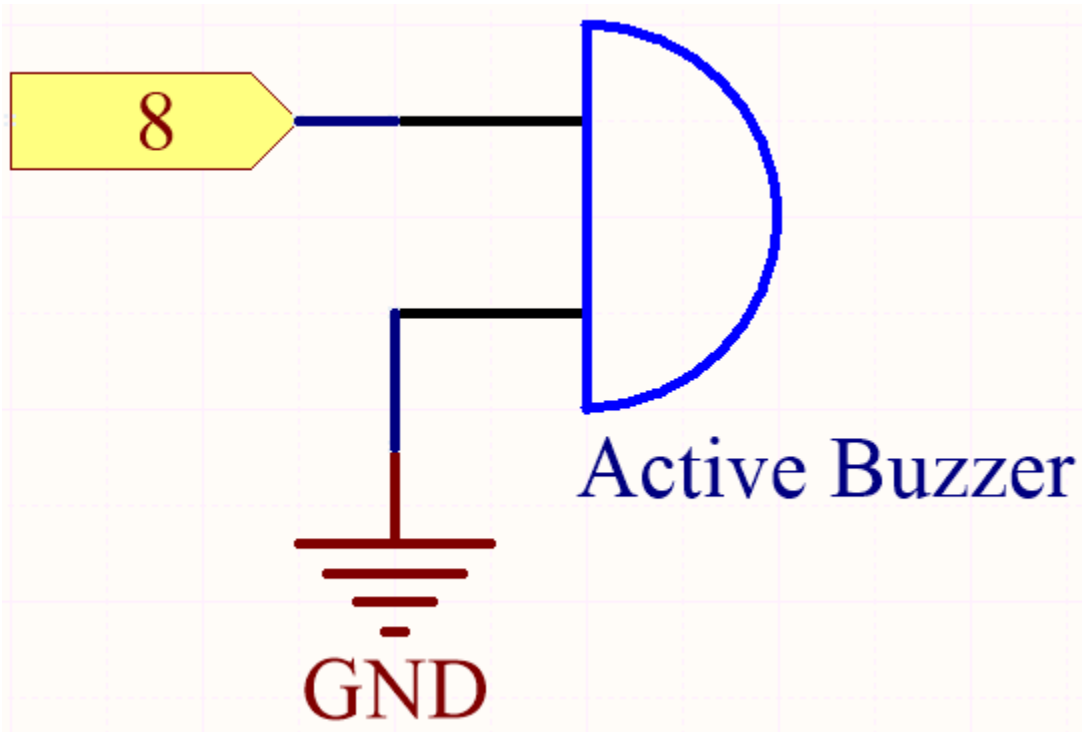
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

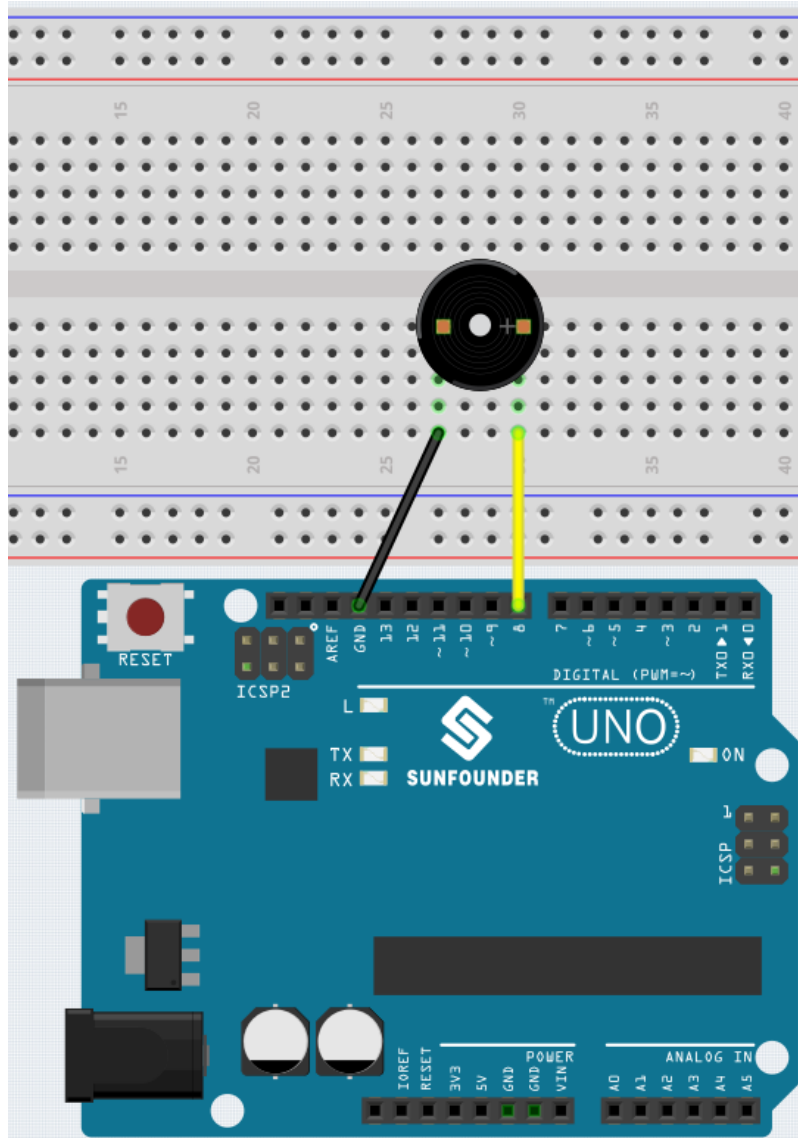
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Buzzer</i>	-

### Schéma



### Câblage



## Code

### Note :

- Vous pouvez ouvrir le fichier 1.2.beep.ino dans le chemin 3in1-kit/basic\_project/1.2.beep.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, vous entendrez un bip toutes les secondes.

### 5.1.3 1.3 Moteur

Un moteur est un dispositif de sortie numérique typique, utilisé de la même manière qu'une LED. Cependant, le moteur nécessite un courant important, et ce dernier peut endommager la carte de contrôle principale, comme la carte R3. Par conséquent, un module L9110 est utilisé dans ce cas, qui est un bon assistant pour la carte R3 afin de contrôler le moteur en toute sécurité.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

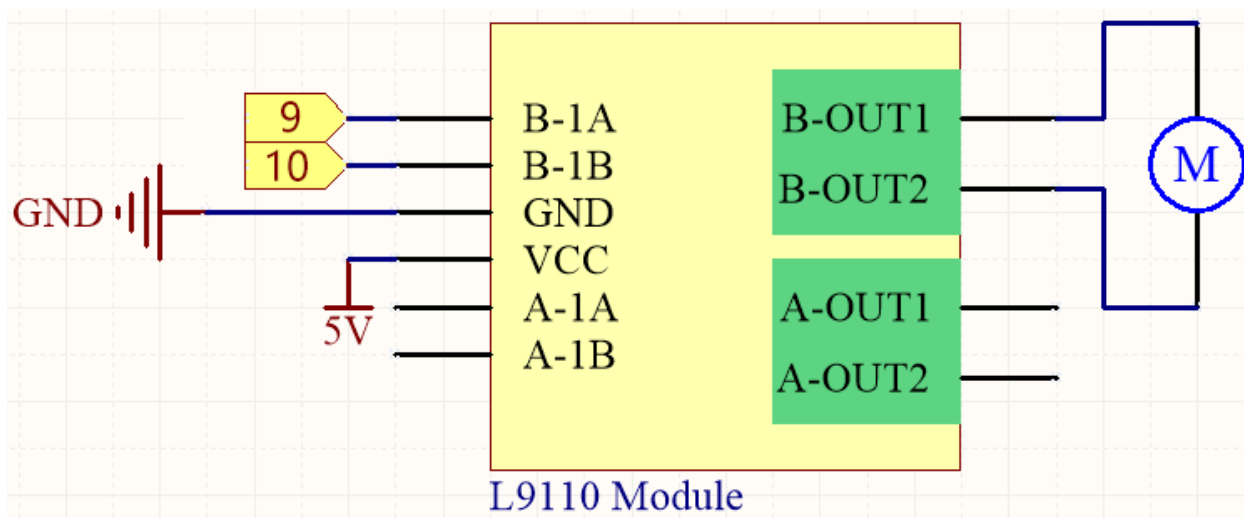
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

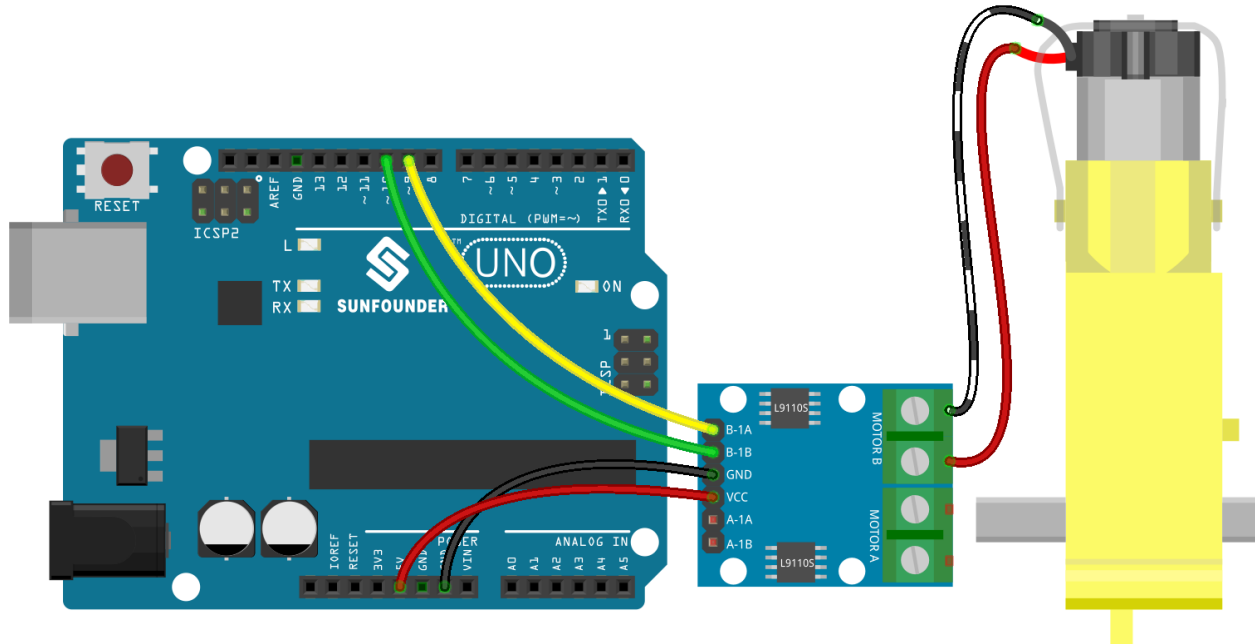
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Moteur TT</i>	-
<i>Module de Contrôle Moteur L9110</i>	-

#### Schéma



#### Câblage



## Code

### Note :

- Vous pouvez ouvrir le fichier 1.3.turn\_the\_wheel.ino dans le chemin 3in1-kit\basic\_project\1.3.turn\_the\_wheel.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

## 5.1.4 1.4 Pompage

La pompe à eau est également un moteur, qui convertit l'énergie mécanique du moteur ou d'une autre source externe grâce à une structure spéciale pour transporter le liquide.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

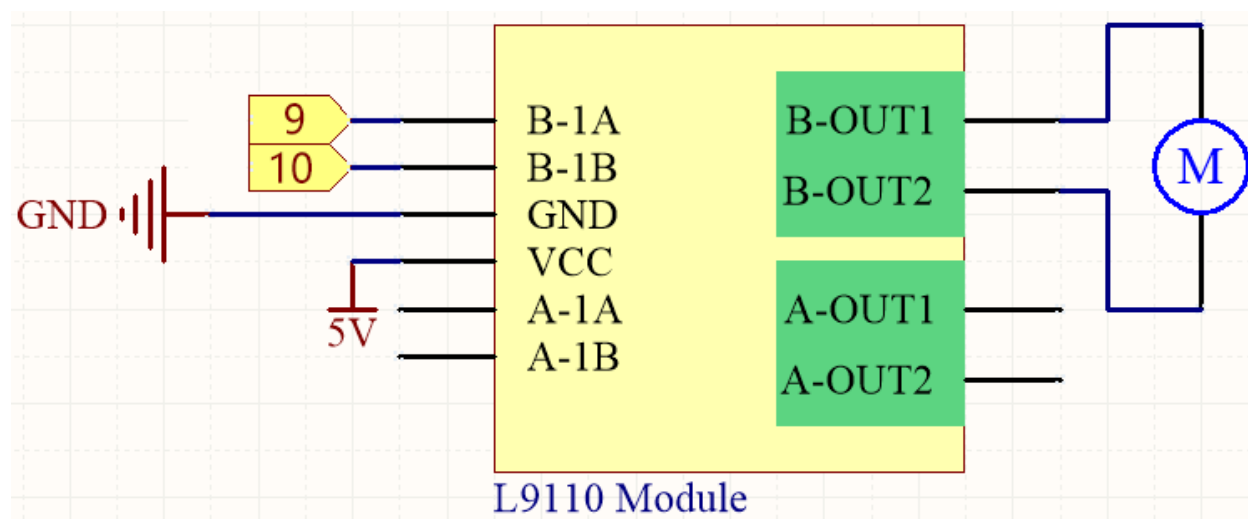
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

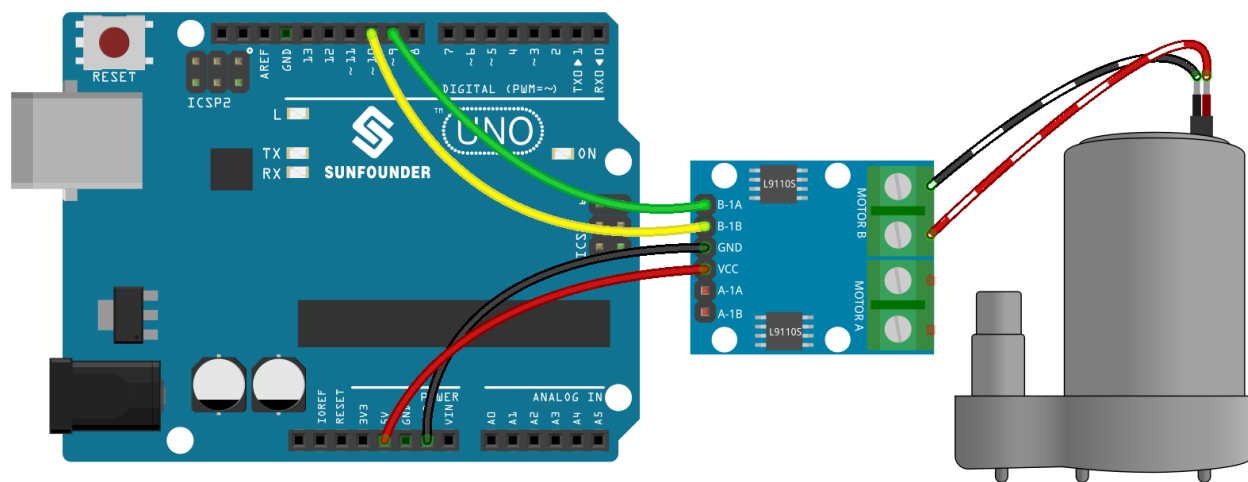
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Pompe Centrifuge</i>	-

### Schéma



### Câblage



### Code

#### Note :

- Vous pouvez ouvrir le fichier 1.4.pumping.ino dans le chemin 3in1-kit/basic\_project/1.4.pumping.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Ajoutez le tuyau à la pompe et placez-la dans le bassin. Après le téléchargement réussi du code, vous pouvez constater



que l'eau dans le bassin est évacuée au bout d'un moment. Lors de cette expérimentation, veuillez tenir le circuit éloigné de l'eau pour éviter un court-circuit !

## 5.2 2. Écriture Analogique

6 des 14 broches numériques de l'Arduino ont également une fonction de sortie PWM. Par conséquent, en plus d'écrire des signaux numériques sur ces 6 broches, vous pouvez également y écrire des signaux analogiques (signaux d'onde PWM). De cette manière, vous pouvez faire varier la luminosité des LED ou faire tourner le moteur à différentes vitesses.

La Modulation de Largeur d'Impulsion, ou **PWM**, est une technique permettant d'obtenir des résultats analogiques avec des moyens numériques. Puisqu'il peut être difficile de saisir le sens littéral, voici un exemple de contrôle de l'intensité d'une LED pour mieux comprendre.

Un signal numérique composé de niveaux haut et bas est appelé une impulsion. La largeur d'impulsion de ces broches peut être ajustée en changeant la vitesse de mise en marche/arrêt. En d'autres termes, lorsque nous allumons et éteignons la LED sur une courte période (comme 20ms, le temps de persistance visuelle de la plupart des gens), nous ne verrons pas qu'elle s'est éteinte, mais la luminosité de la lumière sera légèrement plus faible. Pendant cette période, plus la LED est allumée longtemps, plus elle sera lumineuse. C'est-à-dire, sur une période donnée, plus l'impulsion est large, plus grande sera la « force du signal électrique » sortie par le microcontrôleur.

Voici la fonction nécessaire pour écrire l'onde PWM.

— `analogWrite(pin, value)`

Écrit une valeur analogique (onde PWM) sur une broche. Différentes tensions de sortie (0-5V) peuvent être simulées en générant un signal d'impulsion spécifié. La broche conservera ce signal jusqu'à ce qu'elle soit appelée par une nouvelle instruction de lecture ou d'écriture.

### Syntaxe

`analogWrite(pin, value)`

### Paramètres

- `pin` : la broche Arduino à écrire. Types de données autorisés : `int`.
- `value` : le cycle de travail : entre 0 (toujours éteint) et 255 (toujours allumé). Types de données autorisés : `int`.

### Exemple d'écriture analogique

```
int pin = 9;           //connect to pwm pin

void setup() {
  pinMode(pin, OUTPUT); // sets the pin as output
}

void loop() {
  for (int i = 0 ; i<255 ; i++){
    analogWrite(pin, i); //analogWrite values from 0 to 255
    delay(30);
  }
}
```

### Notes et avertissements

- En regardant de près la carte R3, les broches marquées du symbole « ~ » ont une fonction de sortie analogique.
- Les sorties PWM générées sur les broches 5 et 6 auront des cycles de travail plus élevés que prévu. Cela est dû aux interactions avec les fonctions `millis()` et `delay()`, qui partagent le même minuteur interne utilisé pour générer ces sorties PWM. Cela sera surtout remarqué sur les réglages de cycle de travail faible (par exemple, 0 - 10) et peut entraîner une valeur de 0 ne désactivant pas complètement la sortie sur les broches 5 et 6.

### Composants associés

Ci-dessous les composants associés, vous pouvez cliquer pour apprendre à les utiliser.

## 5.2.1 2.1 Estompage

Ce projet est similaire à *1.1 Bonjour, LED!*, la différence réside dans le type de signal. Le premier consiste à allumer ou éteindre la LED en émettant un signal numérique (0&1), tandis que ce projet contrôle la luminosité de la LED en émettant un signal analogique.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

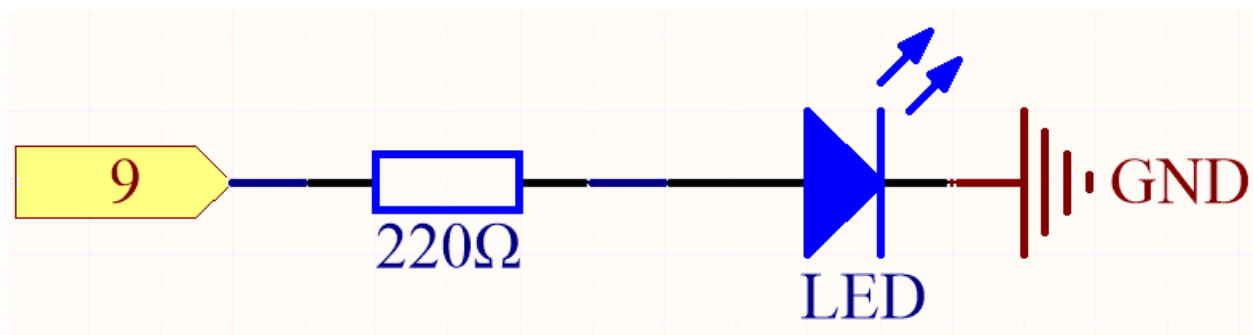
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

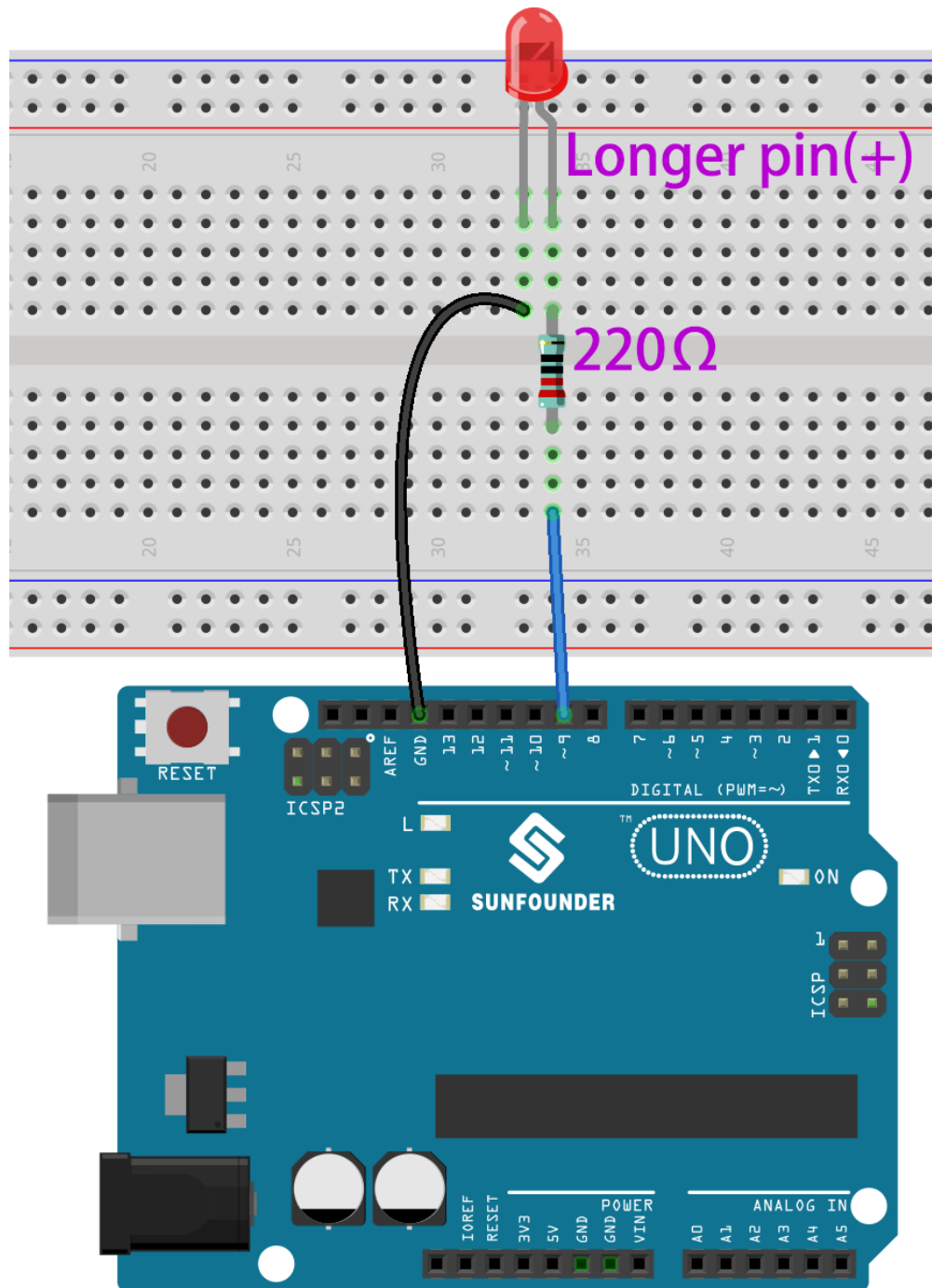
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	

### Schéma



### Câblage



## Code

### Note :

- Vous pouvez ouvrir le fichier 2.1.fading.ino dans le chemin 3in1-kit\basic\_project\2.analogWrite\2.1.fading.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, vous pouvez voir la LED respirer.

### 5.2.2 2.2 Lumière Colorée

Comme nous le savons, la lumière peut être superposée. Par exemple, mélanger de la lumière bleue et verte donne une lumière cyan, du rouge et du vert donne une lumière jaune. Cela s'appelle « La méthode additive de mélange des couleurs ».

— [Couleur additive - Wikipédia](#)

En se basant sur cette méthode, nous pouvons utiliser les trois couleurs primaires pour mélanger la lumière visible de n'importe quelle couleur selon des proportions spécifiques. Par exemple, l'orange peut être produit avec plus de rouge et moins de vert.

Dans ce chapitre, nous utiliserons une LED RVB pour explorer le mystère du mélange des couleurs additives !

La LED RVB équivaut à encapsuler une LED rouge, verte et bleue sous un même capuchon de lampe, et les trois LED partagent une broche de cathode commune. Puisque le signal électrique est fourni à chaque broche d'anode, la lumière de la couleur correspondante peut être affichée. En changeant l'intensité du signal électrique de chaque anode, elle peut produire diverses couleurs.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

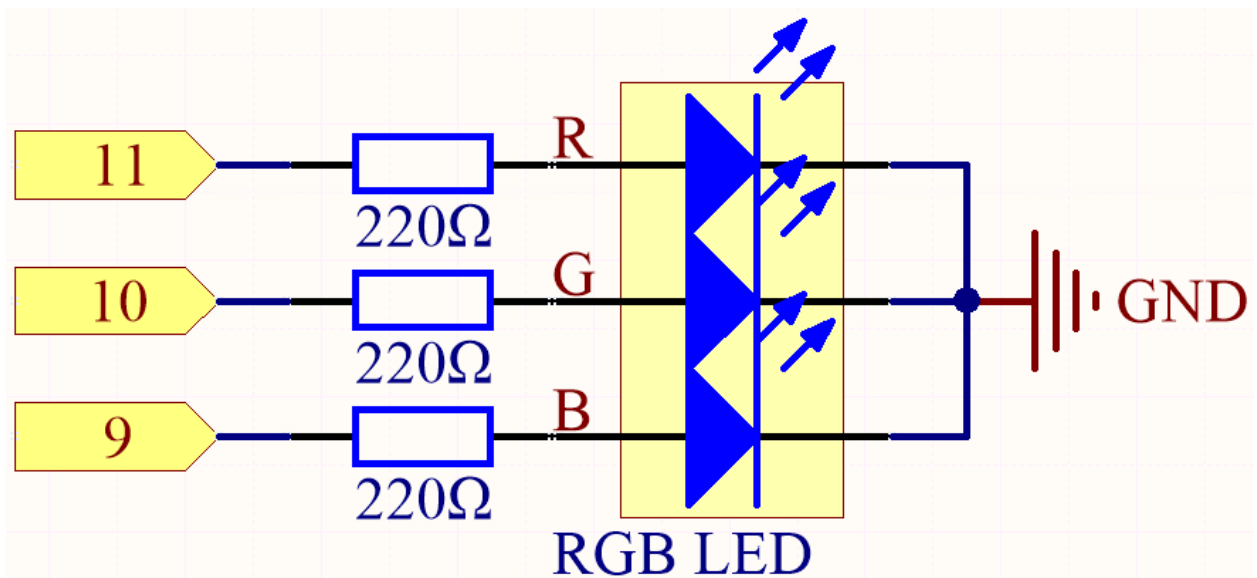
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

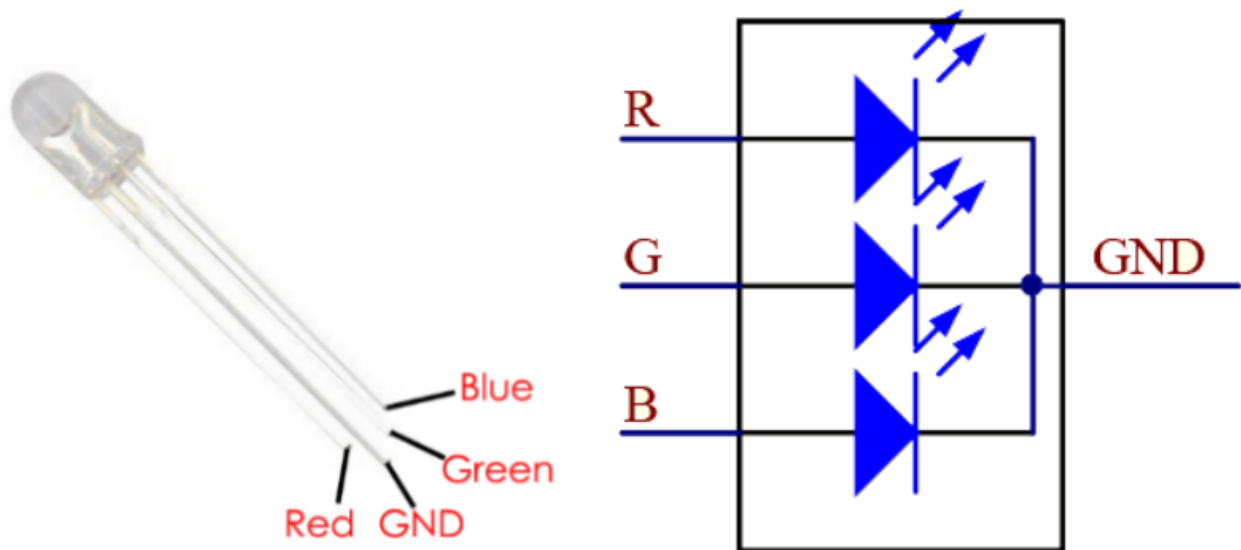
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED RVB</i>	

#### Schéma

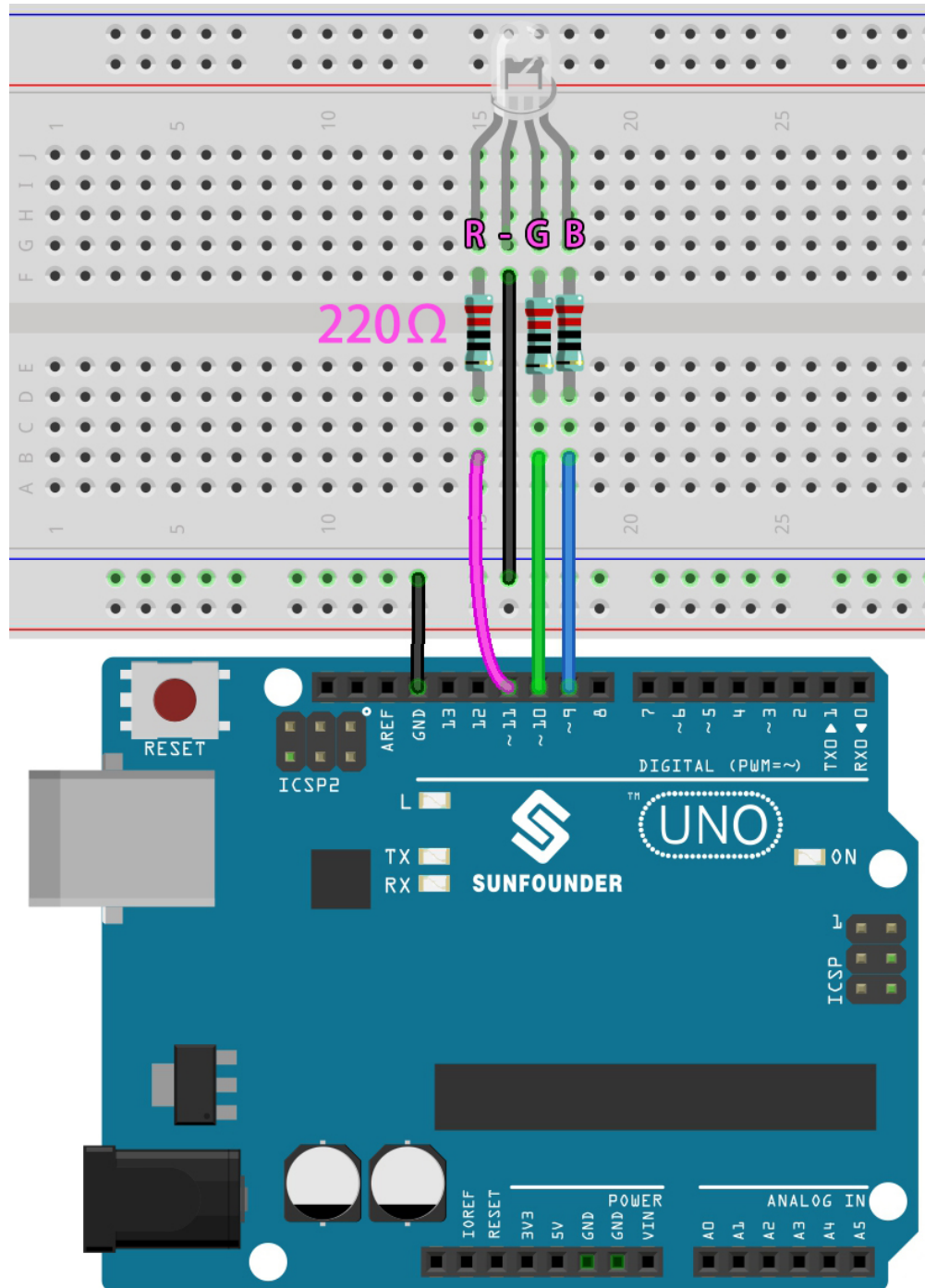


Les broches PWM 11, 10 et 9 contrôlent respectivement les broches Rouge, Verte et Bleue de la LED RVB, et connectent la broche de cathode commune à GND. Cela permet à la LED RVB d'afficher une couleur spécifique en superposant la lumière sur ces broches avec différentes valeurs PWM.

#### Câblage



Une LED RVB a 4 broches : la broche la plus longue est la broche de cathode commune, généralement connectée à GND, la broche à gauche de la plus longue est Rouge, et les 2 broches à droite sont Vert et Bleu.



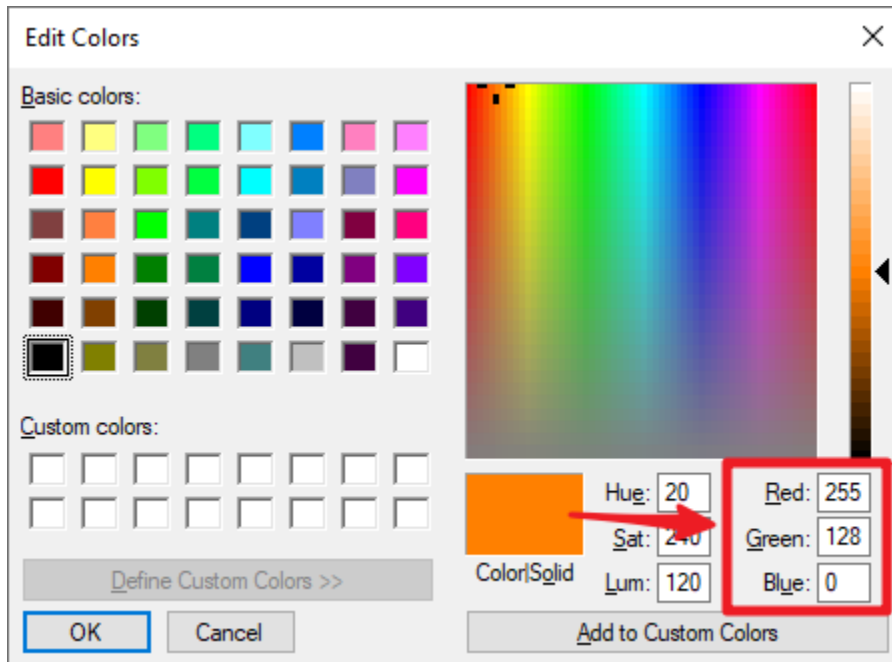
### Code

Ici, nous pouvons choisir notre couleur préférée dans un logiciel de dessin (comme Paint) et l'afficher avec la LED RVB.

### Note :

- Vous pouvez ouvrir le fichier `2.2.colorful_light.ino` dans le chemin `3in1-kit\basic_project\2.analogWrite\2.2.colorful_light`.
- Ou copiez ce code dans **Arduino IDE**.

— Ou téléchargez le code via l'Arduino Web Editor.



Écrivez la valeur RVB dans `color_set()`, et vous pourrez voir la lumière RVB afficher les couleurs que vous souhaitez.

#### Comment ça fonctionne ?

Dans cet exemple, la fonction utilisée pour attribuer des valeurs aux trois broches de RVB est emballée dans une sous-fonction indépendante `color()`.

```
void color (unsigned char red, unsigned char green, unsigned char blue)
{
    analogWrite(redPin, red);
    analogWrite(greenPin, green);
    analogWrite(bluePin, blue);
}
```

Dans `loop()`, la valeur RVB sert d'argument d'entrée pour appeler la fonction `color()` pour réaliser que le RVB peut émettre différentes couleurs.

```
void loop()
{
    color(255, 0, 0); // red
    delay(1000);
    color(0, 255, 0); // green
    delay(1000);
    color(0, 0, 255); // blue
    delay(1000);
}
```

## 5.3 3. Lecture Numérique

Les capteurs capturent les informations du monde réel, qui sont ensuite communiquées à la carte principale via des broches (certaines numériques, d'autres analogiques) afin que l'ordinateur puisse connaître la réalité de la situation.

Ainsi, la carte Arduino peut connaître l'état de fonctionnement des capteurs numériques en lisant la valeur des broches numériques comme les boutons, le module d'évitement d'obstacle IR.

Voici les fonctions requises.

- `pinMode(pin, mode)` : Configure la broche spécifique comme INPUT (entrée) ou OUTPUT (sortie), ici elle doit être réglée sur INPUT.

### Syntaxe

`pinMode(pin, mode)`

### Paramètres

- `pin` : le numéro de broche Arduino à configurer.
  - `mode` : INPUT, OUTPUT, ou INPUT\_PULLUP.
- `digitalRead(pin)` : Lit la valeur (état de niveau) de la broche numérique spécifiée.

### Syntaxe

`digitalRead(pin)`

### Paramètres

- `pin` : le numéro de broche Arduino que vous souhaitez lire

### Retours

HIGH ou LOW

### Exemple de Lecture Numérique

```
int ledPin = 13; // LED connected to digital pin 13
int inPin = 7;   // pushbutton connected to digital pin 7
int val = 0;     // variable to store the read value

void setup() {
  pinMode(ledPin, OUTPUT); // sets the digital pin 13 as output
  pinMode(inPin, INPUT);   // sets the digital pin 7 as input
}

void loop() {
  val = digitalRead(inPin); // read the input pin
  digitalWrite(ledPin, val); // sets the LED to the button's value
}
```

### Notes et Avertissements

1. Pull Up & Pull Down.

`digitalRead()` peut produire des valeurs aléatoires et indéterminées si la broche ne reçoit pas un signal de niveau. Ainsi, diriger les broches d'entrée vers un état connu peut rendre le projet plus fiable. Lors de l'utilisation d'un composant d'entrée tel qu'un bouton, il est généralement nécessaire de connecter une résistance de pull-up ou pull-down en parallèle à la broche d'entrée numérique.

Outre la connexion d'une résistance de pull-up, vous pouvez également régler le mode de la broche sur INPUT\_PULLUP dans le code, par exemple `pinMode(pin, INPUT_PULLUP)`. Dans ce cas, la broche accèdera à la résistance de pull-up intégrée de l'Atmega via le logiciel, et cela aura le même effet que la connexion d'une résistance de pull-up.

2. À propos de la Pin13.



Toutes les broches numériques (1-13) sur la carte R3 peuvent être utilisées comme `digitalRead()`. Cependant, la broche numérique 13 est plus difficile à utiliser comme entrée numérique que les autres broches numériques. Car elle connecte une LED et une résistance, elle est soudée sur la plupart des cartes. Si vous activez sa résistance de pull-up interne de 20k, elle se maintiendra autour de 1,7V au lieu des 5V attendus car la LED embarquée et la résistance en série tirent le niveau de tension vers le bas, ce qui signifie qu'elle retourne toujours LOW. Si vous devez utiliser la pin 13 comme une entrée numérique, réglez son `pinMode()` sur INPUT et utilisez une résistance de pull-down externe.

### 3. Broches analogiques.

Si les broches numériques ne sont pas suffisantes, les broches analogiques (A0-A5) peuvent également être utilisées comme broches numériques. Il faut les régler sur INPUT avec `pinMode(pin,mode)`.

## Composants Associés

Ci-dessous les composants associés, vous pouvez cliquer pour apprendre à les utiliser.

### 5.3.1 3.0 Moniteur Série

Dans l'IDE Arduino, il y a un moniteur série qui vous permet d'envoyer des messages de votre ordinateur vers la carte Arduino (via USB) et également de recevoir des messages de l'Arduino.

Dans ce projet, nous apprendrons donc à recevoir des données de la carte Arduino.

**Note :** Sur les Uno, Nano, Mini et Mega, les broches 0 et 1 sont utilisées pour la communication avec l'ordinateur. Connecter quelque chose à ces broches peut interférer avec cette communication, y compris provoquer des échecs de téléchargement vers la carte.

## Utilisation du Moniteur Série

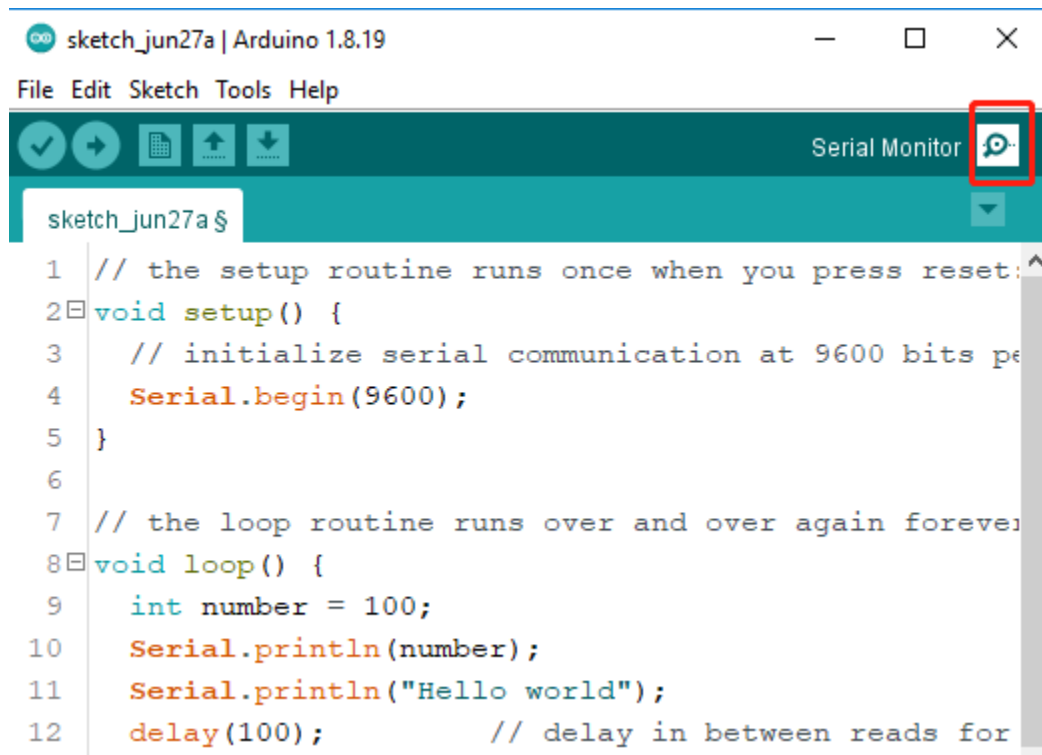
1. Ouvrez l'IDE Arduino et collez le code suivant.

```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

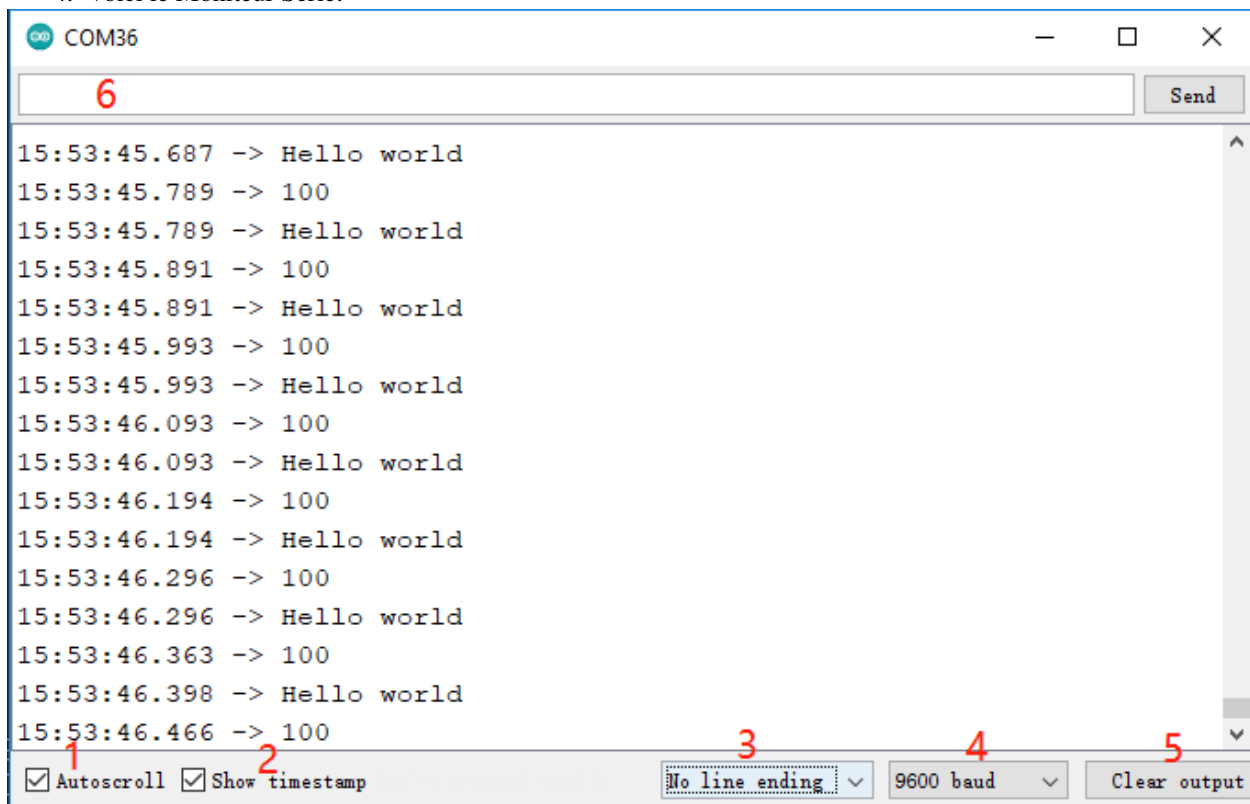
// the loop routine runs over and over again forever:
void loop() {
  int number = 100;
  Serial.println(number);
  Serial.println("Hello world");
  delay(100);          // delay in between reads for stability
}
```

- `Serial.begin()` : Définit le débit de données en bits par seconde (baud) pour la transmission de données série, ici réglé sur 9600.
- `Serial.println()`.

2. Sélectionnez la bonne carte et le bon port pour télécharger le code.
3. Dans la barre d'outils, cliquez sur l'icône de la loupe pour activer le Moniteur Série.



4. Voici le Moniteur Série.



- 1 : Option pour choisir entre défilement automatique ou non.
- 2 : Option pour afficher l'horodatage avant les données affichées sur le Moniteur Série.
- 3 : Sélection de fin, sélectionnez les caractères de fin ajoutés aux données envoyées à l'Arduino. Les sélections comprennent :

- **No line Ending** envoie juste ce que vous tapez ;
- **Newline** est `\n` et envoie un code ASCII de nouvelle ligne après ce que vous tapez ;
- **Carriage Return** est `\r`, qui enverra un caractère de retour chariot ASCII après ce que vous tapez ;
- **Both NL & CR** est `\r\n` qui enverra à la fois un retour chariot et un caractère de nouvelle ligne après ce que vous tapez.
- **4** : Sélectionnez la vitesse de communication entre la carte Arduino et le PC. Cette valeur DOIT être la même que celle définie dans `Serial.begin()`.
- **5** : Effacez tout le texte sur la console de sortie.
- **6** : Une zone de texte pour envoyer des caractères à la carte Arduino, voir [5.12 Lecture Série](#) pour un tutoriel.

### 5.3.2 3.1 Lecture de la Valeur du Bouton

Dans les projets précédents, nous avons utilisé la fonction de sortie, dans ce chapitre, nous utiliserons la fonction d'entrée pour lire la valeur du bouton.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

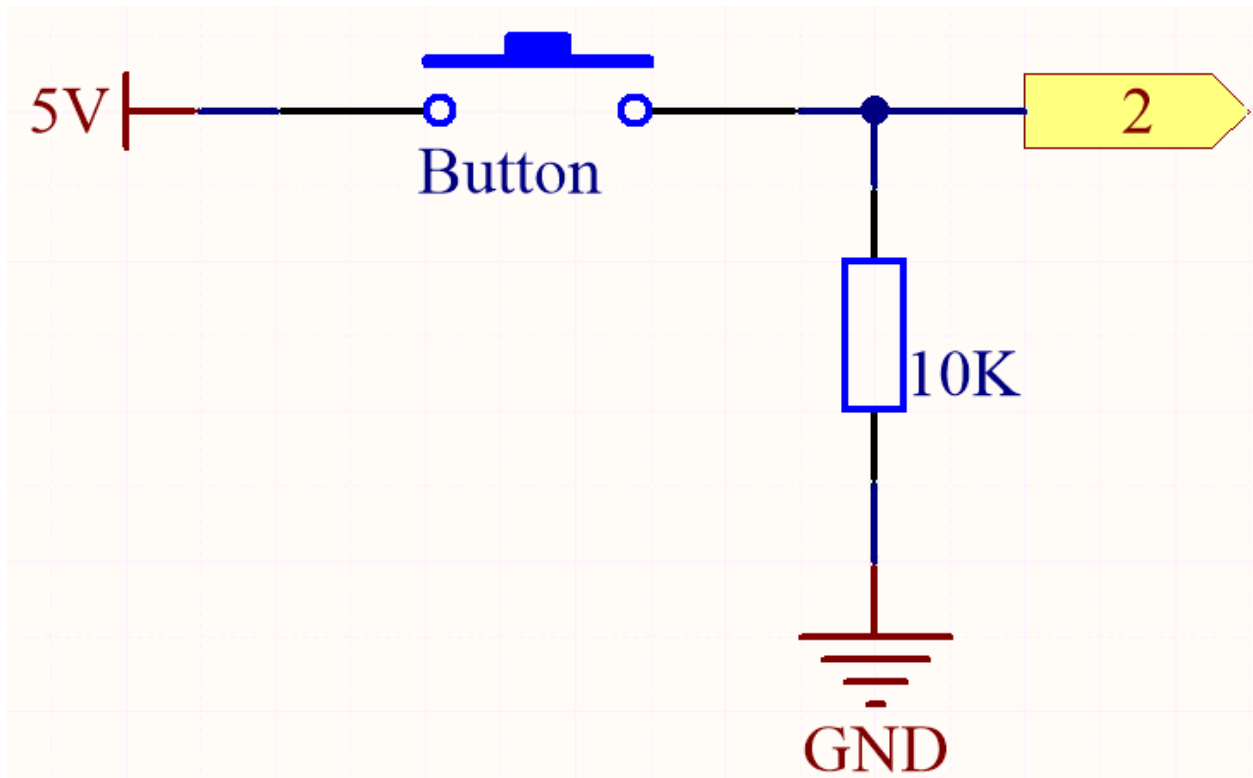
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

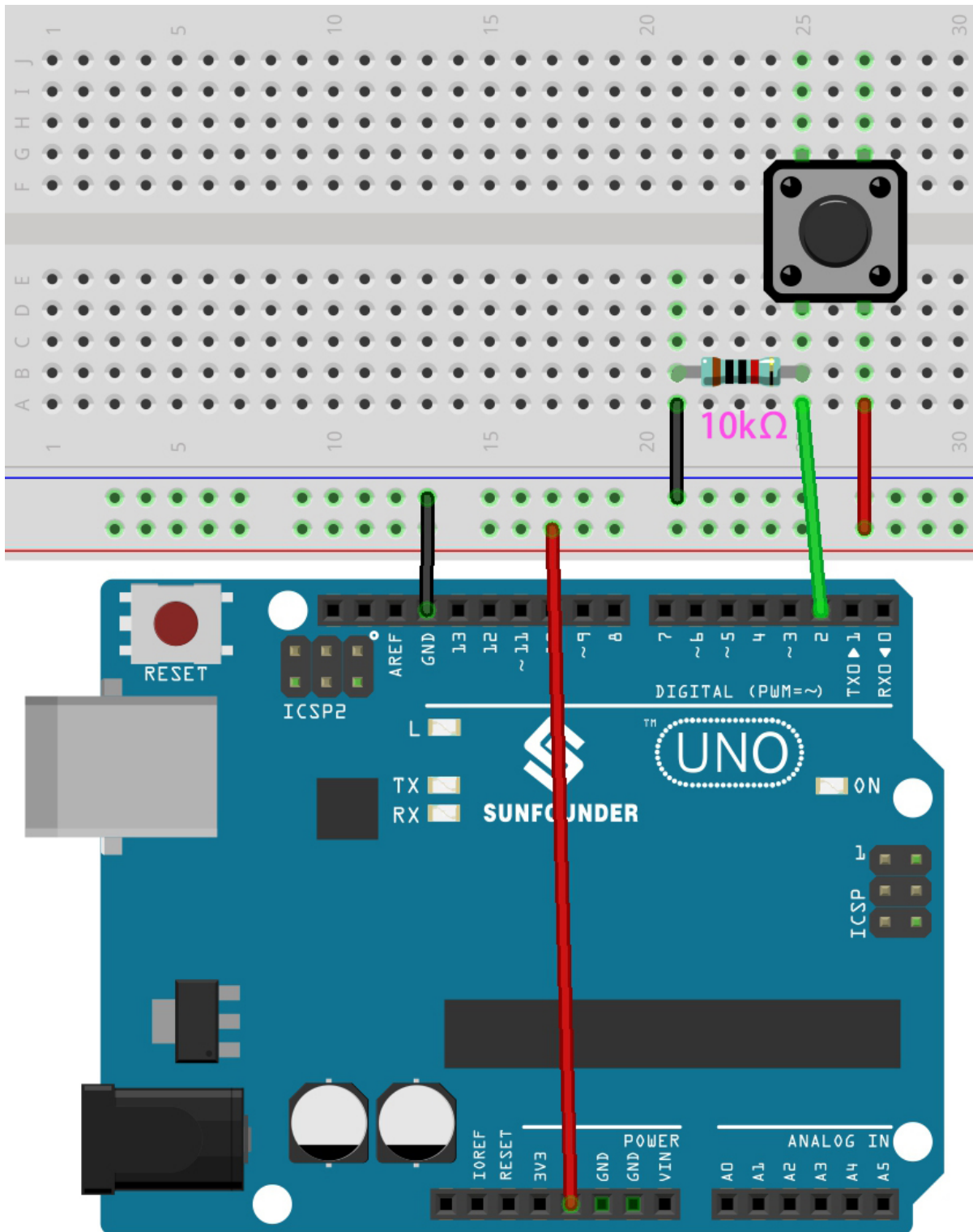
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Bouton</i>	

#### Schéma



Un côté de la broche du bouton est connecté à 5V, et l'autre côté de la broche est connecté à la broche 2, donc lorsque le bouton est pressé, la broche 2 sera haute. Cependant, lorsque le bouton n'est pas pressé, la broche 2 est dans un état suspendu et peut être haute ou basse. Afin d'obtenir un niveau bas stable lorsque le bouton n'est pas pressé, la broche 2 doit être reconnectée à GND via une résistance de pull-down de 10K.

#### Câblage



Code

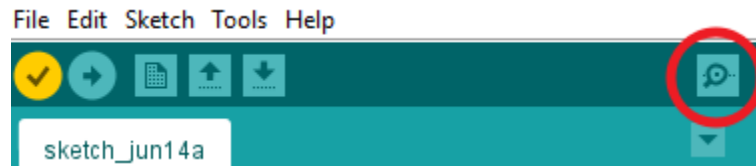
Note :

— Vous pouvez ouvrir le fichier `3.1.read_button_value.ino` dans le chemin `3in1-kit/basic_project\`

3.1.read\_button\_value.

- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, cliquez sur l'icône de la loupe dans le coin supérieur droit de l'IDE Arduino (Moniteur Série).



Lorsque vous appuyez sur le bouton, le Moniteur Série affichera « 1 ».

### 5.3.3 3.2 Ressentir le Magnétisme

Le type le plus courant d'interrupteur à lames souples contient une paire de lames métalliques flexibles magnétisables, dont les extrémités sont séparées par un petit espace lorsque l'interrupteur est ouvert.

Un champ magnétique provenant d'un électroaimant ou d'un aimant permanent fera s'attirer les lames, complétant ainsi un circuit électrique. La force de rappel des lames les amène à se séparer et à ouvrir le circuit, lorsque le champ magnétique cesse.

Un exemple courant d'application d'un interrupteur à lames souples est de détecter l'ouverture d'une porte ou d'une fenêtre, pour une alarme de sécurité.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

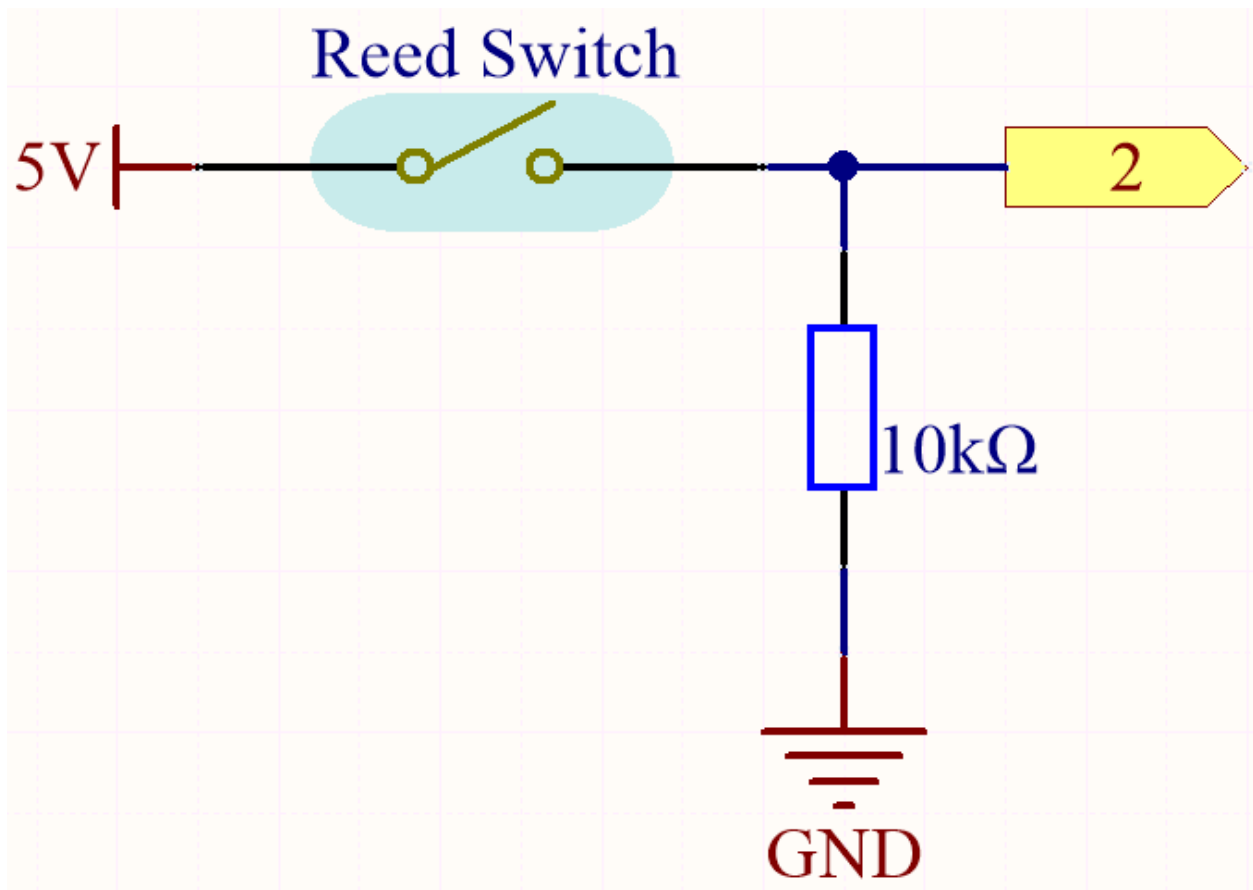
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Interrupteur à Lame Souple</i>	-

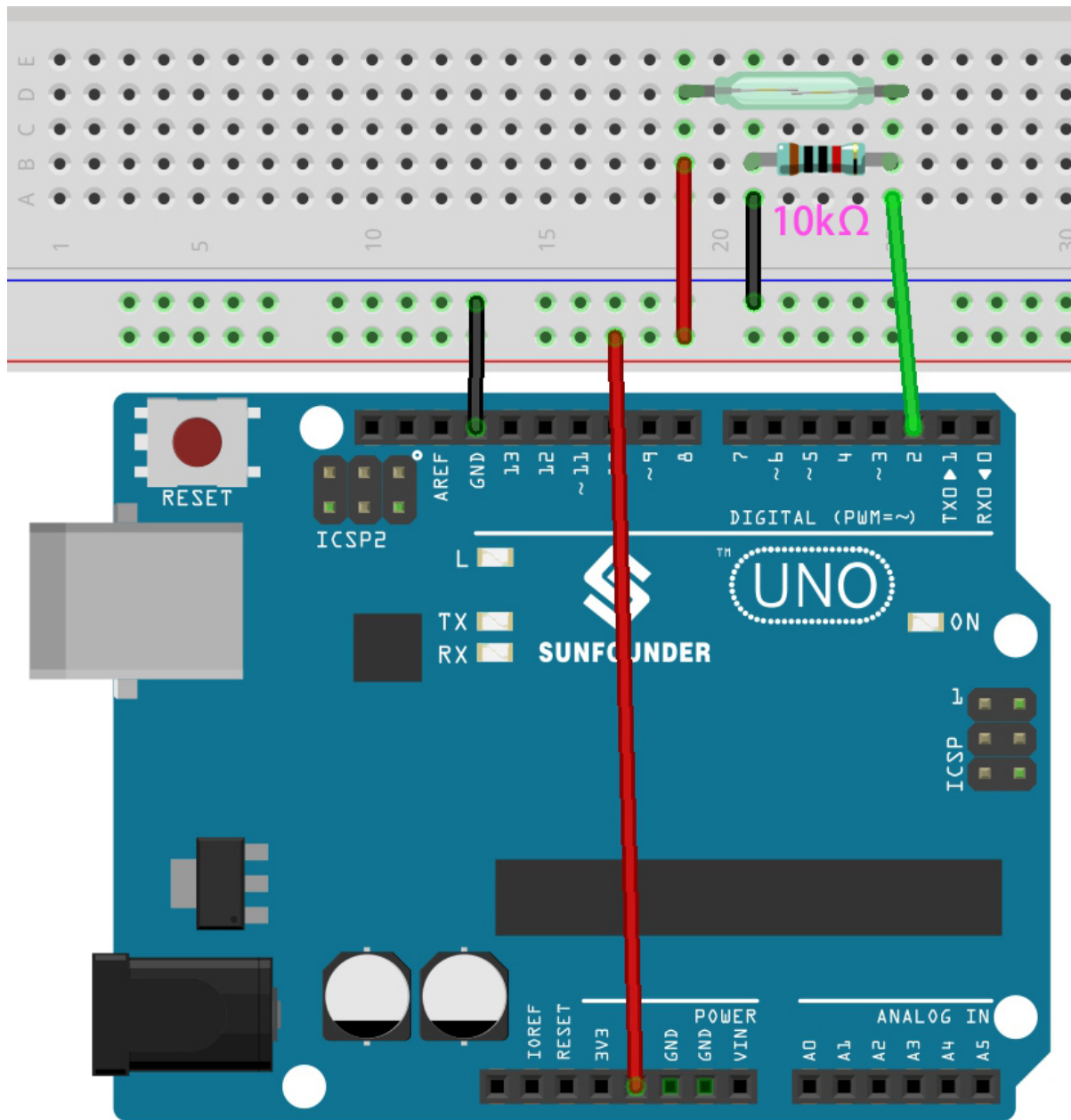
#### Schéma



Par défaut, la broche 2 est basse ; et deviendra haute lorsque l'aimant est près de l'interrupteur à lames.

Le but de la résistance de 10K est de maintenir la broche 2 à un niveau bas stable lorsqu'aucun aimant n'est à proximité.

#### Câblage



## Code

### Note :

- Vous pouvez ouvrir le fichier 3.2.feel\_the\_magnetism.ino dans le chemin 3in1-kit\basic\_project\3.2.feel\_the\_magnetism.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, lorsque un aimant est près de l'interrupteur à lames, le moniteur série affichera 1.



### 5.3.4 3.3 Détecter l'Obstacle

Ce module est couramment installé sur les voitures et les robots pour juger de l'existence d'obstacles devant eux. Il est également largement utilisé dans les appareils portatifs, les robinets d'eau, etc.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

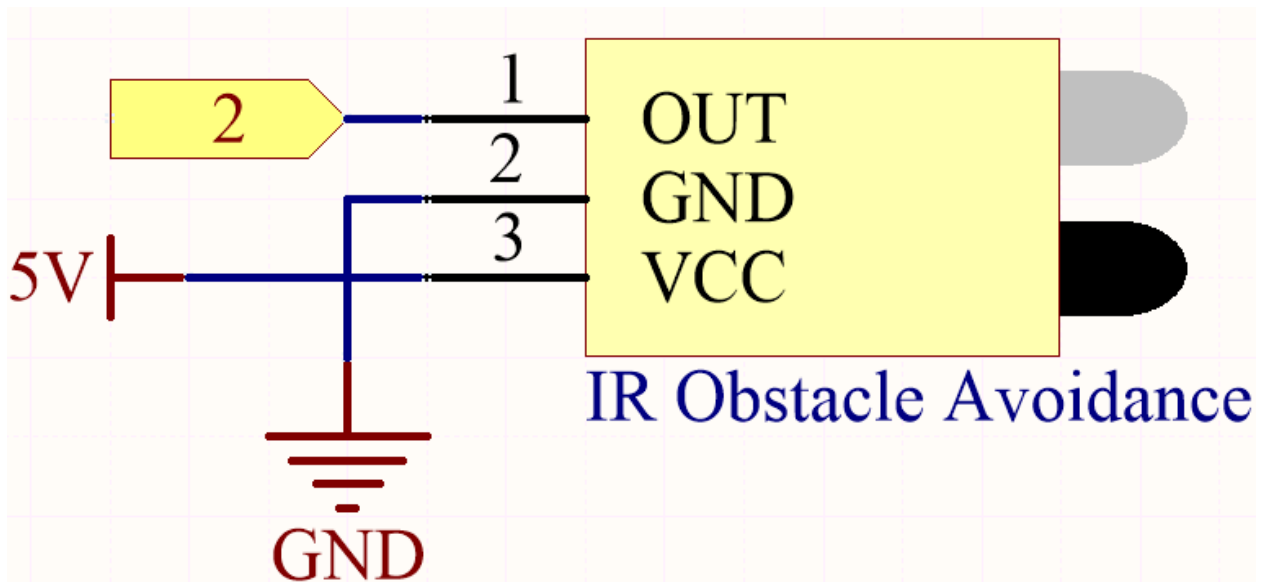
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

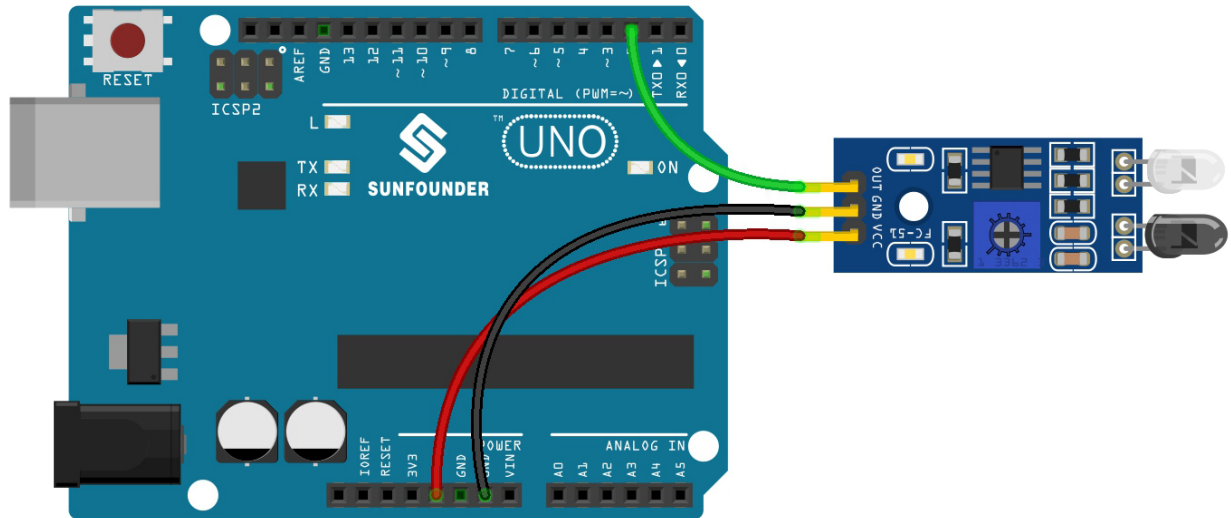
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module d'Évitement d'Obstacle</i>	

#### Schéma



La broche numérique 2 est utilisée pour lire le signal du Module d'Évitement d'Obstacles IR. Nous connectons le VCC du Module Capteur IR à 5V, GND à GND, OUT à la broche numérique 2.

#### Câblage



### Code

#### Note :

- Vous pouvez ouvrir le fichier `3.3.detect_the_obstacle.ino` dans le chemin `3in1-kit\basic_project\3.3.detect_the_obstacle`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Lorsque le module d'évitement d'obstacles IR détecte quelque chose bloquant devant lui, [0] apparaîtra sur le moniteur série, sinon [1] sera affiché.

### 5.3.5 3.4 Détecter la Ligne

Le module de suivi de ligne est utilisé pour détecter s'il y a des zones noires sur le sol, comme des lignes noires collées avec du ruban électrique.

L'une de ses LED émet une lumière infrarouge appropriée vers le sol, et la surface noire a une capacité relativement forte à absorber la lumière et une capacité de réflexion plus faible. Les surfaces blanches sont l'inverse. S'il détecte de la lumière réfléchie, cela signifie que le sol est actuellement blanc. Si elle n'est pas détectée, cela signifie noir.

C'est ainsi que cela fonctionne.

#### Composants requis

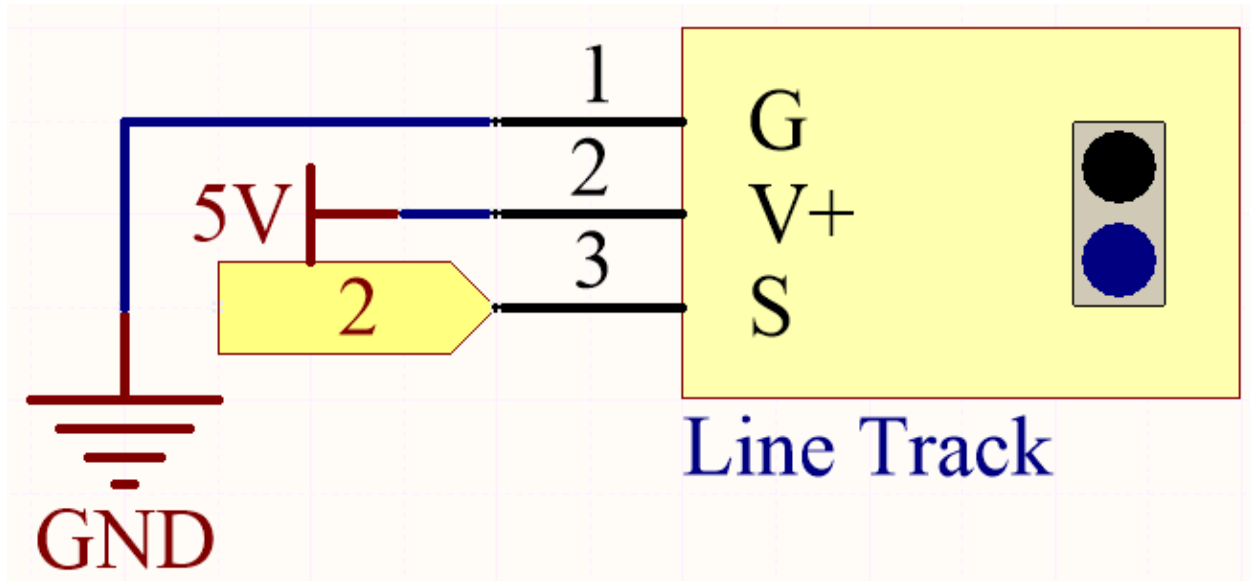
Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

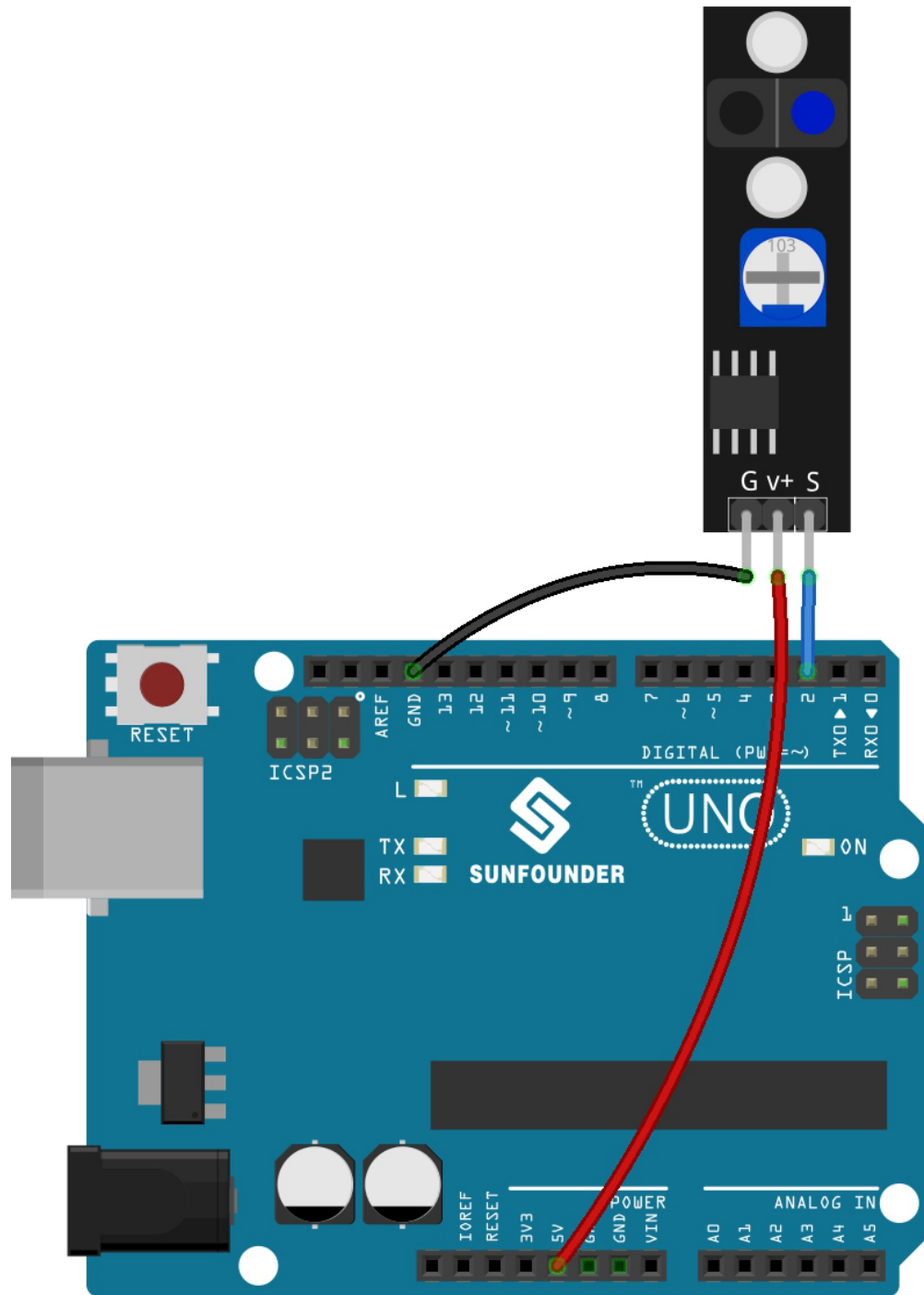
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module de Suivi de Ligne</i>	

**Schéma**

La broche numérique 2 est utilisée pour lire le signal du module de suivi de ligne. Nous connectons le VCC du module à 5V, GND à GND, OUT à la broche numérique 2.

**Câblage**



## Code

### Note :

- Vous pouvez ouvrir le fichier `3.4.detect_the_line.ino` dans le chemin `3in1-kit\basic_project\3.4.detect_the_line`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Lorsque le module de suivi de ligne détecte qu'il y a une ligne noire, il apparaît [1] sur le Moniteur Série; sinon, [0] est affiché.

## 5.4 4. Lecture Analogique

L'Arduino peut lire les capteurs analogiques connectés via les broches analogiques.

La carte R3 contient un convertisseur analogique-numérique multicanal de 10 bits. Cela signifie qu'elle mappe la tension d'entrée entre 0 et la tension de fonctionnement (5V ou 3.3V) en une valeur entière entre 0 et 1023.

Vous avez besoin de la fonction `analogRead(pin)` pour lire la valeur de la broche analogique.

- `analogRead(pin)` : Lit la valeur de la broche analogique spécifiée.

### Syntaxe

`analogRead(pin)`

### Paramètres

- `pin` : le nom de la broche d'entrée analogique à lire (de A0 à A5).

### Retours

0-1023. Type de données : `int`.

### Exemple de Lecture Analogique

```
int analogPin = A0; // device connected to analog pin A0
                    // outside leads to ground and +5V
int val = 0; // variable to store the value read

void setup() {
  Serial.begin(9600); // setup serial
}

void loop() {
  val = analogRead(analogPin); // read the input pin
  Serial.println(val); // debug value
}
```

### Notes et Avertissements

- Les broches analogiques sont A0-A5.
- Vous n'avez pas besoin d'appeler `pinMode()` avant d'appeler la broche analogique, mais si la broche était précédemment réglée sur OUTPUT, la fonction `analogRead()` ne fonctionnera pas correctement. Dans ce cas, vous devez appeler `pinMode()` pour la remettre en INPUT.

### Composants Associés

Ci-dessous les composants associés, vous pouvez cliquer pour apprendre à les utiliser.

### 5.4.1 4.1 Tournez le Bouton

Le potentiomètre est un composant résistif à 3 bornes dont la valeur de résistance peut être ajustée selon une variation régulière.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

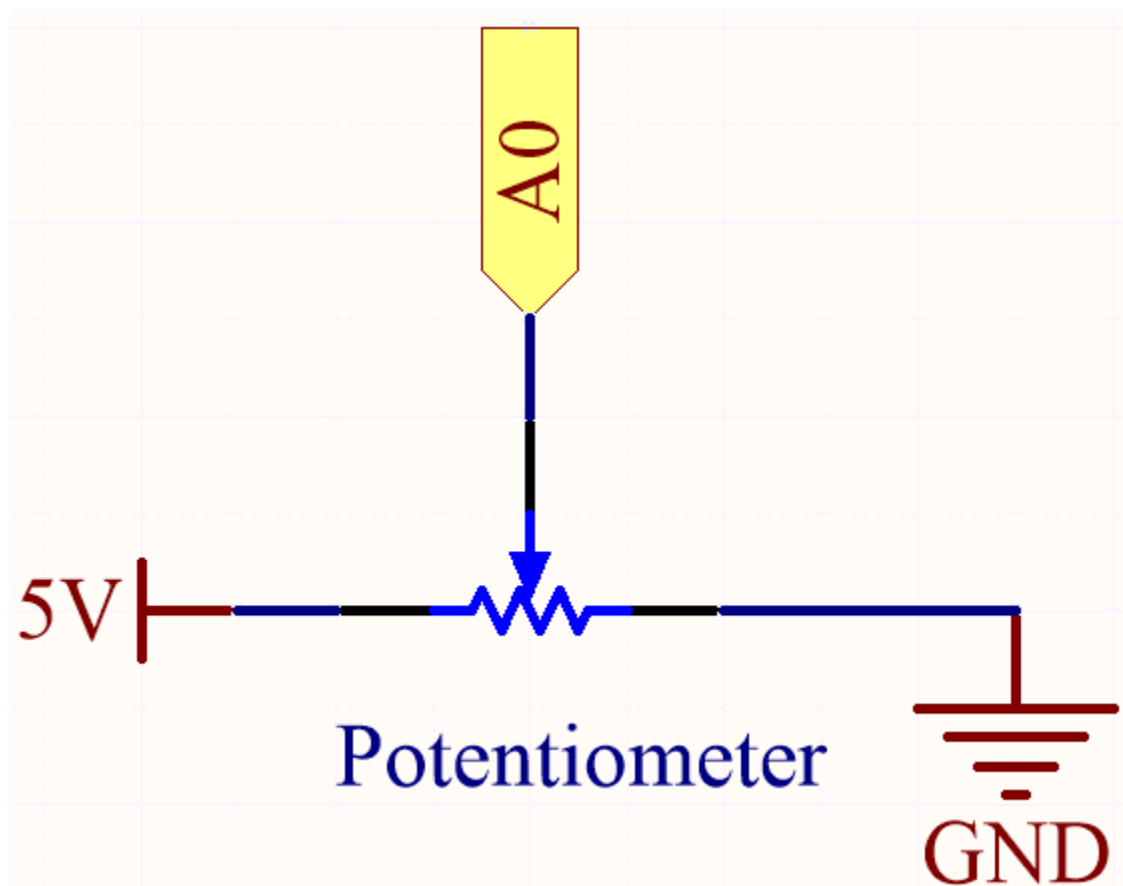
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

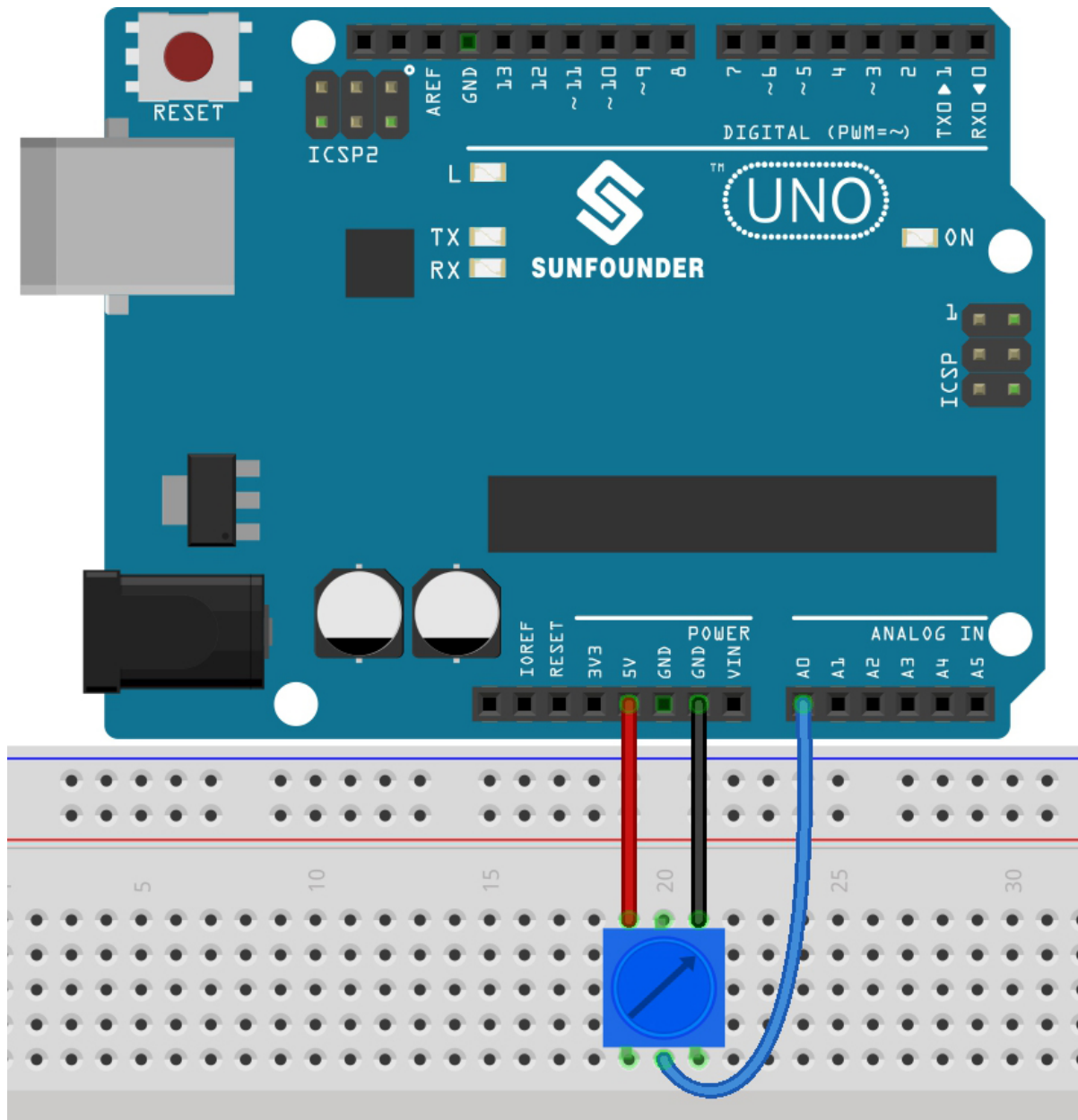
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Potentiomètre</i>	

### Schéma



Dans cet exemple, nous utilisons la broche analogique (A0) pour lire la valeur du potentiomètre. En tournant l'axe du potentiomètre, vous pouvez changer la répartition de la résistance entre ces trois broches, modifiant ainsi la tension sur la broche du milieu. Lorsque la résistance entre la broche du milieu et une broche extérieure connectée à 5V est proche de zéro (et la résistance entre la broche du milieu et l'autre broche extérieure est proche de 10k), la tension à la broche du milieu est proche de 5V. L'opération inverse (la résistance entre la broche du milieu et une broche extérieure connectée à 5V est proche de 10k) rendra la tension à la broche du milieu proche de 0V.

## Câblage



## Code

## Note :

- Vous pouvez ouvrir le fichier 4.1.turn\_the\_knob.ino dans le chemin 3in1-kit\basic\_project\4.1.turn\_the\_knob.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après avoir téléchargé les codes sur la carte, vous pouvez ouvrir le moniteur série pour voir la valeur lue de la broche. En tournant l'axe du potentiomètre, le moniteur série affichera la valeur 0~1023.

### 5.4.2 4.2 Ressentir la Lumière

Le photo-résistor est un dispositif typique pour les entrées analogiques et il est utilisé de manière très similaire à un potentiomètre. Sa valeur de résistance dépend de l'intensité de la lumière, plus la lumière irradiée est forte, plus sa valeur de résistance est petite ; inversement, elle augmente.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

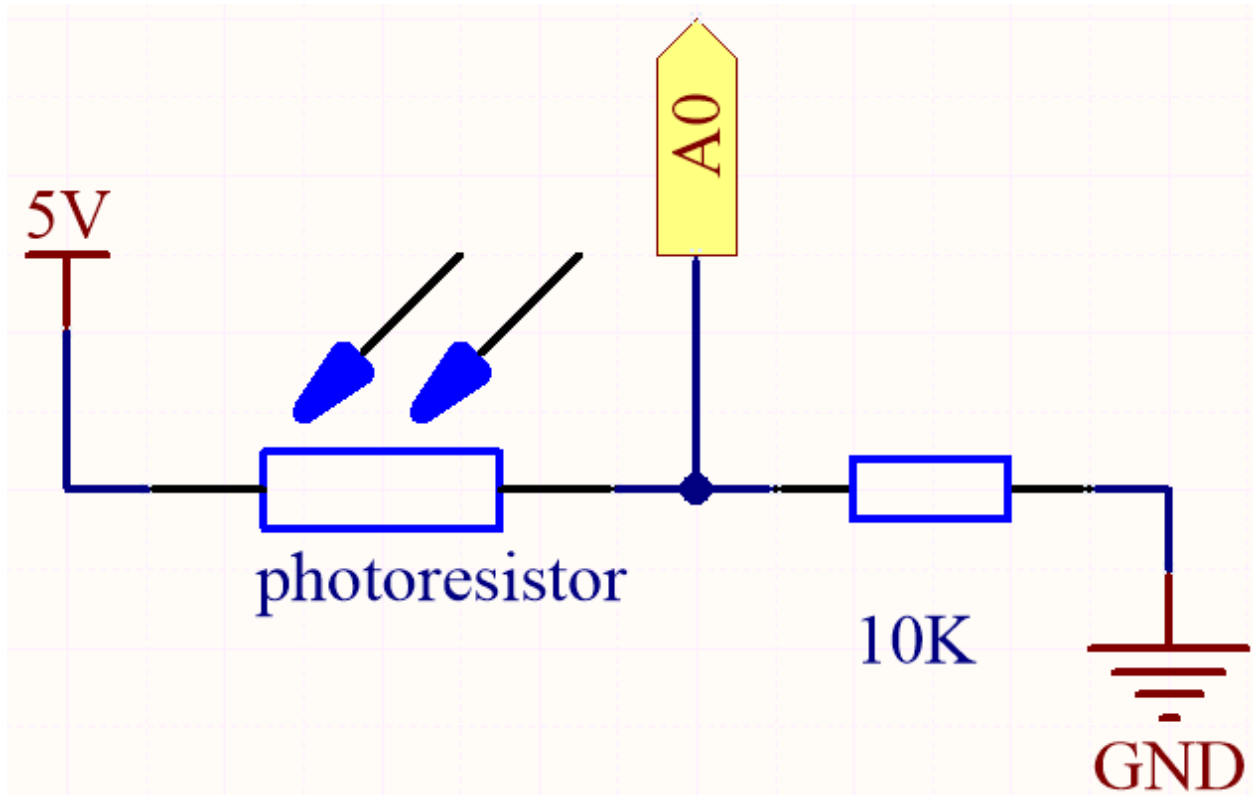
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Photorésistance</i>	

#### Schéma



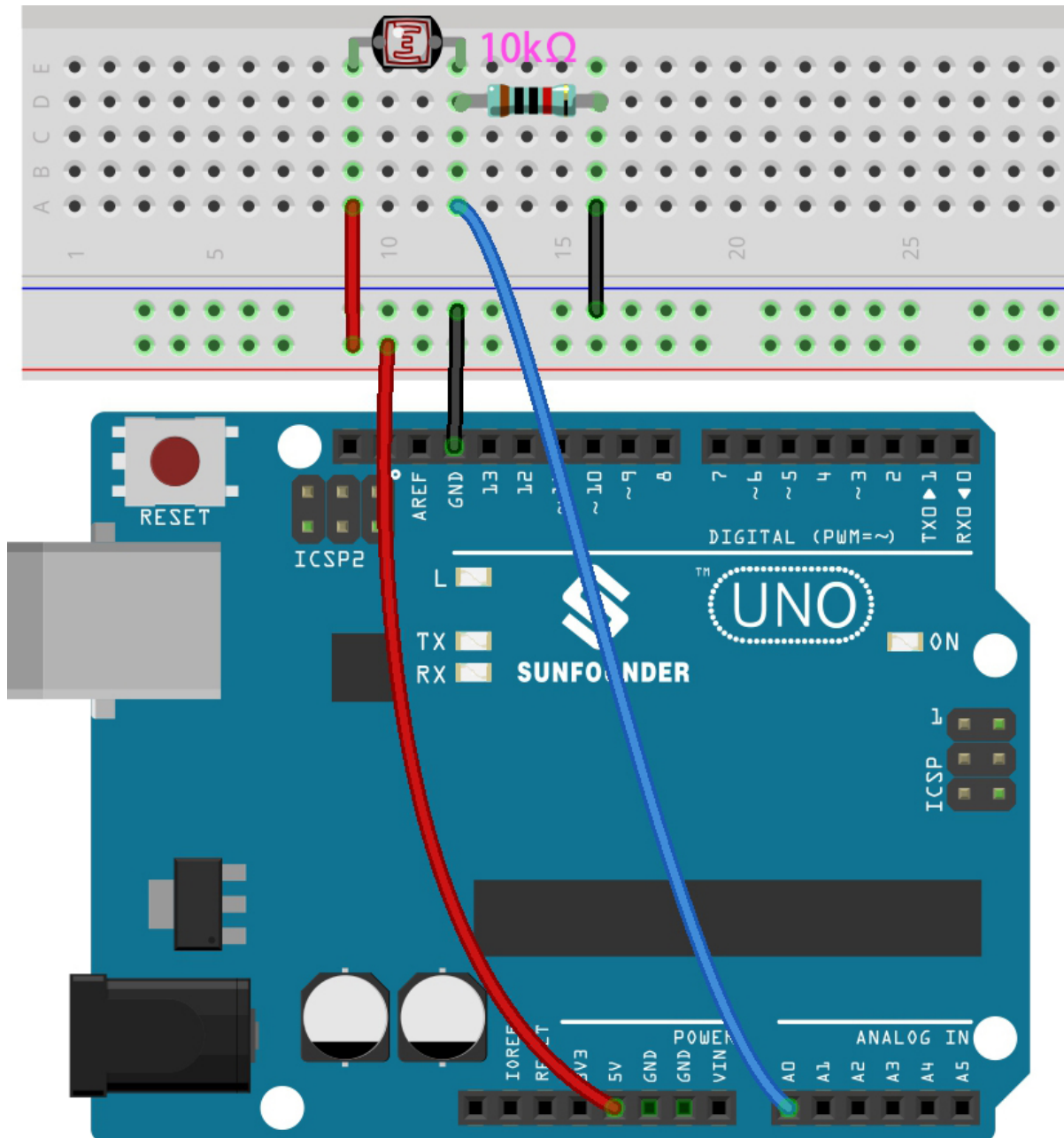


Dans ce circuit, la résistance de 10K et le photo-résistor sont connectés en série, et le courant les traversant est le même. La résistance de 10K agit comme une protection, et la broche A0 lit la valeur après la conversion de tension du photo-résistor.

Lorsque la lumière est renforcée, la résistance du photo-résistor diminue, alors sa tension diminue, donc la valeur de la broche A0 augmentera; si la lumière est suffisamment forte, la résistance du photo-résistor sera proche de 0, et la valeur de la broche A0 sera proche de 1023. À ce moment, la résistance de 10K joue un rôle protecteur, afin que 5V et GND ne soient pas connectés ensemble, entraînant un court-circuit.

Si vous placez le photo-résistor dans une situation sombre, la valeur de la broche A0 diminuera. Dans une situation suffisamment sombre, la résistance du photo-résistor sera infinie, et sa tension sera proche de 5V (la résistance de 10K est négligeable), et la valeur de la broche A0 sera proche de 0.

#### Câblage



## Code

### Note :

- Ouvrez le fichier `4.2.feel_the_light.ino` sous le chemin `3in1-kit\basic_project\4.2.feel_the_light`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, le moniteur série affiche les valeurs du photo-résistor. Plus la luminosité ambiante actuelle est forte, plus la valeur affichée sur le moniteur série est grande.

### 5.4.3 4.3 Manipuler le Joystick

Le joystick devrait être très familier à quiconque joue régulièrement à des jeux vidéo. Il est généralement utilisé pour déplacer des personnages ou faire pivoter l'écran.

Nos mouvements peuvent être lus par le joystick, qui fonctionne sur un principe très simple. Il se compose de deux potentiomètres perpendiculaires l'un à l'autre. Ces deux potentiomètres mesurent la valeur analogique du joystick dans les directions verticale et horizontale, produisant une valeur (x,y) dans un système de coordonnées plan en angle droit.

Ce kit comprend également un joystick avec une entrée numérique. Il est activé lorsque le joystick est pressé.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

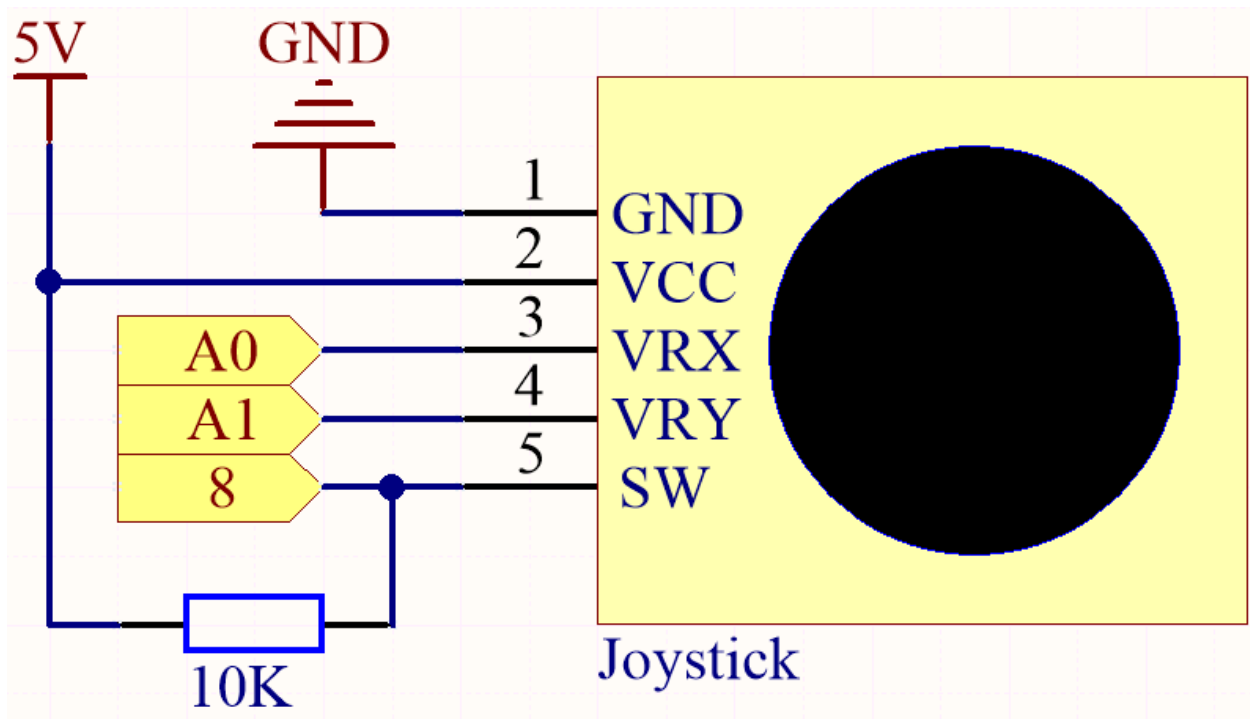
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

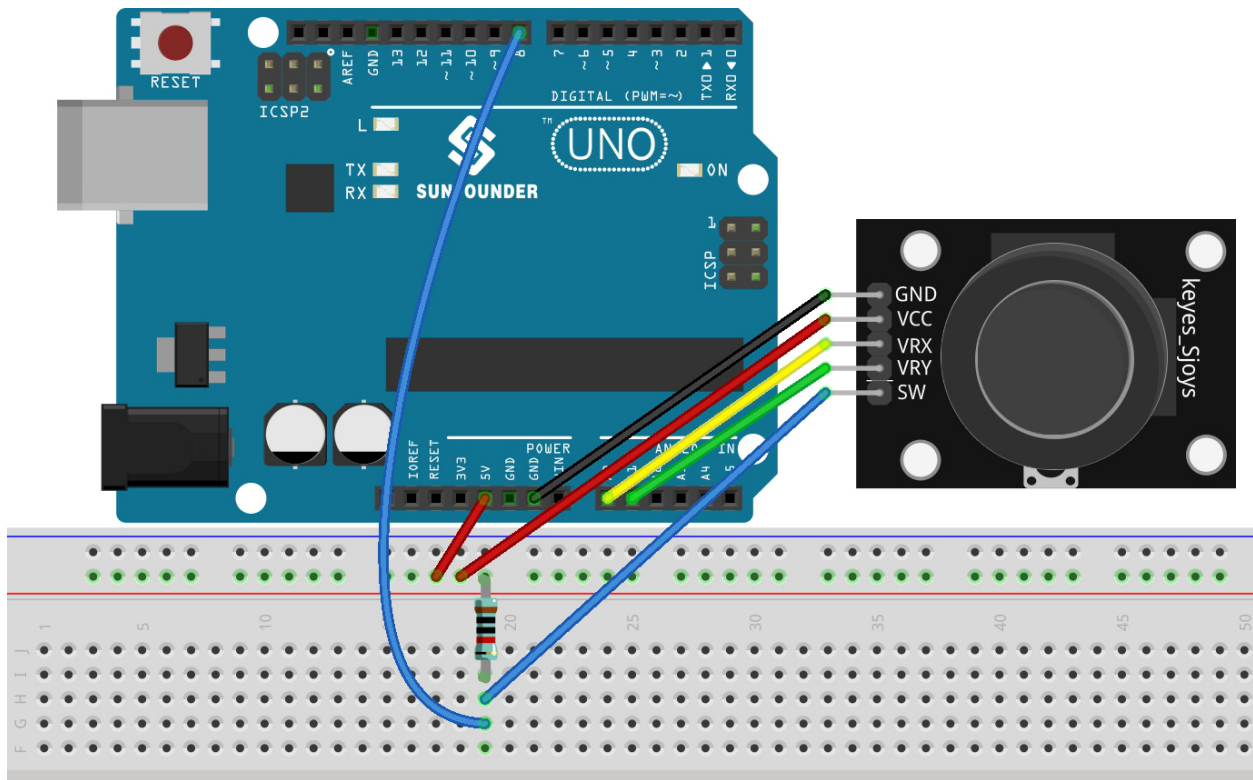
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Module de Joystick</i>	-

#### Schéma



**Note :** La broche SW est connectée à une résistance de pull-up de 10K, la raison est de pouvoir obtenir un niveau haut stable sur la broche SW (axe Z) lorsque le joystick n'est pas pressé ; sinon le SW est dans un état suspendu et la valeur de sortie peut varier entre 0/1.

### Câblage



## Code

### Note :

- Ouvrez le fichier 4.3.toggle\_the\_joystick.ino sous le chemin 3in1-kit\basic\_project\4.3.toggle\_the\_joystick.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Ouvrez le moniteur série après le téléchargement réussi du code pour voir les valeurs x,y,z du joystick.

- Les valeurs des axes x et y sont des valeurs analogiques variant de 0 à 1023.
- L'axe Z est une valeur numérique avec un état de 1 ou 0 (lorsqu'il est pressé, il est 1).

## 5.4.4 4.4 Mesurer l'Humidité du Sol

Dans l'industrie de la plantation, les cultures elles-mêmes ne peuvent pas obtenir directement les éléments inorganiques du sol, l'eau dans le sol agit comme un solvant pour dissoudre ces éléments inorganiques.

Les cultures absorbent l'humidité du sol par le système racinaire, obtiennent des nutriments et favorisent la croissance.

Au cours de la croissance et du développement des cultures, les exigences pour la température du sol sont également différentes. Par conséquent, un capteur d'humidité du sol est nécessaire.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

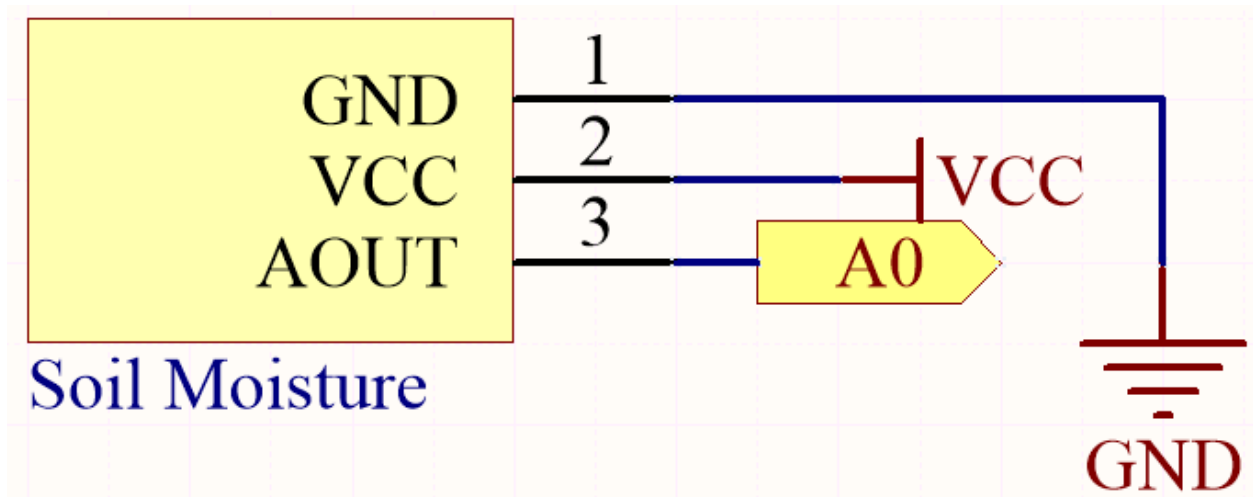
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

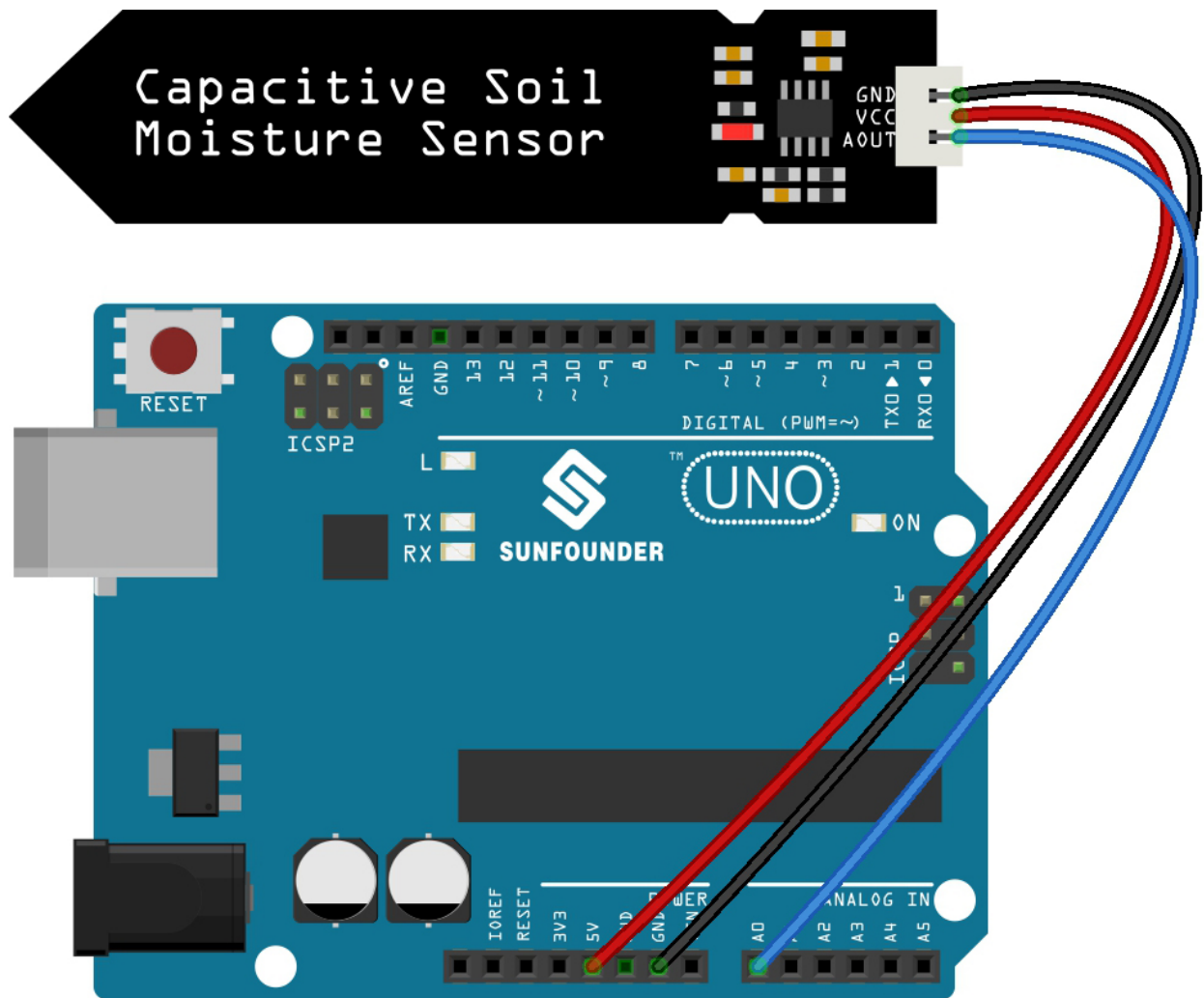
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module d'Humidité du Sol</i>	

### Schéma



### Câblage



## Code

### Note :

- Ouvrez le fichier `4.4.measure_soil_moisture.ino` sous le chemin `3in1-kit\basic_project\4.4.measure_soil_moisture`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Une fois le code téléchargé avec succès, le moniteur série imprimera la valeur de l'humidité du sol.

En insérant le module dans le sol et en l'arrosant, la valeur du capteur d'humidité du sol deviendra plus petite.

### 5.4.5 4.5 Thermomètre

Un thermomètre est un appareil qui mesure la température ou un gradient de température (le degré de chaleur ou de froid d'un objet). Un thermomètre a deux éléments importants : (1) un capteur de température (par exemple, le bulbe d'un thermomètre à mercure ou le capteur pyrométrique dans un thermomètre infrarouge) dans lequel un changement se produit avec un changement de température ; et (2) un moyen de convertir ce changement en une valeur numérique (par exemple, l'échelle visible qui est marquée sur un thermomètre à mercure ou l'affichage numérique sur un modèle infrarouge). Les thermomètres sont largement utilisés dans la technologie et l'industrie pour surveiller les processus, en météorologie, en médecine et dans la recherche scientifique.

Un thermistor est un type de capteur de température dont la résistance dépend fortement de la température, et il a deux types : Coefficient de Température Négatif (NTC) et Coefficient de Température Positif (PTC), également connus sous les noms de NTC et PTC. La résistance du thermistor PTC augmente avec la température, tandis que la condition du NTC est opposée à la précédente.

Dans cette expérience, nous utilisons un **thermistor NTC** pour fabriquer un thermomètre.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

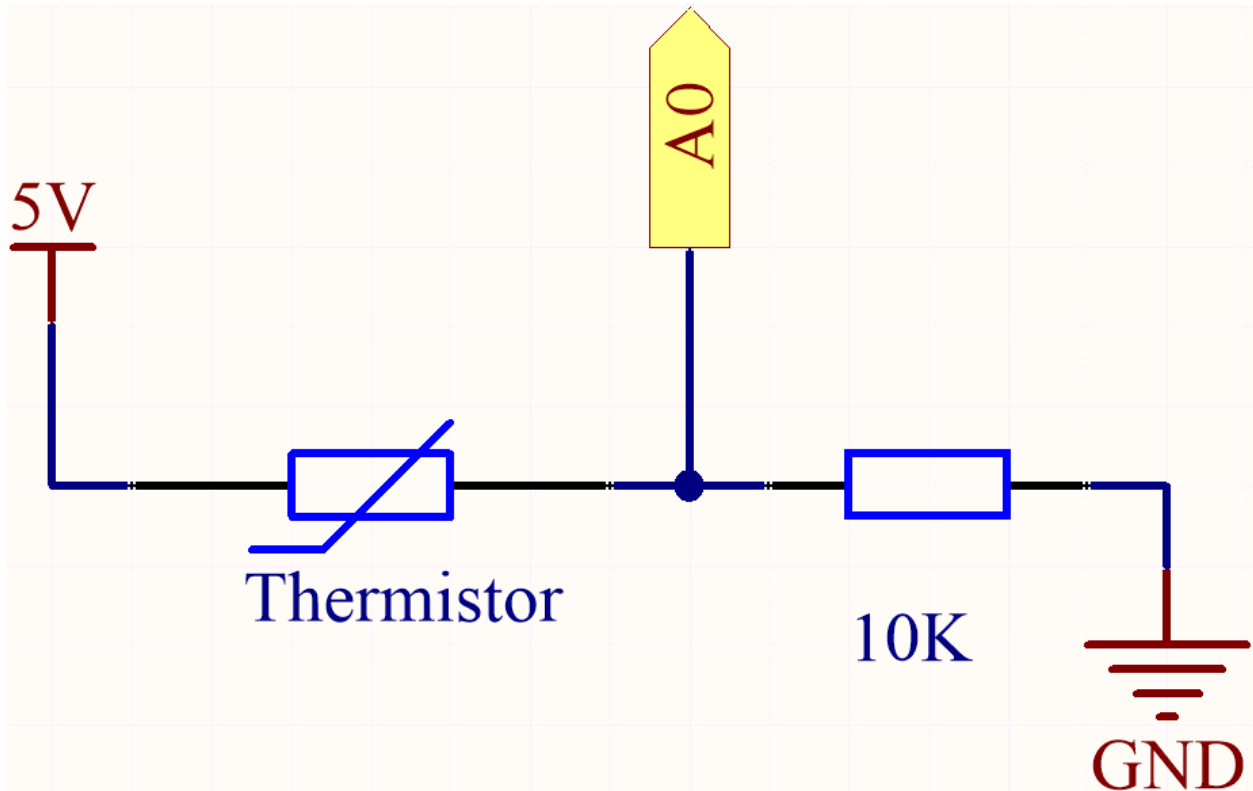
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Thermistance</i>	

#### Schéma





Chaque thermistor a une résistance normale. Ici, elle est de 10k ohms, mesurée à 25 degrés Celsius.

Lorsque la température augmente, la résistance du thermistor diminue. Ensuite, les données de tension sont converties en quantités numériques par l'adaptateur A/D.

La température en degrés Celsius ou Fahrenheit est sortie par programmation.

Voici la relation entre la résistance et la température :

$$RT = RN \exp B(1/TK - 1/TN)$$

- **RT** est la résistance du thermistor NTC lorsque la température est **TK**.
- **RN** est la résistance du thermistor NTC sous la température nominale **TN**. Ici, la valeur numérique de **RN** est de 10k.
- **TK** est une température Kelvin et l'unité est K. Ici, la valeur numérique de **TK** est  $273.15 + \text{degrés Celsius}$ .
- **TN** est une température Kelvin nominale ; l'unité est également K. Ici, la valeur numérique de **TN** est  $273.15 + 25$ .
- Et **B(beta)**, la constante matérielle du thermistor NTC, est également appelée indice de sensibilité thermique avec une valeur numérique 3950.
- **exp** est l'abréviation d'exponentiel, et le nombre de base e est un nombre naturel et vaut approximativement 2.7.

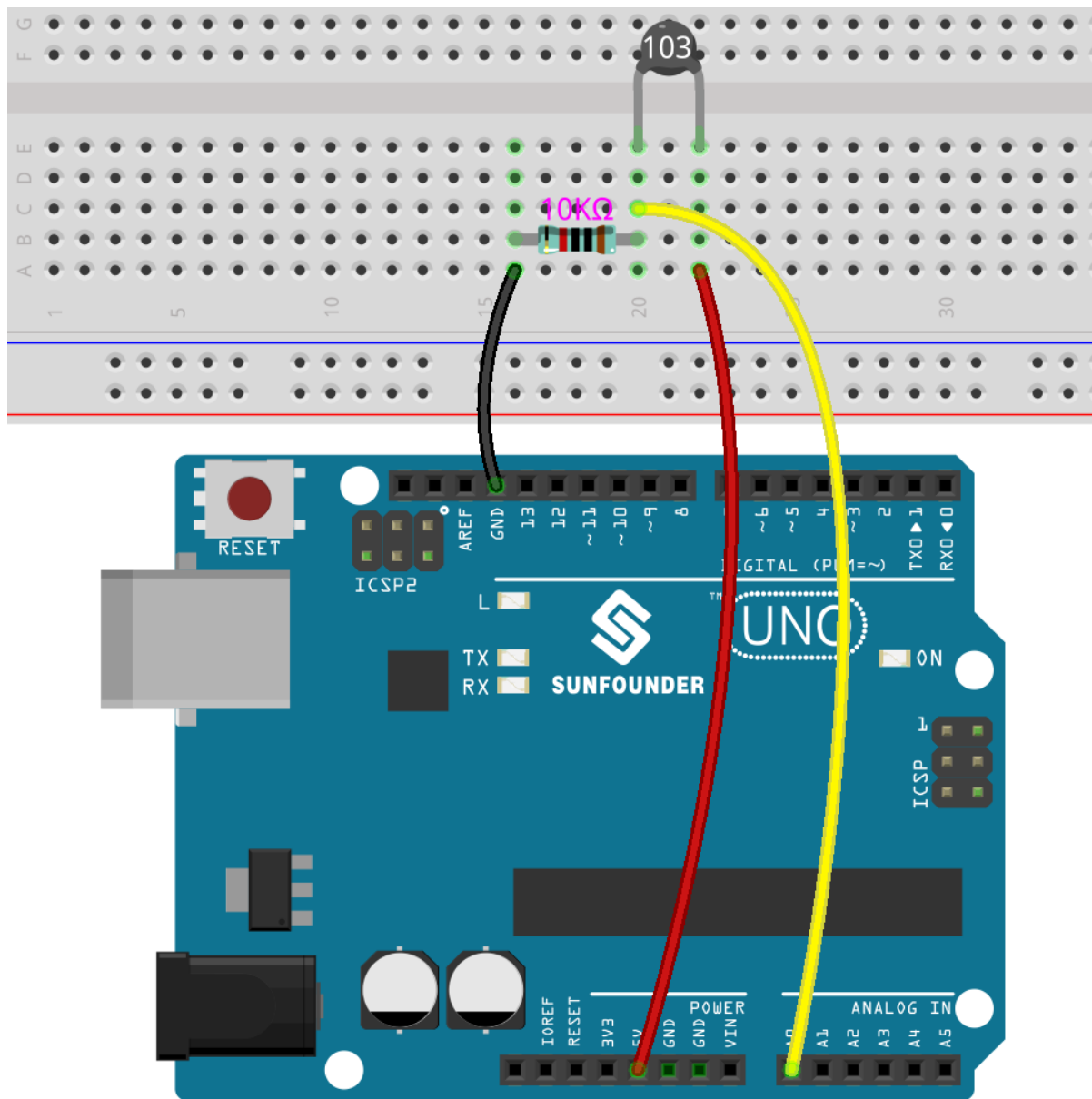
Convertissez cette formule  $TK = 1 / (\ln(RT/RN) / B + 1/TN)$  pour obtenir une température Kelvin qui moins 273.15 équivaut aux degrés Celsius.

Cette relation est une formule empirique. Elle n'est précise que lorsque la température et la résistance sont dans la plage effective.

#### Câblage

#### Note :

- Le thermistor est noir ou vert et marqué 103.



## Code

### Note :

- Ouvrez le fichier `4.5_thermometer.ino` sous le chemin `euler-kit/arduino/4.5_thermometer`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

N'oubliez pas de sélectionner la carte Raspberry Pi Pico et le port correct avant de cliquer sur le bouton Télécharger.

Après le téléchargement réussi du code, le Moniteur Série affichera les températures en degrés Celsius et Fahrenheit.

## 5.5 5. Syntaxe Avancée

Dans ce chapitre, vous trouverez quelques exemples illustrant la logique de base de l'interaction de la plupart des programmes avec la réalité. Cela vous aidera à vous familiariser avec la programmation Arduino. Lorsque vous avez une idée créative en tête, la programmation ne sera plus un défi pour vous.

### 5.5.1 5.1 If else

Habituellement, nous utilisons des jugements conditionnels pour réaliser les projets d'interaction avec la réalité les plus basiques. Ici, nous construisons un système de détection de porte avec un interrupteur à lames souples et une LED pour montrer cette logique.

Fixez l'aimant d'un côté de la porte et l'interrupteur à lames souples (avec circuit) de l'autre côté de la porte. Lorsque la porte est fermée, l'aimant est proche de l'interrupteur à lames souples, ce qui l'activera.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

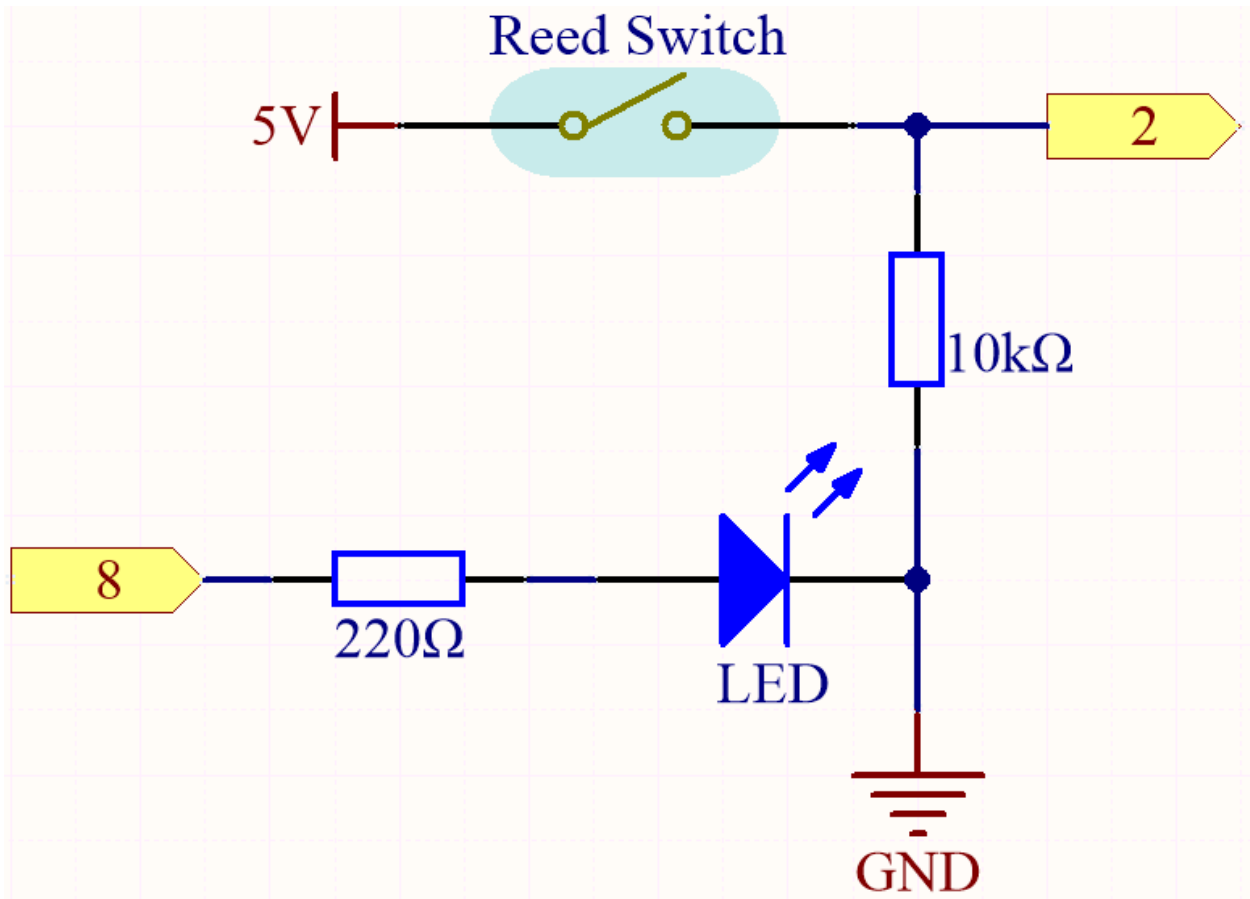
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

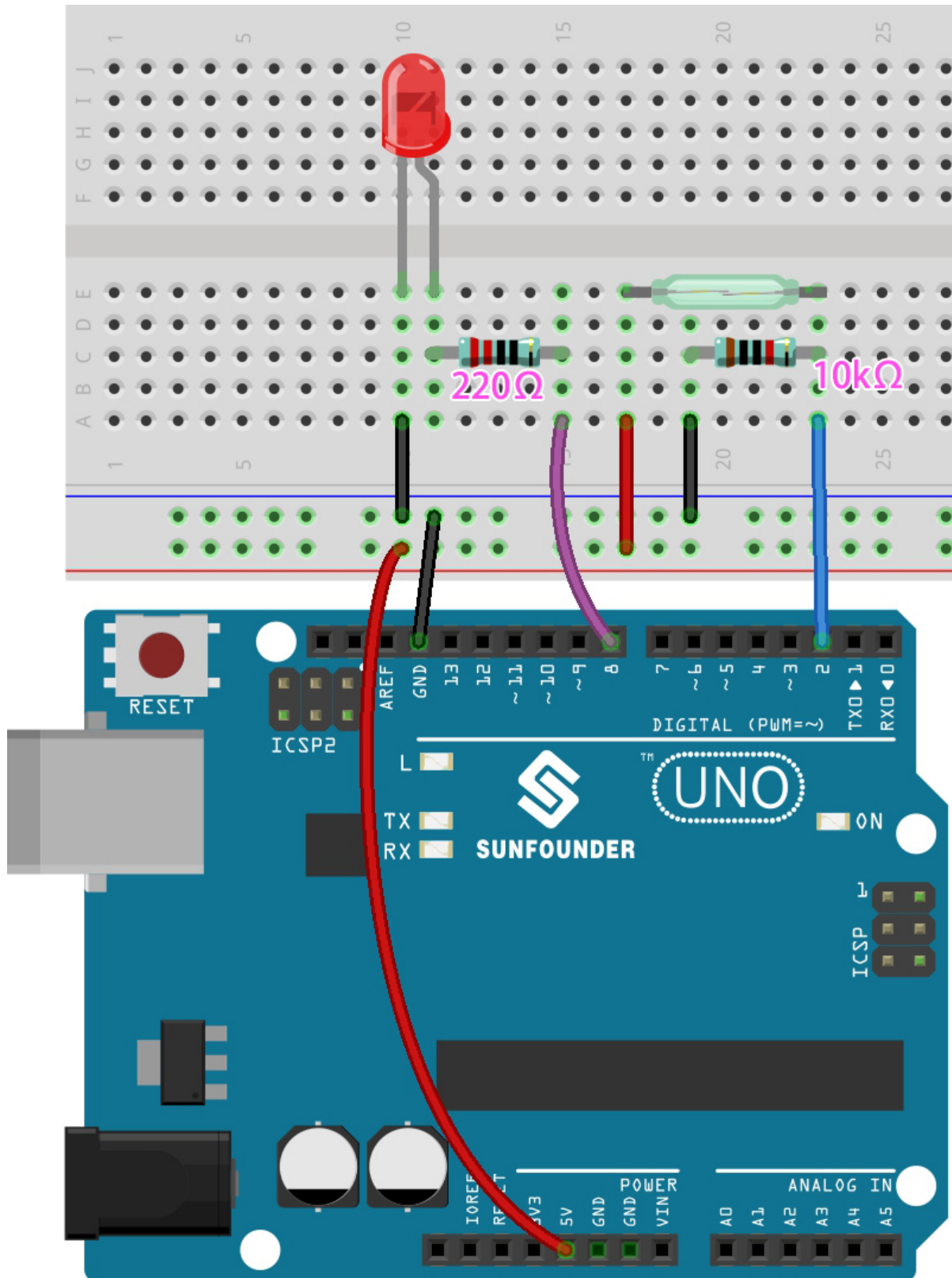
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Interrupteur à Lame Souple</i>	-

#### Schéma



Câblage



Code

---

### Note :

- Ouvrez le fichier `5.1.if_else.ino` sous le chemin `3in1-kit/basic_project/5.1.if_else`.
  - Ou copiez ce code dans **Arduino IDE**.
  - Ou téléchargez le code via l'[Arduino Web Editor](#).
- 

Après le téléchargement réussi du code, si vous ne fermez pas la porte, la LED s'allumera, vous incitant à fermer la porte.

D'ailleurs, si nous avons besoin de l'effet inverse (allumer la LED lorsque la porte est fermée), nous avons juste besoin de modifier la condition dans le `if`.

- `if else`

Le `if else` permet un contrôle plus grand sur le flux du code que la simple instruction `if`, en permettant de regrouper plusieurs tests.

## 5.5.2 5.2 Seuil

Dans de nombreux projets, vous rencontrerez ce besoin. « Lorsque xxx atteint un certain niveau, alors... »

Par exemple, dans une maison intelligente, lorsque l'intensité lumineuse est inférieure à 50Lux, allumez la lumière ; Un autre exemple est sur la carte mère d'un ordinateur, si la température de fonctionnement du CPU est supérieure à 65 degrés Celsius, allumez le ventilateur, et ainsi de suite.

Dans ces exigences, le mot-clé « seuil » est reflété.

Nous pouvons ajuster la valeur du seuil pour que le circuit fonctionne plus en adéquation avec les besoins individuels. Par exemple, si j'aime un environnement de vie plus lumineux, je peux augmenter le seuil des lumières automatiques de la maison intelligente à 80Lux. Un autre exemple est que l'environnement de ventilation de mon studio n'est pas très bon, et la demande de dissipation thermique est plus élevée, donc la valeur seuil de l'ouverture automatique du ventilateur peut être ajustée à 50 degrés Celsius.

Ici, nous utilisons un capteur d'humidité du sol et 2 LED pour fabriquer un moniteur de pot. Si le sol est trop sec, la LED rouge s'allumera ; si le sol est suffisamment humide, la LED verte s'allumera. Vous devez ajuster manuellement les seuils pour déterminer la sécheresse et l'humidité du sol.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

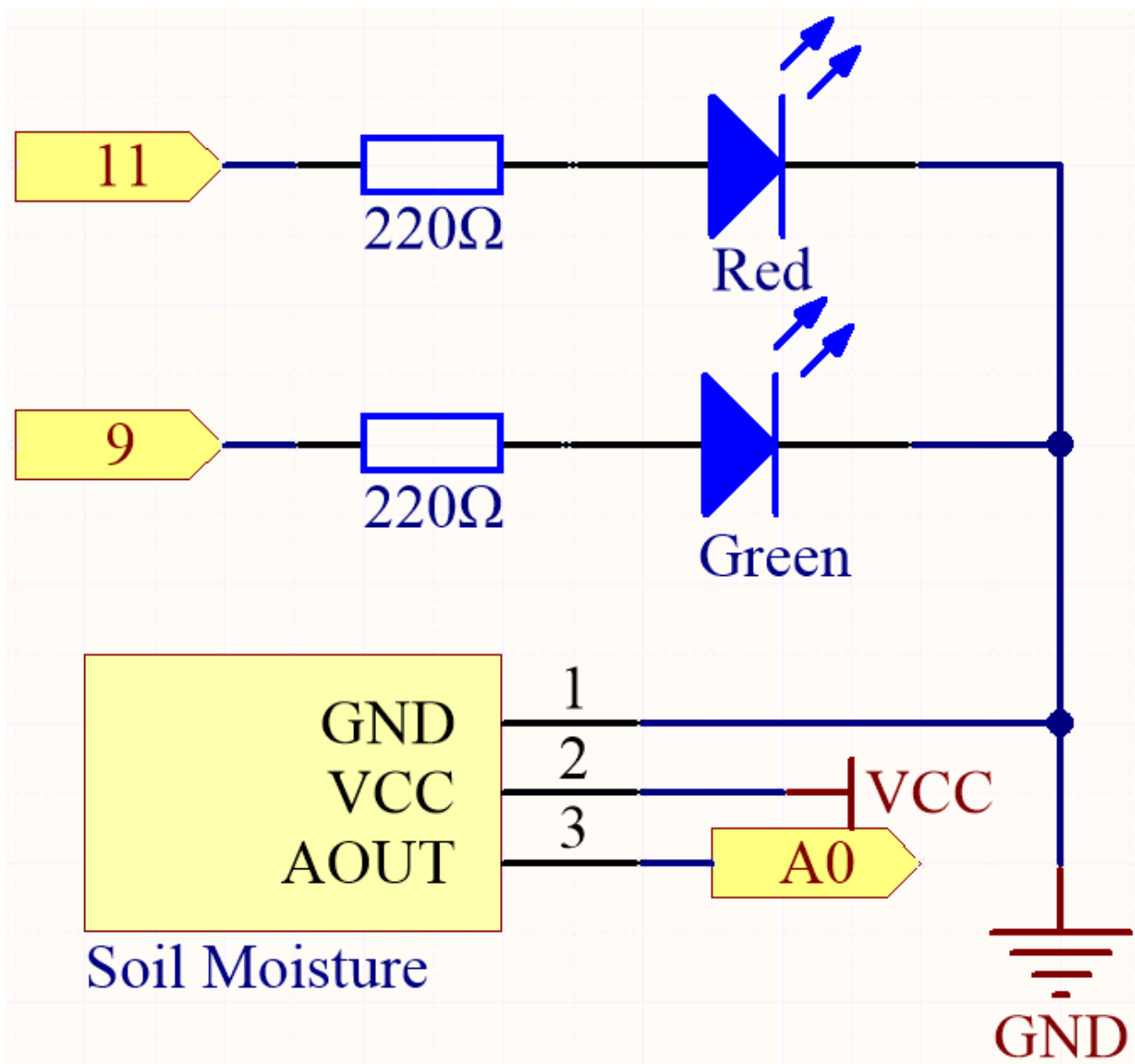
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

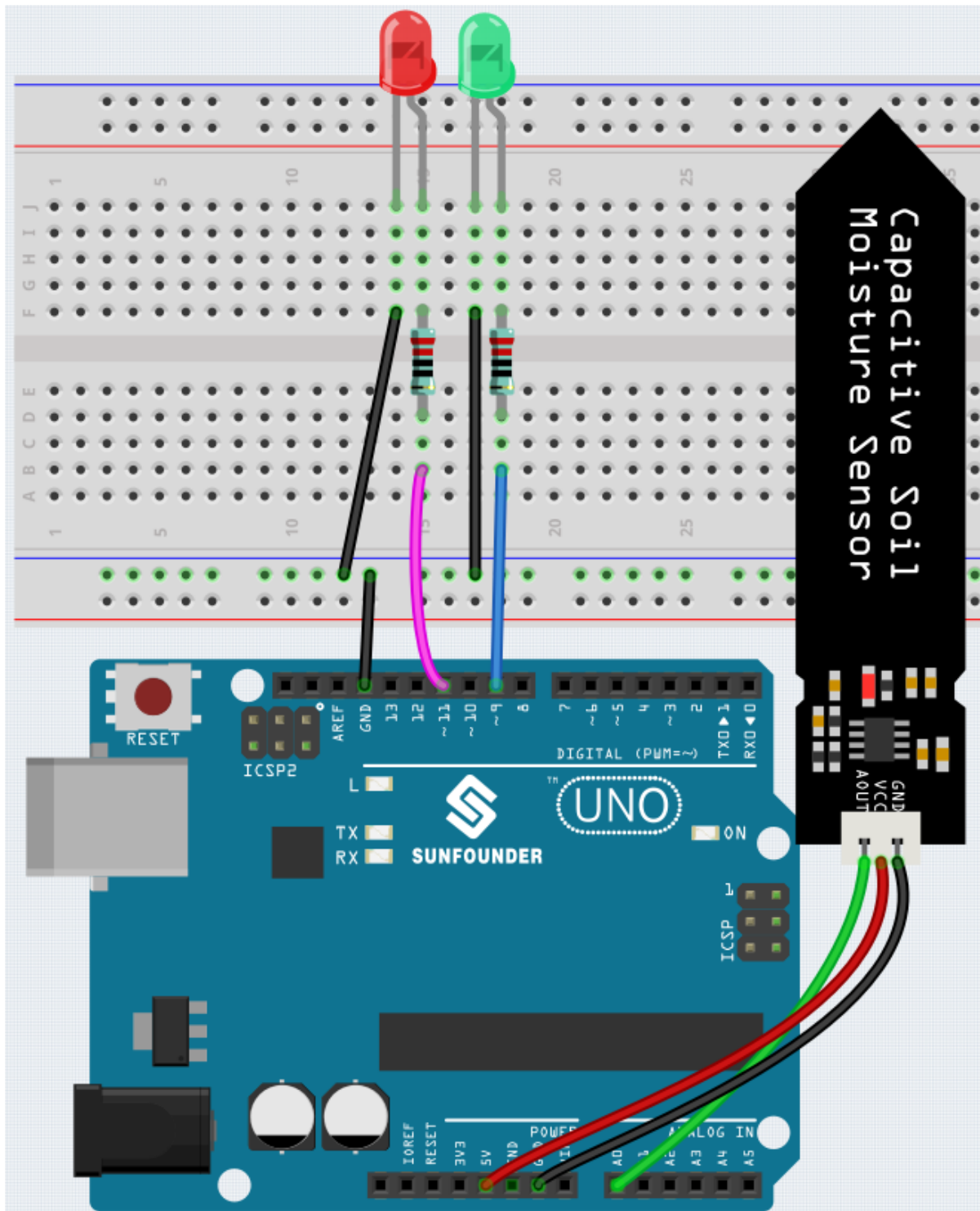
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Module d'Humidité du Sol</i>	

## Schéma



## Câblage



## Code

### Note :

- Ouvrez le fichier `5.2.threshold.ino` sous le chemin `3in1-kit/basic_project/5.2.threshold`.
- Ou copiez ce code dans **Arduino IDE**.



Après le téléchargement réussi du code, si votre seuil est correctement défini, vous verrez la LED rouge s'allumer lorsque le sol est sec pour vous rappeler que vous devez arroser ; après l'arrosage, la LED verte s'allumera.

### Fonctionnement

```
...

void loop() {
  int sensorValue = analogRead(soilMoisture);
  Serial.println(sensorValue);
  if (sensorValue > threshold) {
    digitalWrite(redPin, HIGH); // Turn the red LED
    digitalWrite(greenPin, LOW); // green
  } else {
    digitalWrite(greenPin, HIGH); // Turn on the green LED
    digitalWrite(redPin, LOW); // red
  }
}

...
```

Définissez d'abord une valeur de `threshold` puis lisez la valeur du module d'humidité du sol, sa valeur diminue à mesure que le niveau d'humidité augmente. Si la valeur actuellement lue est supérieure au `threshold` défini, alors laissez la LED rouge s'allumer, sinon allumez la LED verte.

Cette valeur de `threshold` doit être ajustée en fonction de la situation réelle, vous pouvez d'abord télécharger le code, puis ouvrir le moniteur série pour vérifier la valeur, enregistrer la valeur dans des conditions humides et sèches, puis choisir une valeur médiane comme valeur de `threshold`.

### 5.5.3 5.3 Détection de Changement d'État

Lorsque le bouton contrôle d'autres dispositifs, il peut non seulement fonctionner lorsqu'il est pressé, mais aussi s'arrêter lorsqu'il est relâché. Il est également possible de basculer l'état de fonctionnement à chaque fois que le bouton est pressé.

Pour réaliser cet effet, vous devez savoir comment basculer l'état de fonctionnement entre éteint et allumé lorsque le bouton est pressé, C'est la « détection de changement d'état ».

Dans ce projet, nous utiliserons le bouton pour contrôler le moteur.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

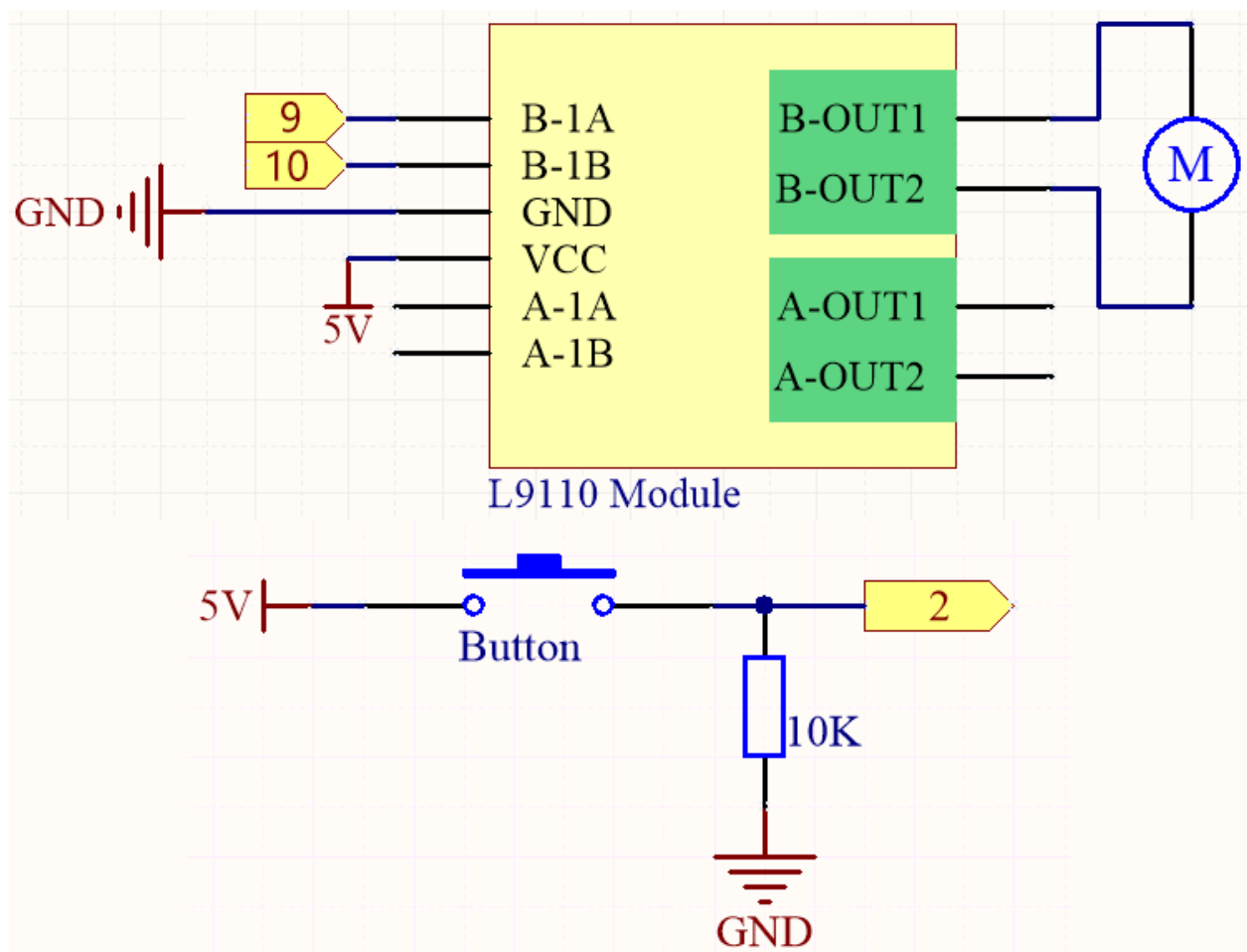
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

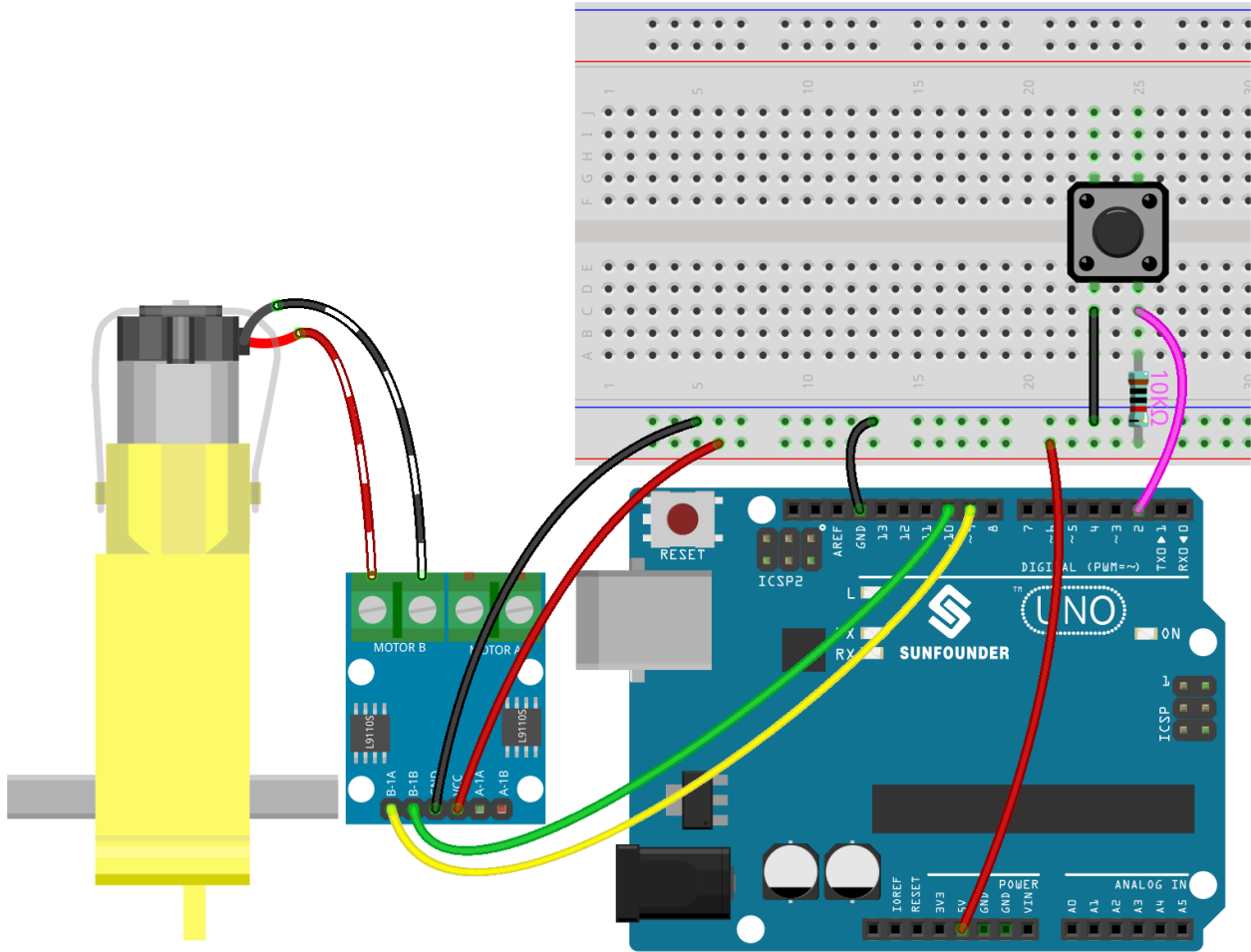
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Bouton</i>	
<i>Moteur TT</i>	-
<i>Module de Contrôle Moteur L9110</i>	-

### Schéma



### Câblage



## Code

### Note :

- Ouvrez le fichier 5.3.state\_change\_detection.ino sous le chemin 3in1-kit\basic\_project\5.3.state\_change\_detection.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, vous appuyez sur le bouton et le moteur tourne ; jusqu'à ce que vous appuyiez à nouveau sur le bouton, le moteur s'arrête.

### Comment ça fonctionne ?

1. Créez des variables et définissez les broches pour le moteur et le bouton.

```
...
int detectionState = 0;
int buttonState = 0;
int lastButtonState = 0;
```

- detectionState est un indicateur dont la valeur change à chaque fois que le bouton est pressé, par exemple, 0 cette fois, 1 la prochaine, et ainsi de suite alternativement.
- buttonState et lastButtonState sont utilisés pour enregistrer l'état du bouton cette fois et la dernière fois, pour comparer si le bouton a été pressé ou relâché.

2. Initialisez chaque broche et réglez le taux de baud du moniteur série.

```
void setup() {
  pinMode(buttonPin, INPUT);
  Serial.begin(9600);
  pinMode(B_1A, OUTPUT);
  pinMode(B_1B, OUTPUT);
}
```

3. Commencez par lire l'état du bouton, et si le bouton est pressé, la variable `detectionState` changera sa valeur de 0 à 1 ou de 1 à 0. Lorsque `detectionState` est à 1, le moteur sera activé. Cela a pour effet que lorsque le bouton est pressé cette fois-ci, le moteur tourne, la prochaine fois que le bouton est pressé, le moteur s'arrête, et ainsi de suite alternativement.

```
void loop() {
  // Toggle the detectionState each time the button is pressed
  buttonState = digitalRead(buttonPin);
  if (buttonState != lastButtonState) {
    if (buttonState == HIGH) {
      detectionState=(detectionState+1)%2;
      Serial.print("The detection state is: ");
      Serial.println(detectionState);
    }
    delay(50);
  }
  lastButtonState = buttonState;

  // According to the detectionState, start the motor
  if(detectionState==1){
    digitalWrite(B_1A,HIGH);
    digitalWrite(B_1B,LOW);
  }else{
    digitalWrite(B_1A,LOW);
    digitalWrite(B_1B,LOW);
  }
}
```

Le flux de travail complet est le suivant.

— Lire la valeur du bouton.

```
buttonState = digitalRead(buttonPin);
```

— Si `buttonState` et `lastButtonState` ne sont pas égaux, cela signifie que l'état du bouton a changé, continuez avec le jugement suivant, et stockez l'état du bouton à ce moment dans la variable `lastButtonState`. `delay(50)` est utilisé pour éliminer le jitter.

```
if (buttonState != lastButtonState) {
  ...
  delay(50);
}
lastButtonState = buttonState;
```

— Lorsque le bouton est pressé, sa valeur est HIGH. Ici, lorsque le bouton est pressé, la valeur de la variable `detectionState` est modifiée, par exemple, de 0 à 1 après une opération.

```

if (buttonState == HIGH) {
    detectionState=(detectionState+1)%2;
    Serial.print("The detection state is: ");
    Serial.println(detectionState);
}

```

— Lorsque la variable `detectionState` est à 1, faire tourner le moteur, sinon l'arrêter.

```

if(detectionState==1){
    digitalWrite(B_1A,HIGH);
    digitalWrite(B_1B,LOW);
}else{
    digitalWrite(B_1A,LOW);
    digitalWrite(B_1B,LOW);
}

```

### 5.5.4 5.4 Intervalle

Parfois, vous avez besoin de faire deux choses en même temps. Par exemple, vous pourriez vouloir faire clignoter une LED tout en lisant un appui sur un bouton. Dans ce cas, vous ne pouvez pas utiliser `delay()`, car Arduino met votre programme en pause pendant le `delay()`. Si le bouton est pressé pendant qu'Arduino est en pause en attendant que le `delay()` se termine, votre programme manquera l'appui sur le bouton.

Une analogie serait de réchauffer une pizza dans votre micro-ondes tout en attendant un email important. Vous mettez la pizza dans le micro-ondes et réglez le temps sur 10 minutes. L'analogie à l'utilisation de `delay()` serait de s'asseoir devant le micro-ondes en regardant le minuteur décompter de 10 minutes jusqu'à ce que le minuteur atteigne zéro. Si l'email important arrive pendant ce temps, vous le manquerez.

Ce que vous feriez dans la vraie vie serait de mettre en marche la pizza, puis de vérifier vos emails, puis peut-être de faire quelque chose d'autre (qui ne prend pas trop de temps !) et de temps en temps, vous reviendriez au micro-ondes pour voir si le minuteur a atteint zéro, indiquant que votre pizza est prête.

Ce sketch démontre comment faire sonner un buzzer sans utiliser `delay()`. Il active le buzzer puis prend note de l'heure. Ensuite, à chaque passage dans `loop()`, il vérifie si le temps d'intervalle souhaité s'est écoulé. Si c'est le cas, il fait sonner le buzzer et prend note du nouveau temps. De cette manière, le buzzer sonne continuellement tandis que l'exécution du sketch ne s'attarde jamais sur une seule instruction.

Sur la base de cette condition, nous pouvons ajouter le code du bouton pour contrôler la LED, il ne sera pas perturbé par le buzzer jouant de la musique.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

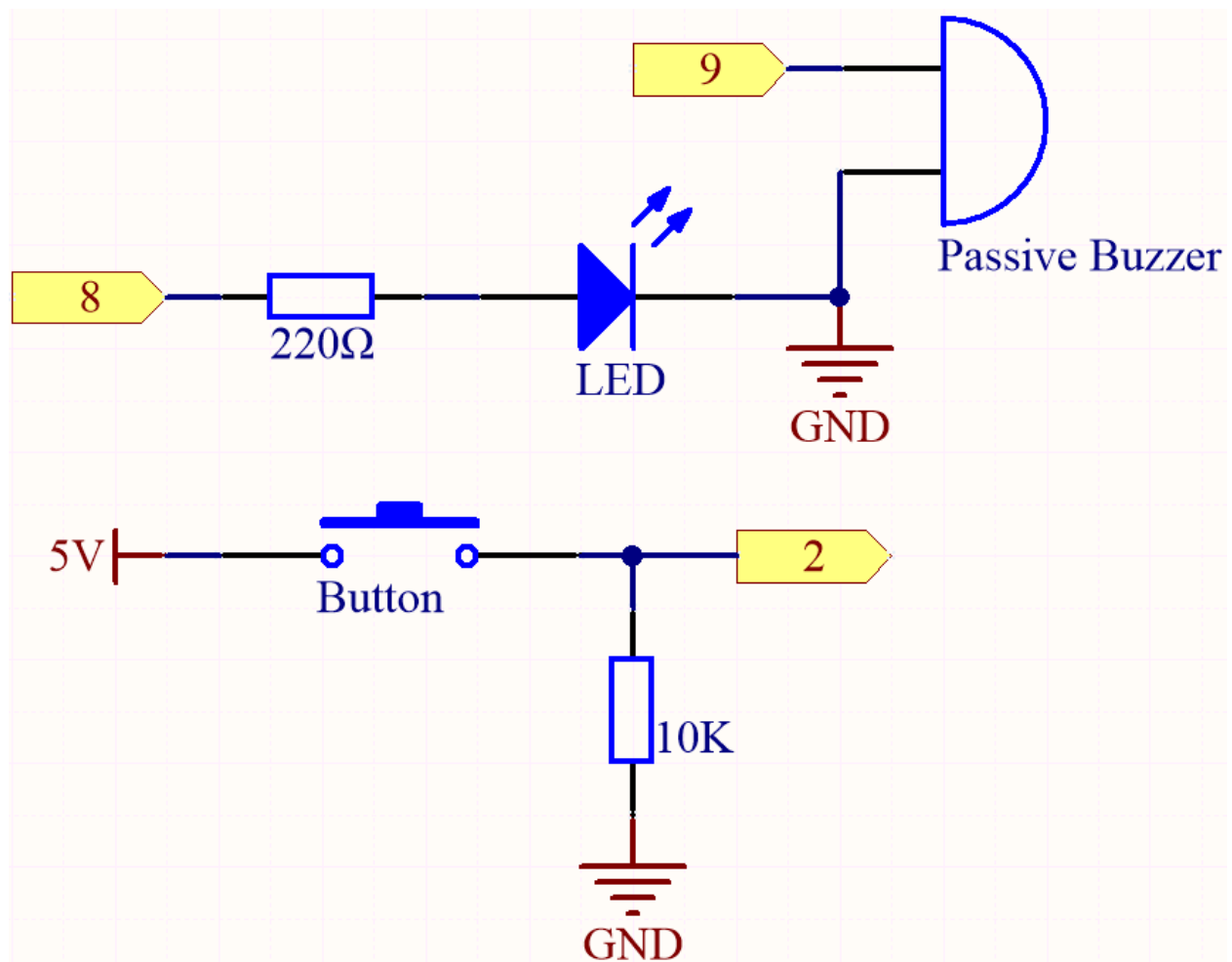
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

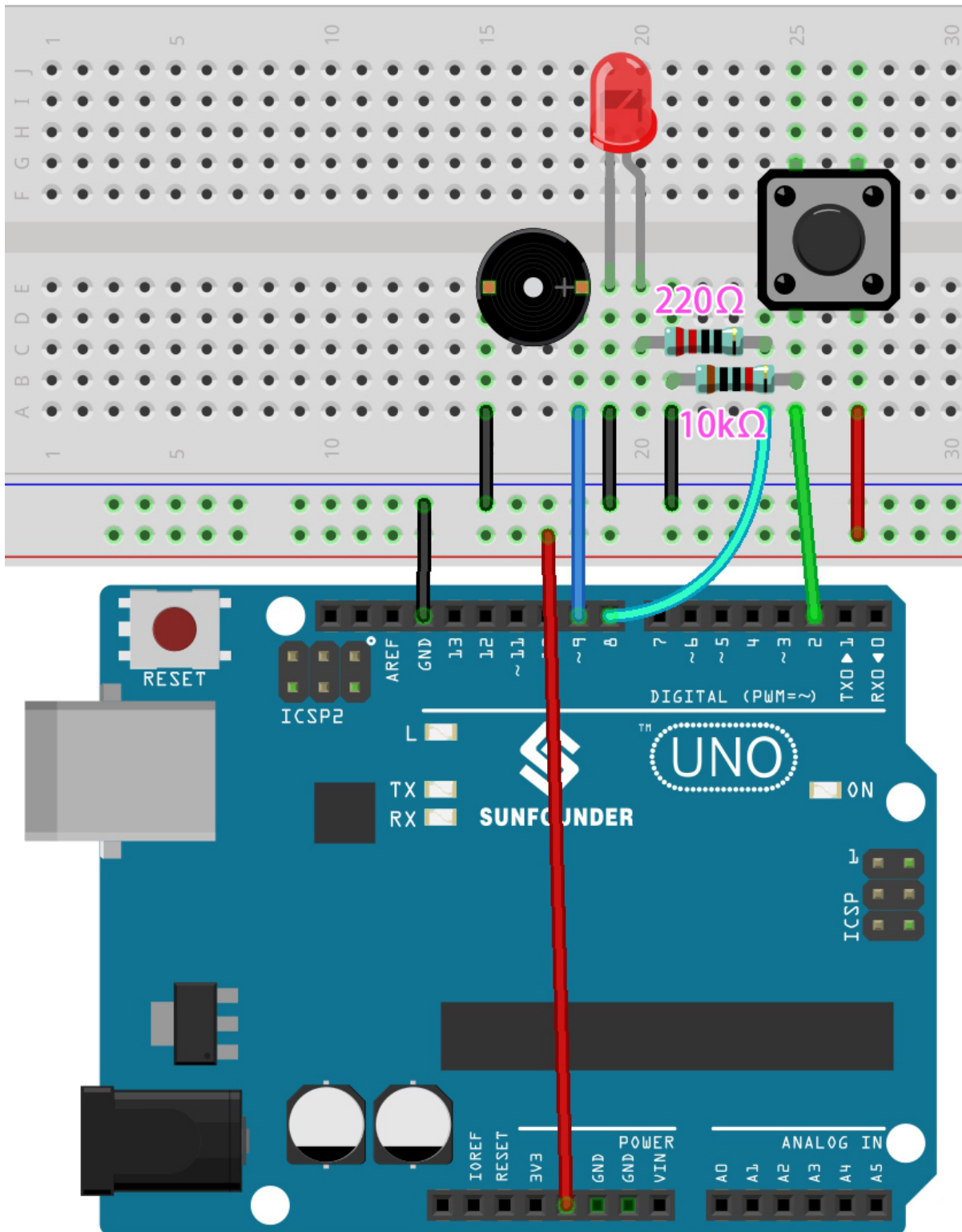
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Bouton</i>	
<i>Buzzer</i>	

### Schéma



### Câblage



Code

Note :

- Ouvrez le fichier `5.4.interval.ino` sous le chemin `3in1-kit/basic_project/5.4.interval`.
  - Ou copiez ce code dans **Arduino IDE**.
  - Ou téléchargez le code via l'[Arduino Web Editor](#).
- 

Après le téléchargement réussi du code, le buzzer jouera de la musique; chaque fois que vous appuyez sur le bouton, la LED s'allumera. Le travail de la LED et du buzzer ne se gêne pas l'un l'autre.

### Comment ça fonctionne ?

Initialisez une variable nommée `previousMillis` pour stocker le temps de fonctionnement précédent du microcontrôleur.

```
unsigned long previousMillis = 0;
```

Marquez quelle note est jouée.

```
int thisNote=0;
```

Le temps d'intervalle de chaque note.

```
long interval = 1000;
```

Dans `loop()`, déclarez `currentMillis` pour stocker le temps actuel.

```
unsigned long currentMillis = millis();
```

Lorsque l'intervalle entre le temps de fonctionnement actuel et le dernier temps de mise à jour est supérieur à 1000ms, certaines fonctions sont déclenchées. En même temps, mettez à jour le `previousMillis` au temps actuel pour le prochain déclenchement qui doit se produire 1 seconde plus tard.

```
if (currentMillis - previousMillis >= interval) {  
    previousMillis = currentMillis; // save the last time of the last tone  
    //...  
}
```

Jouez les notes de la mélodie une par une.

```
tone(buzzerPin,melody[thisNote],100);  
interval=1000/noteDurations[thisNote]; // interval at which to tone  
thisNote=(thisNote+1)%(sizeof(melody)/2); //iterate over the notes of the melody
```

Le bouton contrôle la LED.

```
// play button & led  
digitalWrite(ledPin,digitalRead(buttonPin));
```



### 5.5.5 5.5 Utiliser une Bibliothèque Interne

Dans l'IDE Arduino, vous pouvez utiliser de nombreuses bibliothèques intégrées en ajoutant directement le fichier .h correspondant à votre code.

Ce projet utilise la bibliothèque Servo pour piloter le servomoteur, afin qu'il puisse pivoter entre 0° et 180°.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

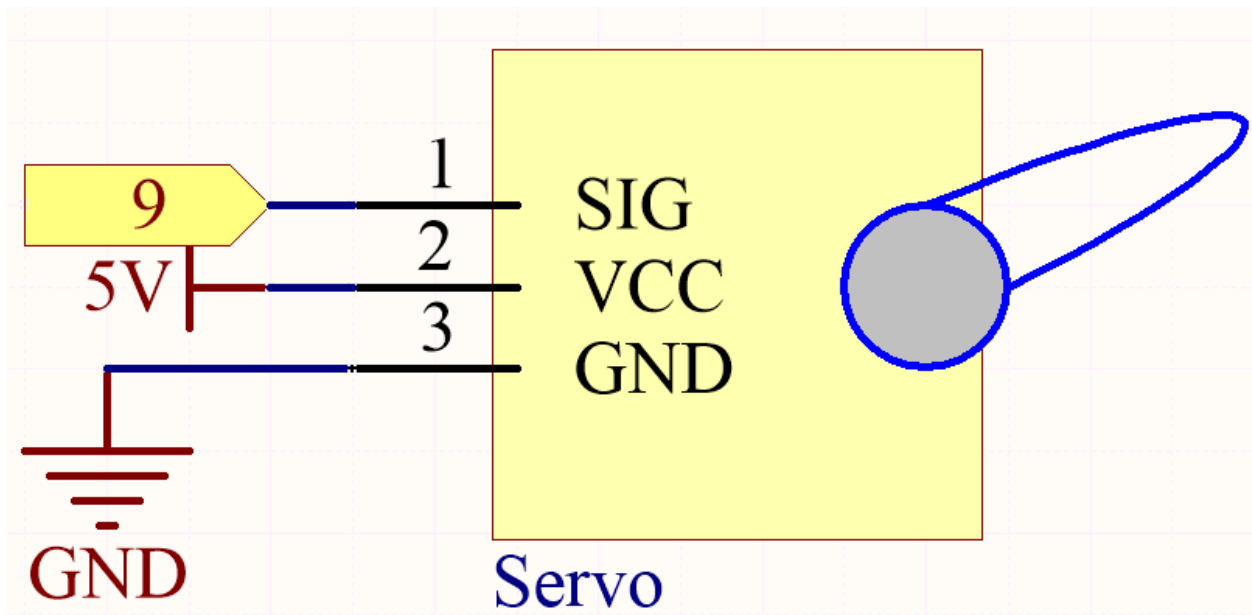
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

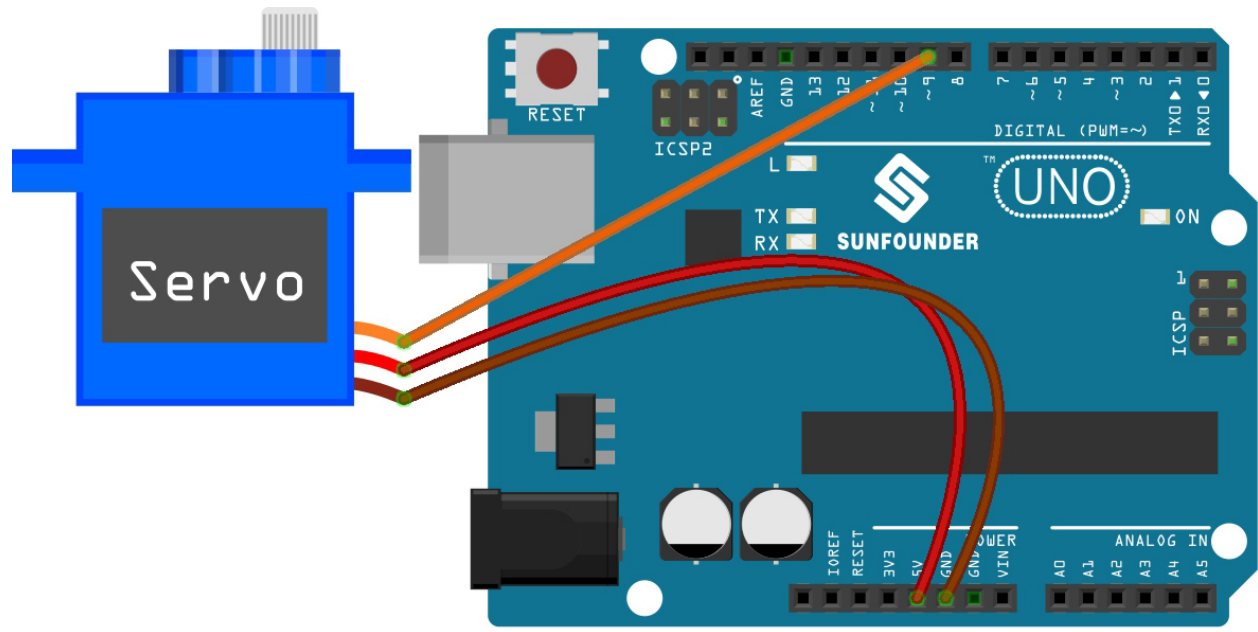
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Servomoteur</i>	

#### Schéma



Dans ce projet, nous utilisons la broche PWM 9 pour piloter le servomoteur, et connectons le fil orange du servomoteur à la broche PWM 9, le fil rouge à 5V et le fil marron à GND.

#### Câblage



### Code

#### Note :

- Ouvrez le fichier 5.5.use\_internal\_library.ino sous le chemin 3in1-kit\basic\_project\5.5.use\_internal\_library.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Une fois que vous avez fini de télécharger les codes sur la carte R3, vous pouvez voir le bras du servo pivoter dans la plage de 0° à 180°.

#### Comment ça fonctionne ?

En appelant la bibliothèque `Servo.h`, vous pouvez facilement piloter le servo.

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

Fonctions de la Bibliothèque :

`Servo`

Créez un objet **Servo** pour contrôler un servo.

```
uint8_t attach(int pin);
```

Appelez `pinMode()` pour transformer une broche en pilote de servo et retourner 0 en cas d'échec.

```
void detach();
```

Libérez une broche du pilotage de servo.

```
void write(int value);
```

Réglez l'angle du servo en degrés, de 0 à 180.

```
int read();
```

Retourne la valeur définie avec le dernier `write()`.

```
bool attached();
```

Retourne 1 si le servo est actuellement attaché.

### 5.5.6 5.6 Cartographie

Si vous observez attentivement, vous remarquerez que de nombreuses valeurs ont des plages différentes en programmation. Par exemple, la plage de valeurs pour les entrées analogiques est de (0~1023). La plage de valeurs pour la sortie analogique est de (0~255). L'angle de sortie du servomoteur est de (0~180).

Cela signifie que si nous voulons utiliser le potentiomètre pour contrôler la luminosité de la LED ou l'angle du servomoteur, nous devons passer par une opération de cartographie.

Voyons maintenant comment y parvenir.

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

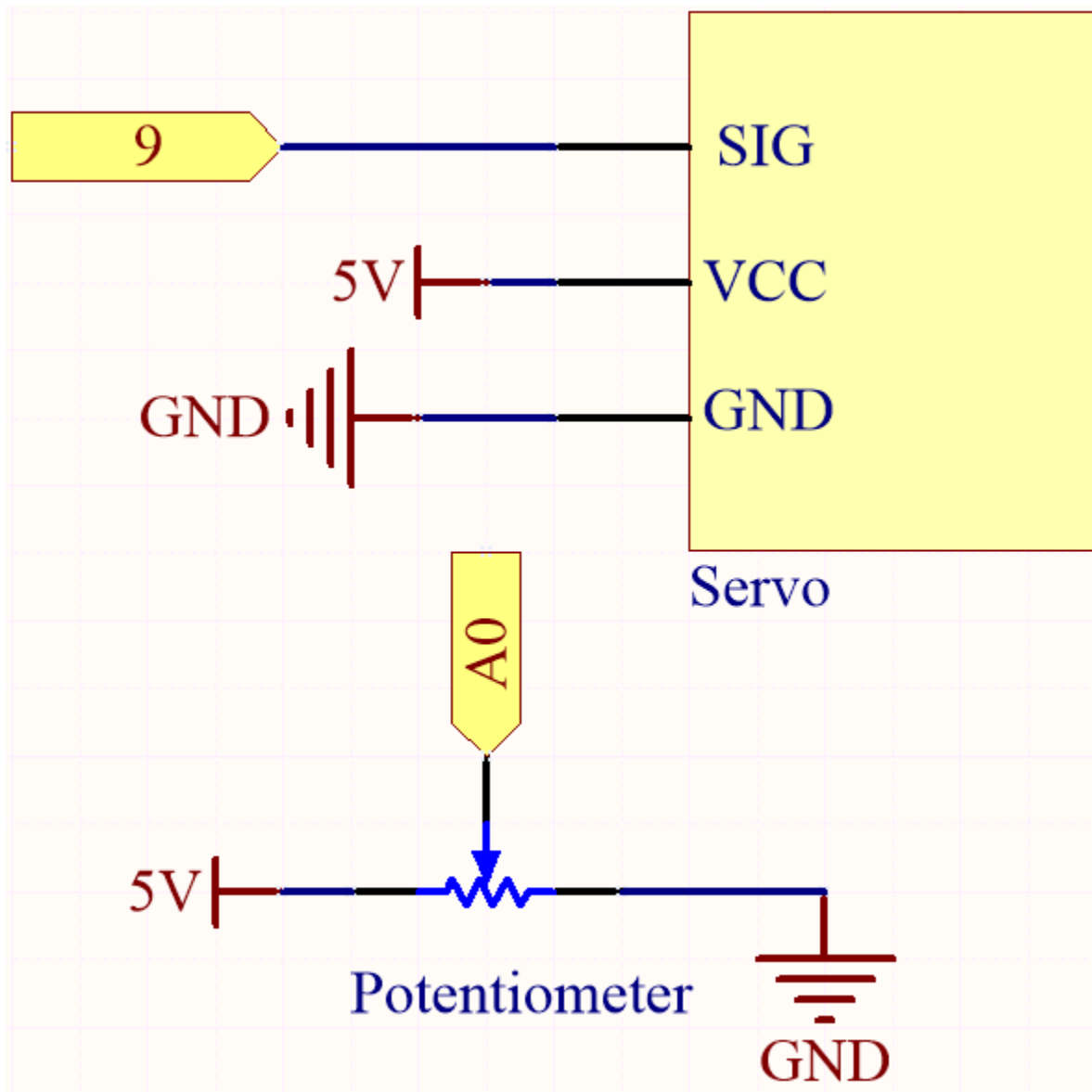
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

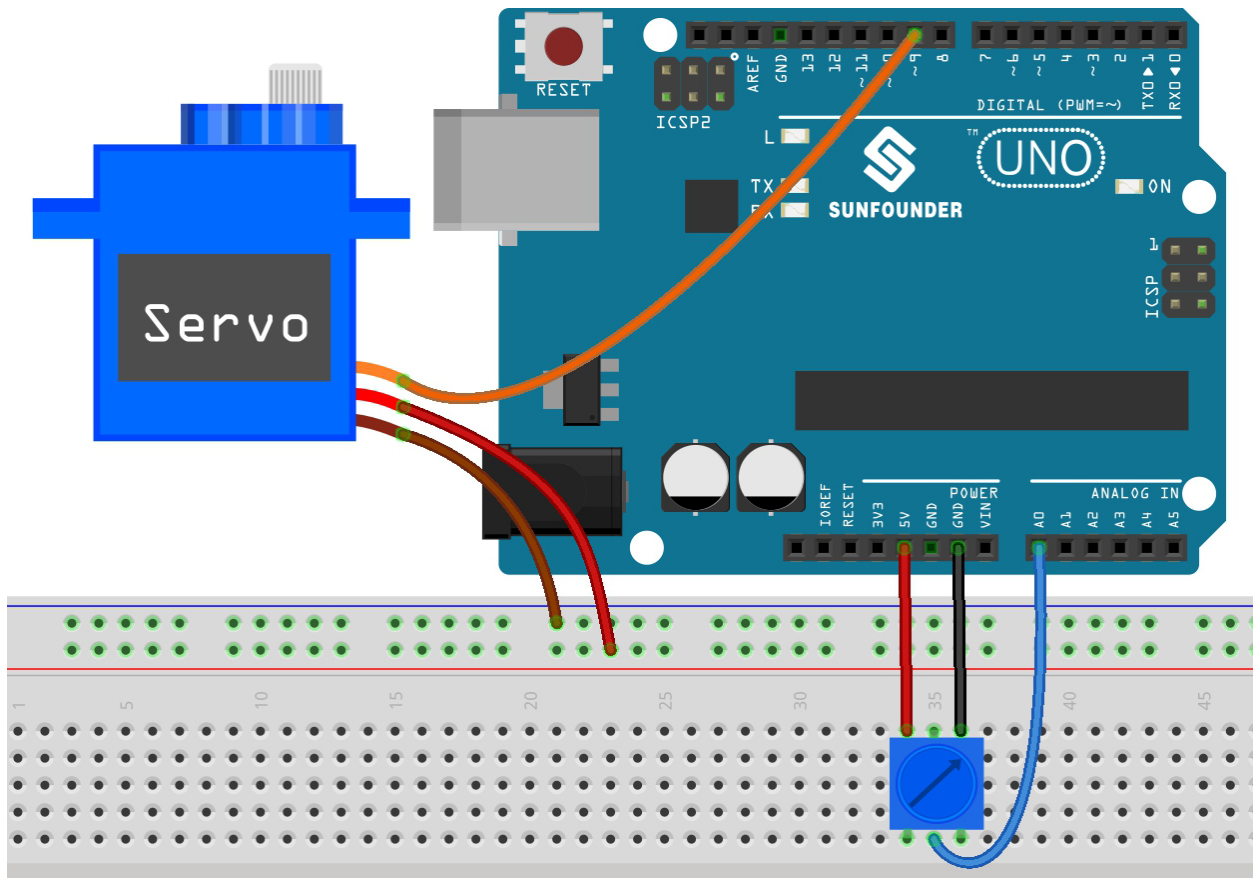
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Servomoteur</i>	
<i>Potentiomètre</i>	

#### Schéma



Câblage



## Code

### Note :

- Ouvrez le fichier 5.6.map.ino sous le chemin 3in1-kit\basic\_project\5.6.map.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Après le téléchargement réussi du code, vous pouvez tourner le potentiomètre d'avant en arrière, et l'arbre de sortie du servo tournera d'avant en arrière.

### Comment ça fonctionne ?

`map(value, fromLow, fromHigh, toLow, toHigh)` : Mappez un nombre d'une plage à une autre. C'est-à-dire qu'une valeur de `fromLow` est mappée sur `toLow`, et une valeur de `fromHigh` est mappée sur `toHigh`.

#### Syntaxe

`map(value, fromLow, fromHigh, toLow, toHigh)`

#### Paramètres

- `value` : le nombre à mapper.
- `fromLow` : la limite inférieure de la plage actuelle de la valeur.
- `fromHigh` : la limite supérieure de la plage actuelle de la valeur.
- `toLow` : la limite inférieure de la plage cible de la valeur.
- `toHigh` : la limite supérieure de la plage cible de la valeur.

Si le potentiomètre contrôle la LED, vous pouvez également utiliser la cartographie pour accomplir la tâche.

```
int x = analogRead(knob);
int y = map(x, 0, 1023, 0, 255);
analogWrite(led, y);
```

#### Notes et Avertissements

- La « limite inférieure » des deux plages peut être plus grande ou plus petite que la « limite supérieure », ce qui signifie que la fonction `map()` peut être utilisée pour inverser une plage de nombres.

```
y = map(x, 0, 180, 180, 0);
```

- La cartographie fonctionne également bien pour les nombres négatifs.

```
y = map(x, 0, 1023, -90, 90);
```

- La cartographie utilise des entiers, et les décimales des nombres flottants sont ignorées.

### 5.5.7 5.7 Tone() ou noTone()

La fonction `Tone()` est utilisée pour générer une onde carrée de la fréquence spécifiée (et un cycle de travail de 50 %) sur une broche. Une durée peut être spécifiée, sinon l'onde continue jusqu'à un appel à `noTone()`.

Dans ce projet, utilisez ces deux fonctions pour faire vibrer le buzzer passif afin de produire un son. Comme le buzzer actif, le buzzer passif utilise également le phénomène d'induction électromagnétique pour fonctionner. La différence est qu'un buzzer passif n'a pas de source oscillante, donc il ne bipera pas si des signaux DC sont utilisés. Mais cela permet au buzzer passif d'ajuster sa propre fréquence d'oscillation et peut émettre différentes notes telles que « do, ré, mi, fa, sol, la, si ».

#### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

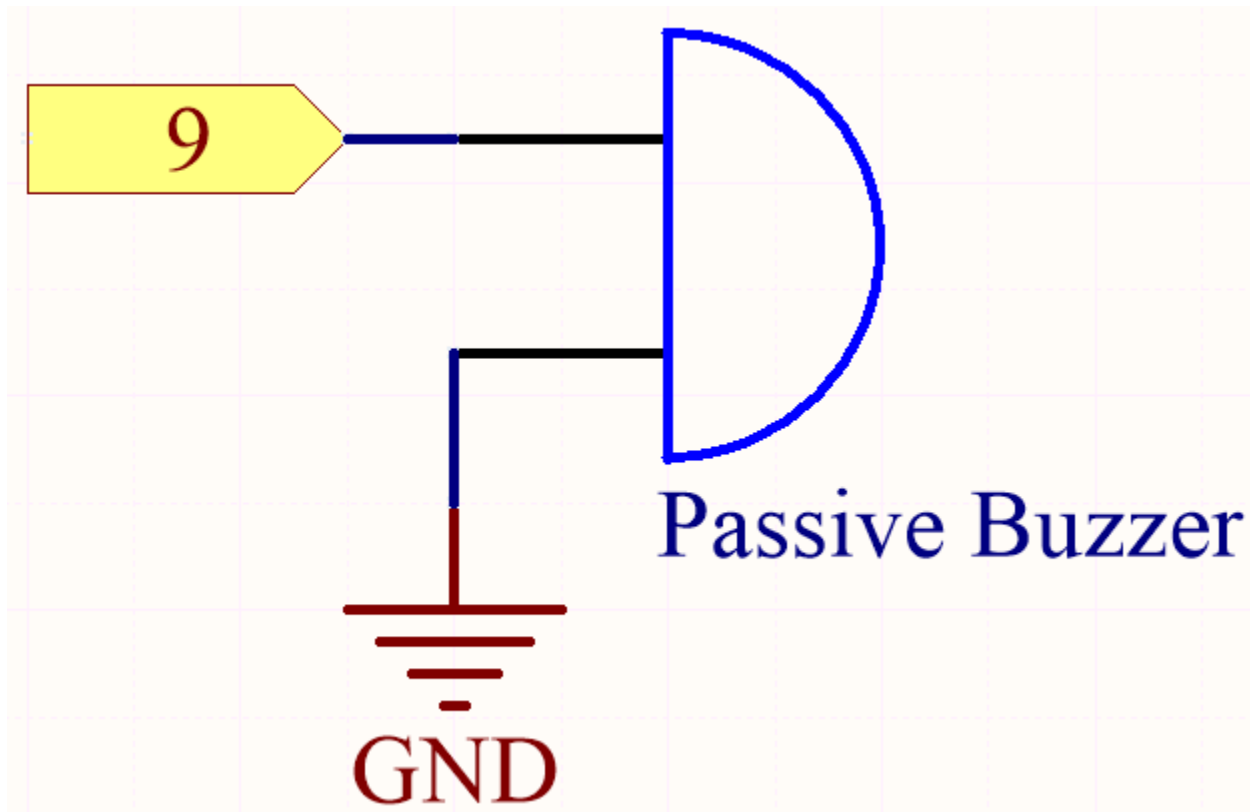
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

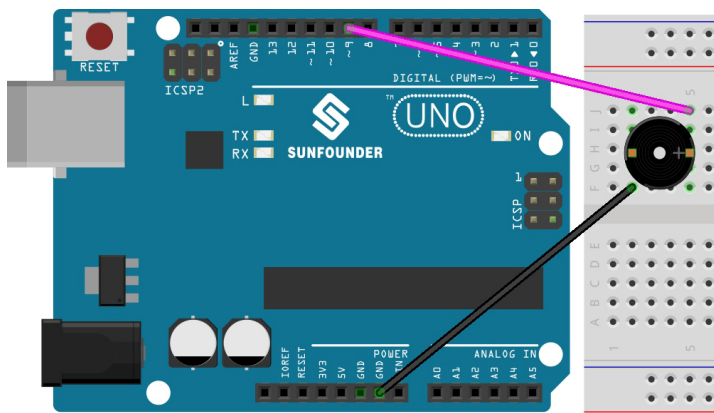
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Buzzer</i>	

#### Schéma



Connectez la cathode du Buzzer à GND, et l'anode à la broche numérique 9.

### Câblage



### Code

#### Note :

- Ouvrez le fichier 5.7.tone\_notone.ino sous le chemin 3in1-kit/basic\_project/5.7.tone\_notone.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via l'[Arduino Web Editor](#).

Lorsque vous aurez fini de télécharger les codes sur la carte R3, vous pourrez entendre une mélodie contenant sept notes.

Comment ça fonctionne ?

Il y a deux points à noter :

1. `tone()` & `noTone()` : Cette fonction est utilisée pour contrôler directement le son du buzzer passif et son prototype est comme suit :

### Syntaxe

```
void tone(int pin, unsigned int frequency)
void tone(int pin, unsigned int frequency, unsigned long duration)
```

### Paramètres

- `pin` : La broche Arduino sur laquelle générer le ton.
- `frequency` : La fréquence du ton en hertz.
- `duration` : La durée du ton en millisecondes (optionnel)

Génère une onde carrée de la fréquence spécifiée (et un cycle de travail de 50 %) sur une broche (afin de faire vibrer le buzzer passif pour produire un son). Une durée peut être spécifiée, sinon l'onde continue jusqu'à un appel à `noTone()`. La broche peut être connectée à un buzzer piézo ou à un autre haut-parleur pour jouer des tons.

Un seul ton peut être généré à la fois. Si un ton est déjà en train de jouer sur une broche différente, l'appel à `tone()` n'aura aucun effet. Si le ton est joué sur la même broche, l'appel définira sa fréquence.

L'utilisation de la fonction `tone()` interférera avec la sortie PWM sur les broches 3 et 11.

Il n'est pas possible de générer des tons inférieurs à 31 Hz.

### Syntaxe

```
void noTone(int pin)
```

### Paramètres

`pin` : La broche Arduino sur laquelle générer le ton.

Arrête la génération d'une onde carrée déclenchée par `tone()`. N'a aucun effet si aucun ton n'est généré.

Après avoir connu les deux fonctions, vous pouvez comprendre les codes - l'installation des tableaux `melody[]` et `noteDurations[]` est la préparation des appels successifs de la fonction `tone()` et le changement de ton et de durée dans la boucle pour un meilleur effet de lecture de musique.

2. `pitches.h` : Le code utilise un fichier supplémentaire, `pitches.h`. Ce fichier contient toutes les valeurs de tonalité pour les notes typiques. Par exemple, `NOTE_C4` est le do médian. `NOTE_FS4` est le fa dièse, et ainsi de suite. Ce tableau de notes a été initialement écrit par Brett Hagman, sur lequel le commande `tone()` a été basée. Vous pouvez le trouver utile chaque fois que vous voulez faire des notes musicales.

```
#include "pitches.h"
```

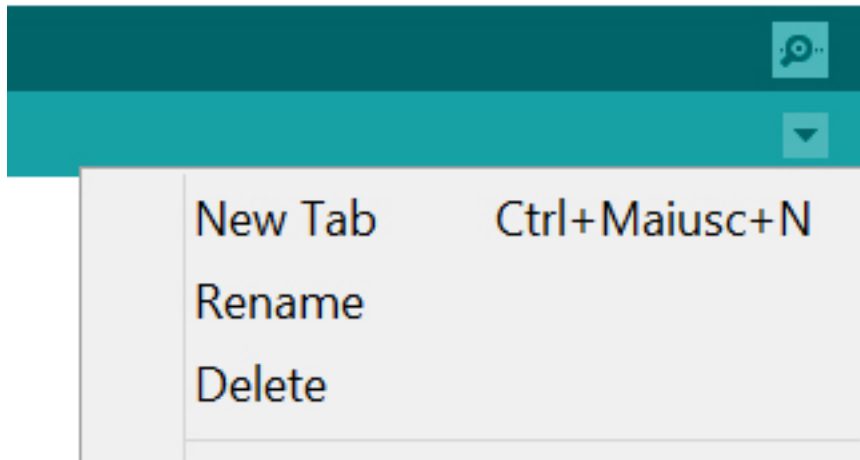
**Note :** Il y a déjà un fichier `pitches.h` dans ce programme d'exemple. Si nous le mettons avec le code principal dans un dossier, les étapes d'installation de `pitches.h` peuvent être omises.





Après avoir ouvert le fichier de code, si vous ne pouvez pas ouvrir le code `pitch.h`, vous pouvez simplement en créer un manuellement. Les étapes sont les suivantes :

Pour créer le fichier `pitch.h`, cliquez soit sur le bouton juste en dessous de l'icône du moniteur série et choisissez **New Tab**, soit utilisez **Ctrl+Shift+N**.



Ensuite, collez le code suivant et enregistrez-le en tant que `pitch.h` :

```

/*****
Public Constants
*****/
#define NOTE_B0  31
#define NOTE_C1  33
#define NOTE_CS1 35
#define NOTE_D1  37
#define NOTE_DS1 39
#define NOTE_E1  41
#define NOTE_F1  44
#define NOTE_FS1 46
#define NOTE_G1  49
#define NOTE_GS1 52
#define NOTE_A1  55
#define NOTE_AS1 58
#define NOTE_B1  62
#define NOTE_C2  65
#define NOTE_CS2 69
#define NOTE_D2  73
#define NOTE_DS2 78
#define NOTE_E2  82
#define NOTE_F2  87
#define NOTE_FS2 93
#define NOTE_G2  98
#define NOTE_GS2 104
#define NOTE_A2  110
#define NOTE_AS2 117
#define NOTE_B2  123
#define NOTE_C3  131
#define NOTE_CS3 139
#define NOTE_D3  147

```

(suite sur la page suivante)

(suite de la page précédente)

```
#define NOTE_DS3 156
#define NOTE_E3 165
#define NOTE_F3 175
#define NOTE_FS3 185
#define NOTE_G3 196
#define NOTE_GS3 208
#define NOTE_A3 220
#define NOTE_AS3 233
#define NOTE_B3 247
#define NOTE_C4 262
#define NOTE_CS4 277
#define NOTE_D4 294
#define NOTE_DS4 311
#define NOTE_E4 330
#define NOTE_F4 349
#define NOTE_FS4 370
#define NOTE_G4 392
#define NOTE_GS4 415
#define NOTE_A4 440
#define NOTE_AS4 466
#define NOTE_B4 494
#define NOTE_C5 523
#define NOTE_CS5 554
#define NOTE_D5 587
#define NOTE_DS5 622
#define NOTE_E5 659
#define NOTE_F5 698
#define NOTE_FS5 740
#define NOTE_G5 784
#define NOTE_GS5 831
#define NOTE_A5 880
#define NOTE_AS5 932
#define NOTE_B5 988
#define NOTE_C6 1047
#define NOTE_CS6 1109
#define NOTE_D6 1175
#define NOTE_DS6 1245
#define NOTE_E6 1319
#define NOTE_F6 1397
#define NOTE_FS6 1480
#define NOTE_G6 1568
#define NOTE_GS6 1661
#define NOTE_A6 1760
#define NOTE_AS6 1865
#define NOTE_B6 1976
#define NOTE_C7 2093
#define NOTE_CS7 2217
#define NOTE_D7 2349
#define NOTE_DS7 2489
#define NOTE_E7 2637
#define NOTE_F7 2794
#define NOTE_FS7 2960
```

(suite sur la page suivante)

(suite de la page précédente)

```
#define NOTE_G7 3136
#define NOTE_GS7 3322
#define NOTE_A7 3520
#define NOTE_AS7 3729
#define NOTE_B7 3951
#define NOTE_C8 4186
#define NOTE_CS8 4435
#define NOTE_D8 4699
#define NOTE_DS8 49
```

### 5.5.8 5.8 Fonction Définie par l'Utilisateur

En C, nous pouvons diviser un grand programme en blocs de construction de base connus sous le nom de fonction. La fonction contient l'ensemble des instructions de programmation entourées par {}. Une fonction peut être appelée plusieurs fois pour fournir une réutilisabilité et une modularité au programme C. En d'autres termes, nous pouvons dire que la collection de fonctions crée un programme. La fonction est également connue sous le nom de procédure ou sous-routine dans d'autres langages de programmation.

Voici les avantages des fonctions :

- En utilisant des fonctions, nous pouvons éviter de réécrire la même logique/code encore et encore dans un programme.
- Nous pouvons appeler des fonctions C autant de fois que nous le souhaitons dans un programme et de n'importe quel endroit dans un programme.
- Nous pouvons facilement suivre un grand programme C lorsqu'il est divisé en plusieurs fonctions.
- La réutilisabilité est la principale réalisation des fonctions C.
- Cependant, l'appel de fonction est toujours un surcoût dans un programme C.

Il existe deux types de fonctions en programmation C :

- **Fonctions de Bibliothèque** : les fonctions qui sont déclarées dans les fichiers d'en-tête C.
- **Fonctions Définies par l'Utilisateur** : les fonctions créées par le programmeur C, afin qu'il/elle puisse les utiliser plusieurs fois. Cela réduit la complexité d'un grand programme et optimise le code.

Dans ce projet, définissez une fonction pour lire la valeur du module ultrasonique.

#### Composants Requis

Dans ce projet, nous avons besoin des composants suivants.

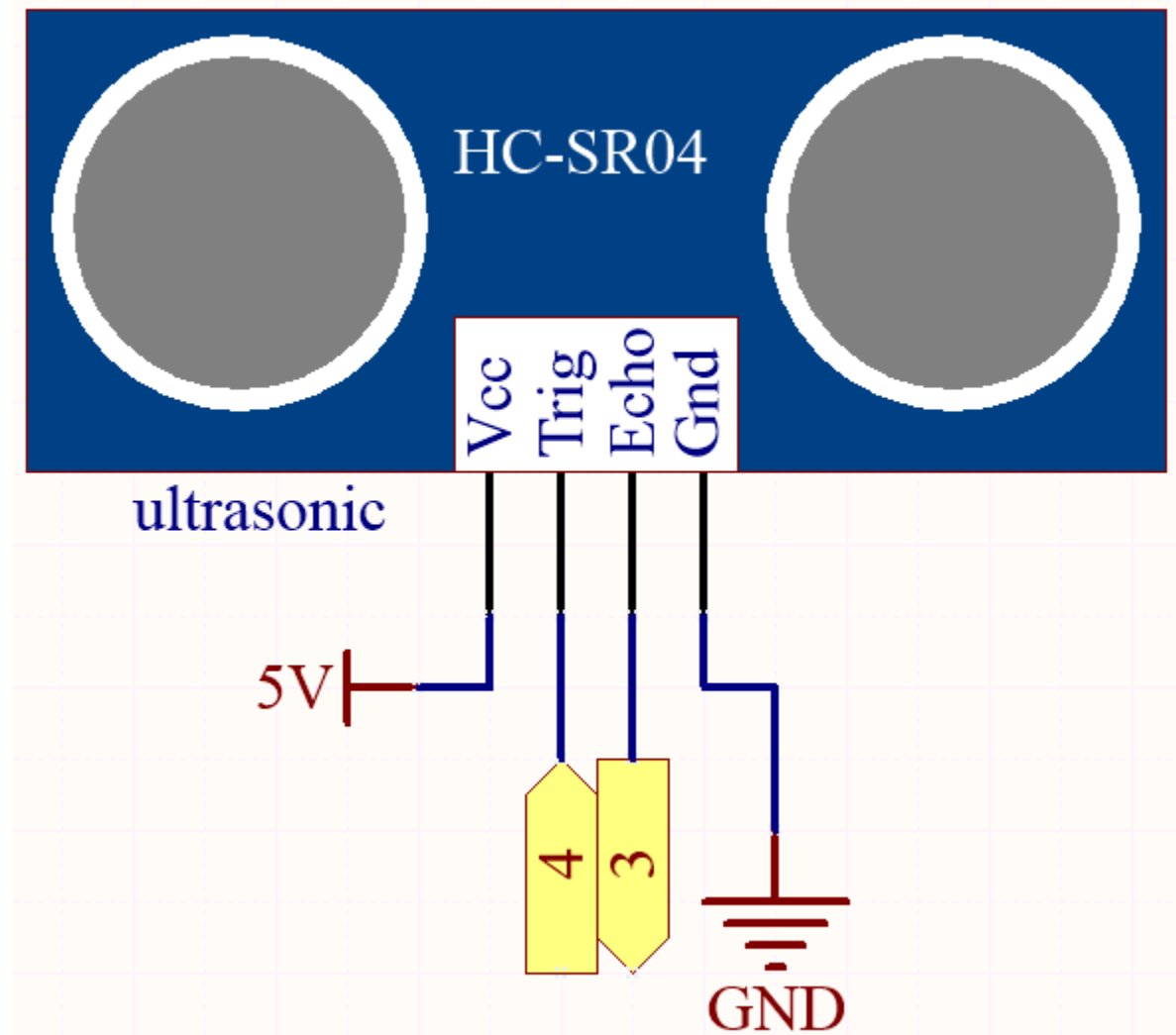
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ARTICLES DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

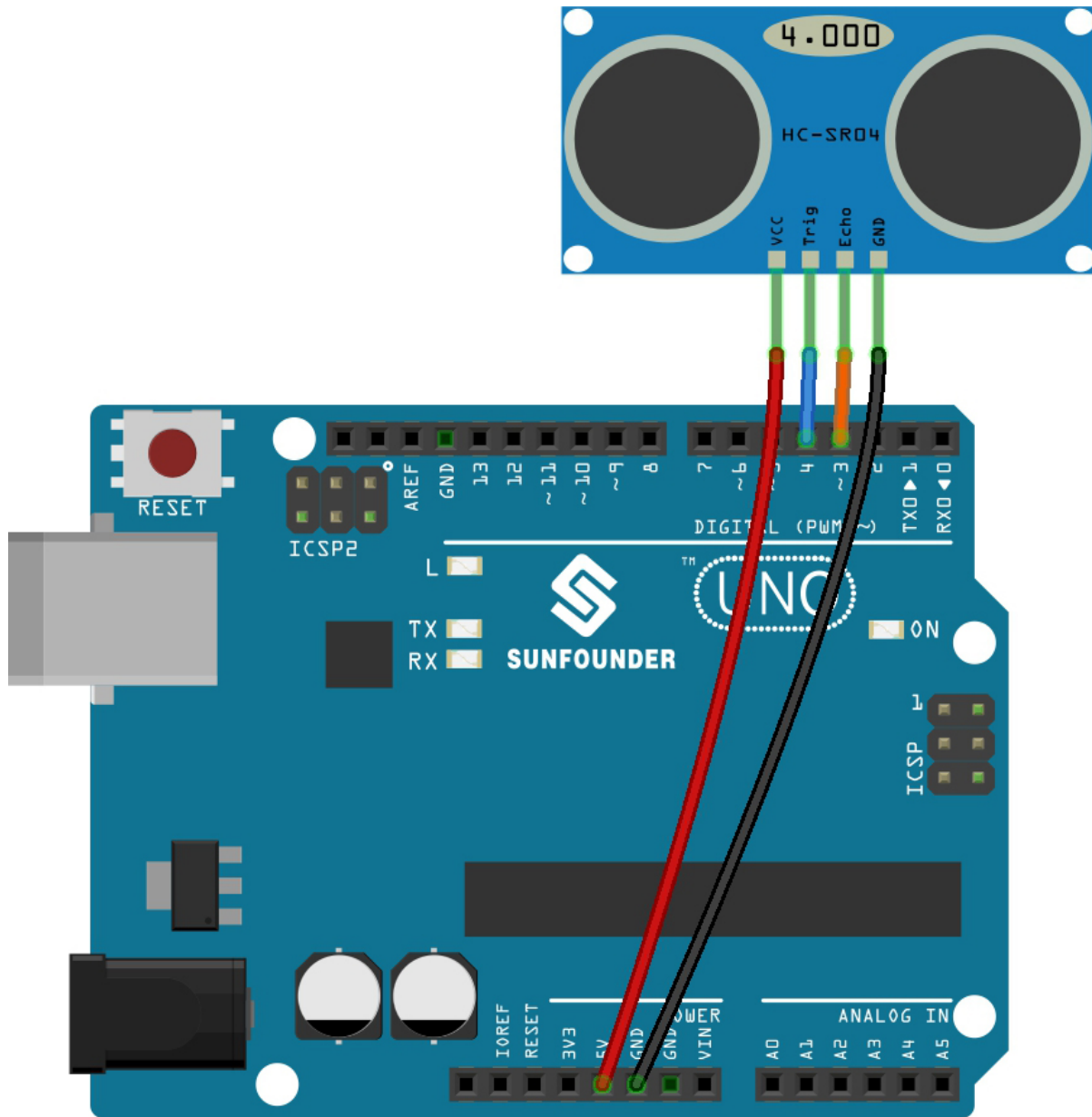
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module Ultrasonique</i>	

## Schéma



## Câblage



### Code

#### Note :

- Ouvrez le fichier 5.8.user\_function.ino sous le chemin de 3in1-kit\basic\_project\5.8.user\_function.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via le [Arduino Web Editor](#).

Après le téléchargement réussi du code, le moniteur série affichera la distance entre le capteur ultrasonique et l'obstacle devant.

#### Comment ça marche ?

Concernant l'application du capteur ultrasonique, nous pouvons directement vérifier la sous-fonction.

```
float readSensorData(){// ...}
```

La broche `trigPin` du module ultrasonique transmet un signal carré de 10µs toutes les 2µs

```
digitalWrite(trigPin, LOW);  
delayMicroseconds(2);  
digitalWrite(trigPin, HIGH);  
delayMicroseconds(10);  
digitalWrite(trigPin, LOW);
```

La broche `echoPin` reçoit un signal de haut niveau s'il y a un obstacle dans la portée et utilise la fonction `pulseIn()` pour enregistrer le temps entre l'envoi et la réception.

```
microsecond=pulseIn(echoPin, HIGH);
```

La vitesse du son est de 340 m/s ou 29 microsecondes par centimètre.

Cela donne la distance parcourue par l'onde carrée, aller et retour, donc nous divisons par 2 pour obtenir la distance de l'obstacle.

```
float distance = microsecond / 29.00 / 2;
```

Notez que le capteur ultrasonique mettra le programme en pause lorsqu'il fonctionne, ce qui peut provoquer des ralentissements lors de l'écriture de projets complexes.

### 5.5.9 5.9 ShiftOut(LED)

La fonction `shiftOut()` permet au 74HC595 de produire 8 signaux numériques. Elle émet le dernier bit du nombre binaire sur Q0, et la sortie du premier bit sur Q7. En d'autres termes, écrire le nombre binaire « 00000001 » fera que Q0 émette un niveau haut et que Q1~Q7 émettent un niveau bas.

Dans ce projet, vous apprendrez à utiliser le 74HC595. Le 74HC595 comprend un registre de décalage de 8 bits et un registre de stockage avec des sorties parallèles à trois états. Il convertit une entrée série en sortie parallèle, vous permettant ainsi d'économiser des ports IO d'un MCU.

Plus précisément, le 74hc595 peut remplacer 8 broches pour la sortie de signal numérique en écrivant un nombre binaire de 8 bits.

— [Nombre binaire - Wikipédia](#)

#### Composants requis

Dans ce projet, nous avons besoin des composants suivants.

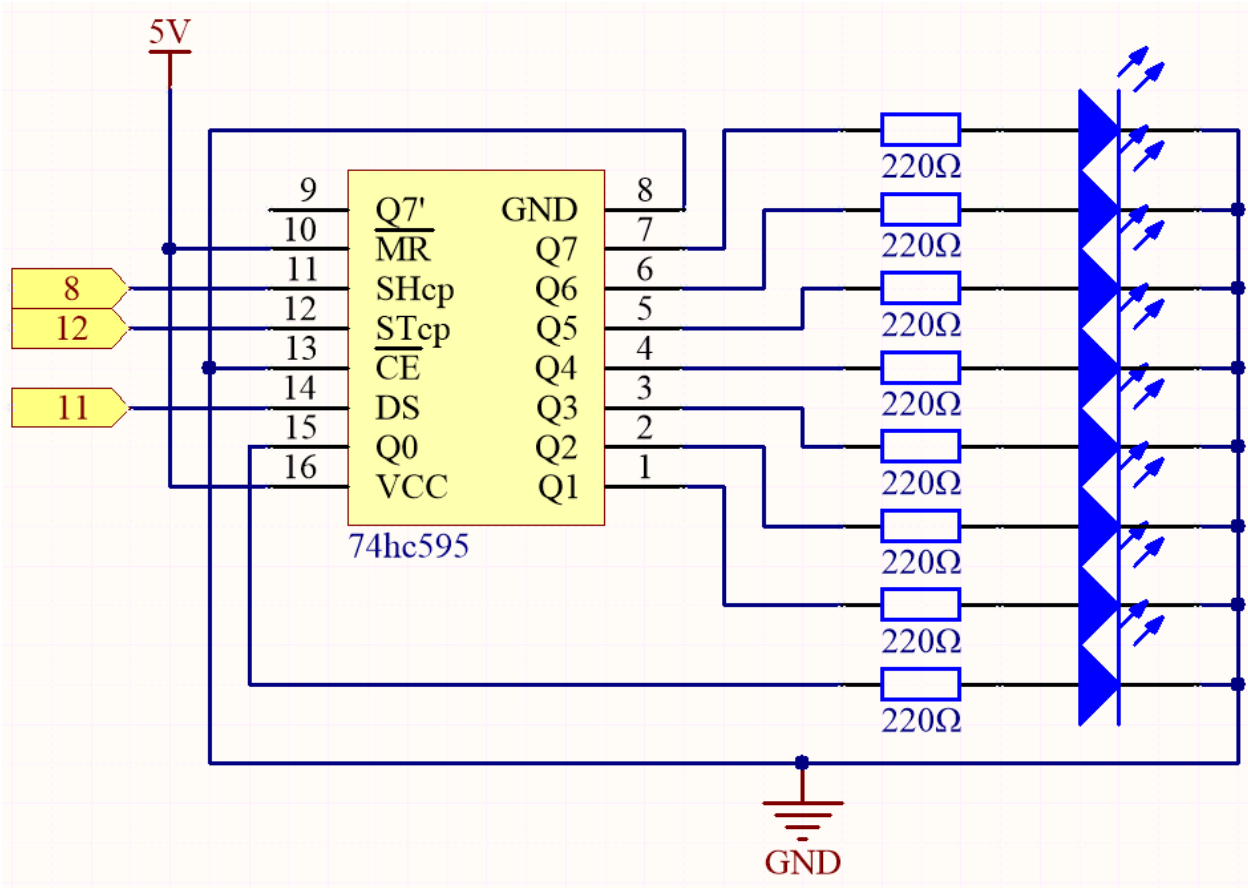
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ARTICLES DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

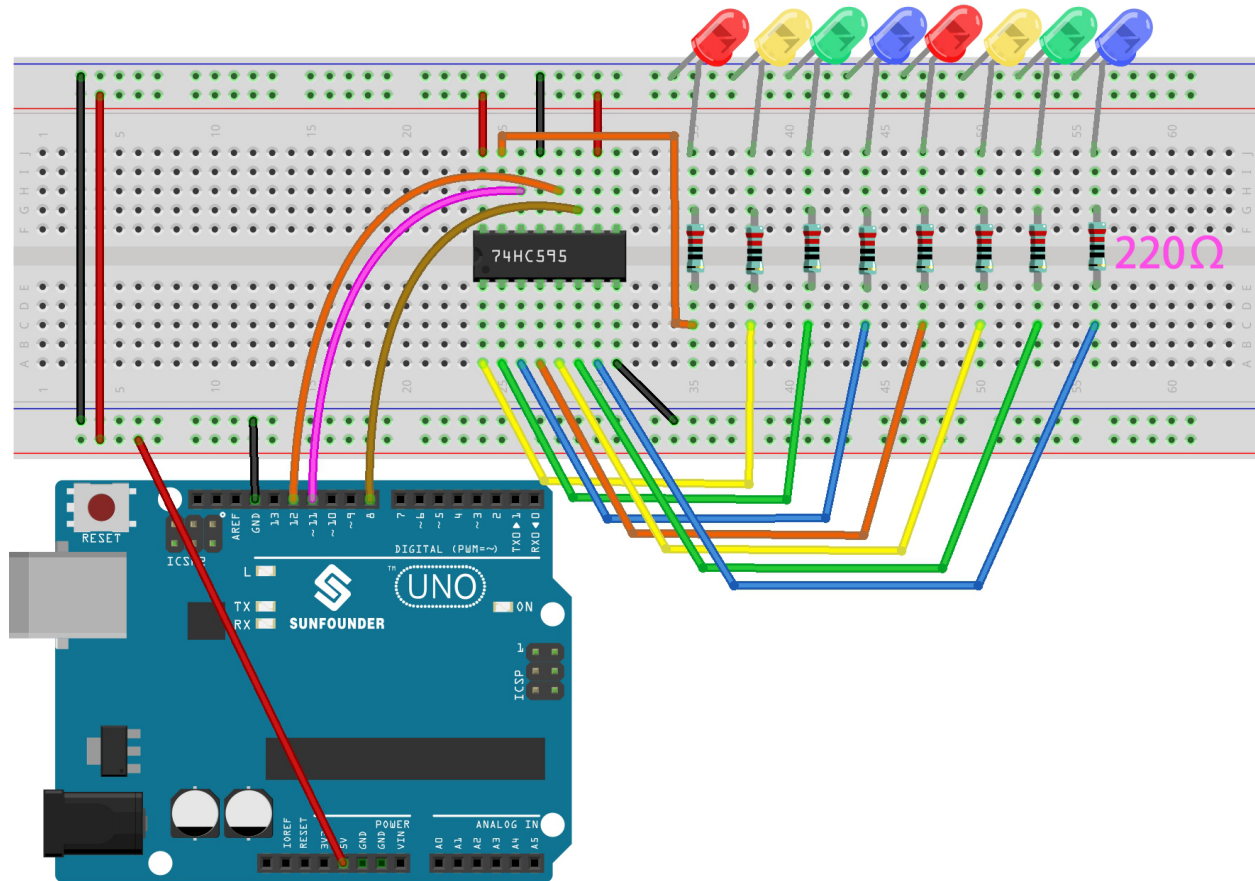
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>74HC595</i>	

## Schéma



- Lorsque MR (broche 10) est au niveau haut et OE (broche 13) au niveau bas, les données sont entrées sur le bord montant de SHcp et vont au registre mémoire via le bord montant de SHcp.
- Si les deux horloges sont connectées ensemble, le registre de décalage est toujours un pulse en avance sur le registre mémoire.
- Il y a une broche d'entrée de décalage série (Ds), une broche de sortie série (Q) et un bouton de réinitialisation asynchrone (niveau bas) dans le registre mémoire.
- Le registre mémoire émet un bus avec un 8 bits parallèle et en trois états.
- Lorsque OE est activé (niveau bas), les données dans le registre mémoire sont émises vers le bus(Q0 ~ Q7).

## Câblage



## Code

### Note :

- Ouvrez le fichier 5.9.shiftout\_led.ino se trouvant dans 3in1-kit\basic\_project\5.9.shiftout\_led.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via [Arduino Web Editor](#).

Une fois les codes téléchargés sur la carte R3, vous pouvez voir les LED s'allumer l'une après l'autre.

### Comment ça fonctionne ?

Déclarez un tableau, stockez plusieurs nombres binaires de 8 bits utilisés pour changer l'état de fonctionnement des huit LED contrôlées par le 74HC595.

```
int dataArray[] = {B00000000, B00000001, B00000011, B00000111, B00001111, B00011111, B00111111, B01111111, B11111111};
```

Mettez STcp à un niveau bas puis à un niveau haut. Cela génèrera une impulsion de bord montant sur STcp.

```
digitalWrite(STcp, LOW);
```

shiftOut() est utilisé pour décaler un octet de données bit par bit, ce qui signifie décaler un octet de données dans dataArray[num] vers le registre de décalage avec la broche DS. **MSBFIRST** indique de déplacer les bits du plus significatif au moins significatif.



```
shiftOut(DS, SHcp, MSBFIRST, dataArray[num]);
```

Après l'exécution de `digitalWrite(STcp, HIGH)`, STcp sera sur un bord montant. À ce moment, les données dans le registre de décalage seront transférées vers le registre mémoire.

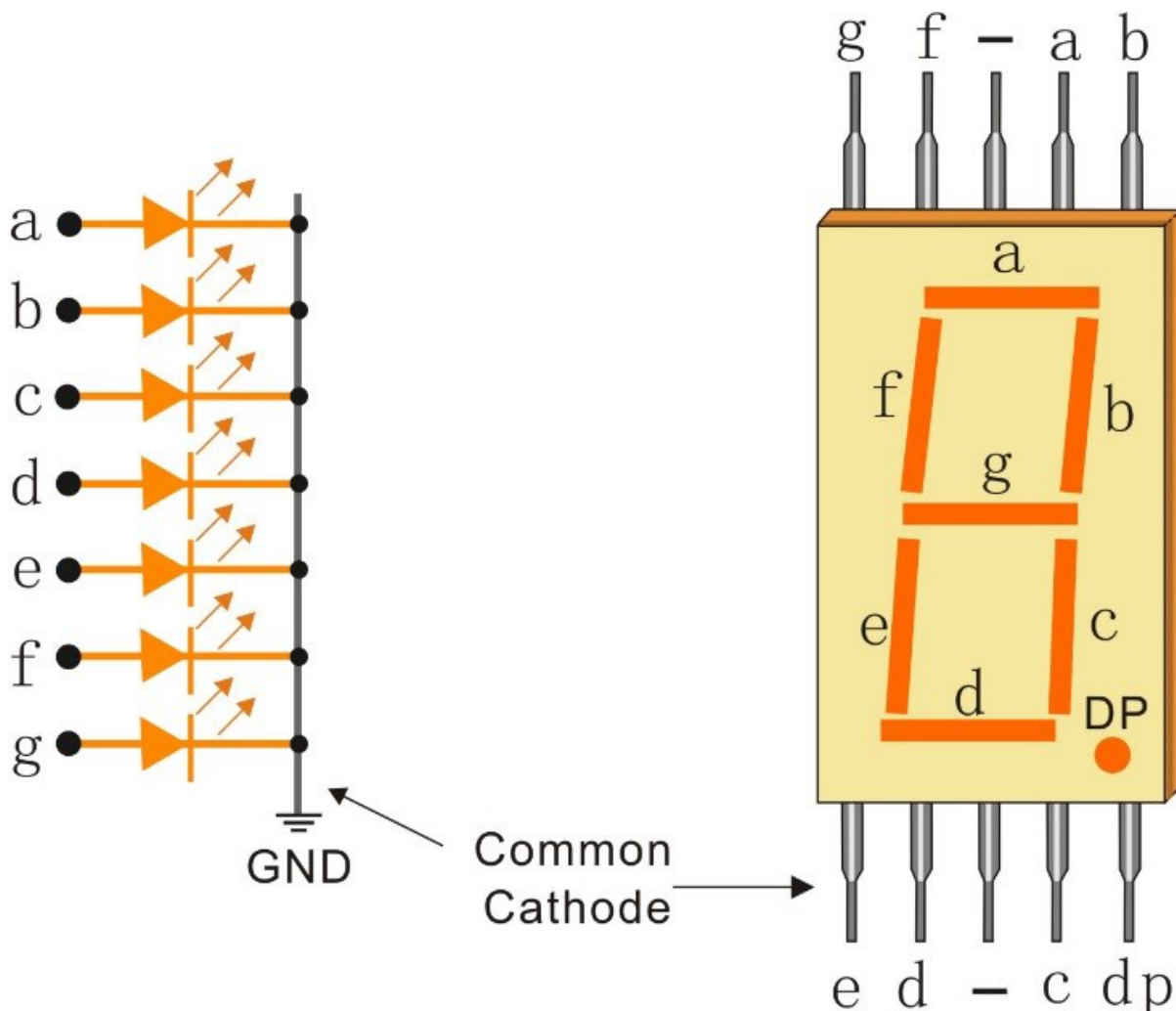
```
digitalWrite(STcp, HIGH);
```

Un octet de données sera transféré dans le registre mémoire après 8 opérations. Ensuite, les données du registre mémoire sont envoyées au bus (Q0-Q7). Par exemple, un `shiftOut` de `B00000001` allumera la LED contrôlée par Q0 et éteindra les LED contrôlées par Q1 à Q7.

### 5.5.10 5.10 ShiftOut(Affichage à segments)

Précédemment, nous avons utilisé la fonction `shiftout()` pour allumer huit LED ; ici, nous l'utilisons pour afficher les chiffres 0 à 9 sur l'affichage à 7 segments.

L'affichage à 7 segments est essentiellement un dispositif composé de 8 LED, dont 7 LED en forme de bande formant un « 8 » et une LED pointillée légèrement plus petite servant de point décimal. Ces LED sont marquées a, b, c, d, e, f, g et dp. Elles ont leurs propres broches d'anode et partagent des cathodes. Leurs emplacements de broches sont indiqués dans la figure ci-dessous.



Composants requis

Pour ce projet, nous avons besoin des composants suivants.

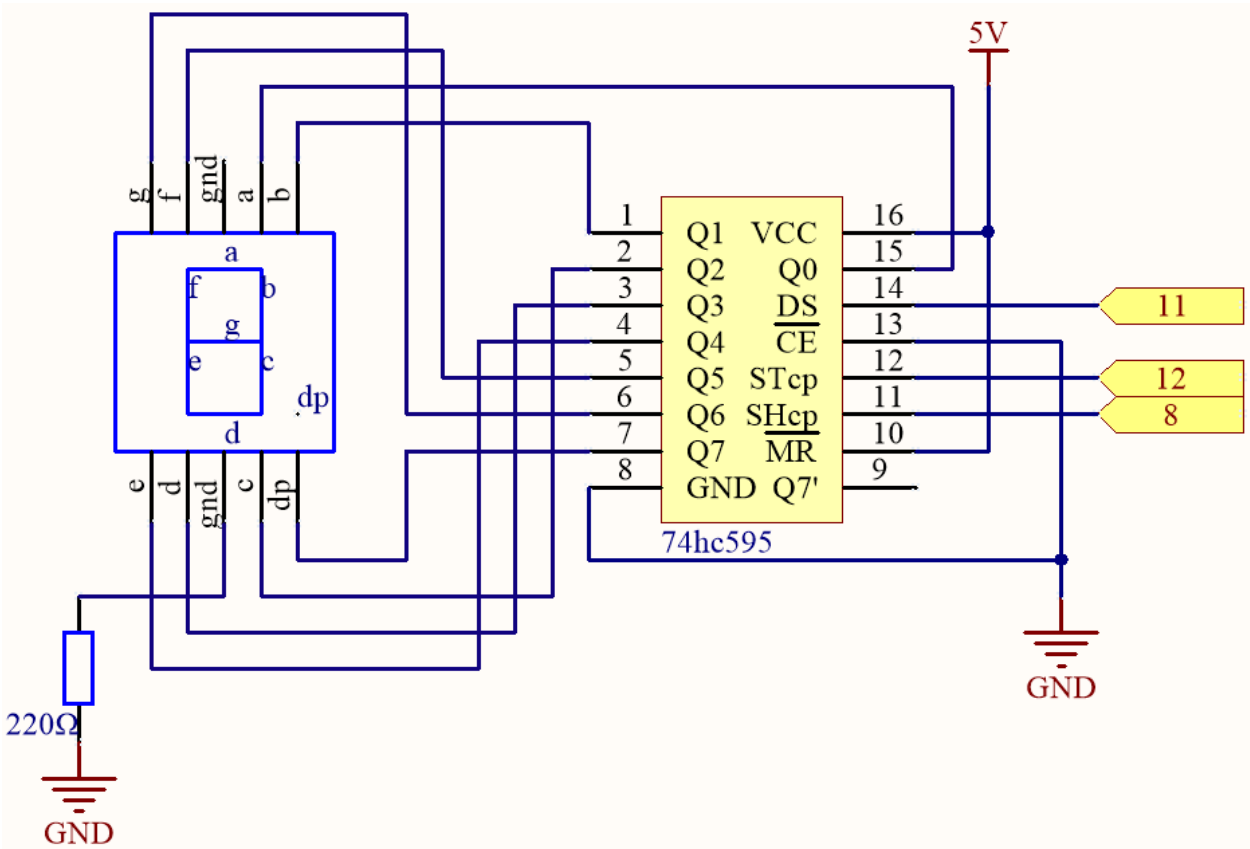
Il est certainement pratique d’acheter un kit complet, voici le lien :

Nom	ARTICLES DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d’essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Affichage 7 segments</i>	
<i>74HC595</i>	

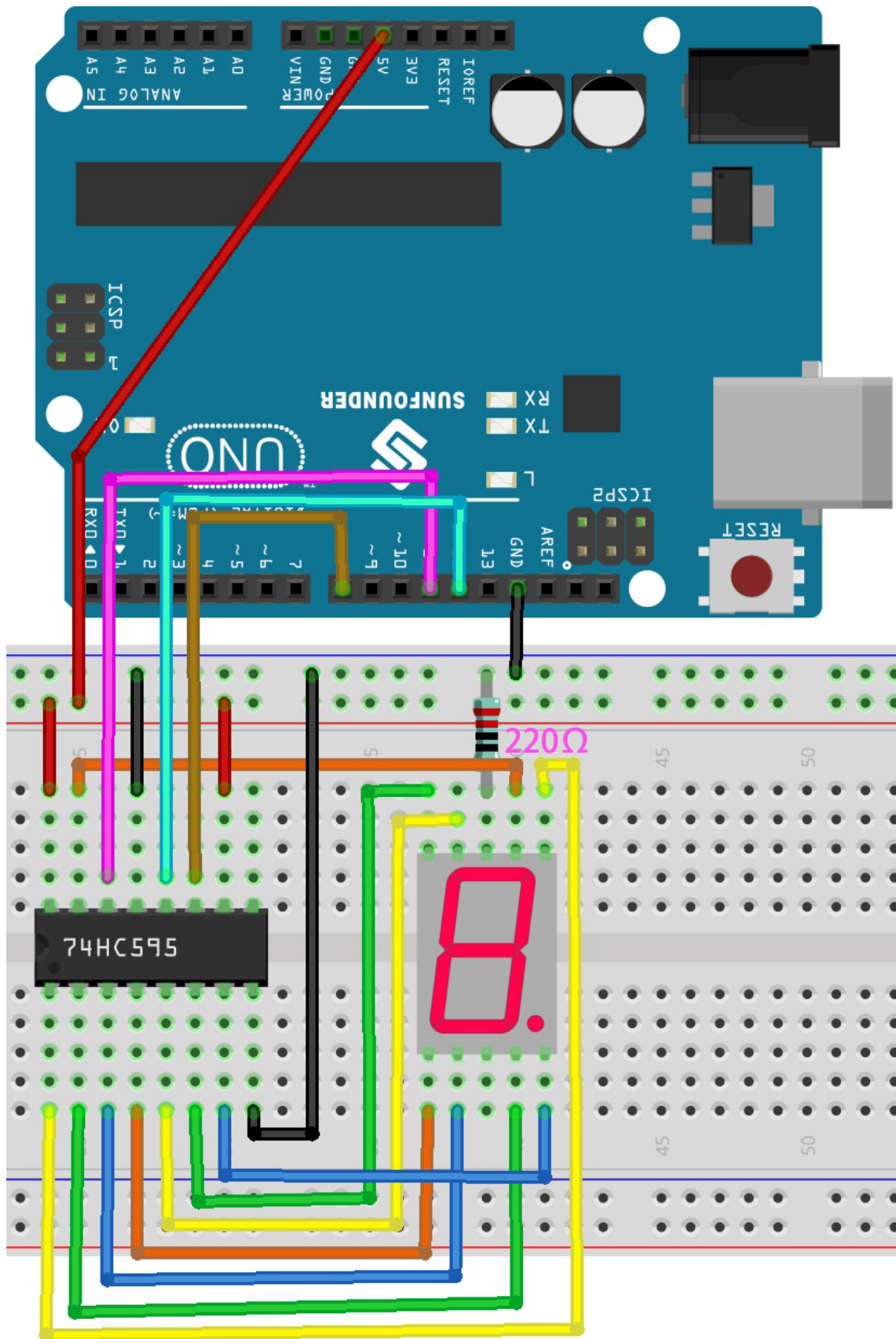
Schéma



**Câblage**

TABLEAU 1 – Wiring

74HC595	LED Segment Display
Q0	a
Q1	b
Q2	c
Q3	d
Q4	e
Q5	f
Q6	g
Q7	dp



## Code

### Note :

- Ouvrez le fichier `5.10.shiftout_segment.ino` situé dans `3in1-kit\basic_project\5.10.shiftout_segment`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via [Arduino Web Editor](#).

Après le téléchargement réussi du code, vous pourrez voir l’Afficheur à Segments LED afficher les chiffres 0 à 9 séquentiellement.

### Comment ça fonctionne ?

`shiftOut()` permet au 74HC595 de produire 8 signaux numériques. Il envoie le dernier bit du nombre binaire à Q0, et le premier bit à Q7. Autrement dit, écrire le nombre binaire « 00000001 » amènera Q0 à produire un niveau haut et Q1 à Q7 un niveau bas.

Supposons que l’afficheur à 7 segments affiche le chiffre « 2 », nous devons écrire un niveau haut pour a, b, d, e et g, et un niveau bas pour c, f et dp. Cela signifie qu’il faut écrire le nombre binaire « 01011011 ». Pour plus de lisibilité, nous utiliserons la notation hexadécimale « 0x5b ».



- [Hexadécimal](#)
- [Convertisseur Binaire-Hexadécimal](#)

De manière similaire, nous pouvons également faire afficher d’autres chiffres à l’afficheur à 7 segments de la même manière. Le tableau suivant montre les codes correspondant à ces chiffres.

TABLEAU 2 – Glyph Code

Numbers	Binary Code	Hex Code
0	00111111	0x3f
1	00000110	0x06
2	01011011	0x5b
3	01001111	0x4f
4	01100110	0x66
5	01101101	0x6d
6	01111101	0x7d
7	00000111	0x07
8	01111111	0x7f
9	01101111	0x6f

Écrivez ces codes dans `shiftOut()` pour faire afficher les nombres correspondants sur l’Afficheur à Segments LED.

### 5.5.11 5.11 Installer des bibliothèques externes

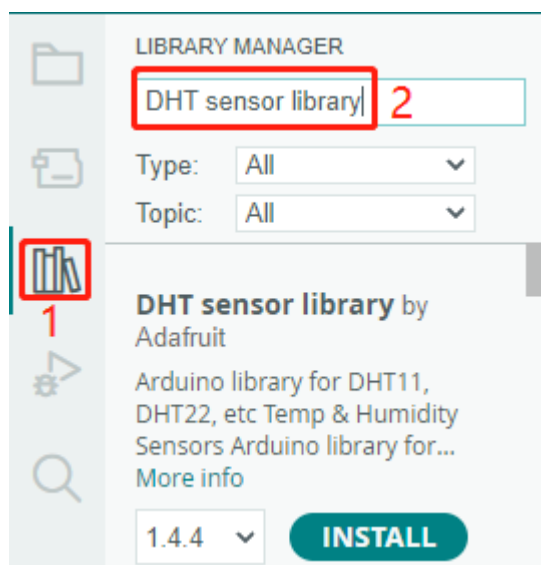
Une bibliothèque est un ensemble de codes ou de fonctions pré-écrits qui étendent les capacités de l’IDE Arduino. Les bibliothèques offrent un code prêt à l’emploi pour diverses fonctionnalités, vous permettant d’économiser du temps et de l’effort dans le codage de caractéristiques complexes.

Il existe deux principales méthodes pour installer des bibliothèques :

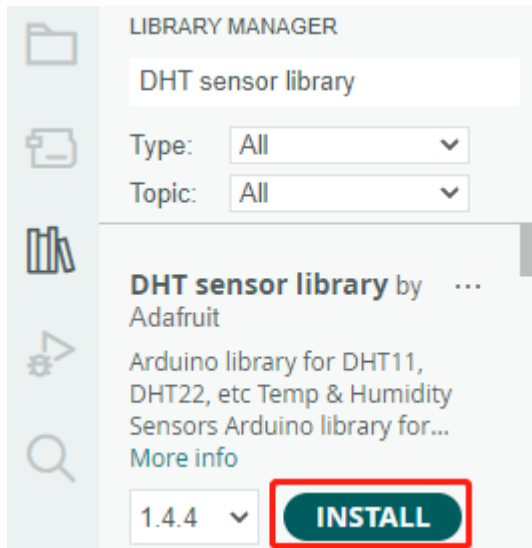
De nombreuses bibliothèques sont disponibles directement via le **Library Manager** d’Arduino. Vous pouvez accéder au **Library Manager** en suivant ces étapes :

1. Dans le **Library Manager**, vous pouvez rechercher la bibliothèque souhaitée par son nom ou parcourir différentes catégories.

**Note :** Dans les projets nécessitant l’installation de bibliothèques, des indications spécifient quelles bibliothèques installer. Suivez les instructions fournies, telles que « La DHT sensor library est utilisée ici, vous pouvez l’installer depuis le **Library Manager**. » Installez simplement les bibliothèques recommandées comme indiqué.



2. Une fois la bibliothèque que vous souhaitez installer trouvée, cliquez dessus puis sur le bouton **Install**.



3. L'IDE Arduino téléchargera et installera automatiquement la bibliothèque pour vous.

### Composants associés

Voici les composants associés, vous pouvez cliquer pour apprendre à les utiliser.

#### 5.11.1 Affichage à Cristaux Liquides

Un LCD1602 I2C est composé d'un LCD1602 et d'un module I2C. Le LCD1602 peut être utilisé pour afficher des caractères, des nombres, etc., mais nécessite l'utilisation de nombreux pins du contrôleur principal. Après configuration d'un module I2C, seulement 2 pins I/O sont nécessaires pour piloter ce LCD1602.

Voyons maintenant comment faire fonctionner ce LCD1602 I2C.

### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

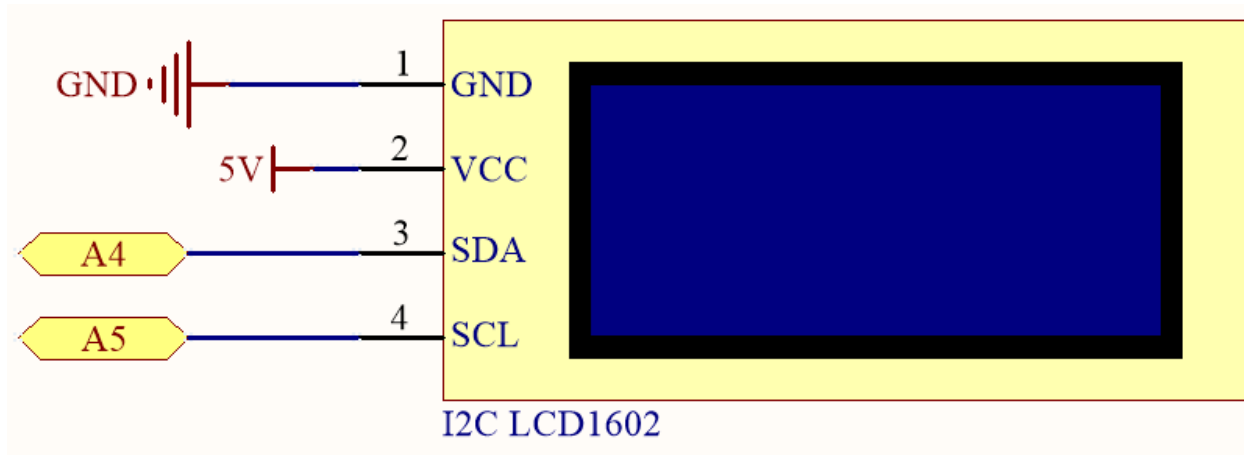
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

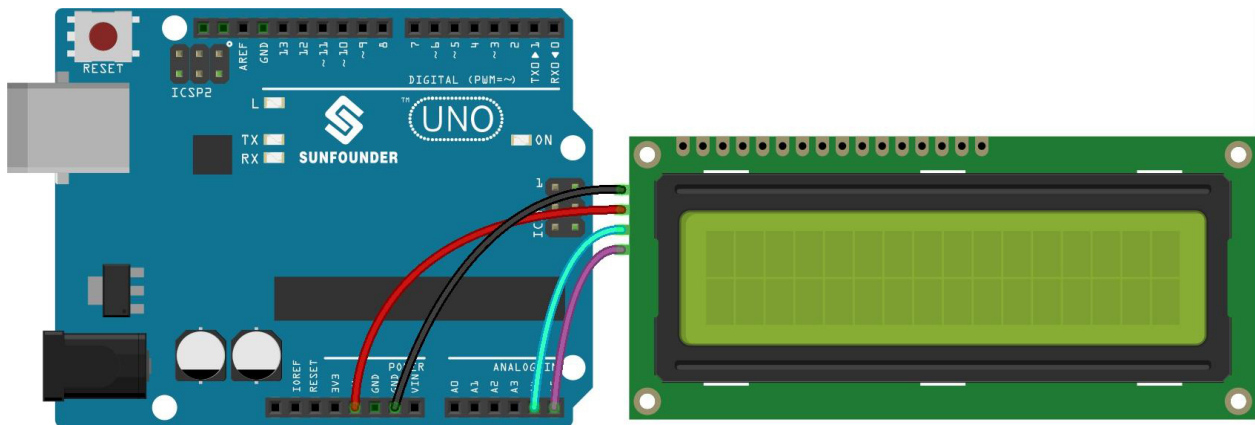
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>I2C LCD1602</i>	

### Schéma



### Câblage



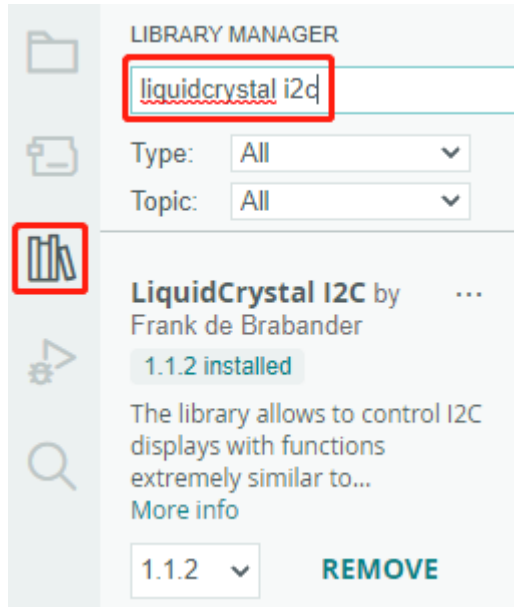
**Note :** Les pins SDA et SCL de la carte R3 sont les pins A4 et A5.

### Code

#### Note :

- Ouvrez le fichier `5.11.liquid_crystal_display.ino` situé dans le dossier `3in1-kit/basic_project/5.11.liquid_crystal_display`.
- Ou copiez ce code dans **Arduino IDE**.
- La bibliothèque `LiquidCrystal I2C` est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.





Après le téléversement réussi du code, vous verrez « SunFounder », « Hello World » sur le LCD1602 I2C.

**Note :** Si le code et le câblage sont corrects, mais que l'écran LCD n'affiche toujours pas de contenu, vous pouvez tourner le potentiomètre situé à l'arrière.

### Comment ça fonctionne ?

En appelant la bibliothèque `LiquidCrystal_I2C.h`, vous pouvez facilement piloter l'écran LCD.

```
#include "LiquidCrystal_I2C.h"
```

Fonctions de la bibliothèque :

```
LiquidCrystal_I2C(uint8_t lcd_Addr, uint8_t lcd_cols, uint8_t lcd_rows)
```

Crée une nouvelle instance de la classe `LiquidCrystal_I2C` qui représente un écran LCD particulier connecté à votre carte Arduino.

- `lcd_Addr` : L'adresse de l'écran LCD est par défaut 0x27.
- `lcd_cols` : Le LCD1602 a 16 colonnes.
- `lcd_rows` : Le LCD1602 a 2 lignes.

```
void init()
```

Initialise l'écran LCD.

```
void backlight()
```

Allume le rétroéclairage (optionnel).

```
void nobacklight()
```

Éteint le rétroéclairage (optionnel).

```
void display()
```

Allume l’affichage de l’écran LCD.

```
void nodisplay()
```

Éteint rapidement l’affichage de l’écran LCD.

```
void clear()
```

Efface l’écran et positionne le curseur à zéro.

```
void setCursor(uint8_t col, uint8_t row)
```

Positionne le curseur à la colonne col et à la ligne row.

```
void print(data, BASE)
```

Affiche du texte sur l’écran LCD.

- data : Les données à afficher (char, byte, int, long ou string).
- BASE (optionnel) : La base dans laquelle afficher les nombres : BIN pour binaire (base 2), DEC pour décimal (base 10), OCT pour octal (base 8), HEX pour hexadécimal (base 16).

### 5.11.2 Récepteur IR

Dans ce projet, vous apprendrez à utiliser un récepteur infrarouge (IR).

Un récepteur infrarouge est un composant qui reçoit des signaux infrarouges et peut recevoir indépendamment des rayons infrarouges et sortir des signaux compatibles avec le niveau TTL. Il est similaire en taille à un transistor classique emballé dans du plastique et convient à toutes sortes de télécommandes infrarouges et de transmissions infrarouges.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

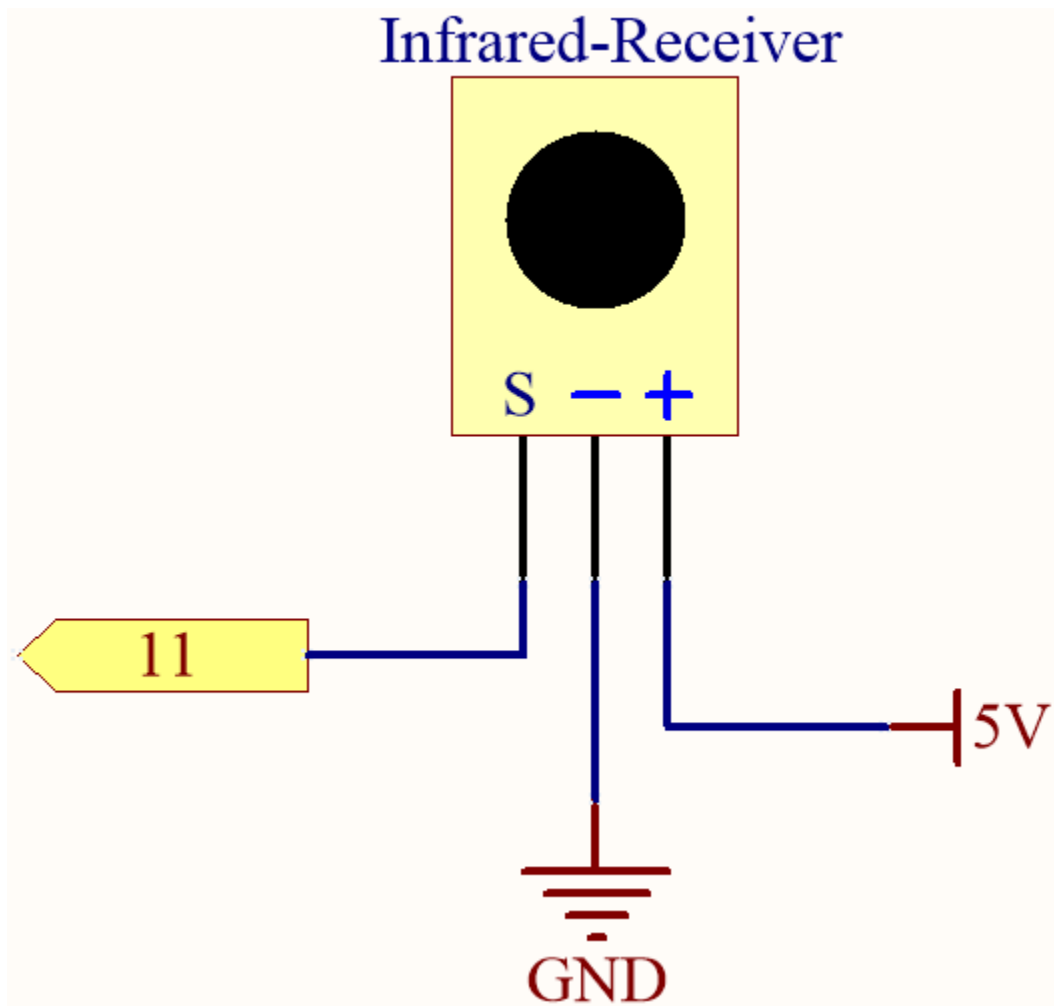
Il est certainement pratique d’acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

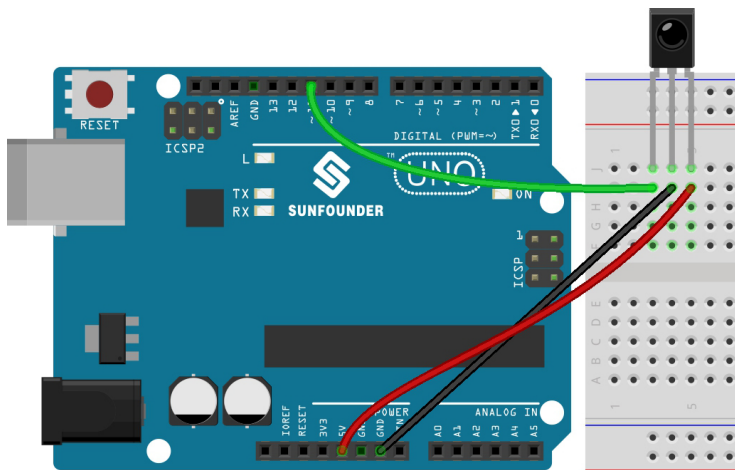
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Récepteur IR</i>	-

#### Schéma



### Câblage

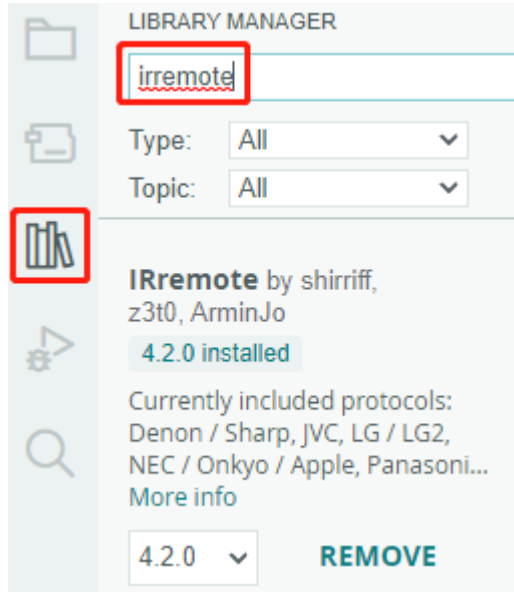
Dans cet exemple, nous connectons la broche gauche du récepteur IR à la broche 11, la broche centrale à GND, et la broche droite à 5V.



### Code

**Note :**

- Ouvrez le fichier 5.11.ir\_receiver.ino situé dans le dossier 3in1-kit\basic\_project\5.11.ir\_receiver.
- Ou copiez ce code dans **Arduino IDE**.
- La bibliothèque IRremote est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après avoir téléversé les codes sur la carte R3, vous pouvez constater que la valeur actuelle du bouton pressé de la télécommande IR s'affiche sur le moniteur série.

**Comment ça fonctionne ?**

Ce code est conçu pour fonctionner avec une télécommande infrarouge (IR) en utilisant la bibliothèque IRremote. Voici le détail :

1. Inclusion des bibliothèques : Cela inclut la bibliothèque IRremote, qui fournit des fonctions pour travailler avec les télécommandes IR.

```
#include <IRremote.h>
```

2. Définit le pin Arduino auquel est connecté le pin de signal du capteur IR.

```
const int IR_RECEIVE_PIN = 11; // Define the pin number for the IR Sensor
```

3. Initialise la communication série à une vitesse de 9600 bauds. Initialise le récepteur IR sur le pin spécifié (IR\_RECEIVE\_PIN) et active le retour LED (si applicable).

```
void setup() {
  Serial.begin(9600); // Start serial
  // communication at 9600 baud rate
  IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK); // Start the IR
  // receiver
}
```

4. La boucle s'exécute continuellement pour traiter les signaux entrants de la télécommande IR.

```

void loop() {
    if (IrReceiver.decode()) {
        String decodedValue = decodeKeyValue(IrReceiver.decodedIRData.
→command);
        if (decodedValue != "ERROR") {
            Serial.println(decodedValue);
            delay(100);
        }
        IrReceiver.resume(); // Enable receiving of the next value
    }
}

```

- Vérifie si un signal IR est reçu et décodé avec succès.
- Décode la commande IR et la stocke dans decodedValue à l'aide d'une fonction personnalisée decodeKeyValue().
- Vérifie si la valeur décodée n'est pas une erreur.
- Affiche la valeur IR décodée sur le moniteur série.
- Reprend la réception du signal IR pour le prochain signal.

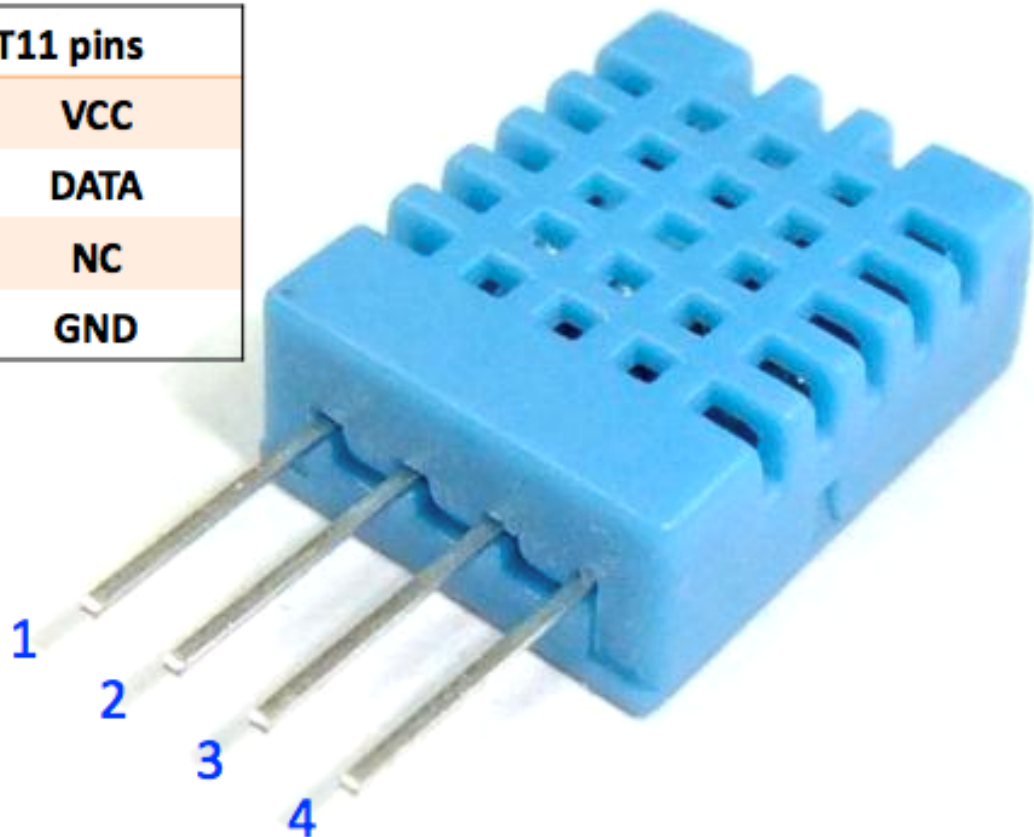
### 5.11.3 Température - Humidité

L'humidité et la température sont étroitement liées, de la quantité physique elle-même à la vie quotidienne des gens. La température et l'humidité de l'environnement humain affectent directement la fonction de thermorégulation et l'effet de transfert de chaleur du corps humain. Cela peut également influencer l'activité de pensée et l'état mental, affectant ainsi l'efficacité de nos études et de notre travail.

La température est l'une des sept quantités physiques de base dans le Système international d'unités, utilisée pour mesurer le degré de chaud et de froid d'un objet. Le degré Celsius est l'une des échelles de température les plus utilisées dans le monde, exprimée par le symbole « °C ».

L'humidité est la concentration de vapeur d'eau présente dans l'air. L'humidité relative de l'air est couramment utilisée dans la vie quotidienne et est exprimée en %HR. L'humidité relative est étroitement liée à la température. Pour un certain volume de gaz scellé, plus la température est élevée, plus l'humidité relative est basse, et inversement.

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



Le dht11, un capteur numérique de température et d'humidité, est fourni dans ce kit. Il utilise un capteur d'humidité capacitif et un thermistor pour mesurer l'air ambiant et produit un signal numérique sur la broche de données.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

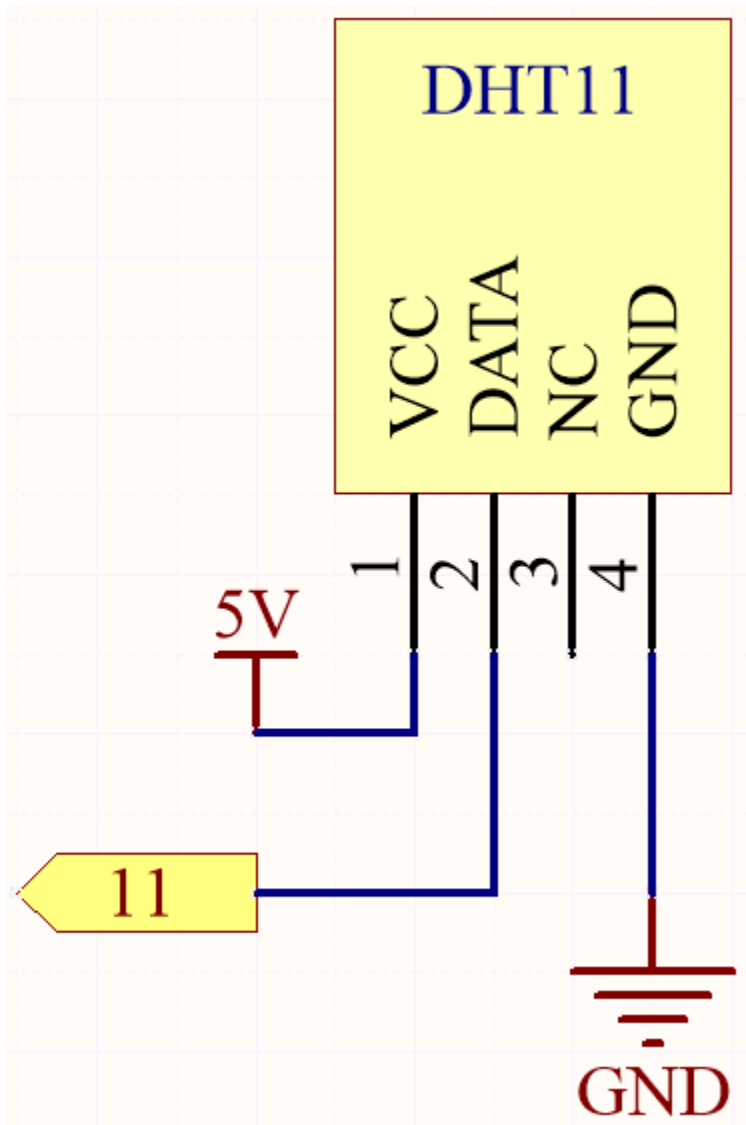
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

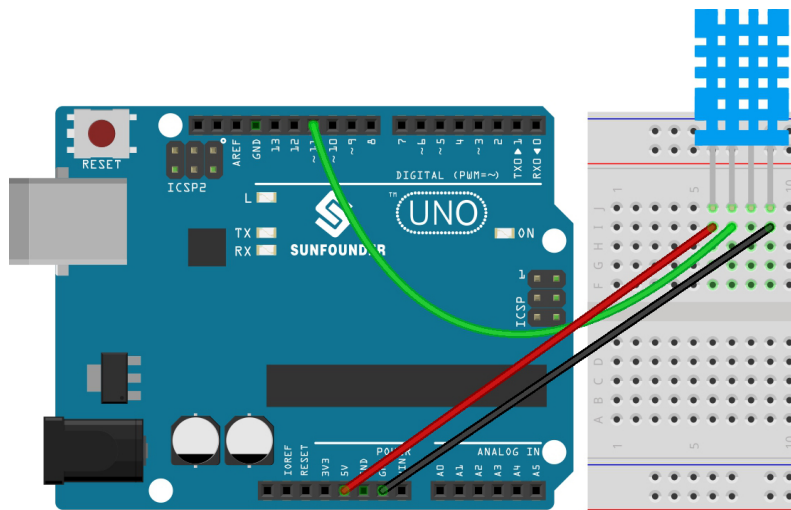
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Capteur d'Humidité et de Température DHT11</i>	-

#### Schéma

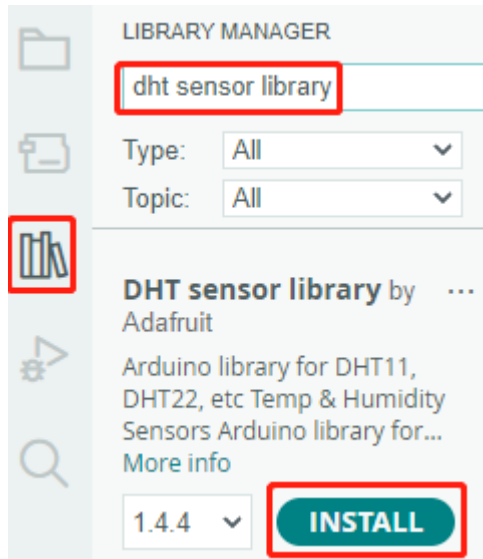


### Câblage



**Code****Note :**

- Ouvrez le fichier `5.11.temperature_humidity.ino` situé dans le dossier `3in1-kit\basic_project\5.11.temperature_humidity`.
- Ou copiez ce code dans **Arduino IDE**.
- La bibliothèque `DHT sensor library` est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après le téléversement réussi du code, vous verrez que le moniteur série affiche en continu la température et l'humidité. Au fur et à mesure que le programme fonctionne de manière stable, ces deux valeurs deviendront de plus en plus précises.

**Comment ça fonctionne ?**

1. Inclut la bibliothèque `DHT.h`, qui offre des fonctions pour interagir avec les capteurs DHT. Ensuite, définit le pin et le type pour le capteur DHT.

```
#include "DHT.h"

#define DHTPIN 11 // Set the pin connected to the DHT11 data pin
#define DHTTYPE DHT11 // DHT 11

DHT dht(DHTPIN, DHTTYPE);
```

2. Initialise la communication série à une vitesse de 115200 bauds et initialise le capteur DHT.

```
void setup() {
  Serial.begin(115200);
  Serial.println("DHT11 test!");
  dht.begin();
}
```

3. Dans la fonction `loop()`, lire les valeurs de température et d'humidité du capteur DHT11, et les afficher sur le moniteur série.



```

void loop() {
    // Wait a few seconds between measurements.
    delay(2000);

    // Reading temperature or humidity takes about 250 milliseconds!
    // Sensor readings may also be up to 2 seconds 'old' (it's a very slow
    ↪ sensor)
    float humidity = dht.readHumidity();
    // Read temperature as Celsius (the default)
    float temperture = dht.readTemperature();

    // Check if any reads failed and exit early (to try again).
    if (isnan(humidity) || isnan(temperture)) {
        Serial.println("Failed to read from DHT sensor!");
        return;
    }
    // Print the humidity and temperature
    Serial.print("Humidity: ");
    Serial.print(humidity);
    Serial.print(" %\t");
    Serial.print("Temperature: ");
    Serial.print(temperture);
    Serial.println(" *C");
}

```

- La fonction `dht.readHumidity()` est appelée pour lire la valeur d'humidité du capteur DHT.
- La fonction `dht.readTemperature()` est appelée pour lire la valeur de température du capteur DHT.
- La fonction `isnan()` est utilisée pour vérifier si les lectures sont valides. Si la valeur d'humidité ou de température est NaN (non un nombre), cela indique une lecture échouée du capteur, et un message d'erreur est imprimé.

### 5.5.12 5.12 Lecture Série

Vous l'avez peut-être remarqué en utilisant la fonction `Serial.print()`. Puisqu'il y a impression, y a-t-il lecture ? À quoi sert la boîte de texte sur le moniteur série ? Oui, vous l'avez deviné, il existe des moyens de contrôler des programmes et des circuits en entrant des informations via la boîte de texte sur le moniteur série.

Dans ce projet, nous utiliserons le LCD1602 I2C pour afficher le texte entré dans le moniteur série afin d'expérimenter l'utilisation de `Serial.read()`.

#### Composants Nécessaires

Pour ce projet, nous aurons besoin des composants suivants.

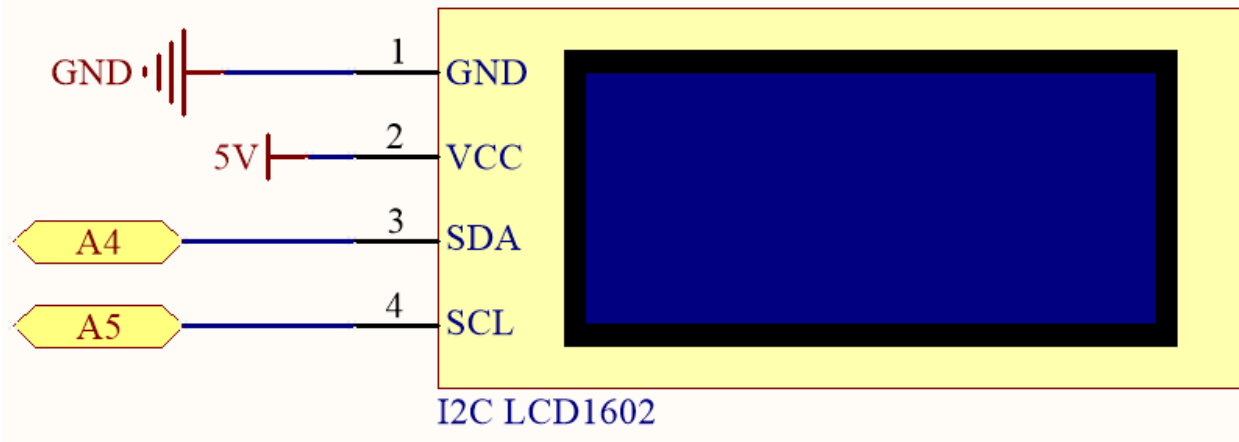
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

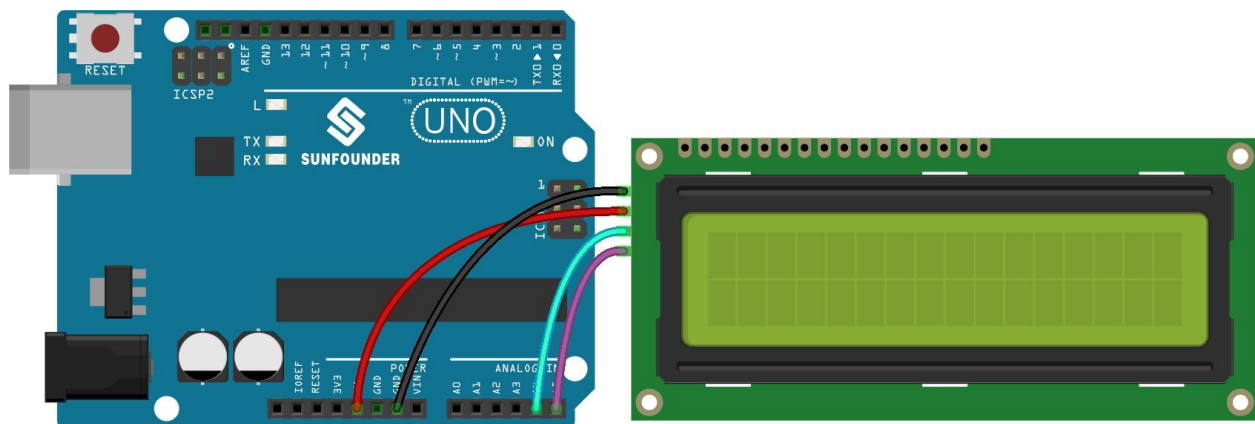
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>I2C LCD1602</i>	

### Schéma



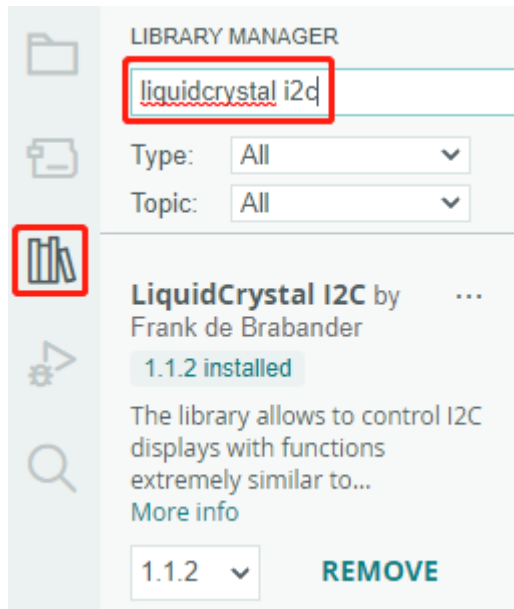
### Câblage



### Code

#### Note :

- Ouvrez le fichier 5.12.serial\_read.ino situé dans le dossier 3in1-kit\basic\_project\5.12.serial\_read.
- Ou copiez ce code dans **Arduino IDE**.
- La bibliothèque LiquidCrystal I2C est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après le téléversement réussi du code, vous pouvez entrer du texte dans la boîte de texte sur le moniteur série, et le LCD affichera les informations.

#### Comment ça fonctionne ?

```
void loop()
{
  // when characters arrive over the serial port...
  if (Serial.available()) {
    // wait a bit for the entire message to arrive
    delay(100);
    // clear the screen
    lcd.clear();
    // read all the available characters
    while (Serial.available() > 0) {
      // display each character to the LCD
      lcd.write(Serial.read());
    }
  }
}
```

- `Serial.available()` peut obtenir le nombre de caractères disponibles dans le flux entrant lorsque vous tapez quelque chose depuis la boîte de texte. Comme il y a deux terminateurs dans l'entrée, vous avez en fait 3 caractères lorsque vous tapez A, et 4 caractères lorsque vous tapez AB.
- `Serial.read()` prendra le premier caractère du flux entrant. Par exemple, si vous avez tapé AB, en appelant `Serial.read()` une seule fois, vous obtiendrez le caractère A; au deuxième appel, vous obtiendrez B; au troisième et quatrième appel, vous obtiendrez deux symboles de fin; appeler cette fonction lorsque le flux entrant n'a pas de caractères disponibles résultera en une erreur.

En résumé, il est courant de combiner les deux ci-dessus, en utilisant une boucle `while` pour lire tous les caractères entrés à chaque fois.

```
while (Serial.available() > 0) {  
    Serial.print(Serial.read());  
}
```

D'ailleurs, si vous n'utilisez pas `Serial.read()` pour obtenir des caractères du flux entrant, les caractères du flux entrant s'empileront. Par exemple, si vous tapez A suivi de AB, le flux entrant accumulera 7 caractères.

### 5.5.13 5.13 Interruption

Si vous utilisez des `delay()` dans un projet utilisant des capteurs, vous pourriez constater que lors du déclenchement de ces capteurs, le programme peut ne pas réagir. Cela est dû au fait que l'instruction `delay` provoque la suspension du programme, et celui-ci ne pourra pas obtenir le signal envoyé par le capteur à la carte de contrôle principale.

Dans ce cas, on peut utiliser l'interruption. L'interruption permet au programme de ne pas manquer une impulsion.

Dans ce chapitre, nous utilisons un buzzer actif et des boutons pour expérimenter le processus d'utilisation de l'interruption.

Dans la fonction `loop()`, `delay(1000)` est utilisé pour compter les secondes. Placez le bouton pour contrôler le buzzer dans l'ISR, afin qu'il ne soit pas perturbé par le `delay` et qu'il puisse accomplir la tâche en douceur.

---

**Note :** Les ISR sont des types spéciaux de fonctions qui ont certaines limitations uniques que la plupart des autres fonctions n'ont pas. Une ISR ne peut pas avoir de paramètres, et elles ne devraient rien retourner. Généralement, une ISR doit être aussi courte et rapide que possible. Si votre croquis utilise plusieurs ISR, une seule peut s'exécuter à la fois, les autres interruptions seront exécutées après la fin de la courante, dans un ordre qui dépend de la priorité qu'elles ont.

---

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

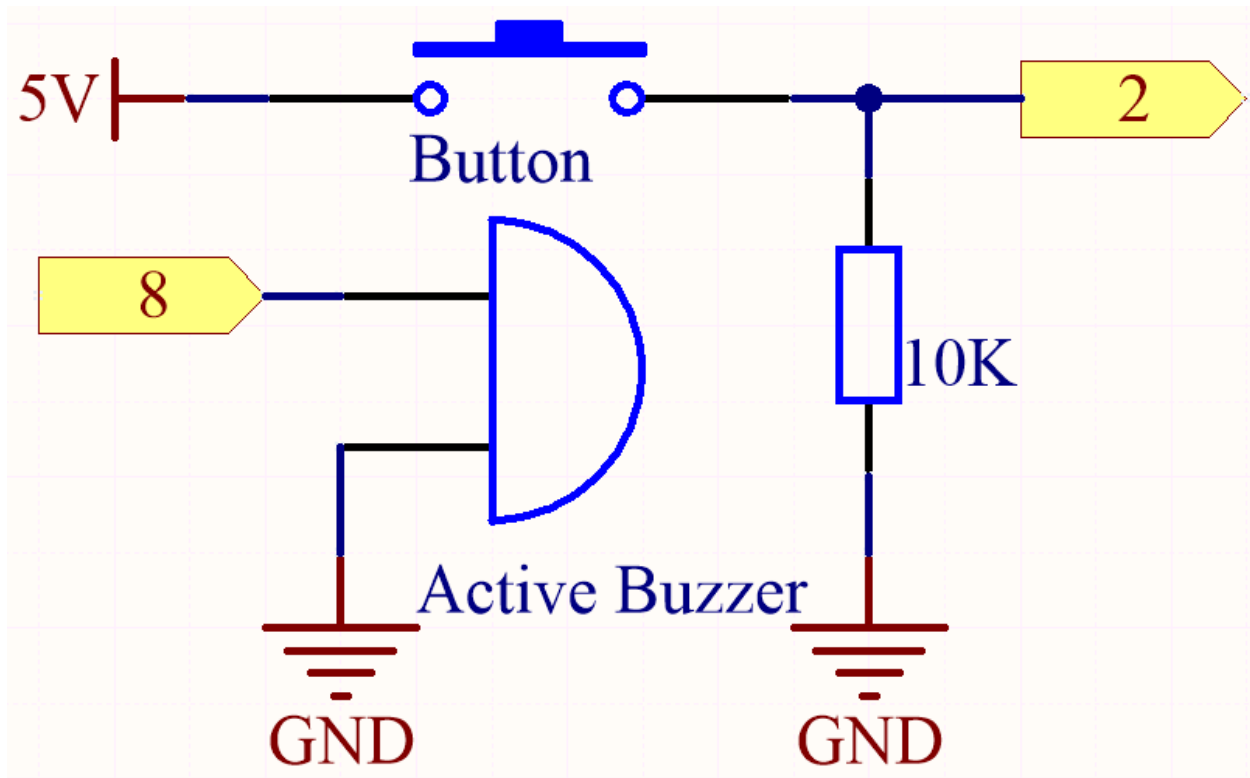
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

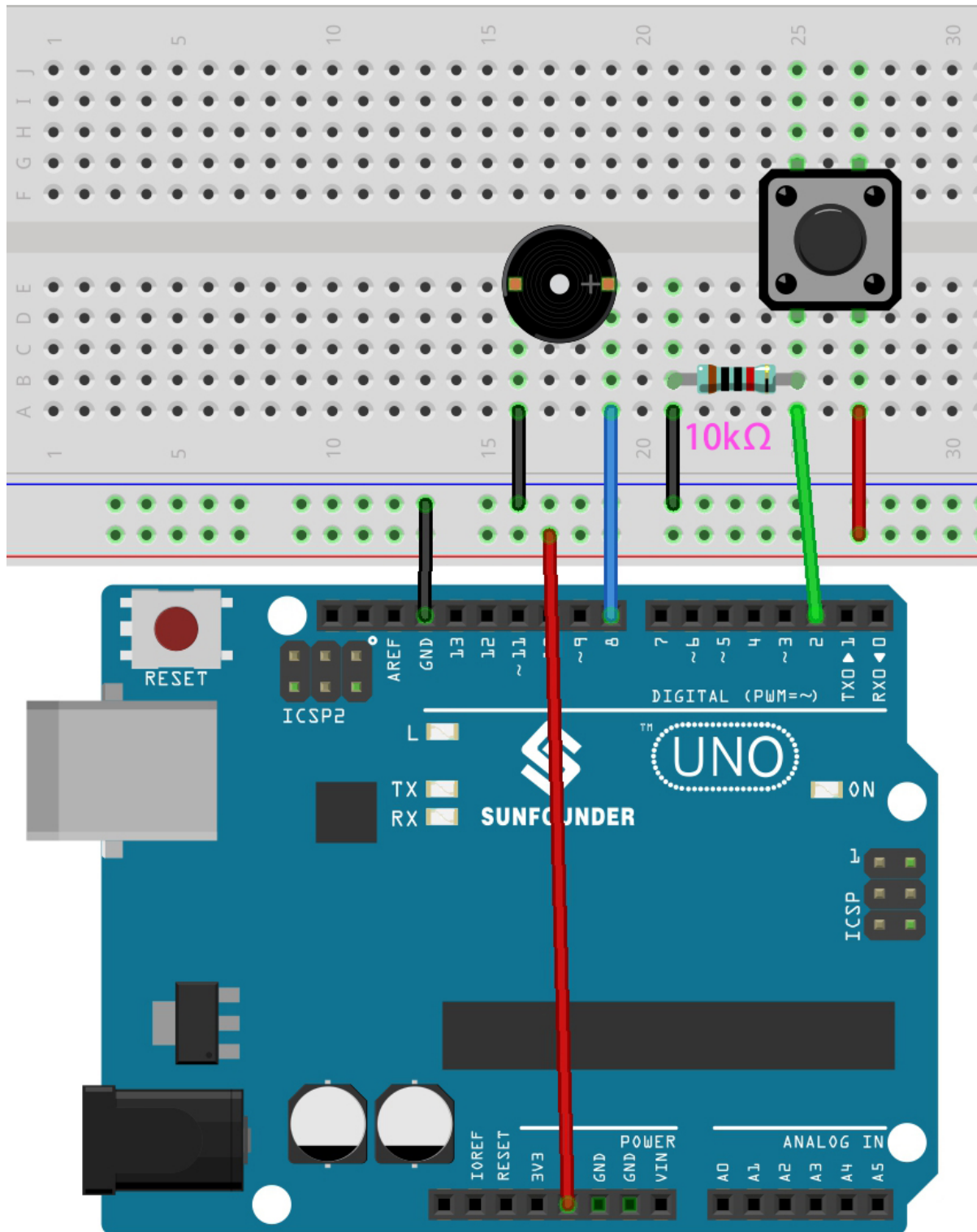
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Bouton</i>	
<i>Buzzer</i>	-

#### Schéma



Câblage



## Code

### Note :

— Ouvrez le fichier 5.13.interrupt.ino situé dans le dossier 3in1-kit\basic\_project\5.13.

`interrupt.`

- Ou copiez ce code dans **Arduino IDE**.
- Ou téléversez le code via l'[Éditeur Web Arduino](#).

Après le téléversement réussi du code, activez le moniteur série et vous verrez un nombre s'incrémentant automatiquement s'afficher chaque seconde. Si vous appuyez sur le bouton, le buzzer émettra un son. La fonction du buzzer contrôlée par le bouton et la fonction de temporisation ne sont pas en conflit l'une avec l'autre.

#### Comment ça fonctionne ?

- `attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)` : Ajouter une interruption.

##### Syntaxe

`attachInterrupt(digitalPinToInterrupt(pin), ISR, mode)`

##### Paramètres

- `pin` : le numéro de pin Arduino. Vous devriez utiliser `digitalPinToInterrupt(pin)` pour convertir le pin numérique réel en un numéro d'interruption spécifique. Par exemple, si vous vous connectez au pin 3, utilisez son `digitalPinToInterrupt(3)` comme premier paramètre.
- `ISR` : l'ISR à appeler lorsque l'interruption se produit ; cette fonction ne doit prendre aucun paramètre et ne rien retourner. Cette fonction est parfois appelée routine de service d'interruption.
- `mode` : définit quand l'interruption doit être déclenchée. Quatre constantes sont prédéfinies comme valeurs valides :
  - `LOW` pour déclencher l'interruption lorsque le pin est bas,
  - `CHANGE` pour déclencher l'interruption chaque fois que la valeur du pin change.
  - `RISING` pour déclencher lorsque le pin passe de bas à haut.
  - `FALLING` pour quand le pin passe de haut à bas.

**Note :** Différentes cartes de contrôle principales peuvent utiliser les pins d'interruption différemment. Sur la carte R3, seuls les pins 2 et 3 peuvent utiliser l'interruption.

## 5.5.14 5.14 Calibration

Lorsque vous utilisez des composants d'entrée analogique, tels que des photorésistances, des capteurs d'humidité du sol, etc., vous pouvez constater que leur plage de lecture n'est pas de 0 à 1023, mais plutôt une plage comme 0 à 800 ou 600 à 1000, car il est impossible d'atteindre les limites de ces dispositifs en utilisation normale.

Dans ce cas, une technique de calibration des entrées du capteur peut être utilisée. Au démarrage, faites mesurer les lectures du capteur par la carte de contrôle pendant cinq secondes et enregistrez les lectures les plus hautes et les plus basses. Cette lecture de cinq secondes définit les valeurs minimales et maximales attendues des lectures prises pendant le cycle.

Dans ce projet, nous utilisons une photorésistance et un buzzer passif pour implémenter un jeu semblable au [theremin](#) en utilisant la technique de calibration décrite ci-dessus.

**Note :** Le [theremin](#) est un instrument de musique électronique qui ne nécessite aucun contact physique. Il génère différents tons en détectant la position des mains du joueur.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

## SunFounder 3in1 Kit

---

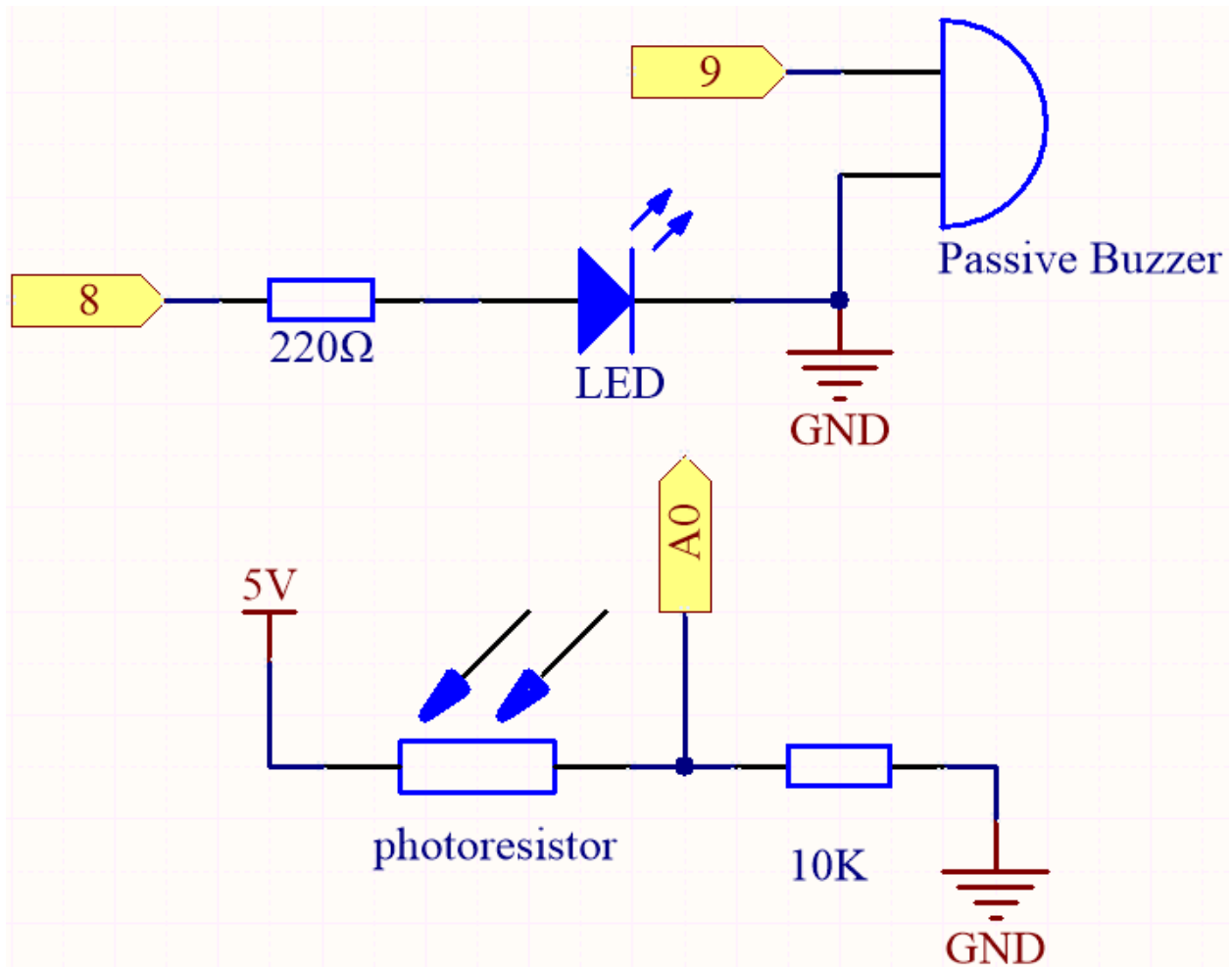
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

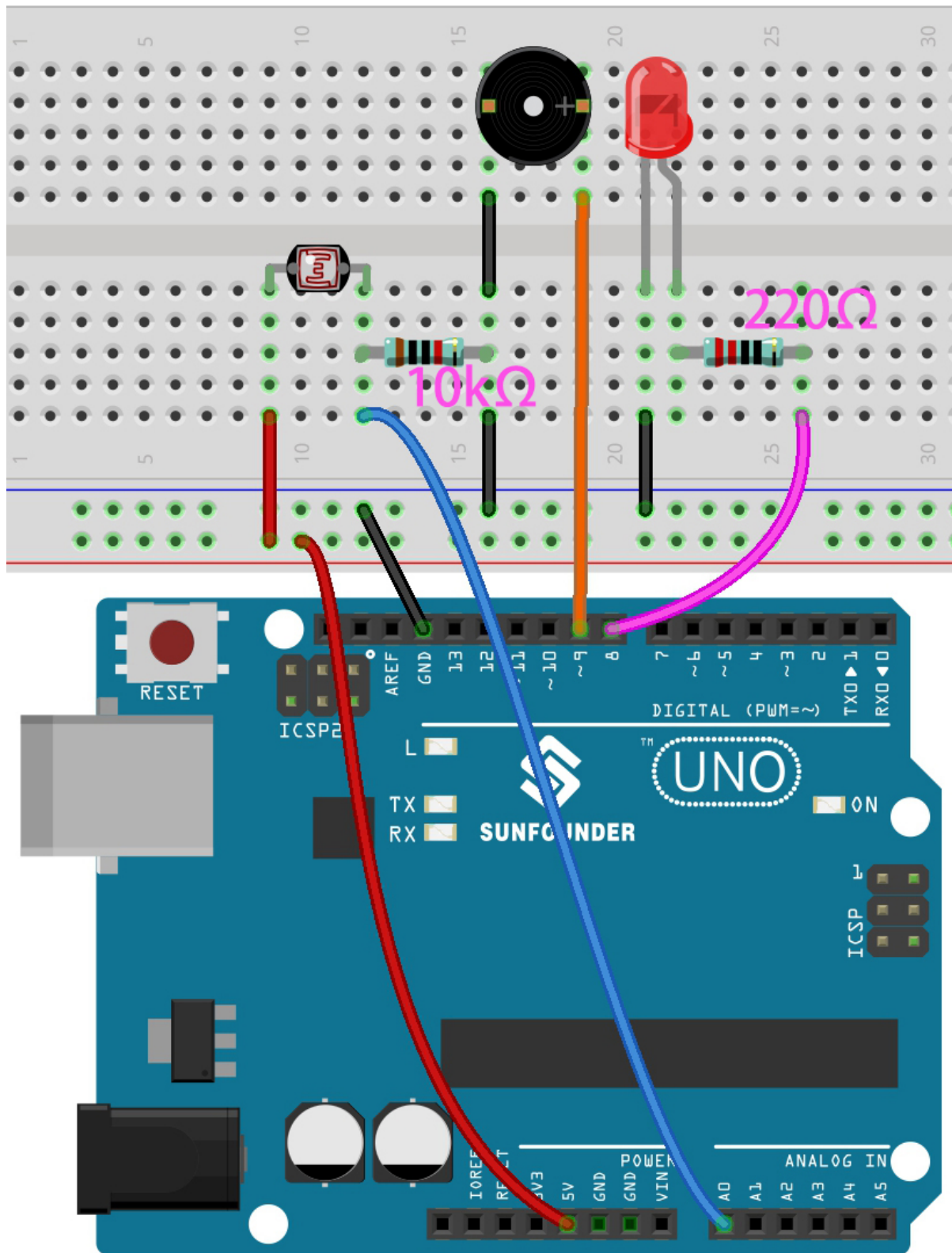
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Buzzer</i>	
<i>LED</i>	
<i>Photorésistance</i>	

## Schéma





Câblage



Code

Note :

- Ouvrez le fichier 5.14.calibration.ino situé dans le dossier 3in1-kit\basic\_project\5.14.calibration.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléversez le code via l'Éditeur Web Arduino.

Après le téléversement réussi du code, la LED s'allumera, et nous aurons 5 secondes pour calibrer la plage de détection de la photorésistance. Cela est dû au fait que nous pouvons être dans un environnement lumineux différent à chaque utilisation (par exemple, l'intensité de la lumière est différente entre midi et le crépuscule).

À ce moment, nous devons balancer nos mains en haut et en bas sur la photorésistance, et la plage de mouvement de la main sera calibrée à la plage de jeu de cet instrument.

Après 5 secondes, la LED s'éteindra et nous pourrons agiter nos mains sur la photorésistance pour jouer.

### Comment ça fonctionne ?

1. Définir les valeurs initiales et les pins de tous les composants.

```
const int buzzerPin = 9;
const int ledPin = 8;
const int photocellPin = A0; //photoresistor attach to A2

int lightLow = 1023;
int lightHigh = 0;

int sensorValue = 0; // value read from the sensor
int pitch = 0; // sensor value converted into LED 'bars'

unsigned long previousMillis = 0;
const long interval = 5000;
```

2. Mettez en place un processus de calibration dans setup().

```
void setup()
{
  pinMode(buzzerPin, OUTPUT); // make buzzer output
  pinMode(ledPin, OUTPUT); // make the LED pin output

  /* calibrate the photoresistor max & min values */
  previousMillis = millis();
  digitalWrite(ledPin, HIGH);
  while (millis() - previousMillis <= interval) {
    sensorValue = analogRead(photocellPin);
    if (sensorValue > lightHigh) {
      lightHigh = sensorValue;
    }
    if (sensorValue < lightLow) {
      lightLow = sensorValue;
    }
  }
  digitalWrite(ledPin, LOW);
}
```

Le déroulement du travail est le suivant.

- en utilisant millis() pour le chronométrage avec un intervalle de 5000 ms.

```
previousMillis = millis();
...
while (millis() - previousMillis <= interval) {
...
}
```

— Pendant ces cinq secondes, agitez une main autour de la photorésistance, les valeurs maximales et minimales de la lumière détectée sont enregistrées et assignées respectivement à `lightHigh` et `lightLow`.

```
sensorValue = analogRead(photocellPin);
if (sensorValue > lightHigh) {
    lightHigh = sensorValue;
}
if (sensorValue < lightLow) {
    lightLow = sensorValue;
}
```

3. Vous pouvez maintenant commencer à jouer ce Theremin. Lisez la valeur de la photorésistance à `sensorValue` et mappez-la de la petite plage à la grande plage pour être utilisée comme fréquence du buzzer.

```
void loop()
{
    /* play*/
    sensorValue = analogRead(photocellPin); //read the value of A0
    pitch = map(sensorValue, lightLow, lightHigh, 50, 6000); // map to the
    ↪ buzzer frequency
    if (pitch > 50) {
        tone(buzzerPin, pitch, 20);
    }
    delay(10);
}
```

### 5.5.15 5.15 EEPROM

**EEPROM** est une mémoire, donc les données qu'elle stocke ne seront pas effacées lorsque la carte de contrôle principale est éteinte. Vous pouvez l'utiliser pour enregistrer certaines données et les lire la prochaine fois que vous l'allumez.

Par exemple, vous pouvez créer un compteur de sport qui suit le nombre de sauts à la corde que vous faites chaque jour.

Vous pouvez également y écrire des données dans un programme et les lire dans un autre. Par exemple, lorsque vous travaillez sur un projet de voiture, les vitesses des deux moteurs sont incohérentes. Vous pouvez écrire un programme de calibration pour enregistrer la valeur de compensation de la vitesse du moteur.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

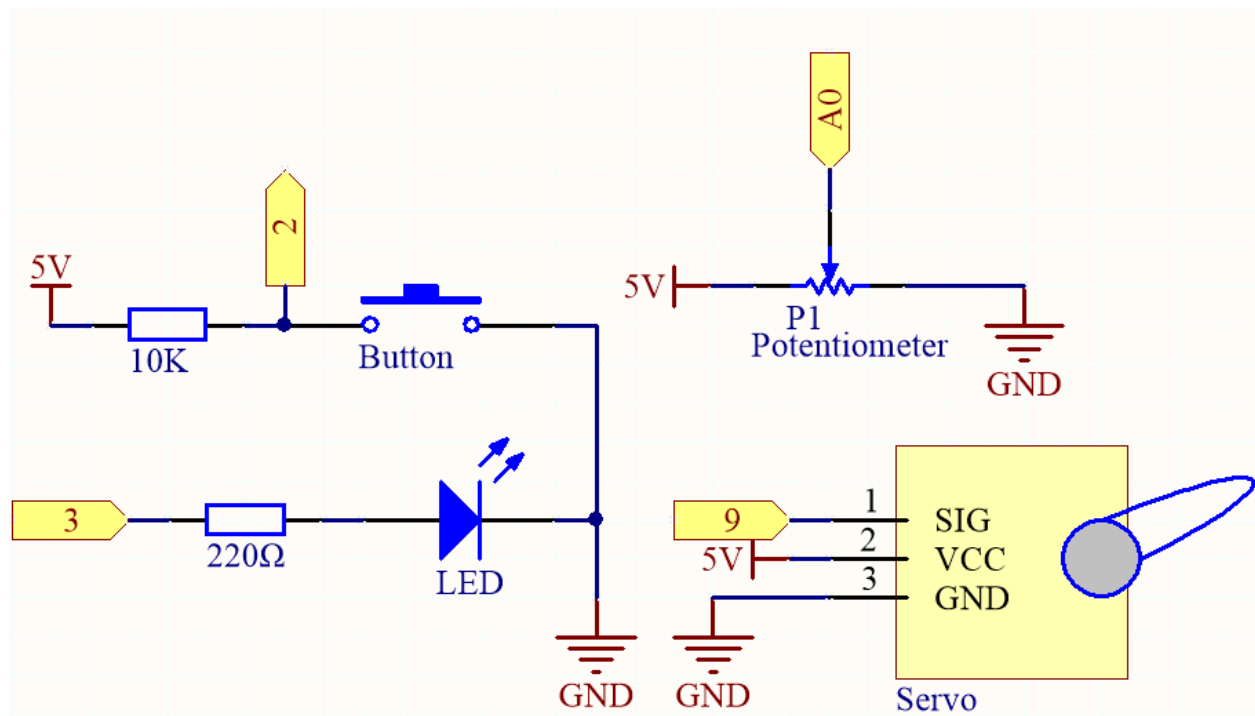
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

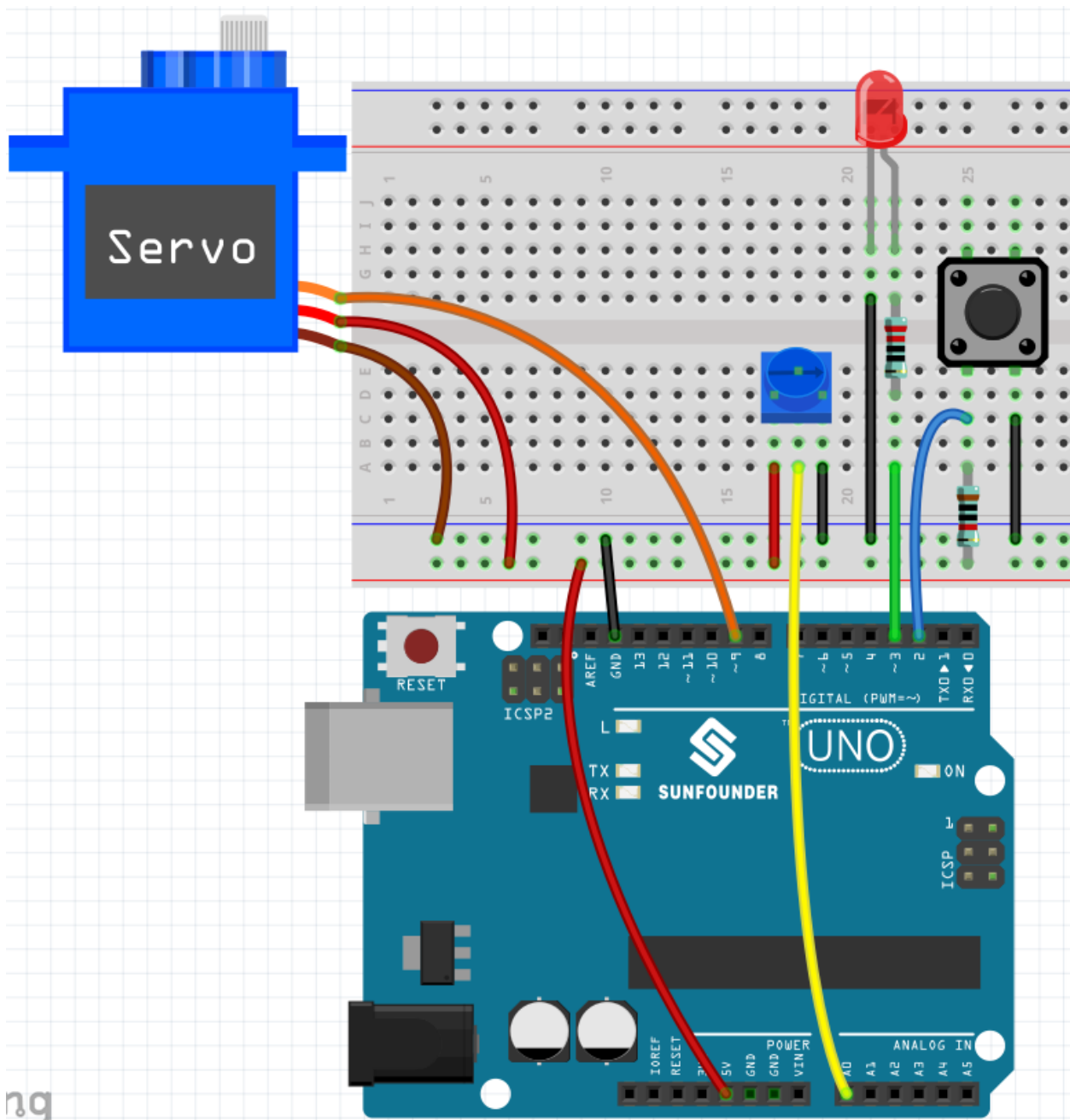
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Servomoteur</i>	
<i>Bouton</i>	
<i>Potentiomètre</i>	

### Schéma



### Câblage



## Code

### Note :

- Ouvrez le fichier 5.15.eeprom.ino situé dans le dossier 3in1-kit\basic\_project\5.15.eeprom.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléversez le code via l'Éditeur Web Arduino.

Pour utiliser ce circuit, vous appuyez simplement sur le bouton pour commencer l'enregistrement et entrer les informations souhaitées via un potentiomètre. Désormais, la carte répétera vos actions indéfiniment (et fera clignoter une LED à chaque itération) jusqu'à ce que vous appuyiez à nouveau sur le bouton pour enregistrer de nouvelles actions. Vous pouvez également varier la durée de l'enregistrement en changeant les valeurs de résolution et de recordTime.

## Comment ça fonctionne ?

1. Importez la bibliothèque `EEPROM.h` et initialisez la mémoire `EEPROM`.

```
...
#include <EEPROM.h> //used to store recorded values

...
float resolution = 1000; //MUST be less than EEPROM.length()
float recordTime = 5; //delay time
bool recording = false;
...
```

Veuillez noter que /DOIT être inférieur à `EEPROM.length()`, dans `setup()` il imprimera la mémoire de l'EEPROM de votre carte, qui devrait être 1024 pour la carte SunFounder R3. Si vous utilisez une autre carte, vous pouvez changer la valeur de la variable `resolution`.

2. Imprimez la mémoire `EEPROM` de votre carte.

```
void setup() {
    ...
    Serial.begin(9600);
    //Serial.println(EEPROM.length());
}
```

Pour trouver la taille de la mémoire `EEPROM` de votre carte, décommentez la ligne `Serial.println(EEPROM.read(i))`. Cela imprimera la taille de l'EEPROM dans le moniteur série, et vous pourrez changer la valeur de la variable `resolution` en conséquence.

3. Dès qu'une pression sur un bouton est détectée, l'enregistrement commence et les informations requises sont entrées via un potentiomètre. Maintenant, la carte répète votre action indéfiniment (et fait clignoter une LED à chaque répétition) jusqu'à ce que vous appuyiez à nouveau sur le bouton, enregistrant une nouvelle action.

```
void loop() {
    if (recording == true) { //record
        for (int i = 1; i <= resolution; i++) {
            digitalWrite(ledPin, HIGH); //light status led
            int val = map(analogRead(A0), 0, 1023, 0, 180);
            EEPROM.write(i, val);
            //Serial.println(EEPROM.read(i));
            myServo.write(val);
            delay(recordTime);
        }
        digitalWrite(ledPin, LOW); //turn off status led
        delay(1000); //give time for person
        recording = false;
    }
    else {
        for (int i = 1; i <= resolution; i++) { //playback
            if (digitalRead(buttonPin) == 0) { //stop playback and record new
↪ values
                recording = true;
                break;
            }
            int readval = EEPROM.read(i);
            myServo.write(readval);
        }
    }
}
```

(suite sur la page suivante)

```

        //Serial.println(readval);
        delay(recordTime);
    }
    digitalWrite(ledPin, HIGH); //show a new repeat
    delay(100);
    digitalWrite(ledPin, LOW);
}
}

```

- Rendez la variable `recording` vraie lorsque le bouton est pressé.
- Lorsque la variable `recording` est vraie, commencez à enregistrer l'action dans la page de mémoire.
- Lisez la valeur du potentiomètre et mappez-la de 0 à 180 pour la stocker dans l'EEPROM et contrôler la rotation du servo.
- La LED s'allume au début de l'enregistrement et s'éteint à la fin.
- Répétez l'action enregistrée avec un clignotement rapide de la LED pour vous rappeler une nouvelle répétition.

#### 4. À propos de la bibliothèque EEPROM.

Voici certaines de ses fonctions.

- `write(address, value)` : Écrivez un octet dans l'EEPROM.
  - `address` : l'emplacement où écrire, à partir de 0 (int)
  - `value` : la valeur à écrire, de 0 à 255 (byte)
  - Une écriture EEPROM prend 3.3 ms pour s'achever. La mémoire EEPROM a une durée de vie spécifiée de 100,000 cycles d'écriture/effacement, donc vous devrez peut-être faire attention à la fréquence à laquelle vous écrivez dedans.
- `Read(address)` : Lit un octet de l'EEPROM. Les emplacements qui n'ont jamais été écrits ont une valeur de 255.
- `update(address, value)` : Écrivez un octet dans l'EEPROM. La valeur est écrite seulement si elle diffère de celle déjà enregistrée à la même adresse.
  - Une écriture EEPROM prend 3.3 ms pour s'achever. La mémoire EEPROM a une durée de vie spécifiée de 100,000 cycles d'écriture/effacement, donc l'utilisation de cette fonction au lieu de `write()` peut économiser des cycles si les données écrites ne changent pas souvent.
- `EEPROM.put(address, data)` : Écrivez n'importe quel type de données ou objet dans l'EEPROM.
  - `address` : l'emplacement à lire, à partir de 0 (int).
  - `data` : les données à lire, peuvent être un type primitif (par exemple, float) ou une structure personnalisée.
  - Cette fonction utilise `EEPROM.update()` pour effectuer l'écriture, donc elle ne réécrit pas la valeur si elle n'a pas changé.
- `EEPROM.get(address, data)` : Lisez n'importe quel type de données ou objet de l'EEPROM.
  - `address` : l'emplacement à lire, à partir de 0 (int).
  - `data` : les données à lire, peuvent être un type primitif (par exemple, float) ou une structure personnalisée.



## 5.6 6. Projet Amusant

Dans ce chapitre, vous trouverez quelques exemples illustrant la logique de base de l'interaction de la plupart des programmes avec la réalité. Cela vous aidera à vous familiariser avec la programmation Arduino. Lorsque vous aurez une idée créative en tête, la programmation ne sera plus un défi pour vous.

### 5.6.1 6.1 Tableau Sensible à la Lumière

Un photo-résistor ou une cellule photoélectrique est une résistance variable contrôlée par la lumière. La résistance d'un photo-résistor diminue avec l'augmentation de l'intensité lumineuse incidente ; en d'autres termes, il présente une photoconductivité. Un photo-résistor peut être appliqué dans des circuits détecteurs sensibles à la lumière, et dans des circuits de commutation activés par la lumière et l'obscurité.

La résistance d'un photo-résistor change avec l'intensité lumineuse incidente. Si l'intensité lumineuse augmente, la résistance diminue ; si elle diminue, la résistance augmente. Dans cette expérience, nous utiliserons huit LEDs pour montrer l'intensité lumineuse. Plus l'intensité lumineuse est élevée, plus de LEDs s'allumeront. Lorsque l'intensité lumineuse est suffisamment élevée, toutes les LEDs seront allumées. Lorsqu'il n'y a pas de lumière, toutes les LEDs s'éteindront.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

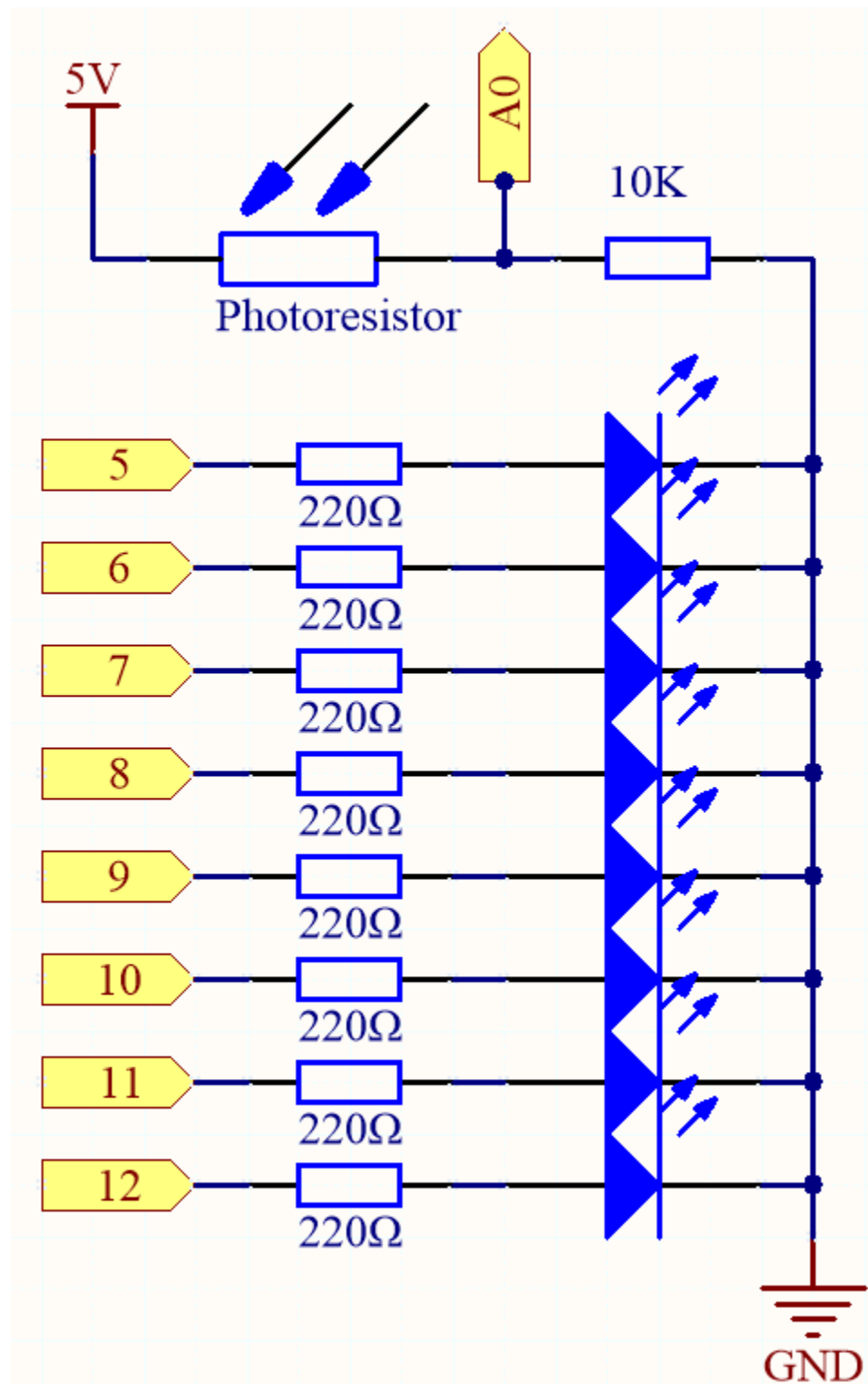
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

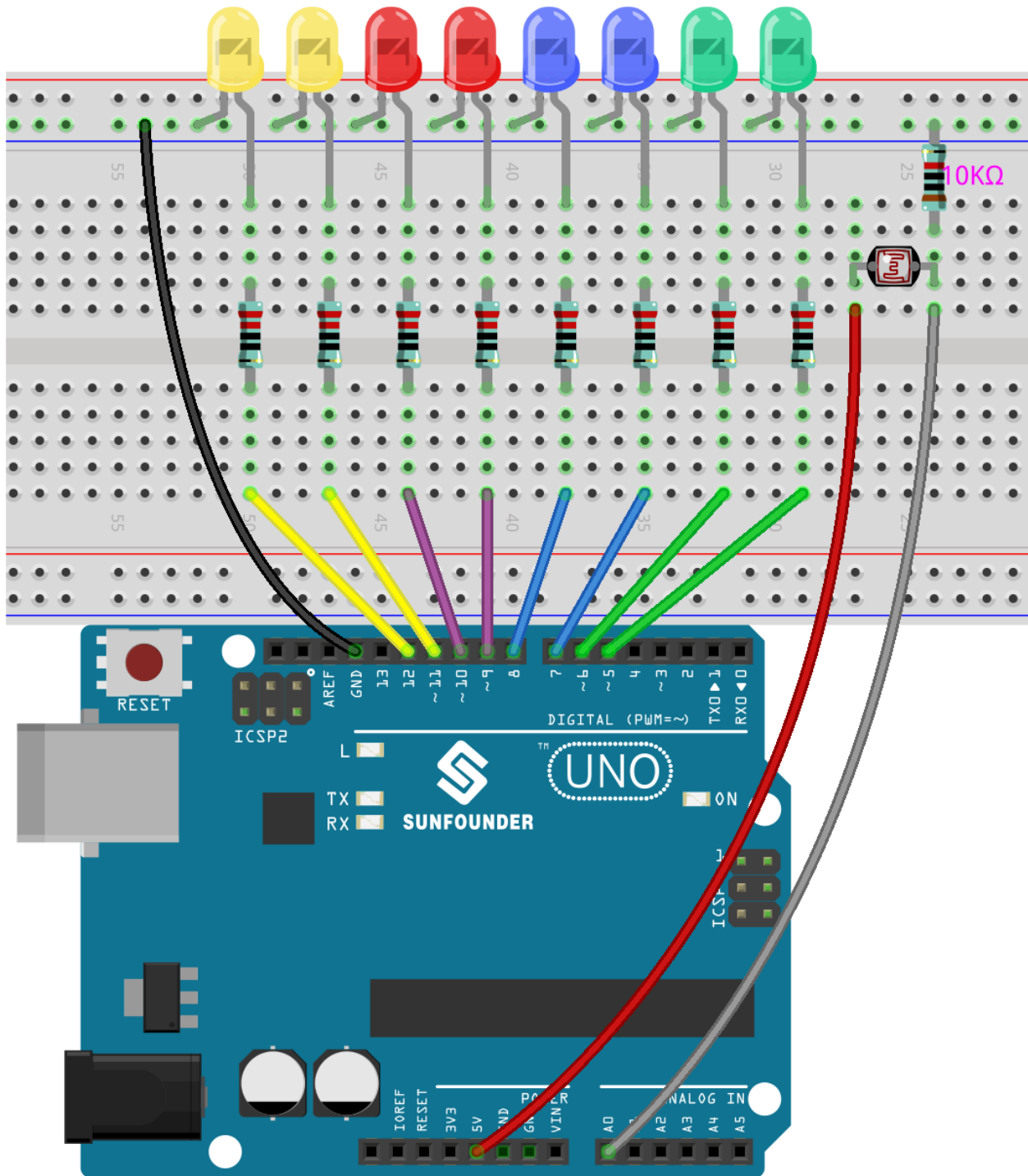
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Photorésistance</i>	

#### Schéma



Câblage



## Code

### Note :

- Ouvrez le fichier 6.1.light\_control\_led.ino situé dans le dossier 3in1-kit\basic\_project\6.1.light\_control\_led.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléversez le code via l'Éditeur Web Arduino.

Maintenant, éclairez le photo-résistor, et vous verrez plusieurs LEDs s'allumer. Éclairez davantage et vous verrez plus de LEDs s'allumer. Quand vous le placez dans un environnement sombre, toutes les LEDs s'éteindront.

### Comment ça fonctionne ?

```
void loop()
{
    sensorValue = analogRead(photocellPin); //read the value of A0
    ledLevel = map(sensorValue, 300, 1023, 0, NbrLEDs); // map to the number of LEDs
    for (int led = 0; led < NbrLEDs; led++)//
    {
        if (led < ledLevel ) //When led is smaller than ledLevel, run the following code.
        {
            digitalWrite(ledPins[led], HIGH); // turn on pins less than the level
        }
        else
        {
            digitalWrite(ledPins[led],LOW); // turn off pins higher than
        }
    }
}
```

En utilisant la fonction map(), vous pouvez mapper la valeur du photo-résistor aux 8 LEDs, par exemple, si sensorValue est 560, alors ledLevel est 4, donc à ce moment, ledPins[0] à ledPins[4] devraient être allumés, et ledPins[5] à ledPins[7] devraient être éteints.

## 5.6.2 6.2 Dé de Numérique

Ici, nous utilisons un bouton, un afficheur 7 segments et un 74hc595 pour créer un dé électronique. Chaque fois que le bouton est pressé, un nombre aléatoire compris entre 1 et 6 est généré et affiché sur l'afficheur 7 segments.

### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

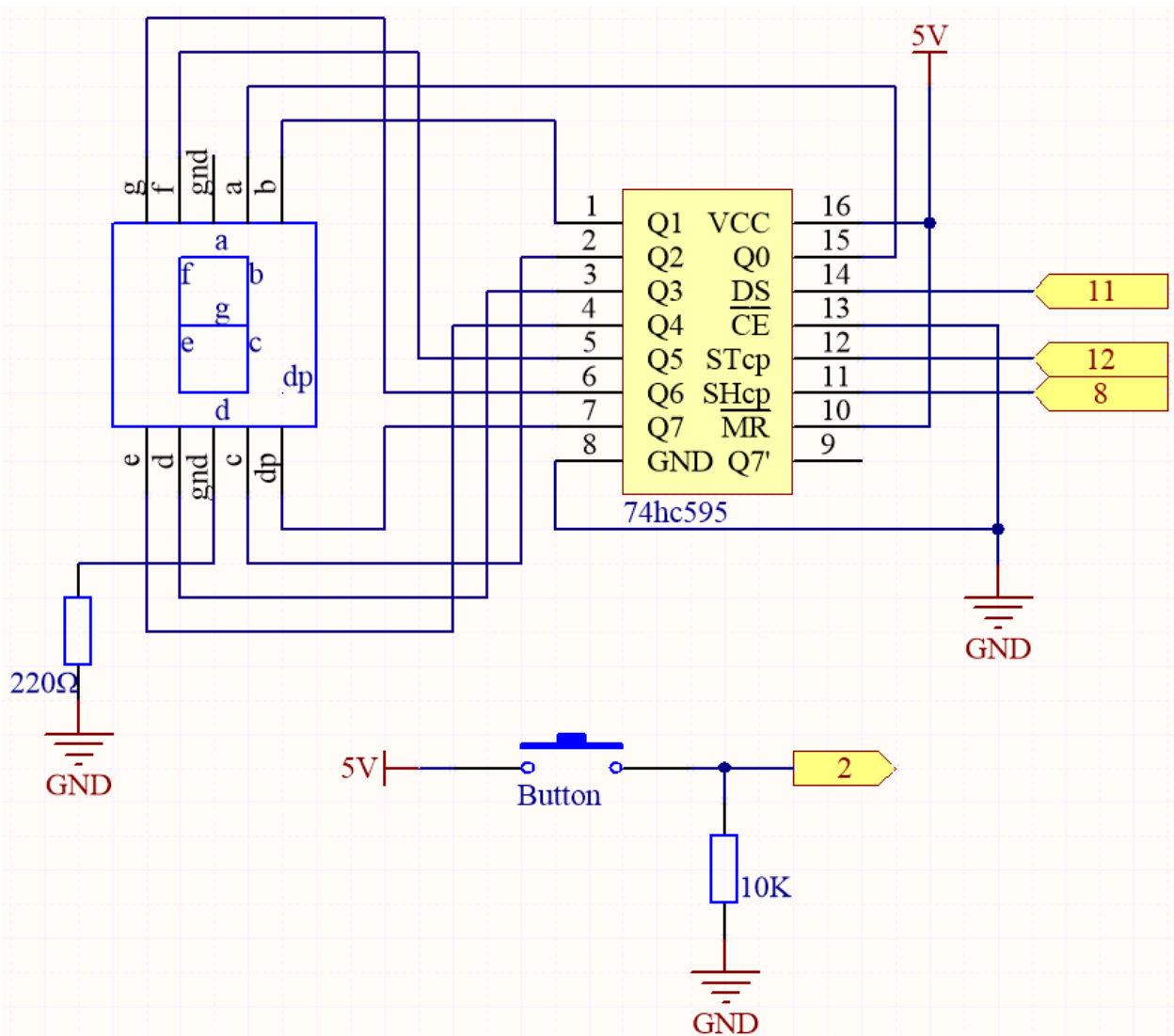
Il est certainement pratique d'acheter un kit complet, voici le lien :

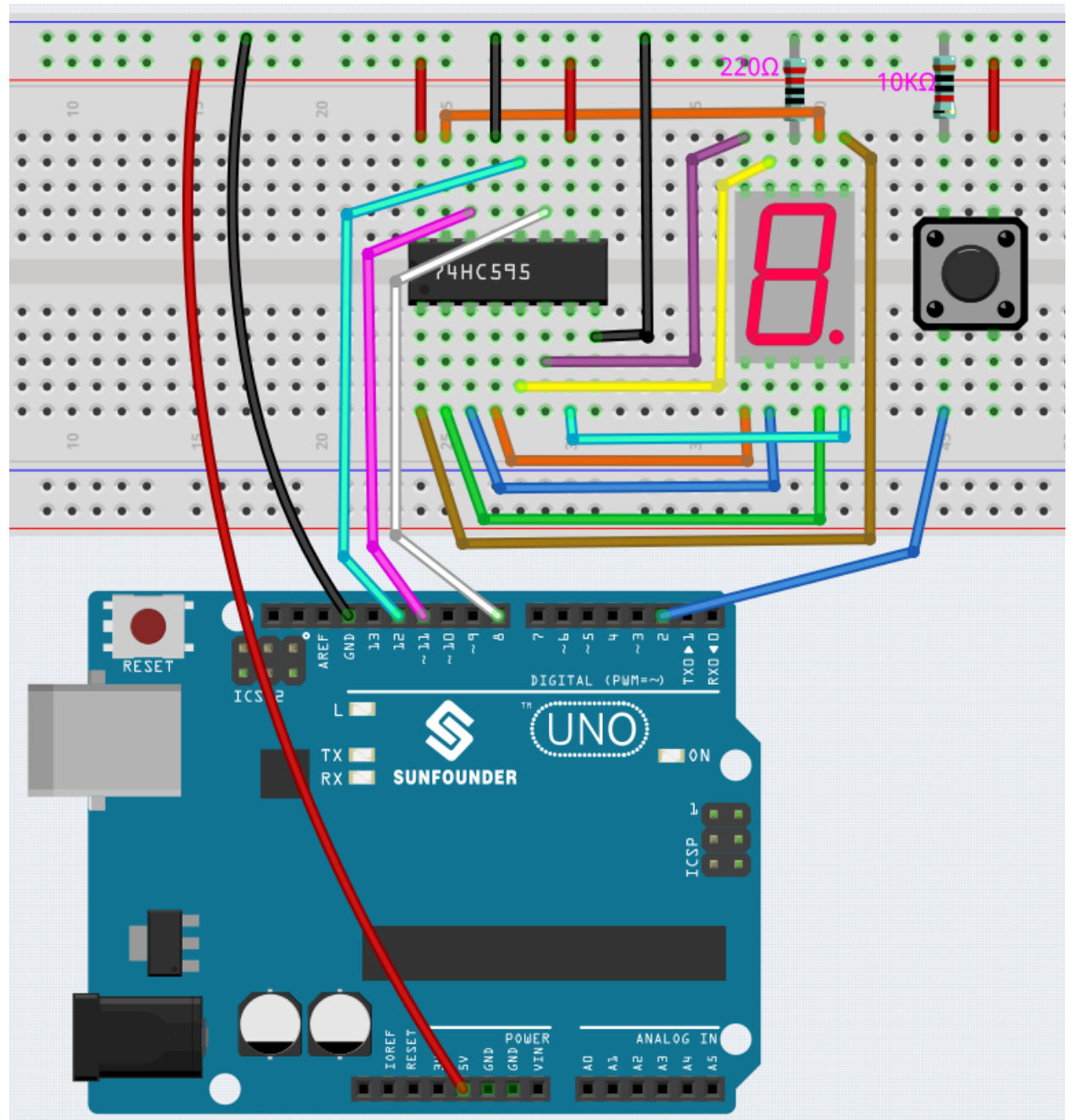
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Bouton</i>	
<i>74HC595</i>	
<i>Affichage 7 segments</i>	

## Schéma





vous appuyerez sur le bouton, il affichera un nombre aléatoire et arrêtera de défiler. L’affichage défilant recommence lorsque vous appuyez à nouveau sur le bouton.

### Comment ça fonctionne ?

Ce projet est basé sur *5.10 ShiftOut(Affichage à segments)* avec un bouton pour démarrer/pauser l’affichage défilant sur l’afficheur 7 segments.

1. Initialisez chaque broche et lisez la valeur du bouton.

```
void setup ()
{
    ...
    attachInterrupt(digitalPinToInterrupt(buttonPin), rollDice, FALLING);
}
```

- L’interruption est utilisée ici pour lire l’état du bouton. La valeur par défaut de `buttonPin` est basse, qui change de bas à haut lorsque le bouton est pressé.
- `rollDice` représente la fonction à appeler lorsque l’interruption est déclenchée, elle est utilisée pour basculer la valeur de la variable `state`.
- `FALLING` signifie que l’interruption est déclenchée lorsque le `buttonPin` passe de bas à haut.

2. Lorsque la variable `state` est à 0, la fonction `showNumber()` est appelée pour faire afficher aléatoirement un nombre entre 1 et 7 sur l’afficheur 7 segments.

```
void loop()
{
    if (state == 0) {
        showNumber((int)random(1, 7));
        delay(50);
    }
}
```

3. À propos de la fonction `rollDice()`.

```
void rollDice() {
    state = !state;
}
```

Lorsque cette fonction est appelée, elle bascule la valeur de `state`, par exemple 1 la dernière fois et 0 cette fois.

4. À propos de la fonction `showNumber()`.

```
void showNumber(int num) {
    digitalWrite(STcp, LOW); //ground ST_CP and hold low for as long as you
    ↪are transmitting
    shiftOut(DS, SHcp, MSBFIRST, dataArray[num]);
    //return the latch pin high to signal chip that it
    //no longer needs to listen for information
    digitalWrite(STcp, HIGH); //pull the ST_CPST_CP to save the data
}
```

Ceci est le code à l’intérieur de `loop()` dans le projet *5.10 ShiftOut(Affichage à segments)* intégré dans la fonction `showNumber()`.

### 5.6.3 6.3 Alarme de Haute Température

Ensuite, nous allons fabriquer un dispositif d'alarme de haute température en utilisant un thermistor, un bouton-poussoir, un potentiomètre et un LCD. Le LCD1602 affiche la température détectée par le thermistor et la valeur seuil de haute température, qui peut être ajustée à l'aide d'un potentiomètre. La valeur seuil est stockée dans l'EEPROM en même temps, donc si la température actuelle dépasse la valeur seuil, le buzzer sonnera.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

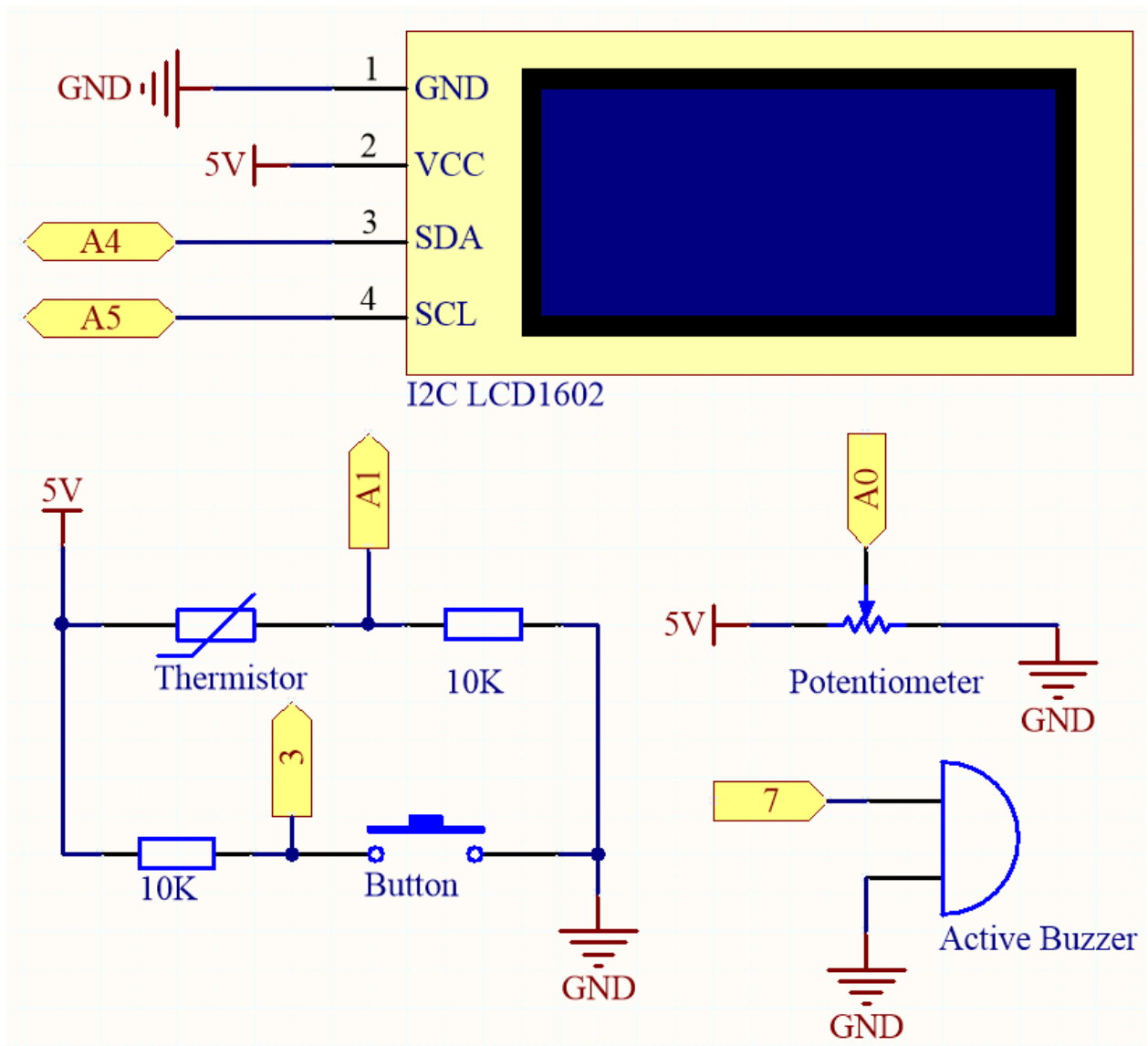
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

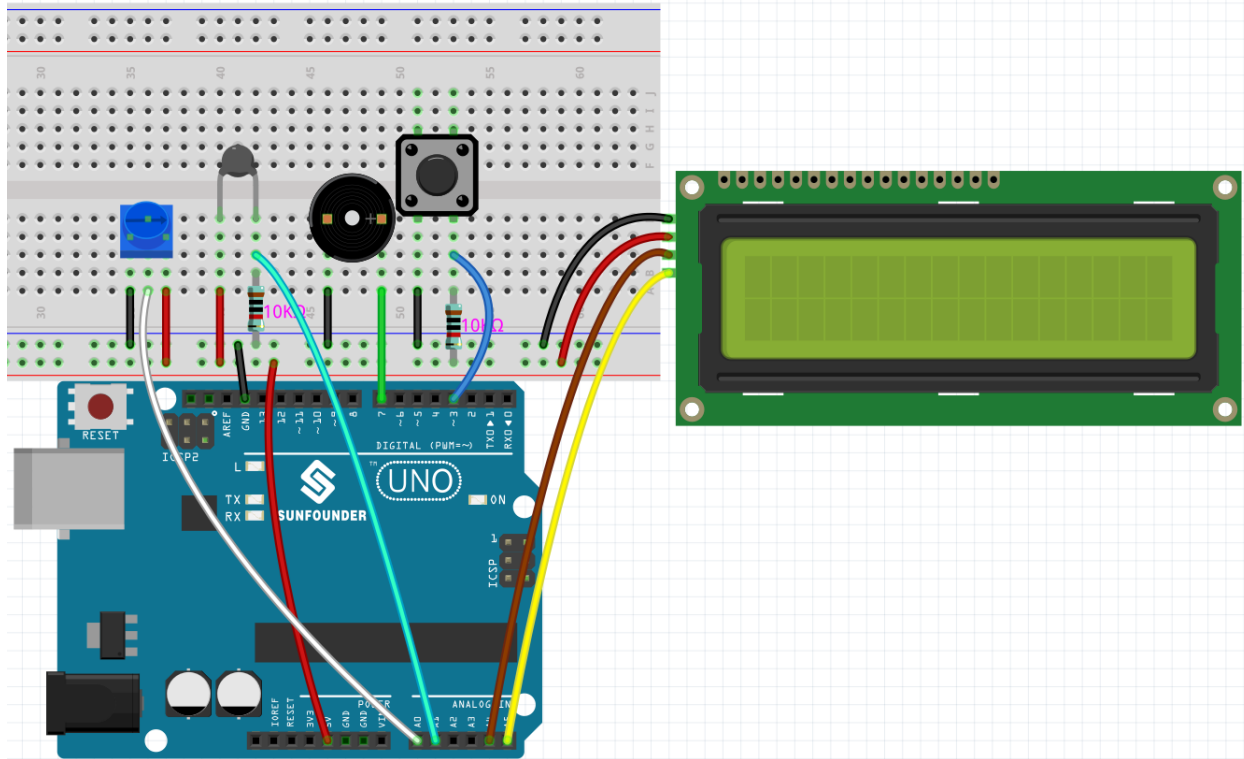
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Buzzer</i>	-
<i>Bouton</i>	
<i>I2C LCD1602</i>	
<i>Thermistance</i>	
<i>Potentiomètre</i>	

#### Schéma





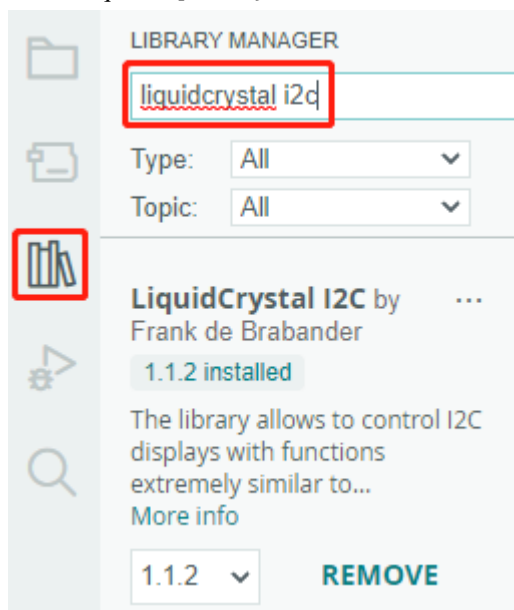
Câblage



## Code

### Note :

- Vous pouvez ouvrir le fichier 6.3.high\_tem\_alarm.ino sous le chemin de 3in1-kit\basic\_project\6.3.high\_tem\_alarm directement.
- Ou copiez ce code dans Arduino IDE .
- La bibliothèque LiquidCrystal I2C est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après le téléchargement réussi du code, le LCD1602 affiche la température détectée par le thermistor et la valeur seuil

de haute température, qui peut être ajustée à l'aide d'un potentiomètre. La valeur seuil est stockée dans l'EEPROM en même temps, donc si la température actuelle dépasse la valeur seuil, le buzzer sonnera.

**Note :** Si le code et le câblage sont corrects, mais que le LCD n'affiche toujours pas de contenu, vous pouvez tourner le potentiomètre à l'arrière.

### Comment ça fonctionne ?

1. Initialisez le bouton, le buzzer et le LCD1602 I2C, et lisez les valeurs de l'EEPROM. Une interruption est également utilisée ici pour lire l'état du bouton.

```
void setup()
{
    pinMode(buzzerPin, OUTPUT);
    pinMode(buttonPin, INPUT);
    lcd.init();
    lcd.backlight();
    upperTem = EEPROM.read(0);
    delay(1000);
    attachInterrupt(digitalPinToInterrupt(buttonPin), buttonState, FALLING);
}
```

- L'interruption est utilisée ici pour lire l'état du bouton. Lorsque le bouton est pressé, buttonPin passe de bas à haut.
- La fonction buttonState est appelée lorsque l'interruption se déclenche, et elle bascule la valeur de la variable state.
- FALLING signifie que l'interruption se produit lorsque buttonPin passe de bas à haut.

2. Pour définir le seuil de haute température, la fonction upperTemSetting() est appelée lorsque state est à 1 (state bascule entre 0 et 1 avec la pression du bouton) dans le programme principal, sinon monitoringTemp() est appelée pour afficher la température actuelle et le seuil défini.

```
void loop()
{
    if (state == 1)
    {
        upperTemSetting();
    }
    else {
        monitoringTemp();
    }
}
```

3. À propos de la fonction upperTemSetting().

```
void upperTemSetting()
{
    int setTem = 0;

    lcd.clear();
    lcd.setCursor(0, 0);
    lcd.print("Adjusting...");
    lcd.setCursor(0, 1);
    lcd.print("Upper Tem: ");
}
```

(suite sur la page suivante)

(suite de la page précédente)

```

while (1) {
    lcd.setCursor(11, 1);
    setTem = map(analogRead(potPin), 0, 1023, 0, 100);
    lcd.print(setTem);
    if (state == 0)
    {
        EEPROM.write(0, setTem);
        upperTem = setTem;
        lcd.clear();
        return;
    }
}
}

```

— Un seuil peut être défini avec cette fonction. Lorsque vous entrez dans cette fonction, le LCD1602 affiche la valeur seuil actuelle, qui peut être modifiée à l'aide du potentiomètre. Cette valeur seuil sera stockée dans l'EEPROM et quittée lorsque le bouton sera pressé à nouveau.

#### 4. À propos de la fonction monitoringTemp().

```

void monitoringTemp()
{
    long a = analogRead(temPin);
    float tempC = beta / (log((1025.0 * 10 / a - 10) / 10) + beta / 298.0) -
    ↪ 273.0;
    float tempF = 1.8 * tempC + 32.0;
    lcd.setCursor(0, 0);
    lcd.print("Temp: ");
    lcd.print(tempC);
    lcd.print(char(223));
    lcd.print("C  ");
    lcd.setCursor(0, 1);
    lcd.print("Upper: ");
    lcd.print(upperTem);
    lcd.print(char(223));
    lcd.print("C  ");
    delay(300);
    if (tempC >= upperTem)
    {
        digitalWrite(buzzerPin, HIGH);
        delay(50);
        digitalWrite(buzzerPin, LOW);
        delay(10);
    }
    else
    {
        digitalWrite(buzzerPin, LOW);
    }
}

```

- En utilisant cette fonction, vous pouvez afficher la température et régler une alarme.
- La valeur du thermistor est lue puis convertie en température Celsius par la formule et affichée sur le LCD1602.
- Le seuil défini est également affiché sur le LCD.
- Si la température actuelle est supérieure au seuil, le buzzer sonnera une alarme.

### 5.6.4 6.4 Aide au Stationnement

Avec le développement de la science et de la technologie, de nombreux produits de haute technologie ont été installés dans les voitures, parmi lesquels le système d'aide au stationnement en marche arrière est l'un d'eux. Ici, nous utilisons un module ultrasonique, un LCD, des LED et un buzzer pour réaliser un système simple d'aide au stationnement ultrasonique.

#### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

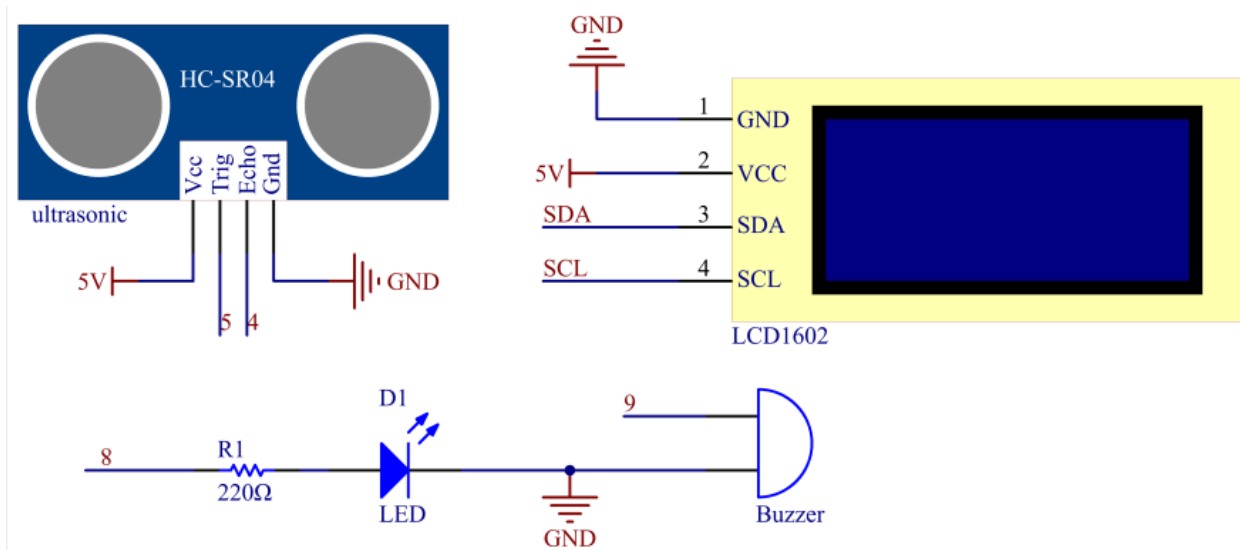
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

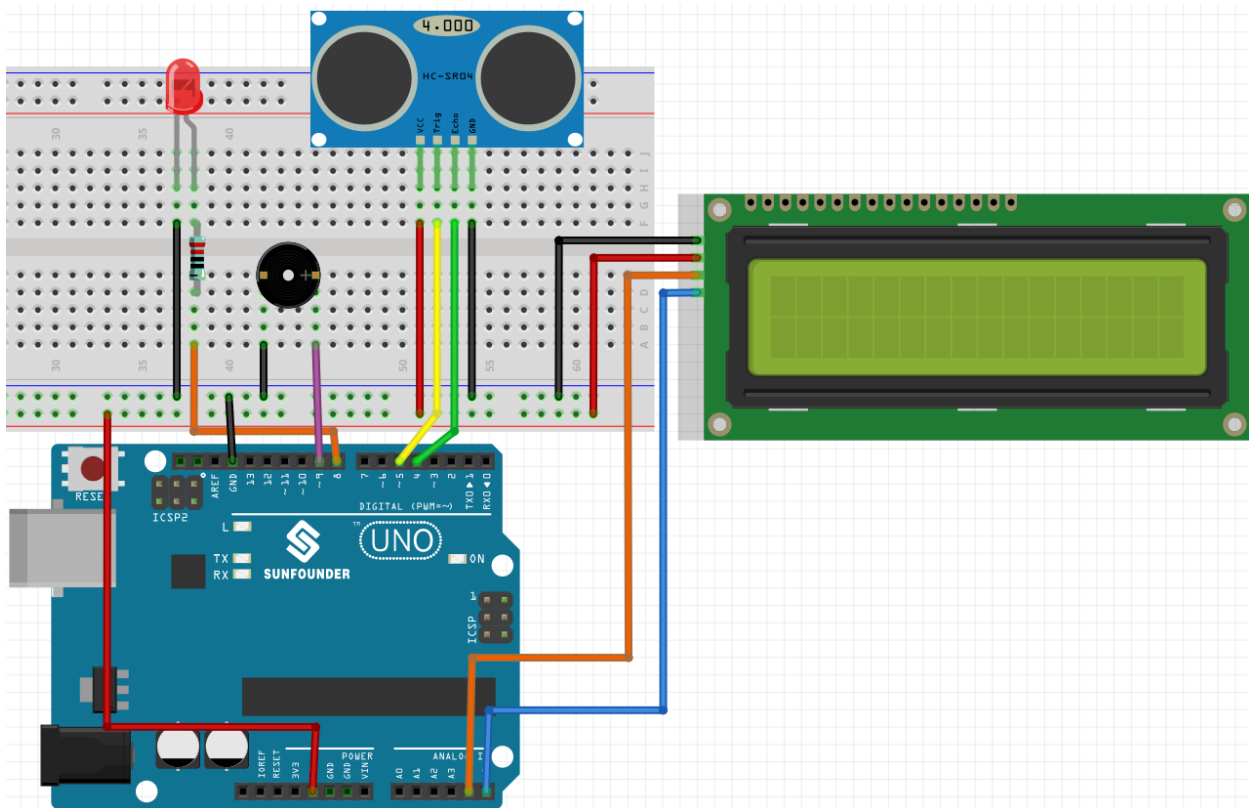
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Buzzer</i>	
<i>I2C LCD1602</i>	
<i>Module Ultrasonique</i>	

#### Schéma



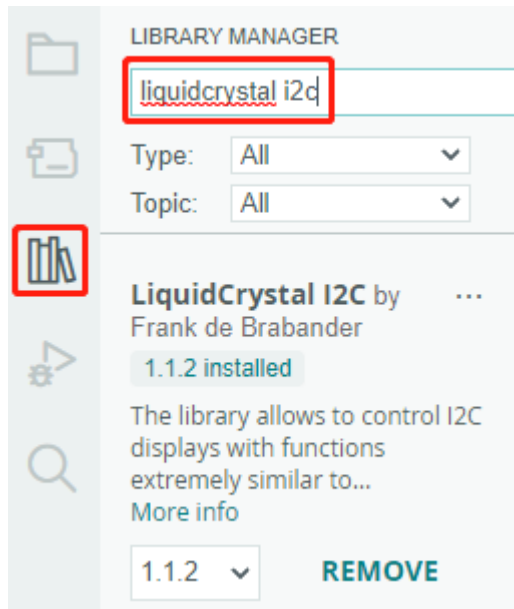
### Câblage



### Code

#### Note :

- Vous pouvez ouvrir le fichier 6.4\_reversingAid.ino sous le chemin de 3in1-kit\basic\_project\6.4\_reversingAid directement.
- Ou copiez ce code dans Arduino IDE .
- La bibliothèque LiquidCrystal I2C est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après le téléchargement réussi du code, la distance détectée actuelle sera affichée sur le LCD. Ensuite, le buzzer changera la fréquence de son en fonction des différentes distances.

**Note :** Si le code et le câblage sont corrects, mais que le LCD n'affiche toujours pas de contenu, vous pouvez tourner le potentiomètre à l'arrière.

### Comment ça fonctionne ?

Ce code nous aide à créer un dispositif simple de mesure de distance capable de mesurer la distance entre des objets et de fournir un retour via un affichage LCD et un buzzer.

La fonction `loop()` contient la logique principale du programme et s'exécute en continu. Examinons de plus près la fonction `loop()`.

1. Boucle pour lire la distance et mettre à jour les paramètres

Dans le `loop`, le code lit d'abord la distance mesurée par le module ultrasonique et met à jour le paramètre d'intervalle en fonction de la distance.

```
// Update the distance
distance = readDistance();

// Update intervals based on distance
if (distance <= 10) {
  intervals = 300;
} else if (distance <= 20) {
  intervals = 500;
} else if (distance <= 50) {
  intervals = 1000;
} else {
  intervals = 2000;
}
```

2. Vérifier s'il est temps de biper

Le code calcule la différence entre l'heure actuelle et l'heure du bip précédent, et si la différence est supérieure ou égale au temps d'intervalle, il déclenche le buzzer et met à jour l'heure du bip précédent.

```
unsigned long currentMillis = millis();
if (currentMillis - previousMillis >= intervals) {
    Serial.println("Beeping!");
    beep();
    previousMillis = currentMillis;
}
```

### 3. Mettre à jour l'affichage LCD

Le code efface l'affichage LCD puis affiche « Dis : » et la distance actuelle en centimètres sur la première ligne.

```
lcd.clear();
lcd.setCursor(0, 0);
lcd.print("Dis: ");
lcd.print(distance);
lcd.print(" cm");

delay(1000);
```

## 5.6.5 6.5 Jeu de Réaction

Notre corps a de nombreux temps de réaction, tels que le temps de réaction auditif, visuel, tactile, etc.

Les temps de réaction ont de nombreux effets sur notre vie quotidienne, par exemple, des temps de réaction plus lents que la normale en conduisant peuvent entraîner de graves conséquences.

Dans ce projet, nous utilisons 3 boutons et 2 LED pour mesurer notre temps de réaction visuel.

Le moniteur série de l'Arduino affiche le message « en attente... » Après avoir appuyé sur le bouton Prêt, l'une des deux LED doit s'allumer aléatoirement après un intervalle de temps aléatoire. Il est important que la personne testée appuie sur le bouton correspondant le plus rapidement possible. L'Arduino enregistre la différence de temps entre le moment où la LED s'allume et lorsque la personne appuie sur le bouton correspondant, et imprime le temps de réponse mesuré sur le moniteur série de l'Arduino.

### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

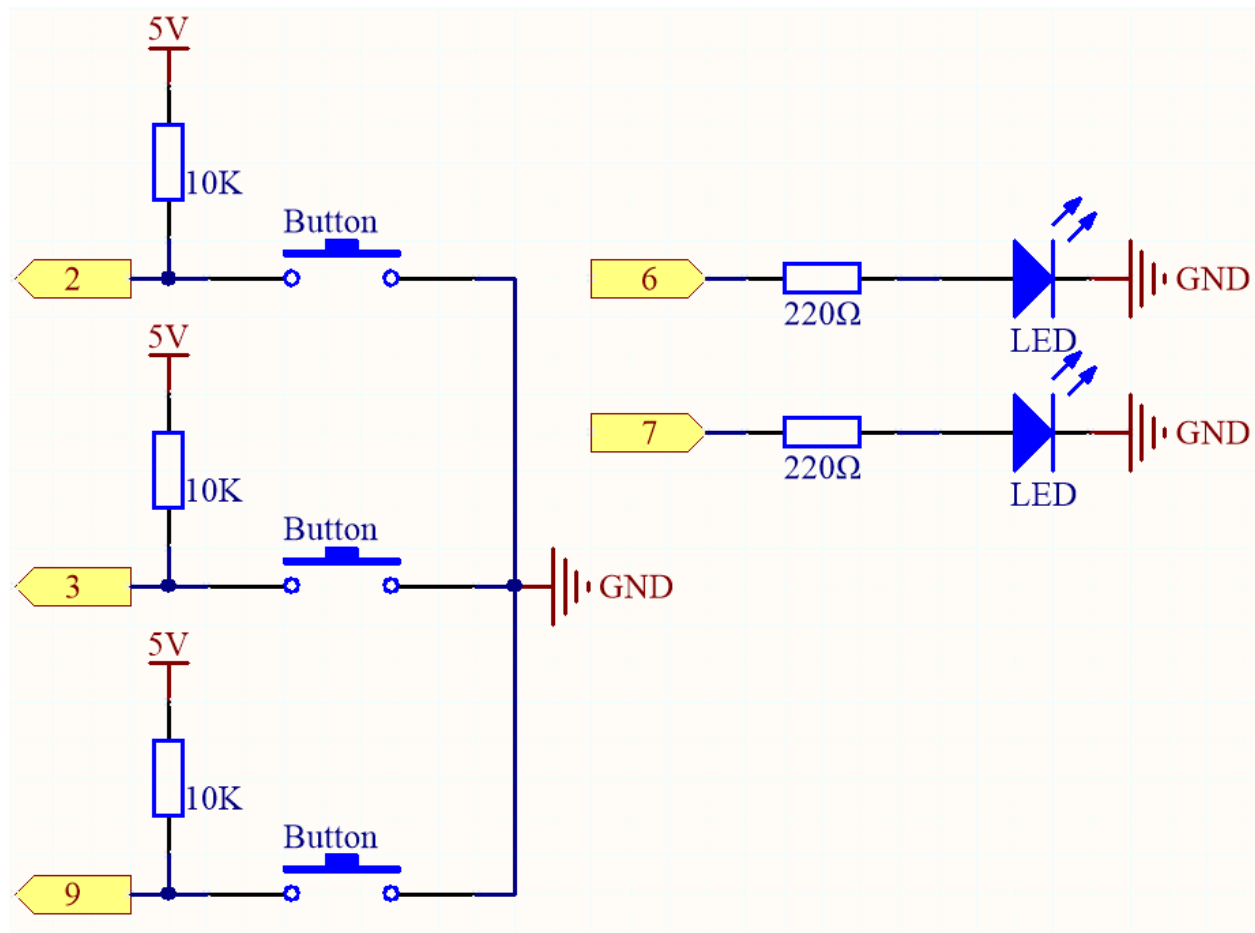
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

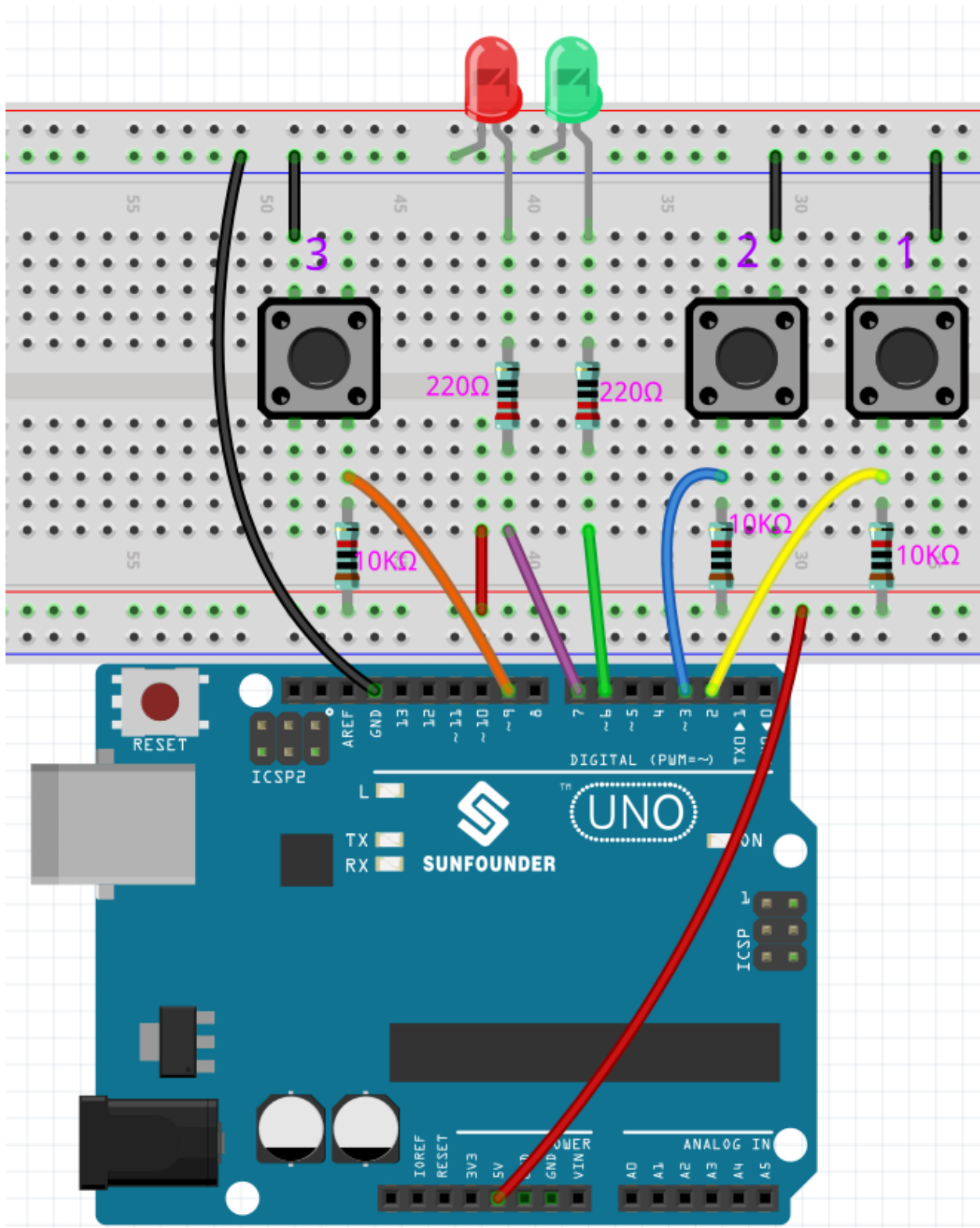


INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	
<i>Bouton</i>	

## Schéma



## Câblage



Code

Note :

- Vous pouvez ouvrir le fichier `6.5_reaction_time.ino` sous le chemin de `3in1-kit/basic_project/6.5_reversingAid` directement.
- Ou copiez ce code dans Arduino IDE.
- Assurez-vous d'avoir ajouté la bibliothèque `LiquidCrystal_I2C`, les tutoriels détaillés se réfèrent à [5.11 Installer des bibliothèques externes](#).

### Comment ça fonctionne ?

1. Initialisez les boutons et les LED, 2 interruptions sont utilisées ici pour lire l'état des boutons.

```
void setup()
{
    ...

    attachInterrupt(digitalPinToInterrupt(buttonPin1), pressed1, FALLING);
    attachInterrupt(digitalPinToInterrupt(buttonPin2), pressed2, FALLING);
    ...
}
```

2. Si le bouton `rstBtn` est pressé, le jeu recommence. À un moment aléatoire entre 2 et 5ms, faites s'allumer l'une des LED.

```
void loop()
{
    if (flag == -1 && digitalRead(rstBtn) == LOW) {
        digitalWrite(ledPin1, LOW);
        digitalWrite(ledPin2, LOW);
        Serial.println("Waiting...");
        int randomTime = random(2000, 5000);
        delay(randomTime);

        timer = millis();
        flag = randomTime % 2;
        Serial.println("Light!");

        if (flag == 0) {
            digitalWrite(ledPin1, HIGH);
        } else if (flag == 1) {
            digitalWrite(ledPin2, HIGH);
        }
    }
    delay(200);
}
```

- Lorsque `flag` est -1 et que le bouton `rstBtn` est pressé, utilisez la fonction `random()` pour générer un temps aléatoire de 2-5s.
  - Ce temps est ensuite utilisé pour contrôler l'allumage des LED.
  - Aussi, l'allumage de 2 LED est généré aléatoirement par `randomTime % 2` avec 0 et 1. Si `flag` est 0, alors LED1 est allumée ; si 1, alors LED2 est allumée.
3. À propos de la fonction `pressed1()`

```
void pressed1() {
    if (flag == -1) {
```

(suite sur la page suivante)

(suite de la page précédente)

```

    return;
}
if (flag == 0) {
    int currentTime = millis();
    Serial.print("Correct! You reaction time is : ");
    Serial.print(currentTime - timer);
    Serial.println(" ms");
} else if (flag == 1) {
    Serial.println("Wrong Click!");
}
flag = -1;
}

```

C'est la fonction qui sera déclenchée lorsque le bouton 1 est pressé. Lorsque le bouton 1 est pressé, si le flag est 0 à ce moment, le temps de réponse sera imprimé, sinon une erreur de pression sera indiquée.

#### 4. À propos de la fonction pressed2()

```

void pressed2() {
    if (flag == -1) {
        return;
    }
    if (flag == 1) {
        int currentTime = millis();
        Serial.print("Correct! You reaction time is : ");
        Serial.print(currentTime - timer);
        Serial.println(" ms");
    } else if (flag == 0) {
        Serial.println("Wrong Click!");
    }
    flag = -1;
}

```

C'est la fonction qui sera déclenchée lorsque le bouton 2 est pressé. Lorsque le bouton 2 est pressé, si le flag est 1 à ce moment, le temps de réponse sera imprimé, sinon une erreur de pression sera indiquée.

## 5.6.6 6.6 Devinez le Nombre

Devinez le Nombre est un jeu de société amusant où vous et vos amis prenez à tour de rôle l'initiative de saisir un nombre (0~99). La plage sera plus petite avec la saisie du nombre jusqu'à ce qu'un joueur réponde correctement à l'énigme. Ensuite, le joueur est battu et puni. Par exemple, si le nombre chanceux est 51, que les joueurs ne peuvent pas voir, et que le joueur 1 saisit 50, l'invite de la plage de nombres change pour 50~99; si le joueur 2 saisit 70, la plage de nombre peut être de 50~70; si le joueur 3 saisit 51, il ou elle est le malchanceux. Ici, nous utilisons une télécommande IR pour saisir les nombres et un LCD pour afficher les résultats.

### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

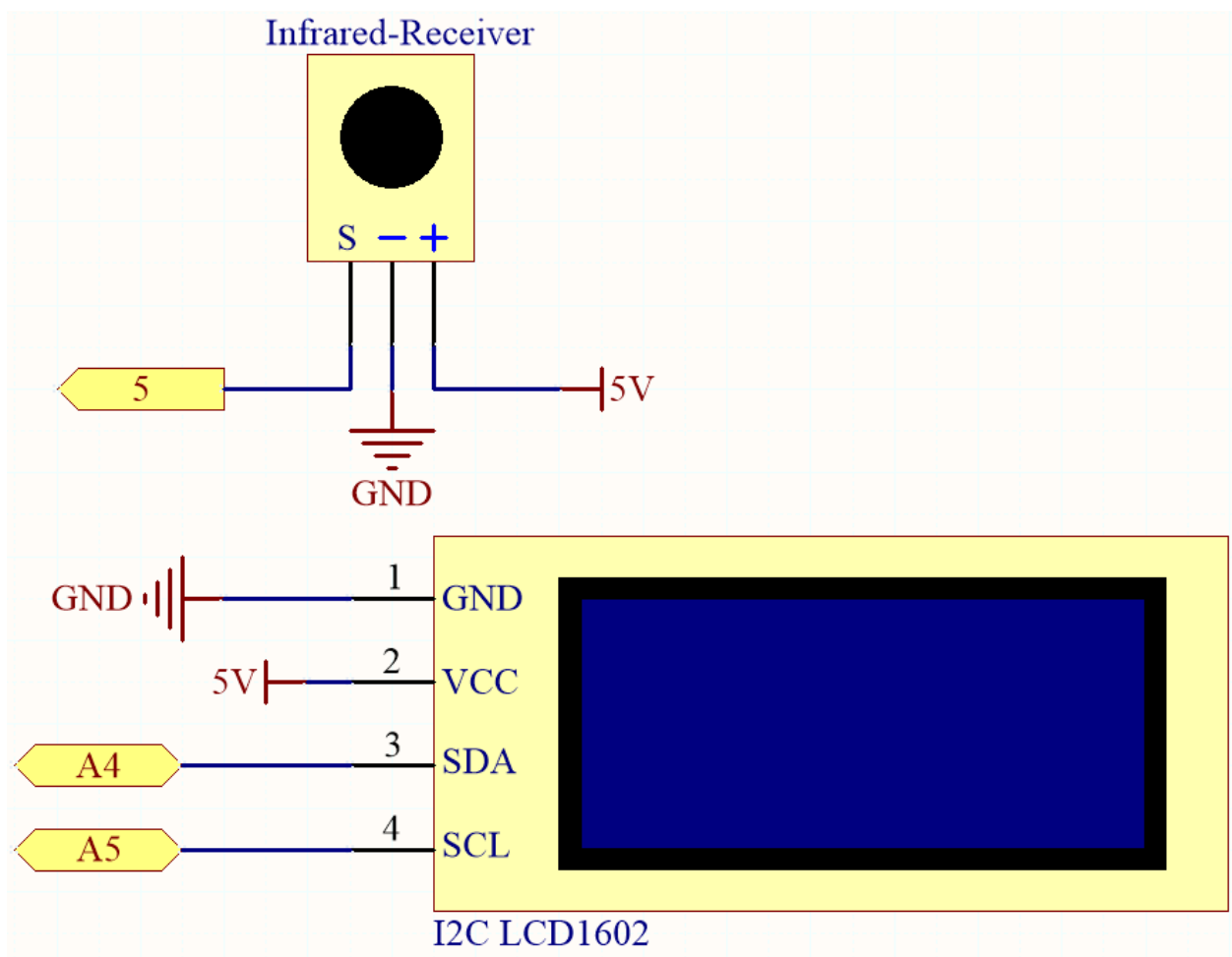
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

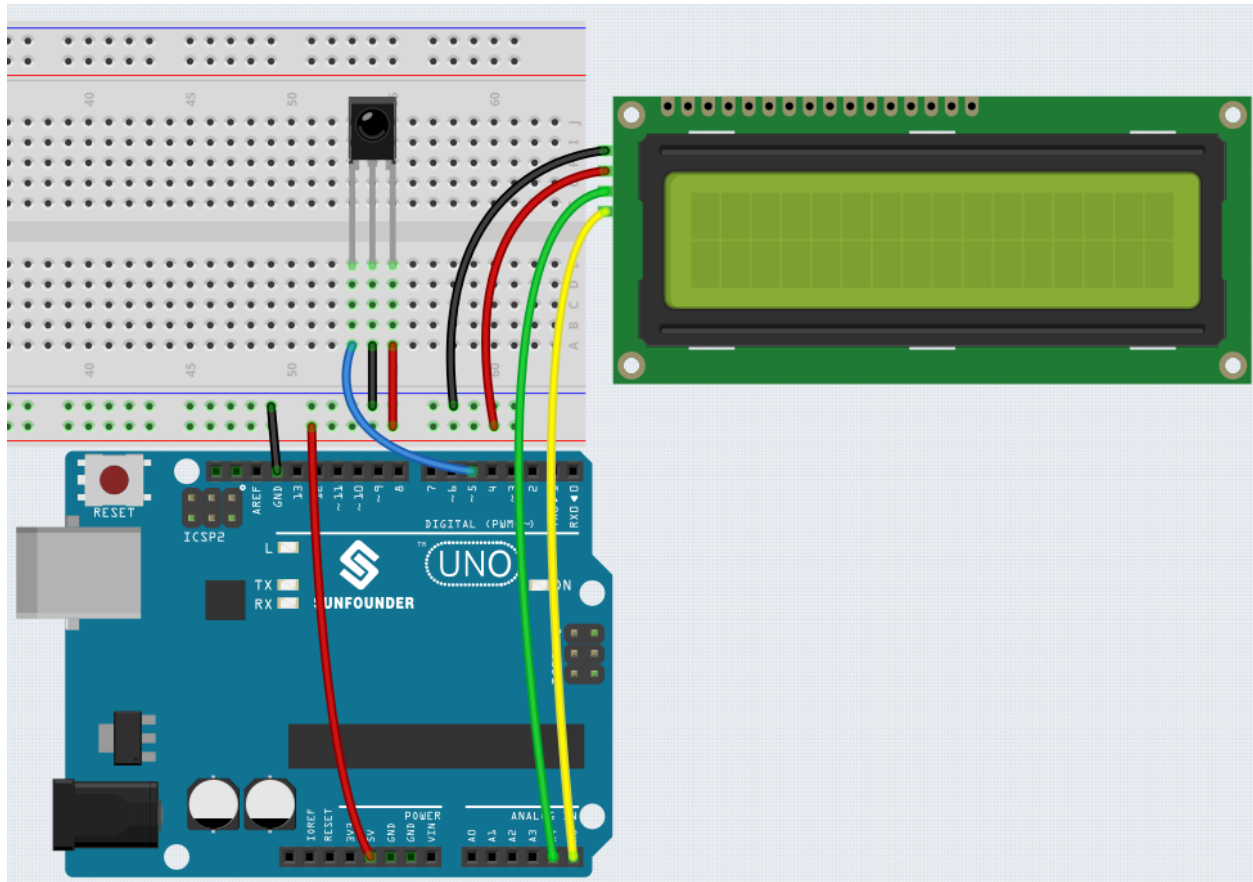
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>I2C LCD1602</i>	
<i>Récepteur IR</i>	-

### Schéma



### Câblage

Dans cet exemple, le câblage du LCD1602 et du module de réception infrarouge est le suivant.



## Code

### Note :

- Vous pouvez ouvrir le fichier `6.6.guess_number.ino` sous le chemin de `3in1-kit\basic_project\6.6.guess_number` directement.
- Ou copiez ce code dans Arduino IDE .
- Les bibliothèques `LiquidCrystal I2C` et `IRremote` sont utilisées ici, vous pouvez les installer depuis le **Library Manager**.

Après le téléchargement réussi du code, les caractères de bienvenue apparaîtront sur le LCD1602. Appuyez maintenant sur le nombre selon l'invite de plage sur l'écran, l'affichage deviendra de plus en plus petit à moins que vous ne devinez ce nombre chanceux.

**Note :** Si le code et le câblage sont corrects, mais que le LCD n'affiche toujours pas de contenu, vous pouvez tourner le potentiomètre à l'arrière pour augmenter le contraste.

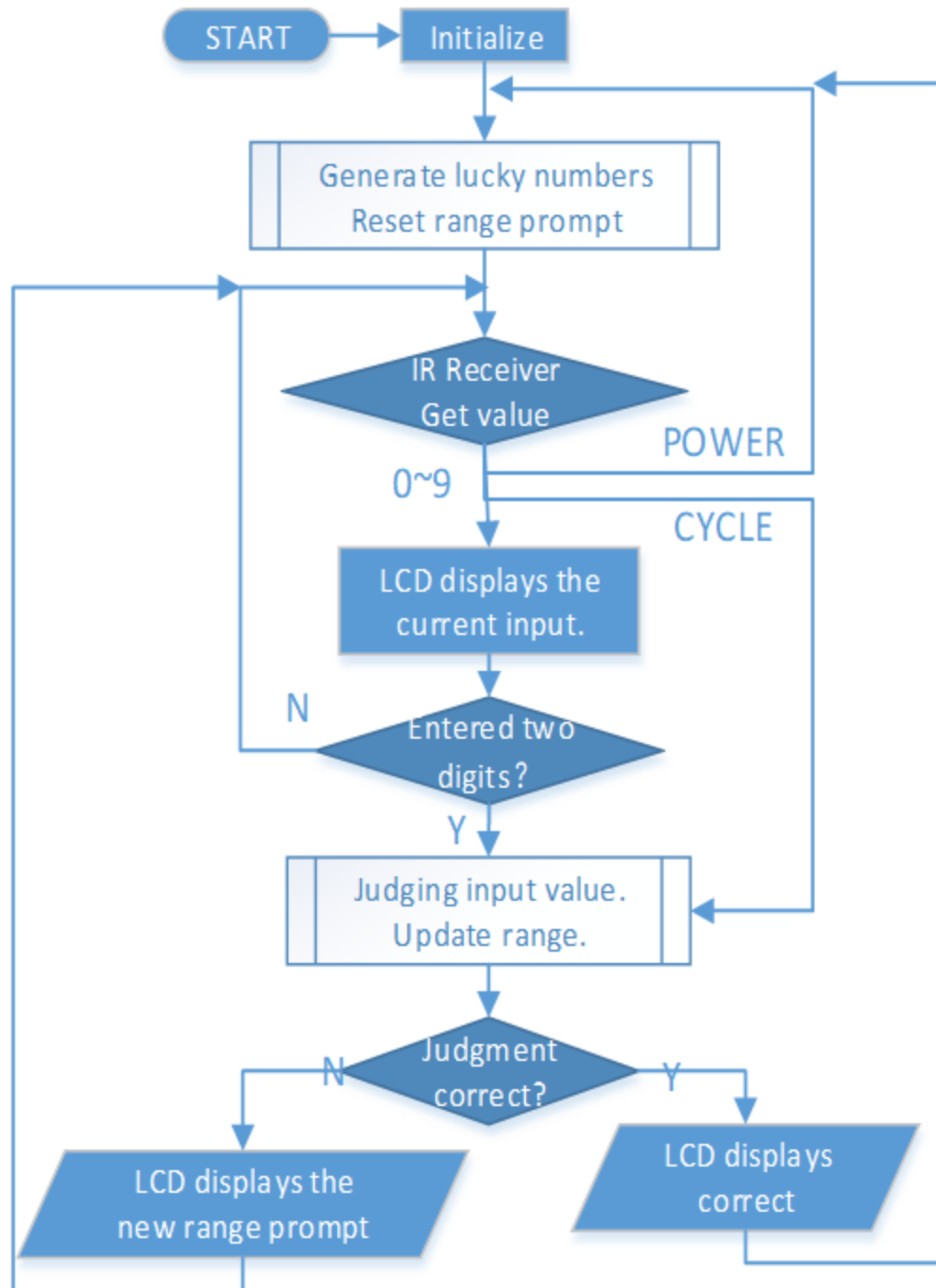
### Comment ça fonctionne ?

Pour rendre le jeu de devinettes de nombres vivant et amusant, nous devons réaliser les fonctions suivantes :

1. Le nombre chanceux sera affiché lorsque nous commençons et réinitialisons le jeu, et l'invite de plage de nombres est réinitialisée à 0 ~ 99.
2. Le LCD affichera le nombre saisi et l'invite de plage de nombres.
3. Après avoir saisi deux chiffres, un jugement de résultat apparaît automatiquement.

4. Si vous saisissez un seul chiffre, vous pouvez appuyer sur la touche CYCLE (la touche au centre de la télécommande) pour démarrer le jugement de résultat.
5. Si la réponse n'est pas devinée, la nouvelle invite de plage de nombres sera affichée (si le nombre chanceux est 51 et que vous entrez 50, l'invite de plage de nombres changera pour 50~99).
6. Le jeu est automatiquement réinitialisé après avoir deviné le nombre chanceux, pour que le joueur puisse jouer une nouvelle manche.
7. Le jeu peut être réinitialisé en appuyant directement sur le bouton POWER (le bouton dans le coin supérieur gauche).

En conclusion, le flux de travail du projet est illustré dans le diagramme de flux.







Je crois que vous avez vu beaucoup de différentes voitures robots intelligentes, leurs fonctions de base sont similaires : mouvement de base, évitement d'obstacles, suivi de ligne, suivi et contrôle par télécommande, etc.

Ici, nous utilisons la structure la plus simple pour construire une voiture robot intelligente, qui peut également réaliser toutes les fonctions ci-dessus. De plus, vous pouvez la contrôler avec votre téléphone portable, veuillez vous référer à [8. Voiture IoT](#) pour le tutoriel.

### Instructions de montage

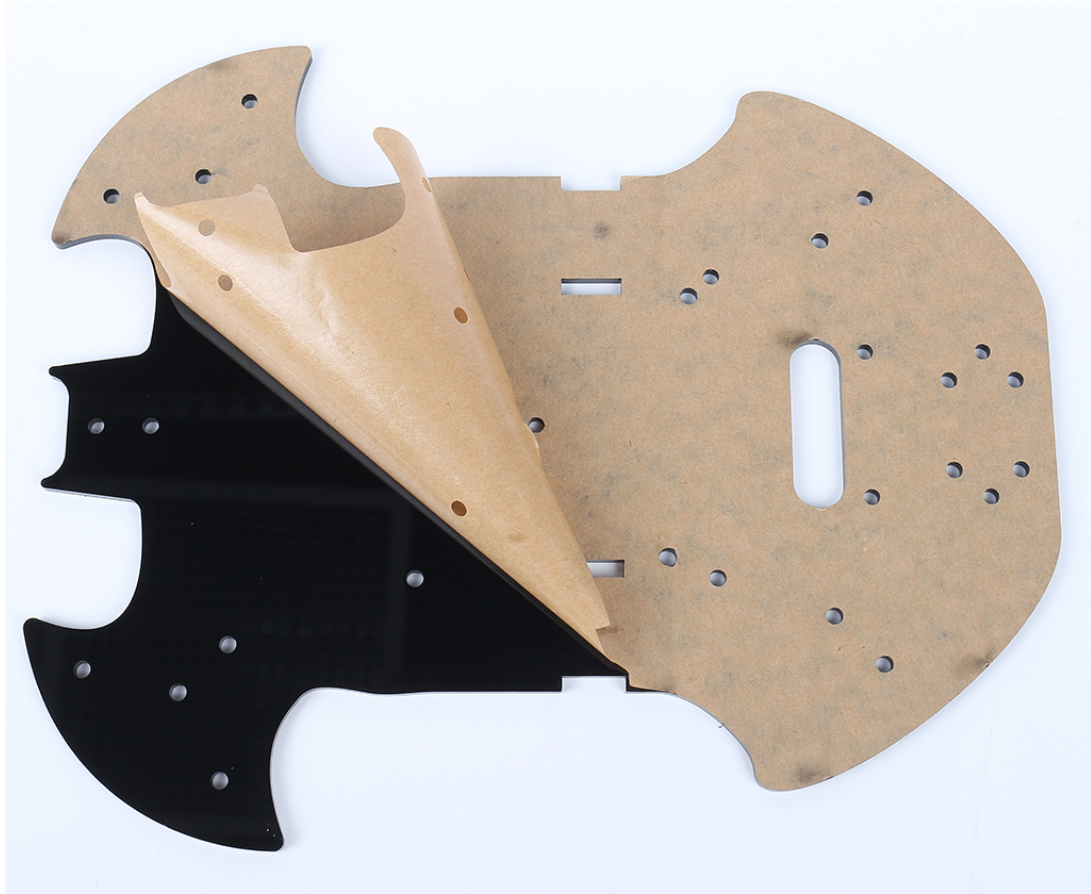
## 6.1 Assembler la Voiture

### Vidéo

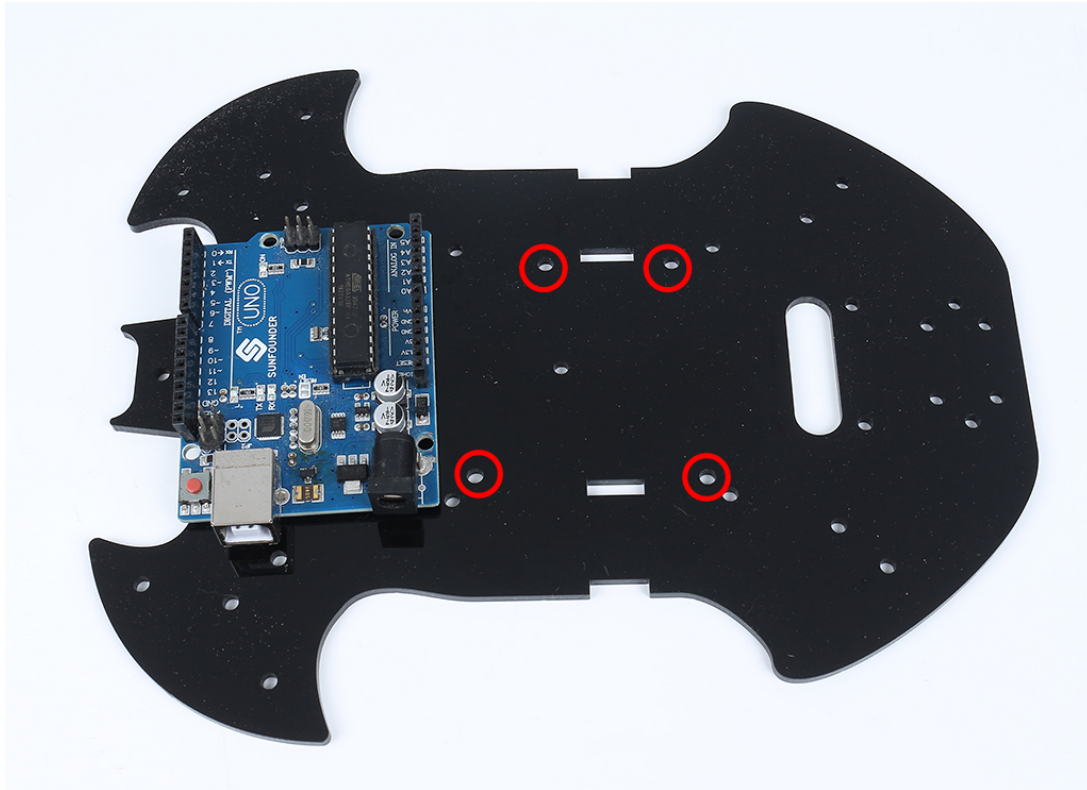
### Étapes

Veuillez suivre les étapes ci-dessous pour terminer l'assemblage de la voiture.

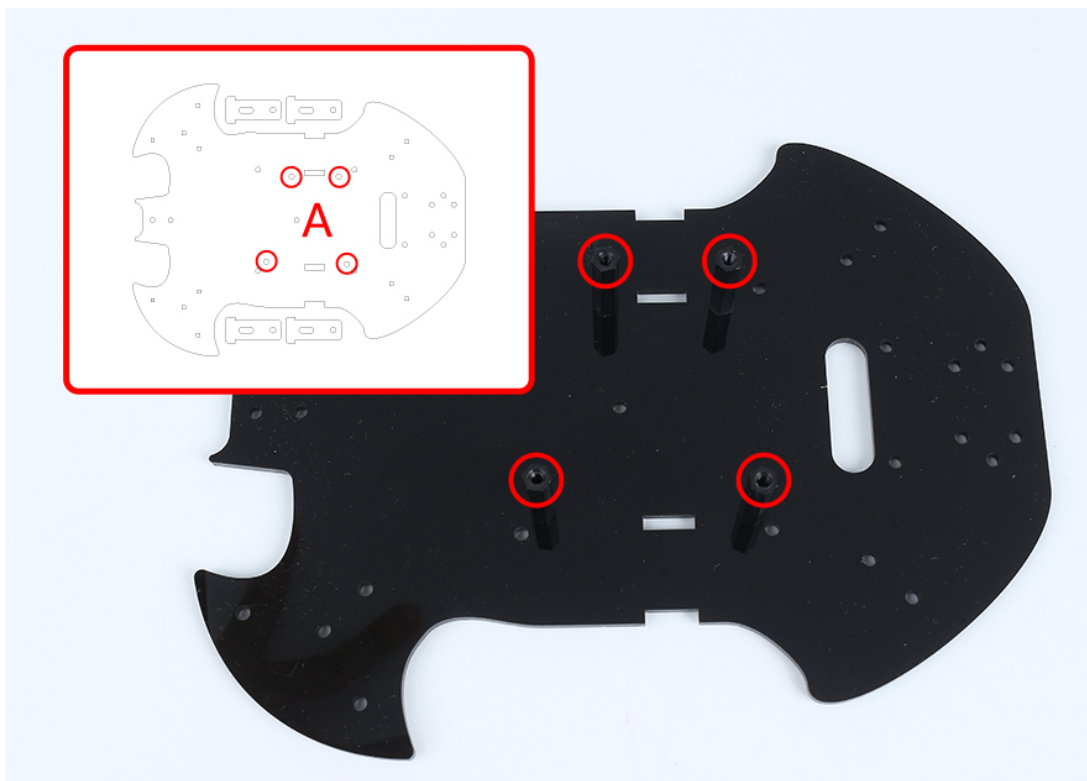
1. Retirez le film protecteur sur l'acrylique.



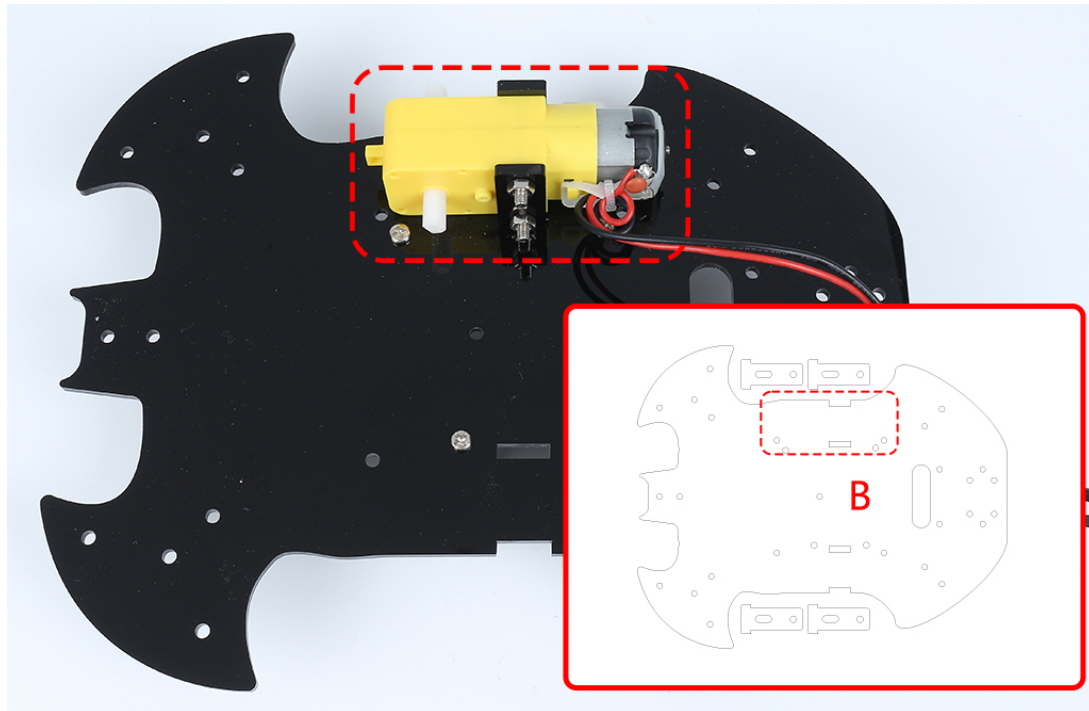
2. Placez la planche sur la table comme indiqué sur la photo, le côté avec le même trou que la carte R3, nous l'appelons A ; l'arrière est B. Cela vous aidera à éviter les erreurs pendant l'assemblage.



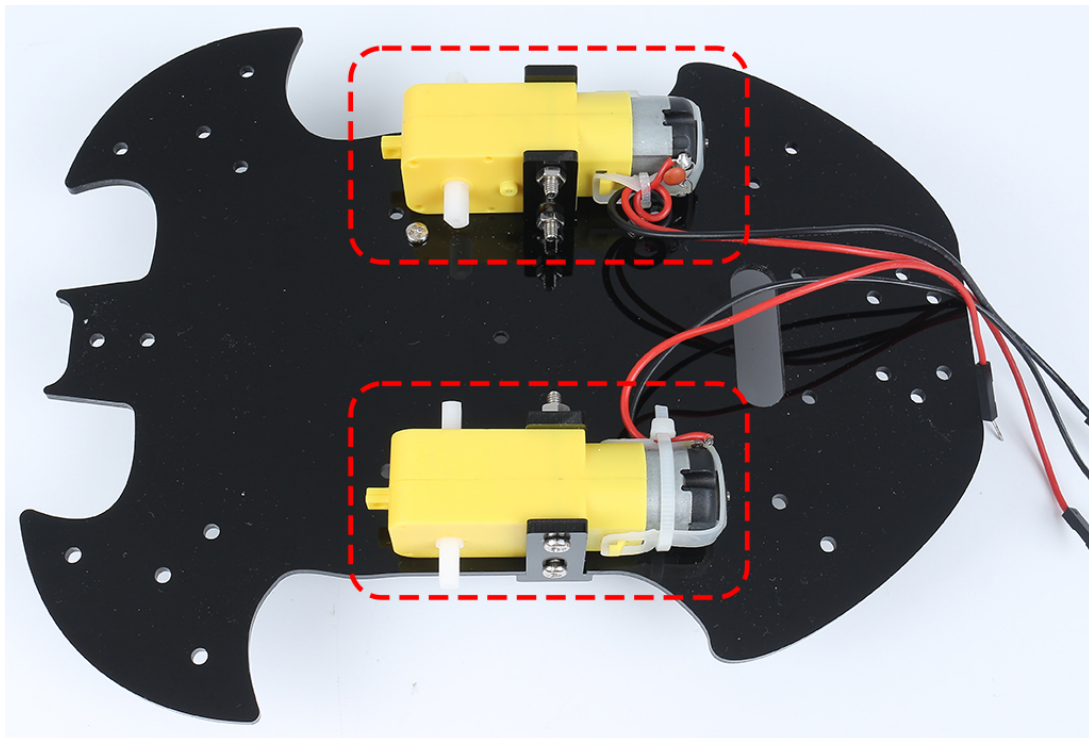
3. Montez le **support M3x24mm** avec des **vis M3x6mm** à la position indiquée ci-dessous.



4. Retournez sur le côté B, utilisez des **vis M3x30mm** et des **écrous M3** pour fixer le moteur TT. 2 détails ici :  
 1 - l'arbre de sortie est orienté vers le côté en forme de chauve-souris ; 2 - le câble du moteur est orienté vers l'intérieur.

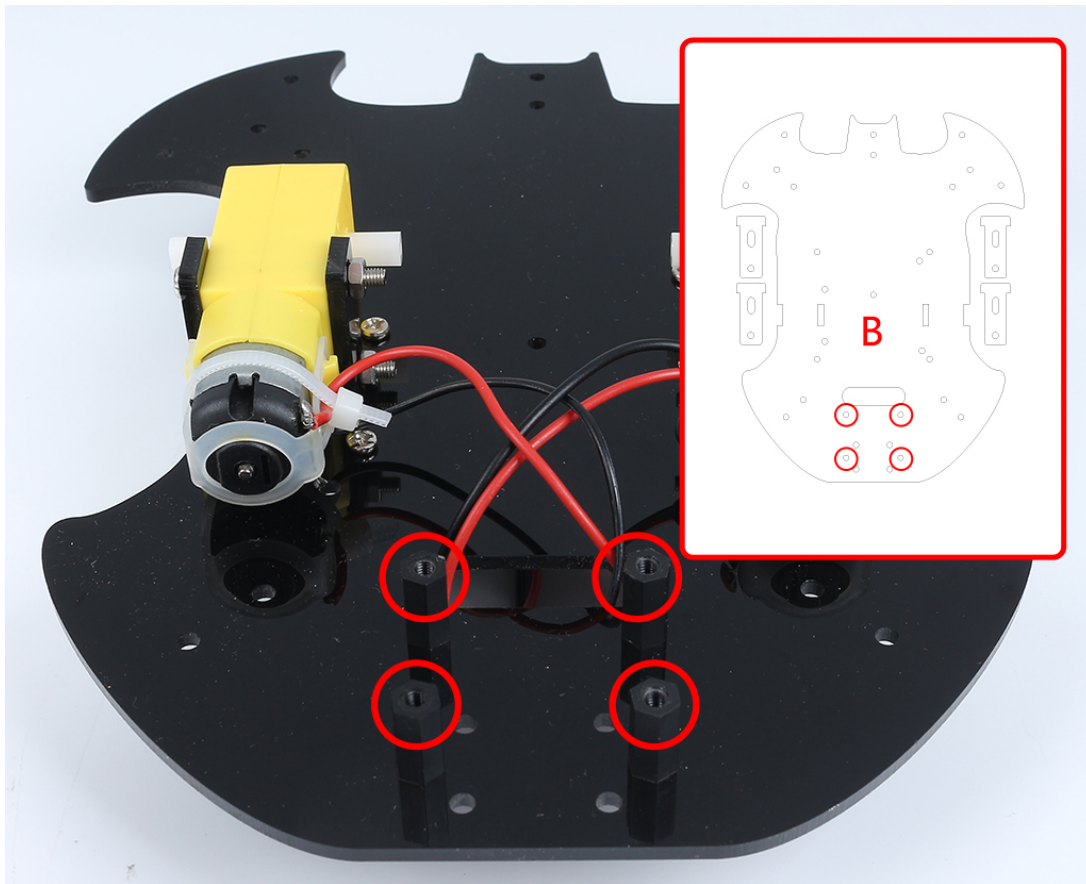


5. Montez un autre moteur TT, la même attention doit être portée à la direction de l'arbre de sortie et à la direction du câble.

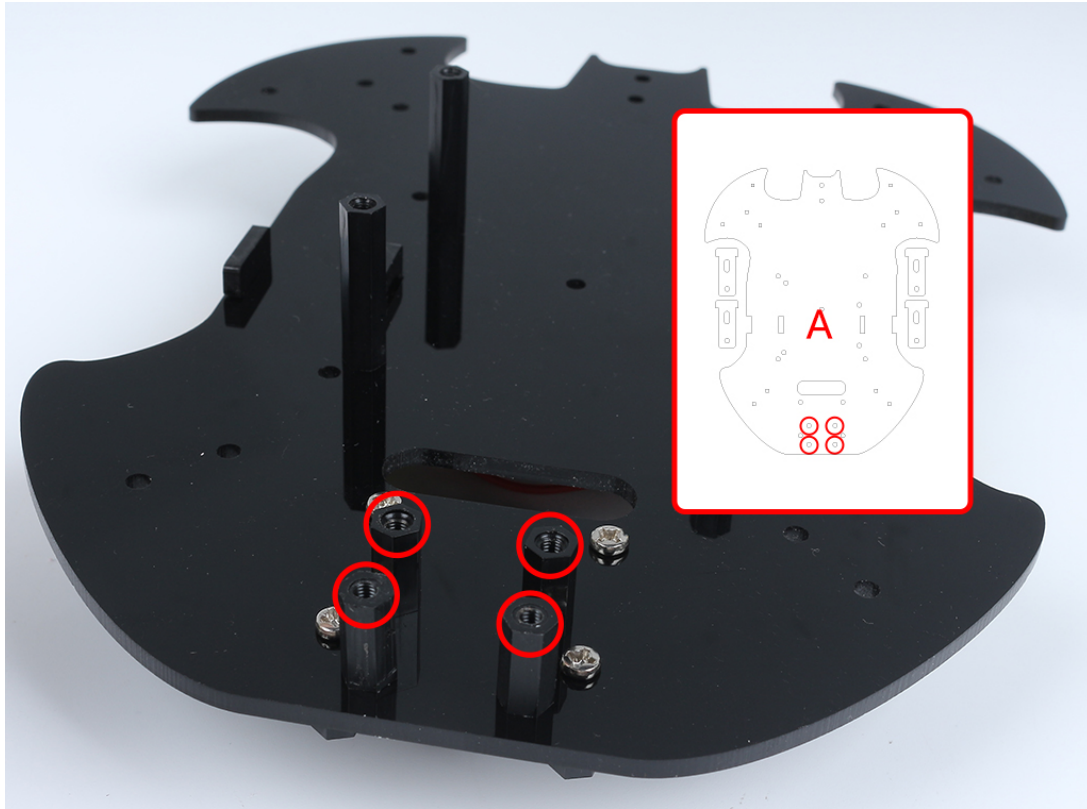


6. Utilisez des vis **M3x6mm** pour monter le **support M3x10mm** à la position indiquée ci-dessous.

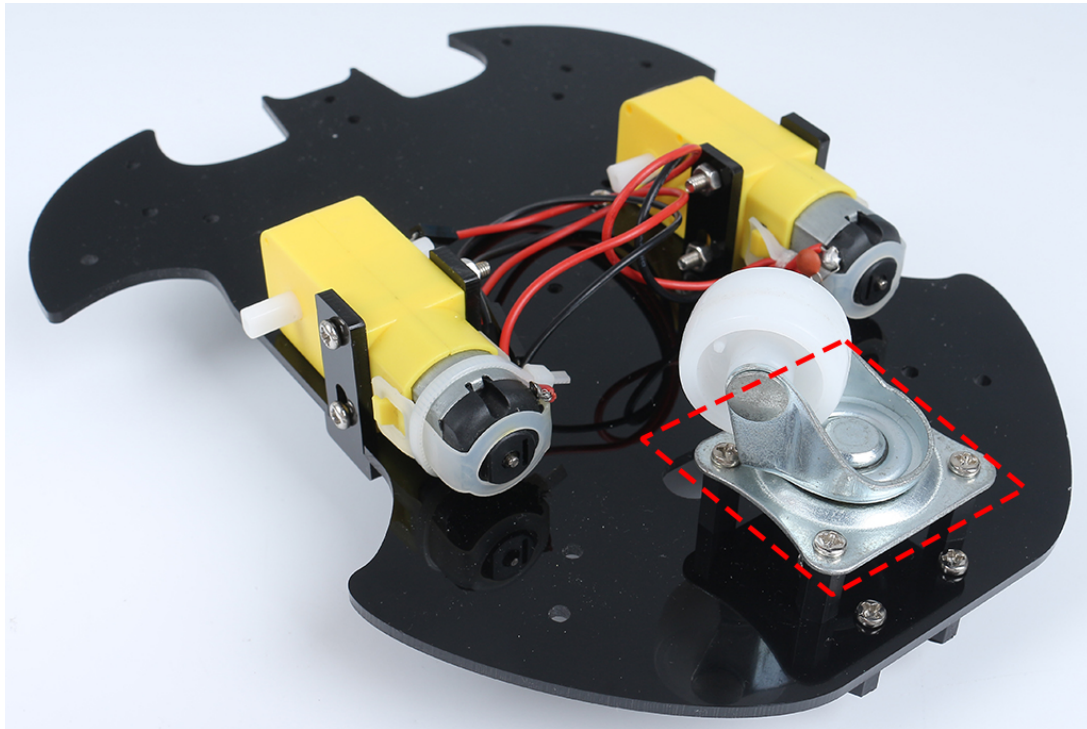




7. Fixez le **support M2.5x11mm** à l'arrière de la voiture avec des **vis M2.5x6mm**.

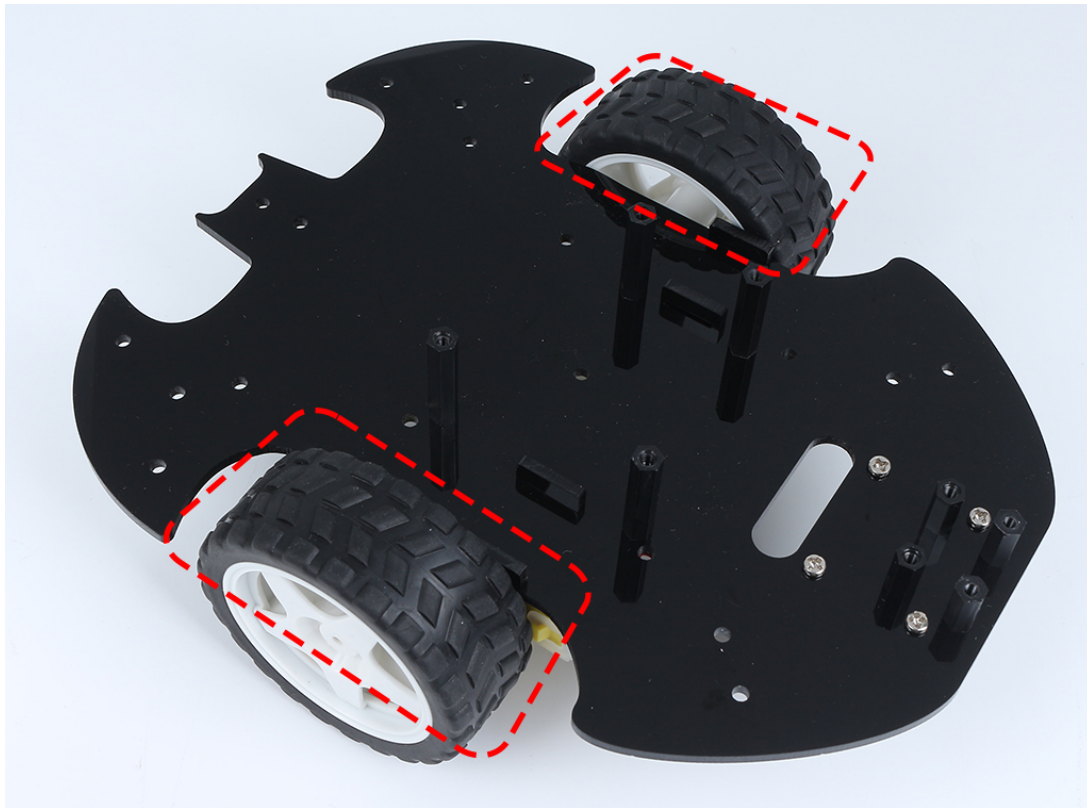


8. Utilisez des vis **M3x6mm** pour monter la roue universelle.



9. Mettez les 2 roues et la structure de base de la voiture est terminée.

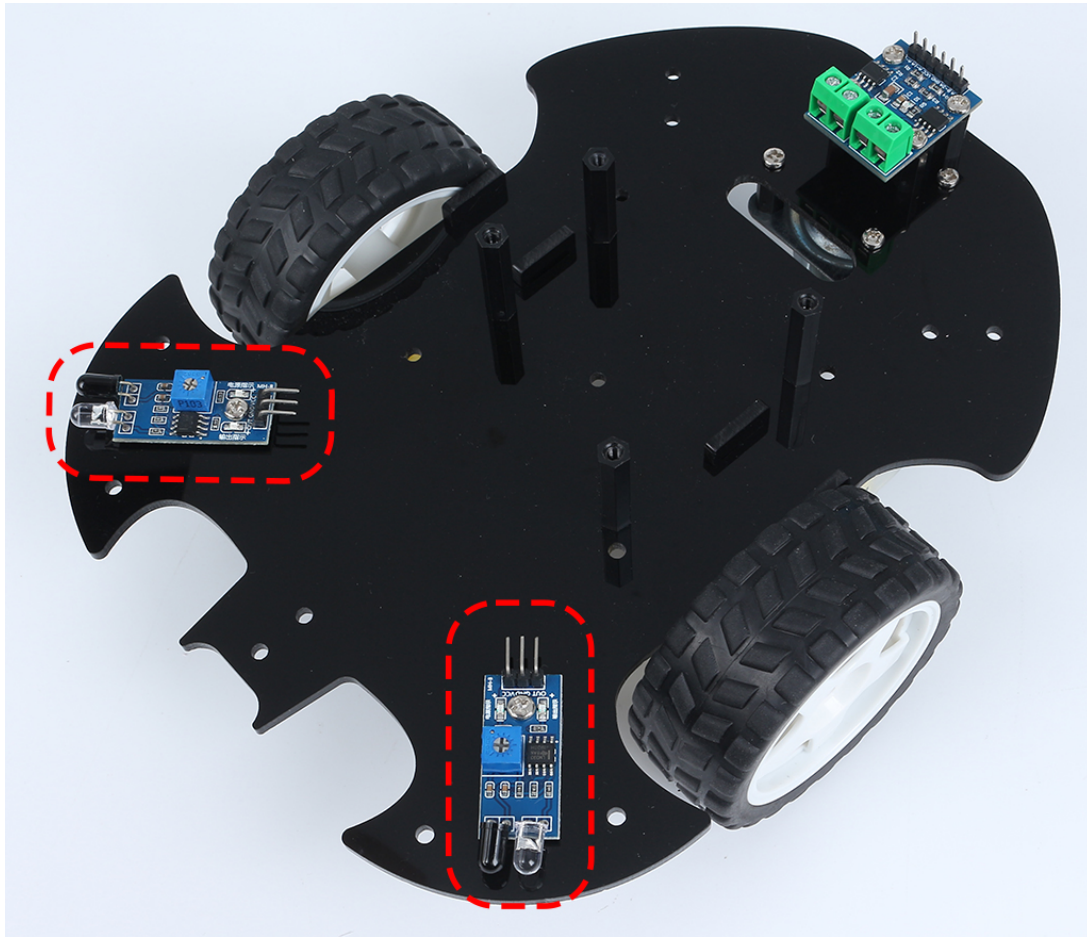




10. Fixez le module L9110 avec des vis M2.5x6mm.



11. Assemblez les deux modules de détection d'obstacles IR avec des **vis M3x10mm** et des **écrous M3**.



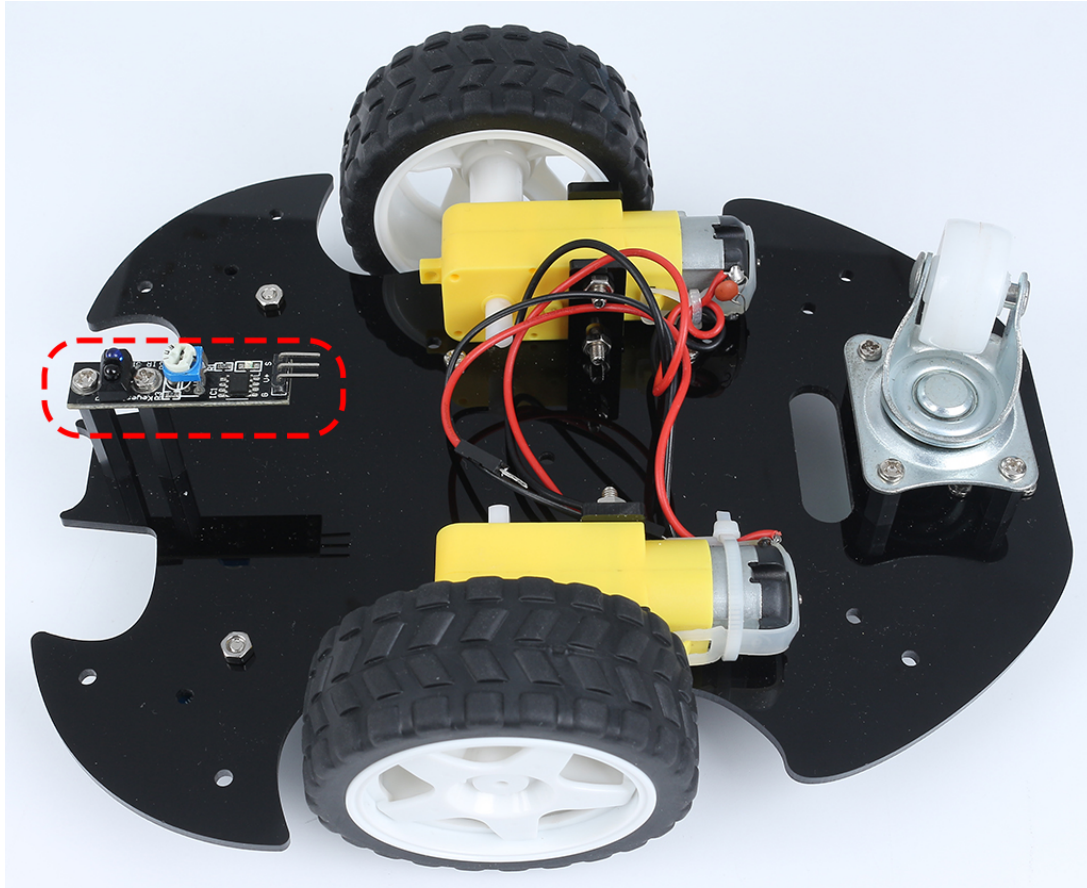
12. Retournez sur le côté B et attachez le module de suivi de ligne avec quatre **vis M3x6mm** et deux **supports M3x24mm**.

---

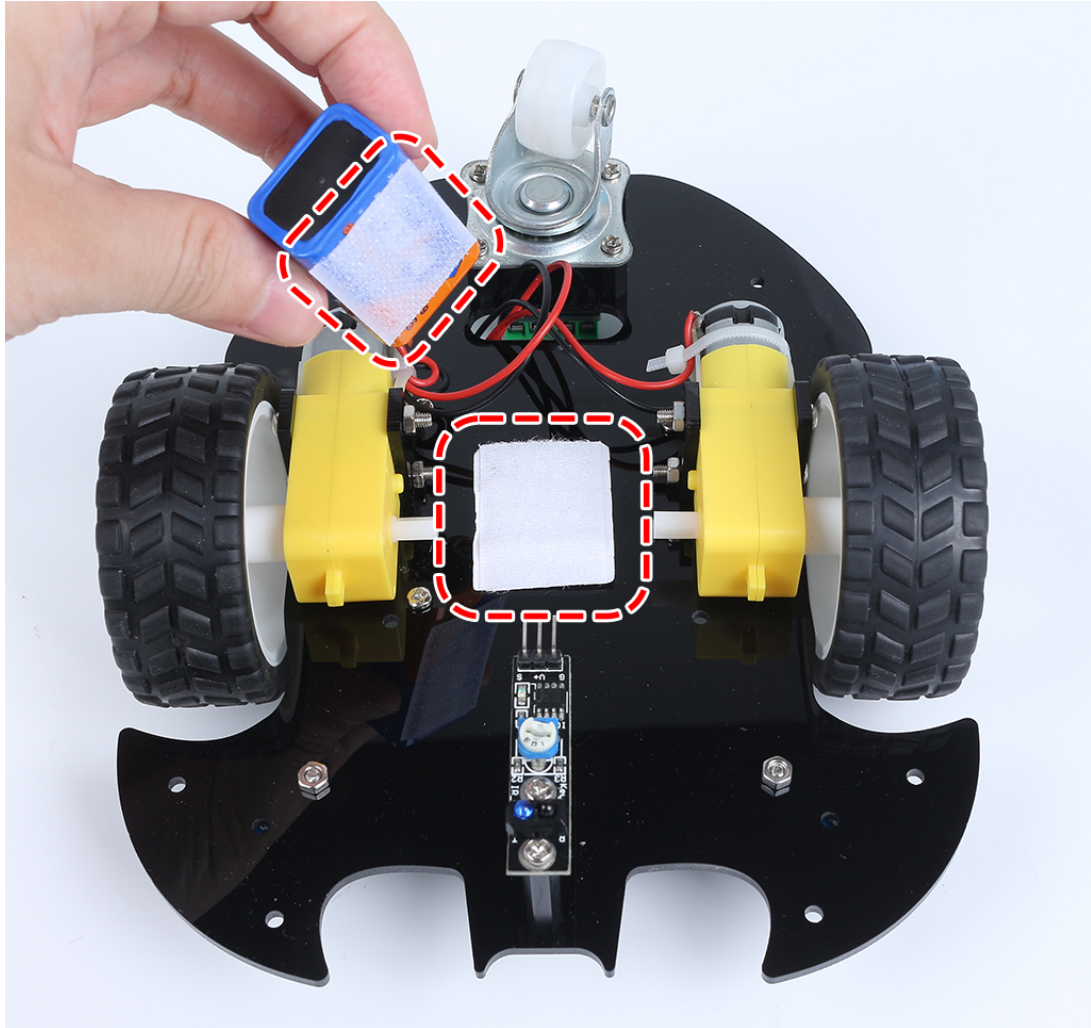
**Note :** Il est conseillé de fixer d'abord les **supports M3x24mm** sur le module de suivi de ligne.  
Une note importante à garder à l'esprit : les broches du capteur de ligne sont légèrement souples et dépassent un peu vers les trous. Lors du vissage des **supports M3x24mm**, appliquez une légère pression pour écarter légèrement les broches du capteur.

---

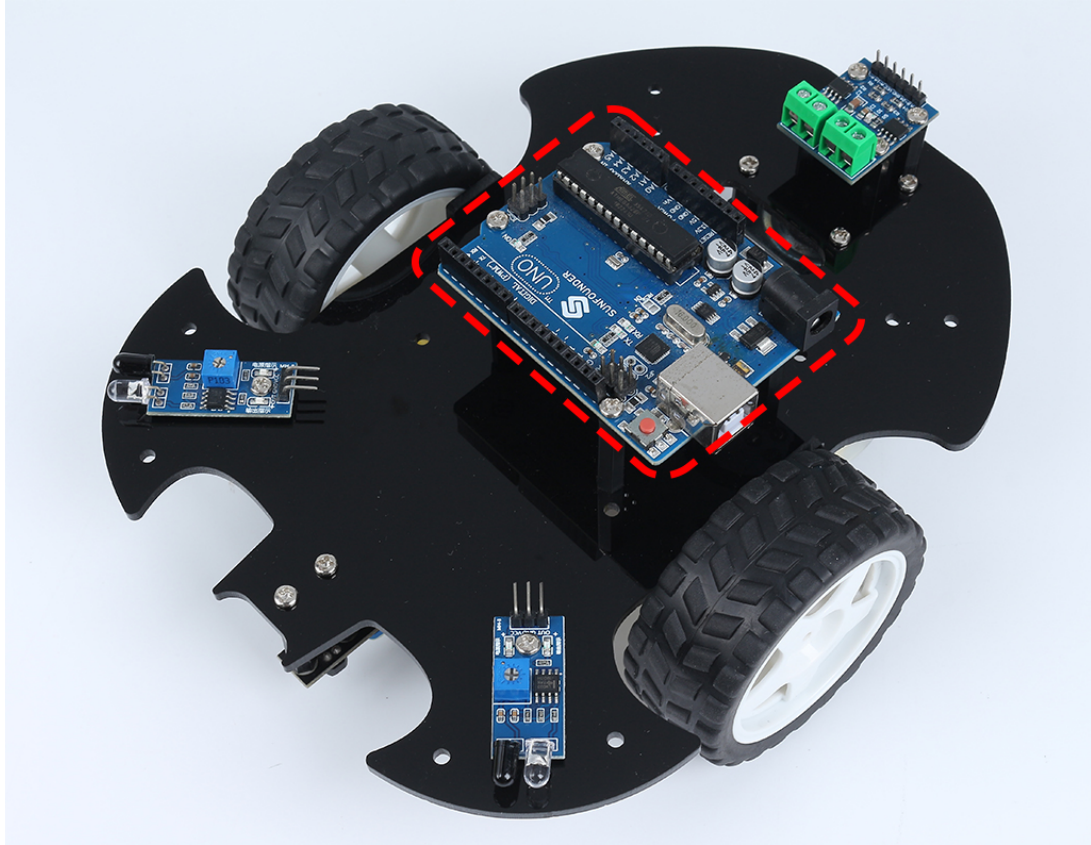




13. Collez le velcro sur la batterie 9V et mettez le clip de batterie. Collez l'autre section du Velcro sur la voiture pour fixer la batterie.

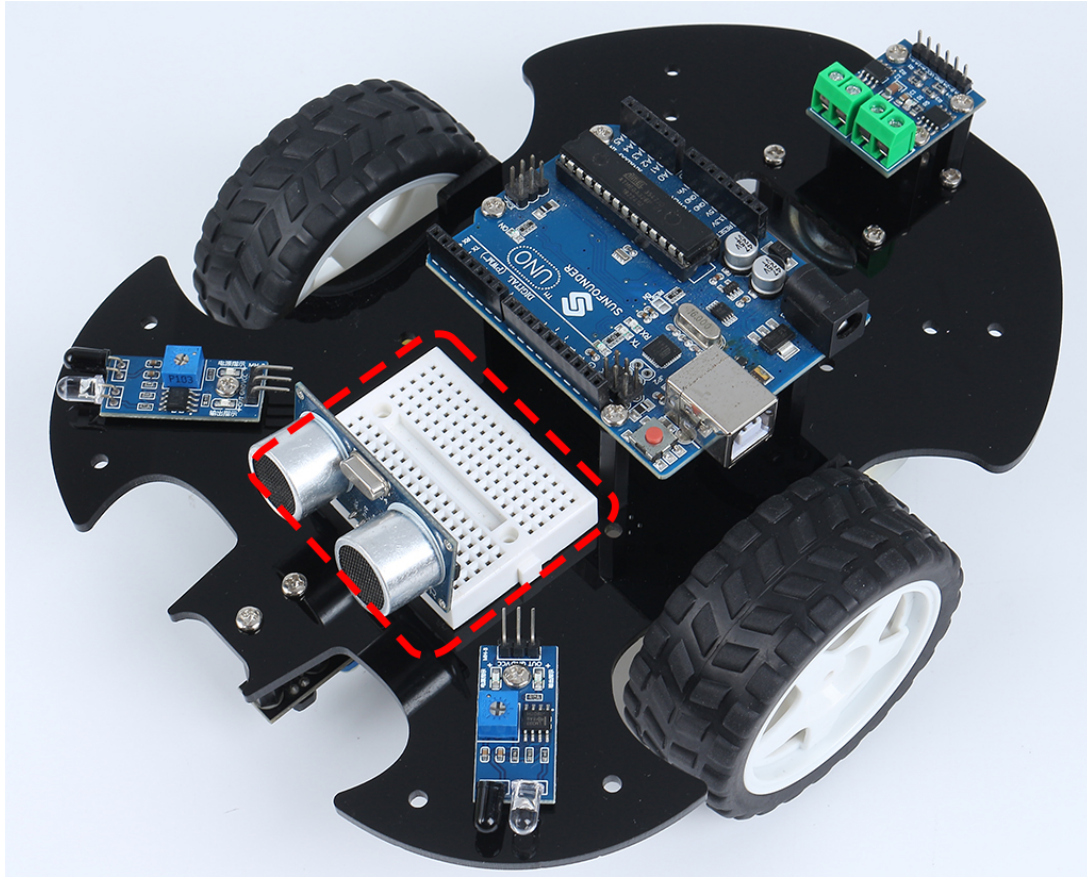


14. Retournez sur le côté A et montez la carte R3 avec des vis M3x6mm.



15. Attachez la plaque de pain à l'avant de la voiture. Par la suite, vous pourrez ajouter différents composants (par exemple, module ultrasonique) à la plaque de pain selon les besoins de votre projet.





16. Pour faire fonctionner la voiture, il faudra également la câbler et écrire du code, ce qui sera fait dans les sections suivantes.

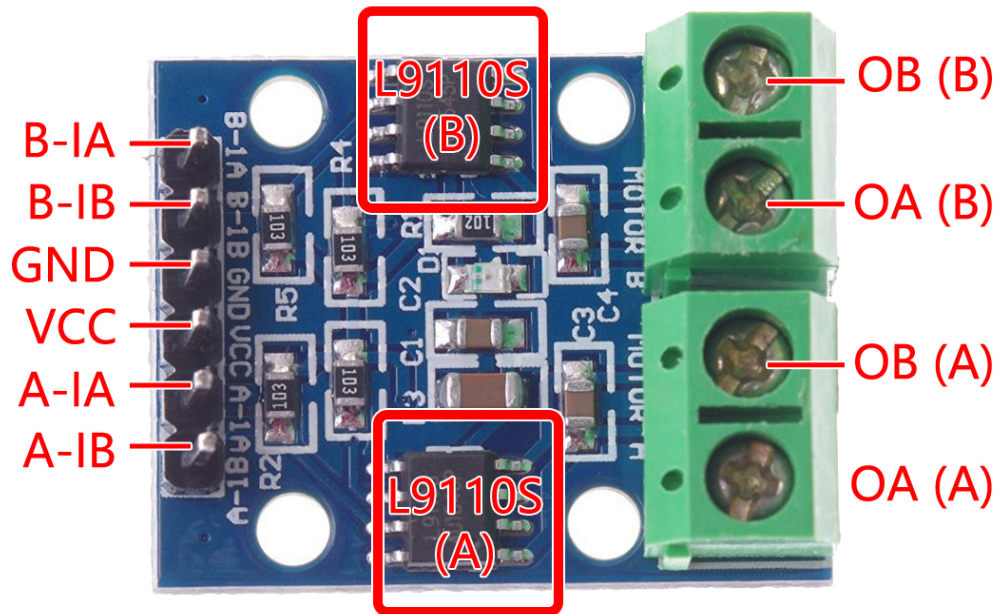
### Projets

Voici quelques projets pour la voiture, programmés en C avec l'IDE Arduino. Si vous n'êtes pas particulièrement compétent avec Arduino, vous pouvez vous référer à *Découverte d'Arduino*.

Les projets suivants sont rédigés par ordre de difficulté de programmation, il est recommandé de lire ces livres dans l'ordre.

Si vous souhaitez programmer une voiture avec Scratch, veuillez vous référer à : *Jouez avec Scratch*.

## 6.2 1. Mouvement



Avant de commencer la programmation, revoyons le principe de fonctionnement du module L9110.

Voici la table de vérité du moteur B :

B-1A	B-1B(B-2A)	État du moteur B
1	0	Rotation dans le sens des aiguilles d'une montre
0	1	Rotation dans le sens inverse des aiguilles d'une montre
0	0	Freinage
1	1	Arrêt

Voici la table de vérité du moteur A :

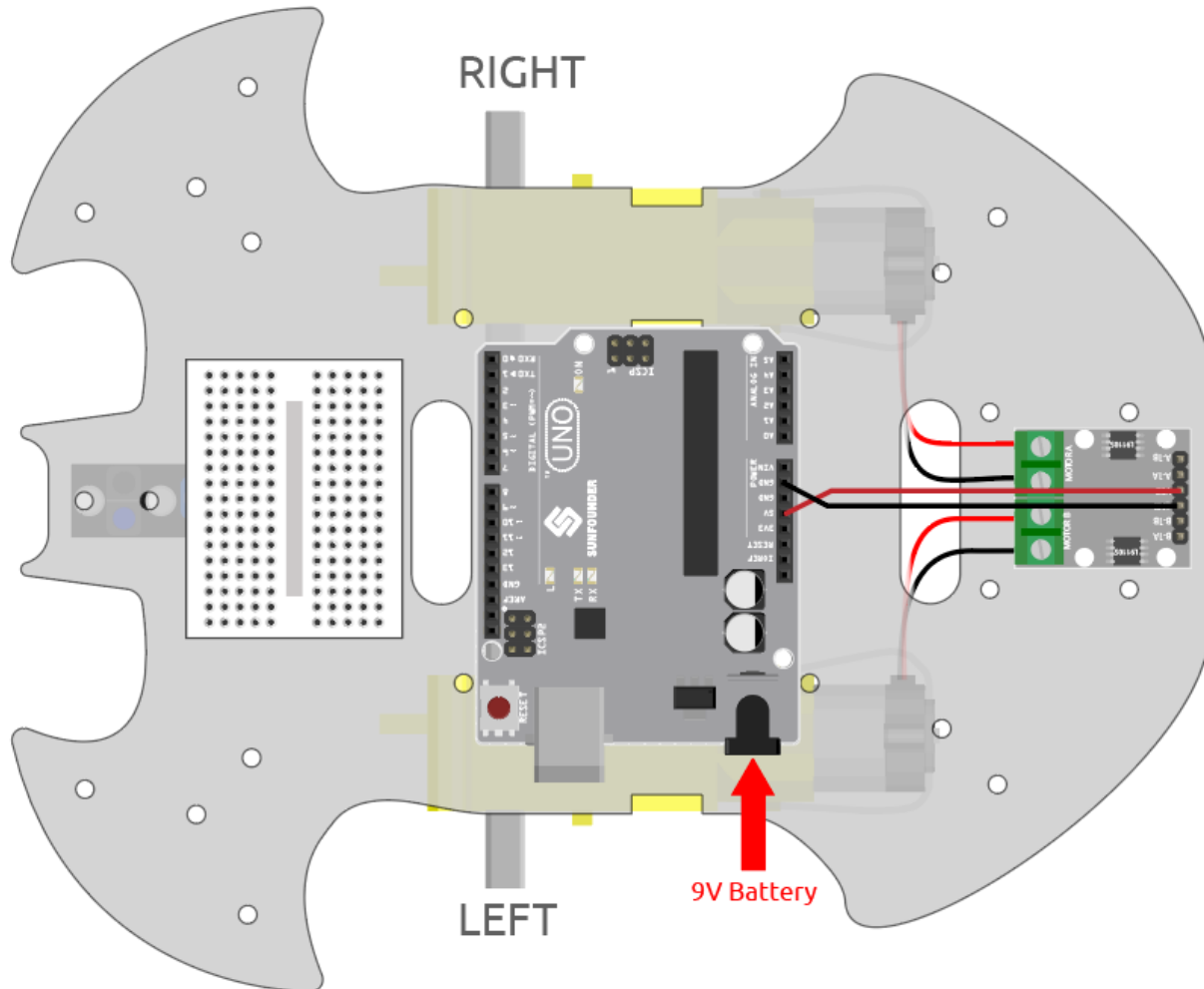
A-1A	A-1B	État du moteur B
1	0	Rotation dans le sens des aiguilles d'une montre
0	1	Rotation dans le sens inverse des aiguilles d'une montre
0	0	Freinage
1	1	Arrêt

— *Module de Contrôle Moteur L9110*

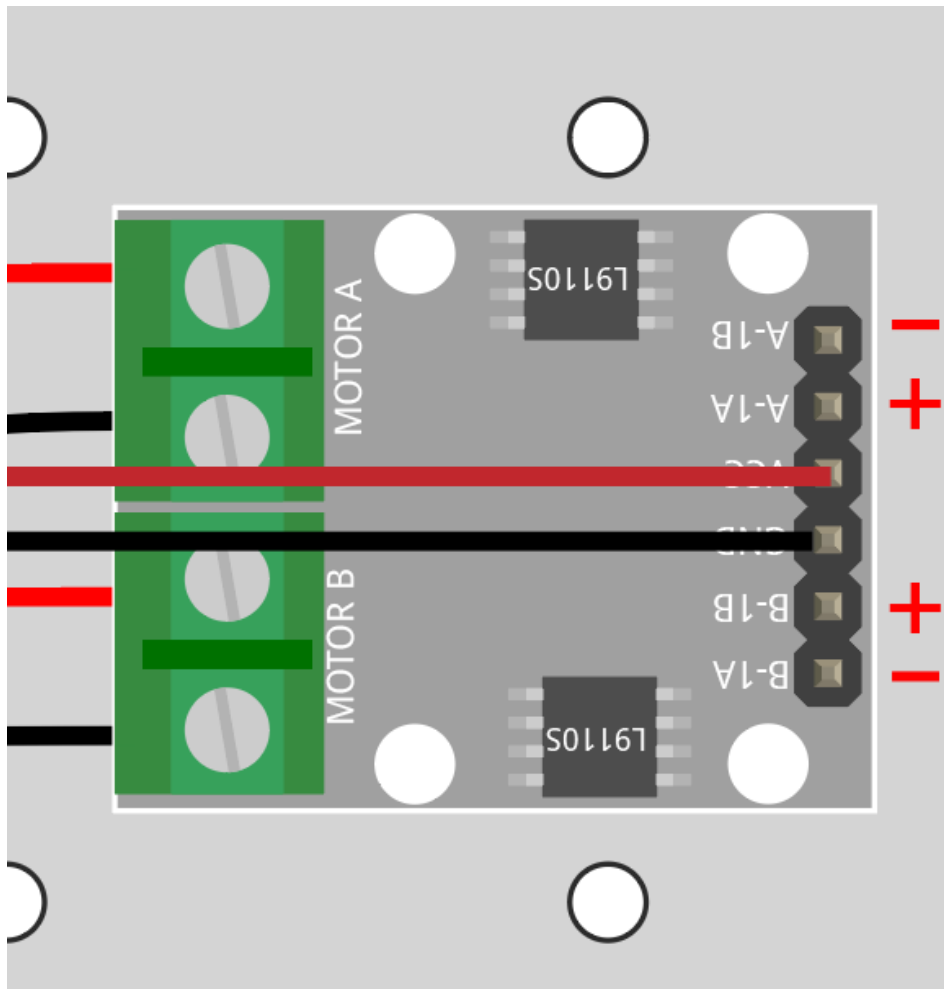
### Avancer

Connectons maintenant l'entrée du module L9110 directement à 12V et GND respectivement pour faire avancer la voiture.

1. Connectez la carte R3, le module L9110 et 2 moteurs.



2. Connectez B-1B(B-2A) et A-1A à VCC, et B-1A et A-1B à GND, alors vous pourrez voir la voiture avancer.

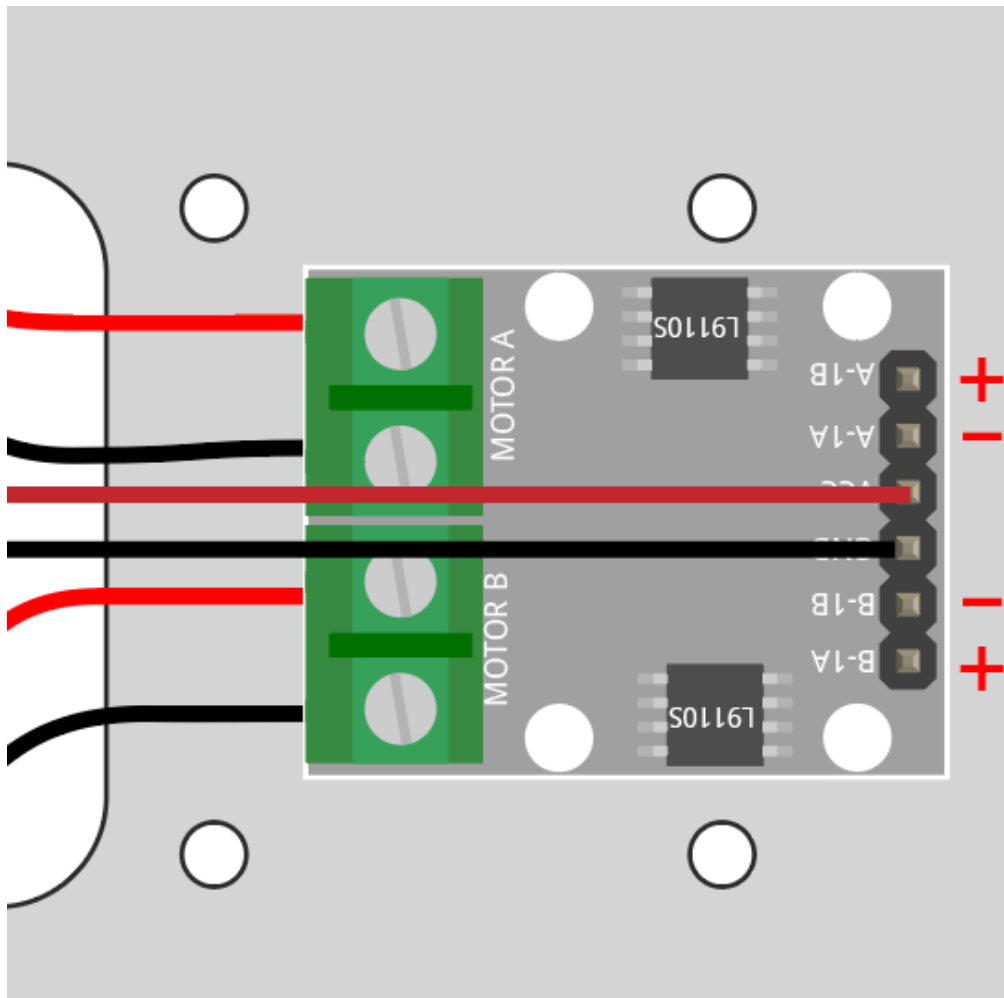


Si les deux ne tournent pas vers l'avant, mais que les situations suivantes se produisent, vous devrez réajuster le câblage des deux moteurs.

- Si les deux moteurs tournent en arrière en même temps (le moteur gauche tourne dans le sens des aiguilles d'une montre, le moteur droit tourne dans le sens inverse), échangez le câblage des moteurs gauche et droit en même temps, OA(A) et OB(A) échangent, OA(B) et OB(B) échangent.
- Si le moteur gauche tourne en arrière (rotation dans le sens des aiguilles d'une montre), échangez le câblage de OA(B) et OB(B) du moteur gauche.
- Si le moteur droit tourne en arrière (rotation dans le sens inverse des aiguilles d'une montre), échangez le câblage de OA(A) et OB(A) du moteur droit.

#### Arrière

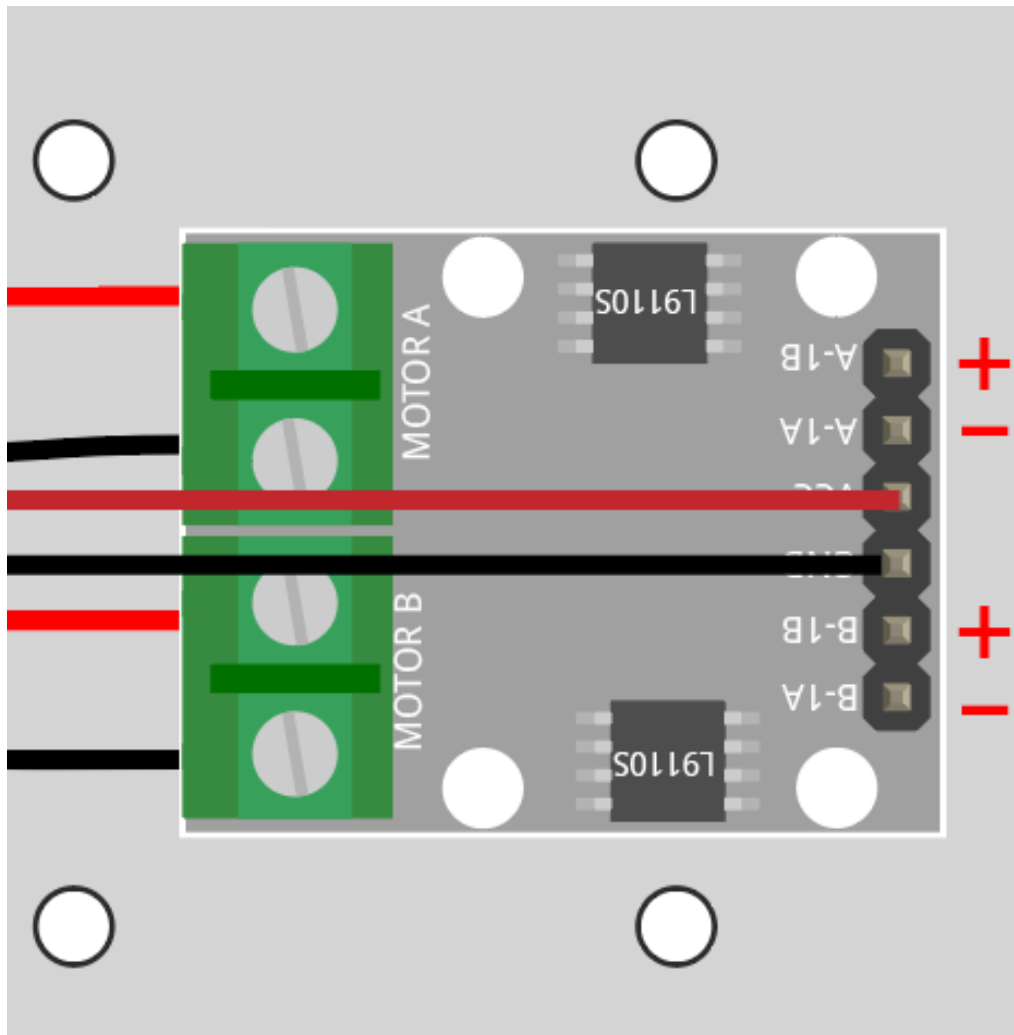
Connectez B-1B(B-2A) et A-1A à GND, et B-1A et A-1B à VCC, alors vous pourrez voir la voiture reculer.



### Tourner à Gauche

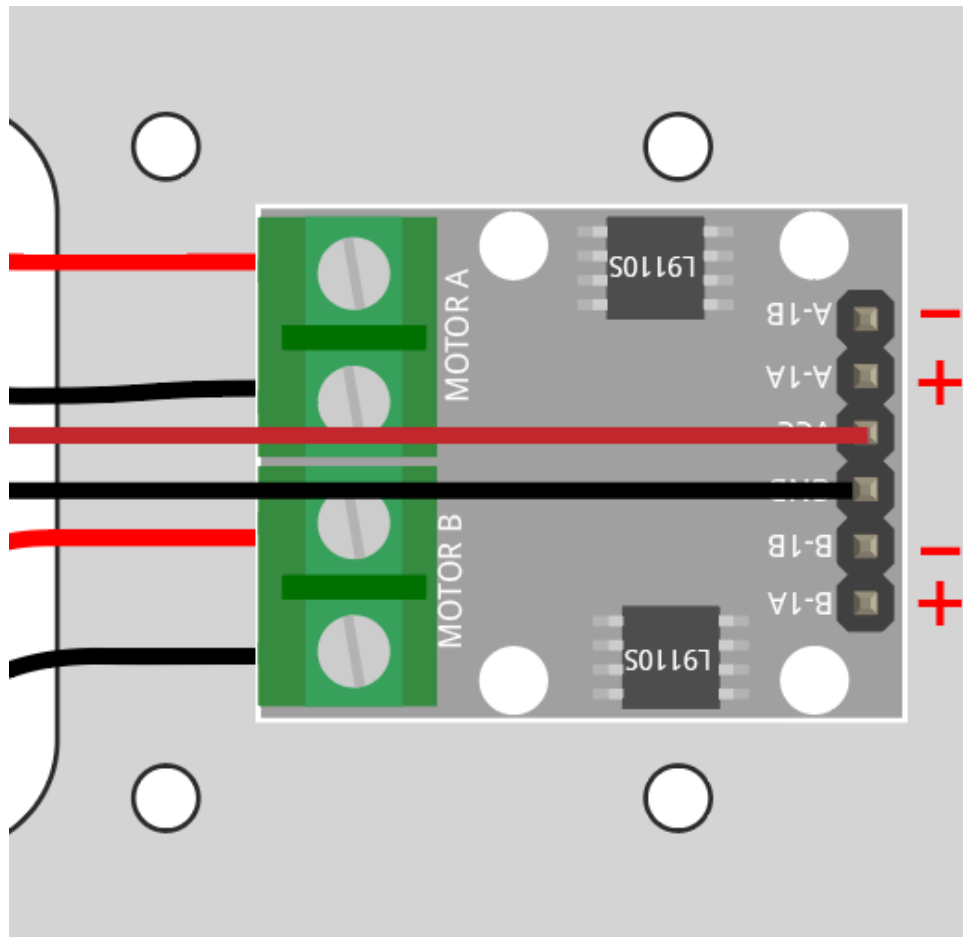
Si vous voulez faire tourner la voiture à gauche, c'est-à-dire faire tourner les deux moteurs dans le sens des aiguilles d'une montre. Vous devez connecter B-1A et A-1A à GND, et B-1B(B-2A) et A-1B à VCC.





### Tourner à Droite

Inversement, si vous voulez tourner la voiture à droite, c'est-à-dire faire tourner les deux moteurs dans le sens inverse des aiguilles d'une montre. Vous devez connecter B-1A et A-1A à VCC et B-1B(B-2A) et A-1B à GND.



### Arrêt

Pour arrêter le moteur, connectez les entrées du même côté à 12V ou GND en même temps, par exemple connectez B-1A et B-1B(B-2A) à 12V ou 5V en même temps, et de même pour A-1A et A-1B.

Ceci est bien sûr théorique et nécessaire plus tard lors de la commande avec du code. Ici, retirez l'alimentation de la voiture pour l'arrêter.

## 6.3 2. Mouvement par Code

Dans le projet précédent, nous avons essayé de contrôler le fonctionnement du moteur en utilisant différents signaux de niveau pour l'entrée du module L9110.

Si nous modifions les signaux de niveau par le programme, alors nous pouvons contrôler le mouvement de la voiture de manière flexible.

### Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

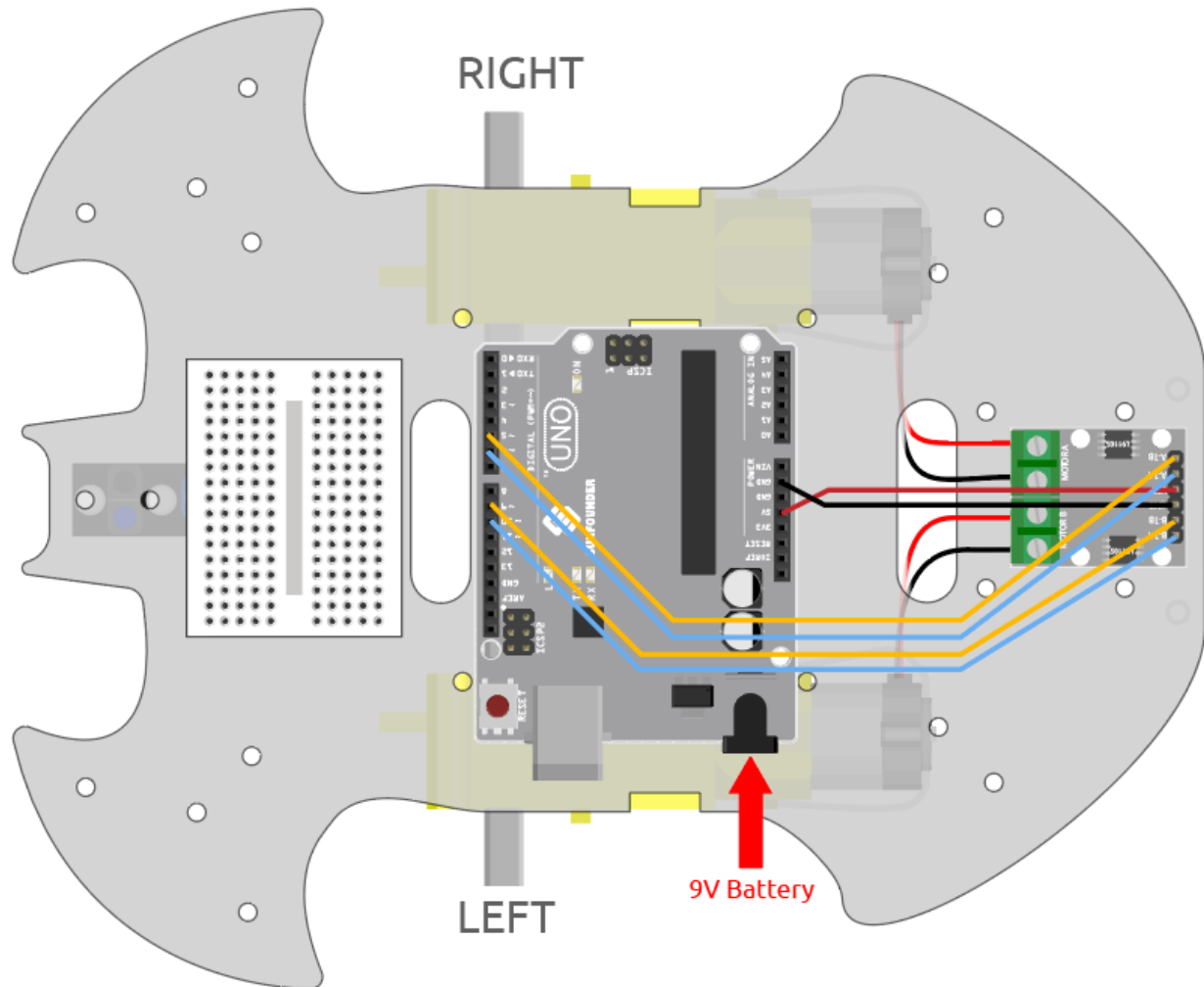
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-

### Câblage

Connectez les fils entre le module L9110 et la carte R3 selon le schéma ci-dessous.

L9110 Module	R3 Board	Motor
A-1B	5	
A-1A	6	
B-1B(B-2A)	9	
B-1A	10	
OB(B)		Black wire of right motor
OA(B)		Red wire of right motor
OB(A)		Black wire of left motor
OA(A)		Red wire of left motor



## Code

### Note :

- Ouvrez le fichier 2.move.ino dans le chemin 3in1-kit\car\_project\2.move.
- Ou copiez ce code dans **Arduino IDE**.

Après le téléchargement du code, la voiture se déplacera respectivement vers l'avant, l'arrière, la gauche et la droite pendant deux secondes.

### Comment ça fonctionne ?

Ce projet est essentiellement le même que le précédent, impliquant de faire avancer, reculer, tourner à gauche et à droite la voiture, ainsi que de l'arrêter en fournissant différents niveaux de signal aux broches d'entrée du module L9110.

1. Initialisez les broches du module L9110.

```
const int A_1B = 5;
const int A_1A = 6;
const int B_1B = 9;
const int B_1A = 10;
```

(suite sur la page suivante)

(suite de la page précédente)

```
void setup() {
    pinMode(A_1B, OUTPUT);
    pinMode(A_1A, OUTPUT);
    pinMode(B_1B, OUTPUT);
    pinMode(B_1A, OUTPUT);
}
```

2. Réglez les broches d'entrée à différents niveaux hauts ou bas pour contrôler la rotation des moteurs gauche et droit, puis encapsulez-les dans des fonctions individuelles.

```
void moveForward() {
    digitalWrite(A_1B, LOW);
    digitalWrite(A_1A, HIGH);
    digitalWrite(B_1B, HIGH);
    digitalWrite(B_1A, LOW);
}

void moveBackward() {
    digitalWrite(A_1B, HIGH);
    digitalWrite(A_1A, LOW);
    digitalWrite(B_1B, LOW);
    digitalWrite(B_1A, HIGH);
}

...
```

3. Appelez ces fonctions dans loop().

```
void loop() {
    moveForward();
    delay(2000);
    stopMove();
    delay(500);

    moveBackward();
    delay(2000);
    stopMove();
    delay(500);

    ...
}
```

— `digitalWrite(pin, value)`

- `pin` : le numéro de la broche Arduino.
- `value` : HIGH ou LOW.

Écrit une valeur HIGH ou LOW sur une broche numérique. Si la broche a été configurée comme une OUTPUT avec `pinMode()`, sa tension sera réglée sur la valeur correspondante : 5V (ou 3,3V sur les cartes 3,3V) pour HIGH, 0V (masse) pour LOW.

— `pinMode(pin, mode)`

- `pin` : le numéro de la broche Arduino à configurer.
- `mode` : INPUT, OUTPUT ou INPUT\_PULLUP.

Configure la broche spécifiée pour se comporter soit comme une entrée, soit comme une sortie.

— `delay(ms)`

- `ms` : le nombre de millisecondes de pause. Types de données autorisés : unsigned long.
- Interrompt le programme pendant la durée (en millisecondes) spécifiée en paramètre. (Il y a 1000 millisecondes dans une seconde.)

## 6.4 3. Accélération

En plus du signal numérique (HIGH/LOW), l'entrée du module L9110 peut également recevoir un signal PWM pour contrôler la vitesse de sortie.

En d'autres termes, nous pouvons utiliser `AnalogWrite()` pour contrôler la vitesse de déplacement de la voiture.

Dans ce projet, nous faisons varier progressivement la vitesse de la voiture vers l'avant, en accélérant d'abord puis en décélérant.

### Câblage

Ce projet utilise le même câblage que [2. Mouvement par Code](#).

### Code

---

#### Note :

- Ouvrez le fichier `3.speed_up.ino` dans le chemin `3in1-kit\car_project\3.speed_up`.
  - Ou copiez ce code dans **Arduino IDE**.
  - Ou téléchargez le code via le [Arduino Web Editor](#).
- 

Après l'exécution du programme, la voiture accélérera progressivement puis décélérera graduellement.

#### Comment ça fonctionne ?

Le but de ce projet est d'écrire différentes valeurs PWM aux broches d'entrée du module L9110 pour contrôler la vitesse avant de la voiture.

1. Utilisez la déclaration `for()` pour donner `speed` par paliers de 5, en écrivant des valeurs de 0 à 255 afin de voir le changement de vitesse de la voiture.

```
void loop() {  
    for(int i=0;i<=255;i+=5){  
        moveForward(i);  
        delay(500);  
    }  
    for(int i=255;i>=0;i-=5){  
        moveForward(i);  
        delay(500);  
    }  
}
```

2. À propos de la fonction `moveForward()`.

Contrairement à [2. Mouvement par Code](#) qui donne directement des niveaux hauts/bas aux broches d'entrée du module L9110, ici nous passons un paramètre `speed` là où nous devons donner des niveaux hauts.

```
void moveForward(int speed) {  
    analogWrite(A_1B, 0);  
    analogWrite(A_1A, speed);  
    analogWrite(B_1B, speed);  
    analogWrite(B_1A, 0);  
}
```

- `for`

La déclaration `for` est utilisée pour répéter un bloc d'instructions entre accolades. Un compteur d'incrément est généralement utilisé pour incrémenter et terminer la boucle.

```
for (initialization; condition; increment) {
  // statement(s);
}
```

- **initialization** : se produit d'abord et une seule fois.
- **condition** : à chaque passage dans la boucle, la condition est testée; si elle est vraie, le bloc d'instructions et l'incrément sont exécutés, puis la condition est à nouveau testée. Lorsque la condition devient fausse, la boucle se termine.
- **increment** : exécuté à chaque passage dans la boucle lorsque la condition est vraie.

## 6.5 4. Suivre la ligne

La voiture est équipée d'un module de suivi de ligne, qui peut être utilisé pour faire suivre à la voiture une ligne noire.

Lorsque le module de suivi de ligne détecte la ligne noire, le moteur droit tourne tandis que le moteur gauche reste immobile, permettant ainsi à la voiture de se déplacer d'un pas vers l'avant gauche. Au fur et à mesure que la voiture se déplace, le module de ligne sera déplacé hors de la ligne, puis le moteur gauche tourne et le moteur droit ne tourne pas, la voiture se déplace d'un pas vers la droite pour revenir sur la ligne. En répétant les deux étapes ci-dessus, la voiture peut se déplacer le long de la ligne noire.

Avant de commencer le projet, vous devez construire une carte de courbe avec du ruban adhésif noir pour ligne, la largeur de ligne recommandée est entre 0,8-1,5 cm et l'angle du virage ne doit pas être inférieur à 90 degrés.

### Composants requis

Dans ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ARTICLES DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

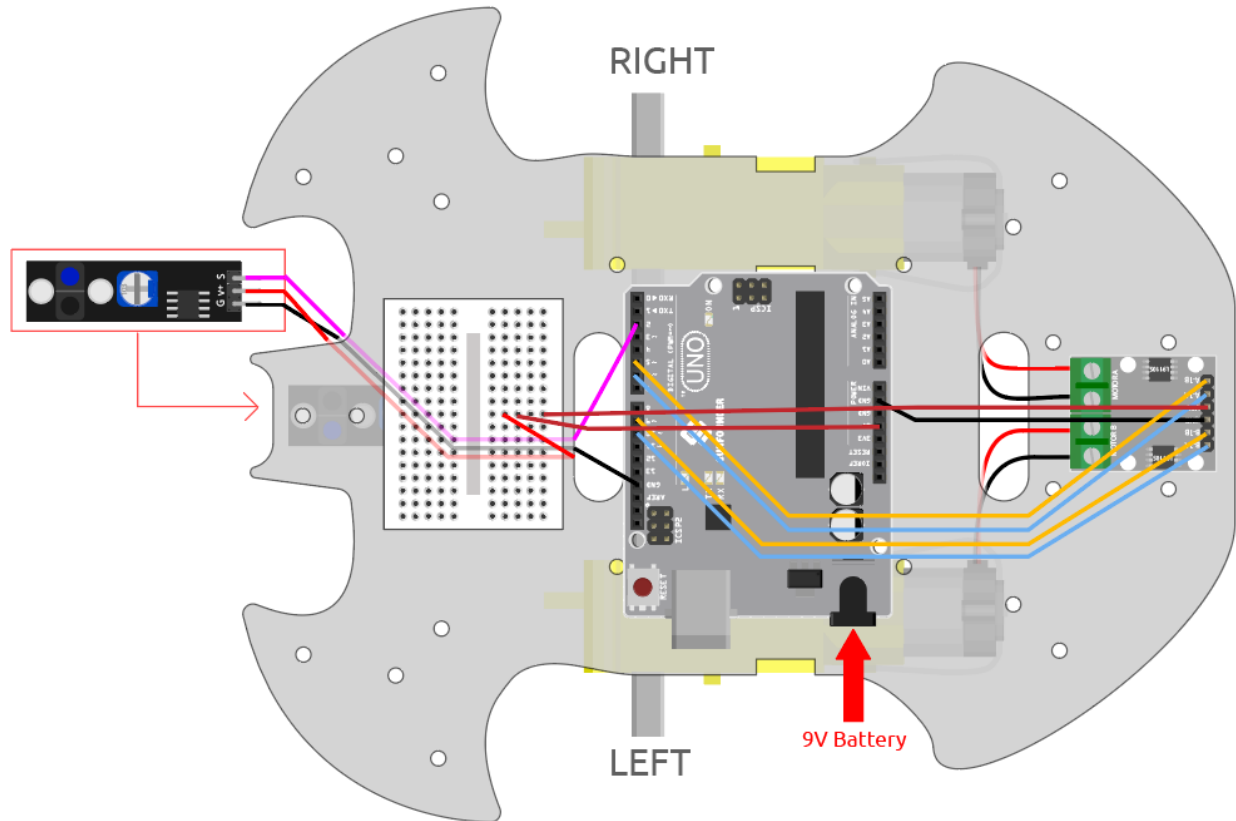
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module de Suivi de Ligne</i>	

### Câblage

C'est un module numérique de suivi de ligne, lorsqu'une ligne noire est détectée, il émet 1 ; lorsqu'une ligne blanche est détectée, il émet une valeur de 0. De plus, vous pouvez ajuster sa distance de détection grâce au potentiomètre sur le module.

Construisez le circuit selon le schéma suivant.

Module de suivi de ligne	Carte R3
S	2
V+	5V
G	GND



### Ajuster le module

Avant de commencer le projet, vous devez ajuster la sensibilité du module.

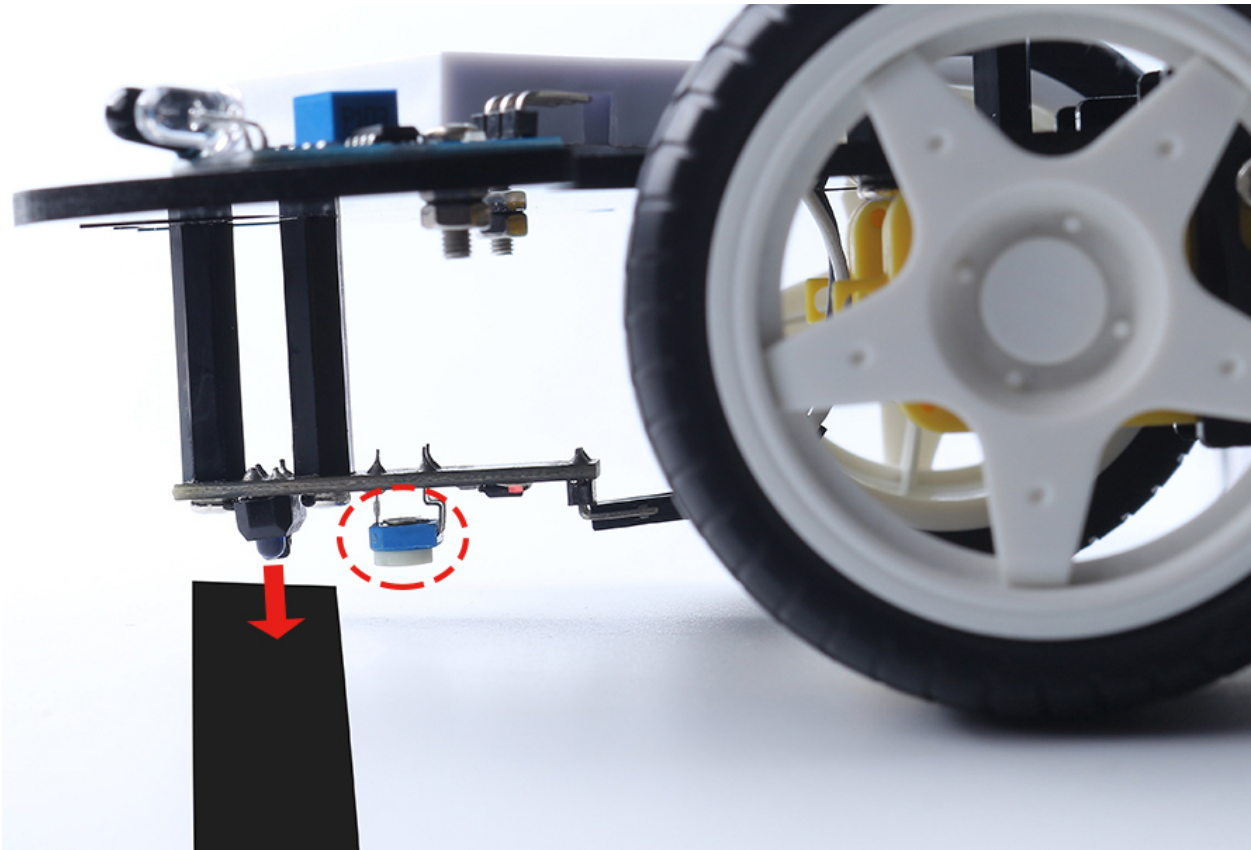
Câblez selon le schéma ci-dessus, puis alimentez la carte R3 (soit directement dans le câble USB soit dans le câble du bouton de la batterie 9V), sans télécharger le code.

Collez un ruban électrique noir sur la table et placez la voiture dessus.

Observez la LED de signal sur le module pour vous assurer qu'elle s'allume sur la table blanche et s'éteint sur le ruban noir.

Si ce n'est pas le cas, vous devez ajuster le potentiomètre sur le module, afin qu'il puisse faire l'effet ci-dessus.





## Code

### Note :

- Ouvrez le fichier `4.follow_the_line.ino` sous le chemin de `3in1-kit\car_project\4.follow_the_line`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via le [Arduino Web Editor](#).

Après avoir téléchargé le code sur la carte R3, alignez le module de suivi de ligne sous la voiture avec la ligne noire, et vous verrez la voiture suivre la ligne.

### Comment ça fonctionne ?

Dans ce code, il permet aux deux moteurs de micro-tourner à gauche et à droite en fonction de la valeur du module de suivi de ligne, afin que vous puissiez voir la voiture suivre la ligne noire.

1. Ajoutez la définition de pin pour le module de suivi de ligne, ici il est réglé sur `INPUT`. Initialisez également le moniteur série et réglez le débit en bauds à 9600bps.

```
...  
const int lineTrack = 2;  
Serial.begin(9600);  
void setup() {  
    ...  
    pinMode(lineTrack, INPUT);  
}
```

2. Lisez la valeur du module de suivi de ligne, si elle est 1, alors laissez la voiture aller vers l'avant gauche ; sinon, aller vers l'avant droite. Vous pouvez également ouvrir le moniteur série en cliquant sur l'icône de la loupe dans le coin supérieur droit pour voir le changement de la valeur du module de suivi de ligne sur la ligne noire et blanche avant de débrancher le câble USB.

```
void loop() {

    int speed = 150;

    int lineColor = digitalRead(lineTrack); // 0:white    1:black
    Serial.println(lineColor);
    if (lineColor) {
        moveLeft(speed);
    } else {
        moveRight(speed);
    }
}
```

3. À propos des fonctions `moveLeft()` et `moveRight()`.

Contrairement à la fonction de virage gauche-droite dans le projet 2. *Mouvement par Code*, seuls de petits virages gauche-droite sont nécessaires ici, donc vous n'avez besoin d'ajuster la valeur de A\_1A ou B\_1B à chaque fois. Par exemple, si vous vous déplacez vers l'avant gauche (`moveLeft()`), vous devez uniquement régler la vitesse sur A\_1A et toutes les autres à 0, ce qui fera tourner le moteur droit dans le sens des aiguilles d'une montre et le moteur gauche immobile.

```
void moveLeft(int speed) {
    analogWrite(A_1B, 0);
    analogWrite(A_1A, speed);
    analogWrite(B_1B, 0);
    analogWrite(B_1A, 0);
}

void moveRight(int speed) {
    analogWrite(A_1B, 0);
    analogWrite(A_1A, 0);
    analogWrite(B_1B, speed);
    analogWrite(B_1A, 0);
}
```

#### — Serial

Utilisé pour la communication entre la carte Arduino et un ordinateur ou d'autres appareils.

- `Serial.begin()` : Définit le taux de données en bits par seconde (baud) pour la transmission de données série.
- `Serial.println()` : Imprime des données sur le port série sous forme de texte ASCII lisible par l'homme suivi d'un caractère de retour chariot (ASCII 13, ou "r") et d'un caractère de saut de ligne (ASCII 10, ou "n").

#### — if else

Le `if else` permet un contrôle plus grand sur le flux de code que la simple instruction `if`, en permettant de regrouper plusieurs tests.

## 6.6 5. Jouer avec le module d'évitement d'obstacles

Deux modules infrarouges d'évitement d'obstacles sont installés à l'avant de la voiture, ils permettent de détecter certains obstacles proches.

Dans ce projet, la voiture se déplace librement vers l'avant et, lorsqu'elle rencontre un obstacle, elle est capable de l'éviter et de continuer à se déplacer dans d'autres directions.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module d'Évitement d'Obstacle</i>	

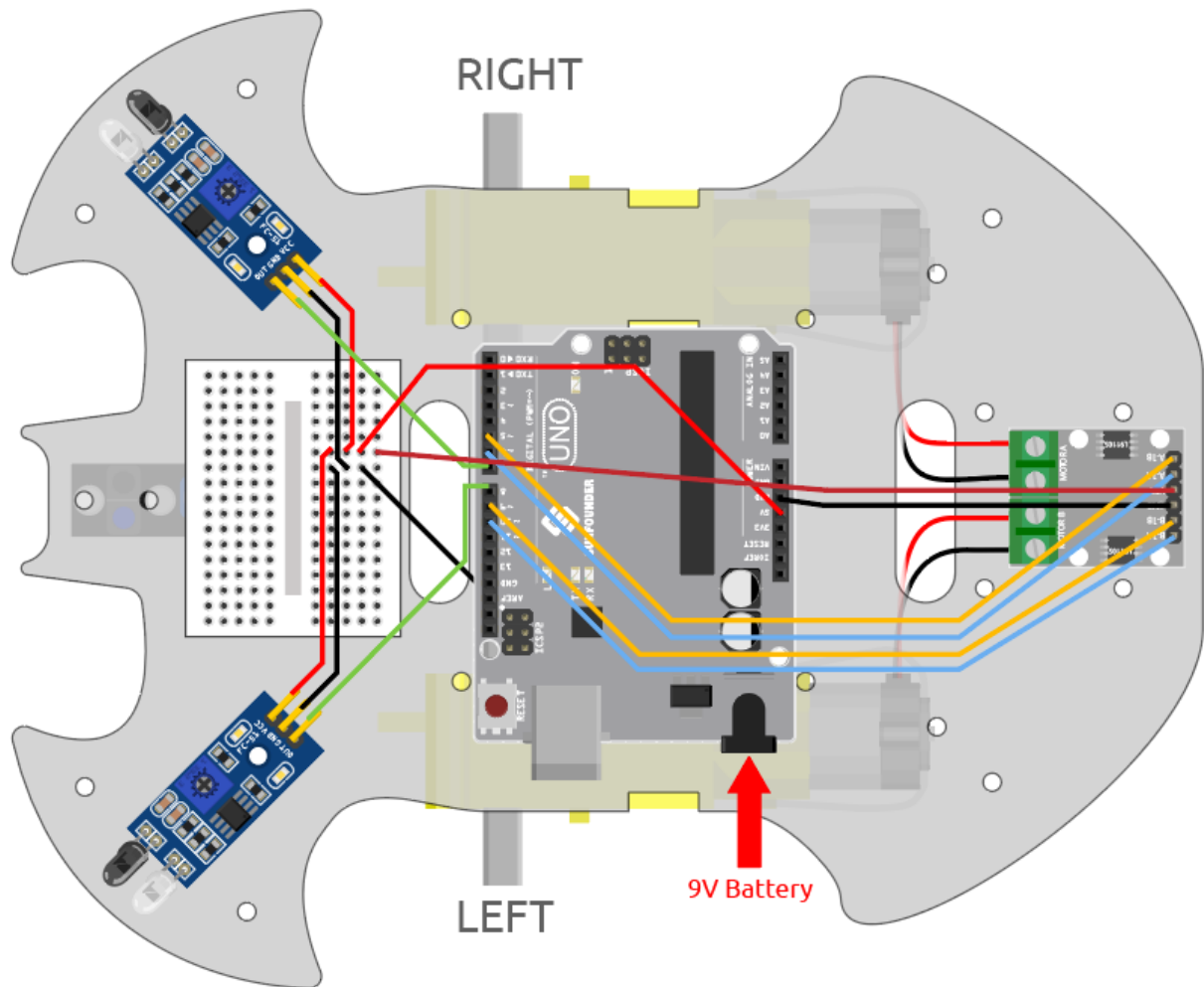
### Câblage

Le module d'évitement d'obstacle est un capteur de proximité infrarouge à distance réglable dont la sortie est normalement haute et basse lorsqu'un obstacle est détecté.

Construisez maintenant le circuit selon le schéma ci-dessous.

Module IR gauche	Carte R3
OUT	8
GND	GND
VCC	5V

Module IR droit	Carte R3
OUT	7
GND	GND
VCC	5V



### Ajuster le module

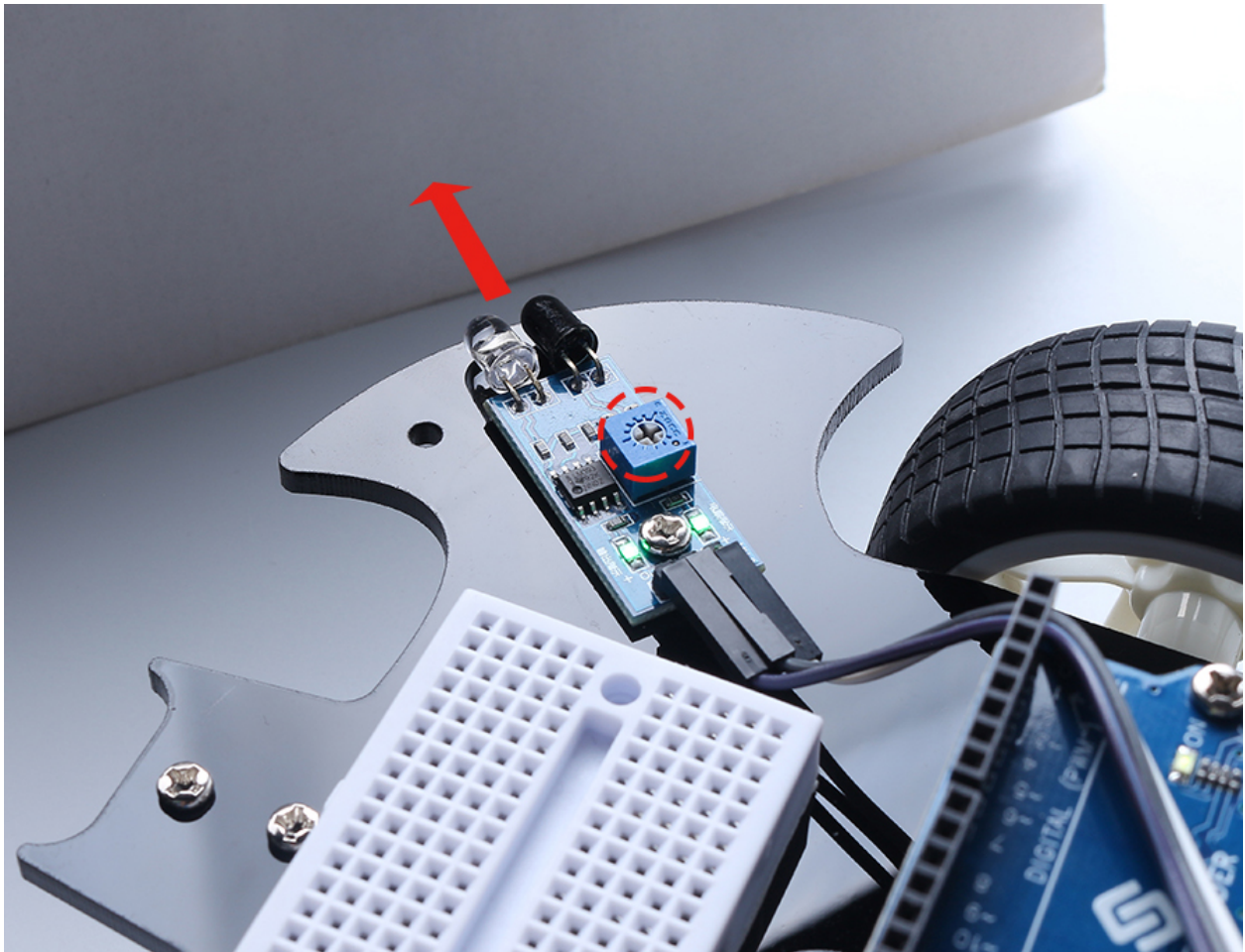
Avant de commencer le projet, vous devez régler la distance de détection du module.

Câblez selon le schéma ci-dessus, alimentez la carte R3 (soit en branchant directement le câble USB, soit en connectant le câble de la batterie 9V), sans télécharger le code.

Placez un carnet ou tout autre objet plat à environ 5 cm devant le module d'évitement d'obstacle IR.

Utilisez ensuite un tournevis pour tourner le potentiomètre sur le module jusqu'à ce que l'indicateur de signal sur le module s'allume, afin d'ajuster sa distance maximale de détection à 5 cm.

Suivez la même méthode pour régler l'autre module infrarouge.



### Code

#### Note :

- Ouvrez le fichier `5.obstacle_avoidance_module.ino` situé dans le dossier `3in1-kit\car_project\5.obstacle_avoidance_module`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via [Arduino Web Editor](#).

La voiture avancera une fois le code téléchargé avec succès. Lorsque le module infrarouge gauche détecte un obstacle, elle reculera vers la gauche ; lorsque le module infrarouge droit détecte un obstacle, elle reculera vers la droite ; si les deux côtés détectent un obstacle, elle reculera directement en arrière.

#### Comment ça fonctionne ?

Ce projet se base sur la valeur des modules d'évitement d'obstacles infrarouges gauche et droit pour faire prendre à la voiture l'action appropriée.

1. Ajoutez la définition de pin pour les 2 modules d'évitement d'obstacles, ici ils sont réglés sur INPUT.

```
...
const int rightIR = 7;
const int leftIR = 8;
```

(suite sur la page suivante)

(suite de la page précédente)

```

void setup() {
  ...

  //IR obstacle
  pinMode(leftIR, INPUT);
  pinMode(rightIR, INPUT);
}

```

2. Lisez les valeurs des modules infrarouges gauche et droit et laissez la voiture prendre l'action correspondante.

```

void loop() {

  int left = digitalRead(leftIR);  // 0: Obstructed  1: Empty
  int right = digitalRead(rightIR);
  int speed = 150;

  if (!left && right) {
    backLeft(speed);
  } else if (left && !right) {
    backRight(speed);
  } else if (!left && !right) {
    moveBackward(speed);
  } else {
    moveForward(speed);
  }
}

```

- Si le module IR gauche est à 0 (obstacle détecté) et le module IR droit est à 1, laissez la voiture reculer à gauche.
- Si le module IR droit est à 0 (obstacle détecté), laissez la voiture reculer à droite.
- Si les 2 modules IR détectent l'obstacle en même temps, la voiture reculera.
- Sinon, la voiture continuera à avancer.

3. À propos de la fonction backLeft().

Lorsque le moteur droit tourne dans le sens antihoraire et que le moteur gauche ne tourne pas, la voiture reculera vers la gauche.

```

void backLeft(int speed) {
  analogWrite(A_1B, speed);
  analogWrite(A_1A, 0);
  analogWrite(B_1B, 0);
  analogWrite(B_1A, 0);
}

```

4. À propos de la fonction backRight().

Lorsque le moteur gauche tourne dans le sens horaire et que le moteur droit ne tourne pas, la voiture reculera vers la droite.

```

void backRight(int speed) {
  analogWrite(A_1B, 0);
  analogWrite(A_1A, 0);
  analogWrite(B_1B, 0);
  analogWrite(B_1A, speed);
}

```



- && : L'opérateur logique ET donne un résultat vrai seulement si les deux opérandes sont vrais.
- ! : L'opérateur logique NON donne un résultat vrai si l'opérande est faux et vice versa.

## 6.7 6. Jouer avec le module ultrasonique

Dans le projet 5. *Jouer avec le module d'évitement d'obstacles*, les 2 modules infrarouges d'évitement d'obstacles sont utilisés pour éviter les obstacles, mais la distance de détection du module d'évitement d'obstacle IR est courte, ce qui peut rendre la voiture trop tardive pour éviter les obstacles.

Dans ce projet, nous utilisons un module ultrasonique pour effectuer une détection à longue distance, afin que la voiture puisse détecter les obstacles à une distance plus éloignée et prendre une décision.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

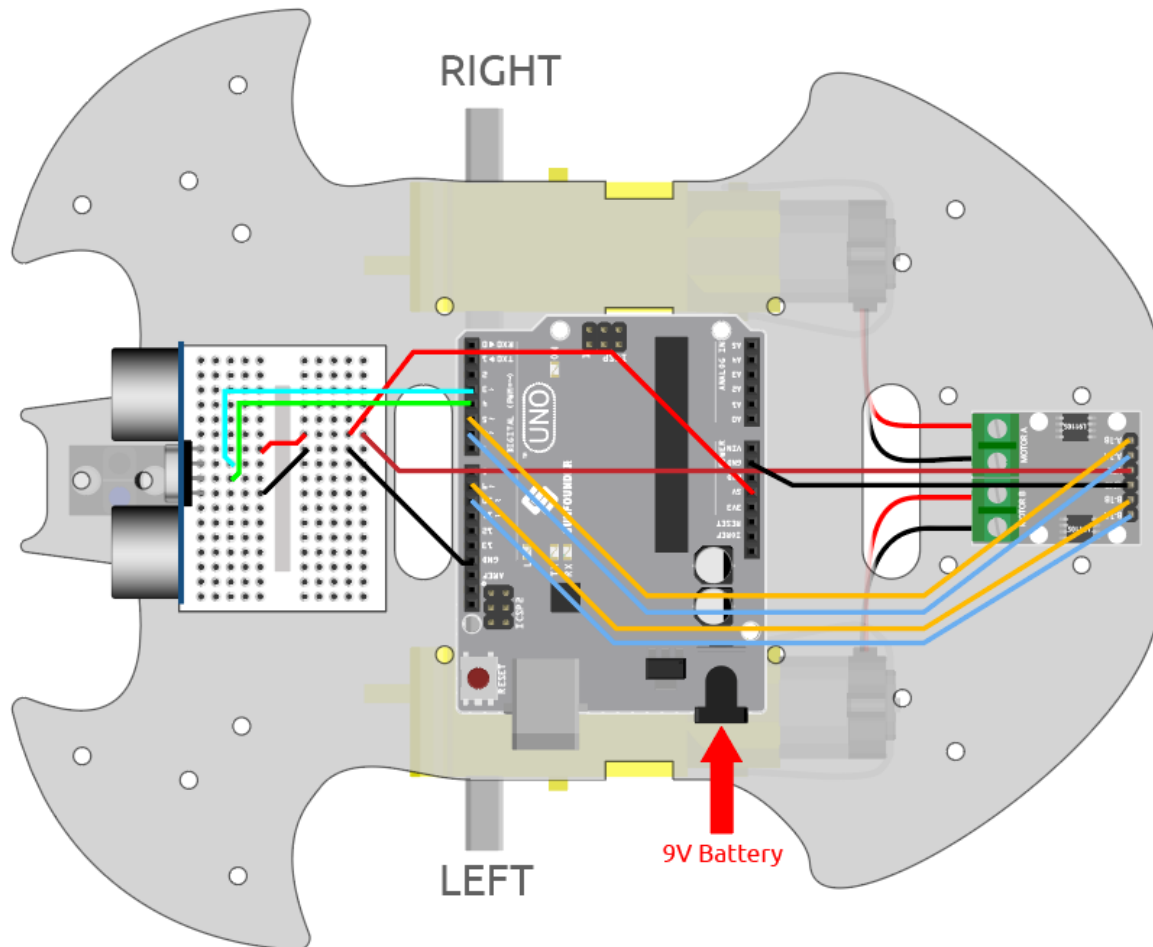
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module Ultrasonique</i>	

### Câblage

Un module de capteur ultrasonique est un instrument qui mesure la distance à un objet à l'aide d'ondes sonores ultrasoniques. Il possède deux sondes. L'une est pour envoyer des ondes ultrasoniques et l'autre pour recevoir les ondes et transformer le temps d'envoi et de réception en une distance, détectant ainsi la distance entre l'appareil et un obstacle.

Construisez maintenant le circuit selon le schéma suivant.

Ultrasonic Module	R3 Board
Vcc	5V
Trig	3
Echo	4
Gnd	GND



## Code

### Note :

- Ouvrez le fichier 6.ultrasonic\_module.ino situé dans le dossier 3in1-kit\car\_project\6.ultrasonic\_module.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via [Arduino Web Editor](#).

Après le téléchargement réussi du code, orientez la voiture vers un mur. Si la distance est trop grande, elle avancera ; si elle est trop proche, elle reculera ; si elle est à une distance sûre, elle s'arrêtera.

### Comment ça fonctionne ?

Ce projet se base sur la distance lue par le module ultrasonique pour faire bouger la voiture en conséquence.

1. Ajoutez la définition de pin pour le module ultrasonique, `trigPin` est utilisé pour transmettre l'onde ultrasonique, réglez-le sur OUTPUT ; `echoPin` est réglé sur INPUT pour recevoir l'onde ultrasonique.

```
...
const int trigPin = 3;
const int echoPin = 4;

void setup() {
```

(suite sur la page suivante)



(suite de la page précédente)

```
...

//ultrasonic
pinMode(echoPin, INPUT);
pinMode(trigPin, OUTPUT);
}
```

2. Lisez d'abord la valeur de distance obtenue à partir du module ultrasonique, si la distance est supérieure à 25, laissez la voiture avancer; si la distance est entre 2-10cm, laissez la voiture reculer, sinon (entre 10~25) arrêtez.

```
void loop() {
    float distance = readSensorData();
    if (distance > 25) {
        moveForward(200);
    }
    else if (distance < 10 && distance > 2) {
        moveBackward(200);
    } else {
        stopMove();
    }
}
```

3. À propos de la fonction readSensorData().

L'émetteur du module ultrasonique transmet un signal carré de 10us toutes les 2us, et le récepteur reçoit un signal de niveau haut s'il y a un obstacle dans la portée. Utilisez la fonction pulseIn() pour enregistrer le temps de l'envoi à la réception, divisez par la vitesse du son 340m/s, puis divisez par 2, le résultat est la distance entre ce module et l'obstacle en unités : cm.

```
float readSensorData() {
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);
    float distance = pulseIn(echoPin, HIGH) / 58.00; //Equivalent to (340m/
    ↪ s*1us)/2
    return distance;
}
```

— pulseIn(pin, value)

— pin : le numéro du pin Arduino sur lequel vous voulez lire l'impulsion. Types de données autorisés : int.

— value : type d'impulsion à lire : soit HIGH, soit LOW. Types de données autorisés : int.

Lit une impulsion (soit HIGH, soit LOW) sur un pin. Par exemple, si la valeur est HIGH, pulseIn() attend que le pin passe de LOW à HIGH, commence à chronométrer, puis attend que le pin redevienne LOW et arrête le chronométrage.

## 6.8 7. Suivez Votre Main

Considérez cette voiture comme votre animal de compagnie ici, et lorsqu'elle vous verra lui faire signe, elle viendra en courant vers vous.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module Ultrasonique</i>	
<i>Module d'Évitement d'Obstacle</i>	

### Câblage

Connectez le module ultrasonique et les 2 modules d'évitement d'obstacles IR en même temps.

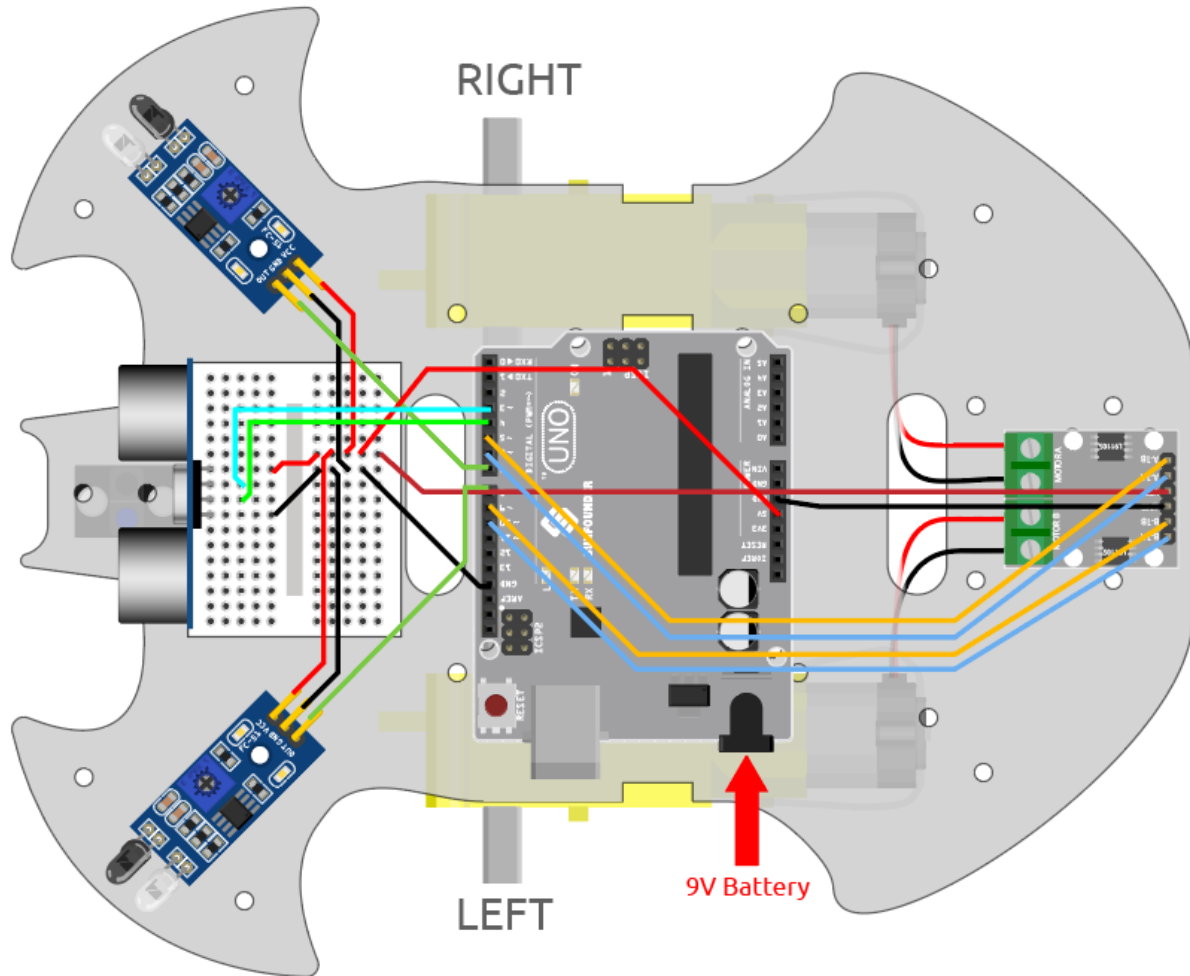
Connectez l'ultrasonique à la carte R3 comme suit.

Module ultrasonique	Carte R3
Vcc	5V
Trig	3
Echo	4
Gnd	GND

Le câblage des 2 modules d'évitement d'obstacles IR à la carte R3 est le suivant.

Left IR Module	R3 Board
OUT	8
GND	GND
VCC	5V

Right IR Module	R3 Board
OUT	7
GND	GND
VCC	5V



### Code

#### Note :

- Ouvrez le fichier `7.follow_your_hand.ino` situé dans le dossier `3in1-kit\car_project\7.follow_your_hand`.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via [Arduino Web Editor](#).

Placez la voiture sur le sol après le téléchargement réussi du code. Placez votre main à environ 5\*10cm devant la voiture, et elle suivra votre main vers l'avant. Si vous placez votre main près du module d'obstacle IR de chaque côté, elle tournera également dans la direction correspondante.

#### Comment ça fonctionne ?

Ce projet est une combinaison des deux projets précédents *6. Jouer avec le module ultrasonique* et *5. Jouer avec le module d'évitement d'obstacles*, mais l'effet implémenté est différent. Les 2 projets précédents détectent un obstacle en arrière, mais ici, il détecte que votre main suivra la direction avant ou tournera. Le déroulement de ce projet est le suivant.

- Lisez la distance détectée par le module ultrasonique et la valeur des deux modules infrarouges.
- Si la distance est de 5~10cm, laissez la voiture se déplacer avec votre main.
- Si le module IR gauche détecte votre main, tournez à gauche.
- Si le module IR droit détecte votre main, tournez à droite.

— Si ni le module infrarouge ni le module ultrasonique ne détectent votre main, laissez la voiture s’arrêter.

```
void loop() {
    float distance = readSensorData();

    int left = digitalRead(leftIR); // 0: Obstructed 1: Empty
    int right = digitalRead(rightIR);
    int speed = 150;

    if (distance>5 && distance<10){
        moveForward(speed);
    }
    if(!left&&right){
        turnLeft(speed);
    }else if(left&&!right){
        turnRight(speed);
    }else{
        stopMove();
    }
}
```

## 6.9 8. Voiture Autonome

Ce projet est une combinaison des deux projets *6. Jouer avec le module ultrasonique* et *5. Jouer avec le module d’évitement d’obstacles*. 2 modules infrarouges d’évitement d’obstacles effectuent une détection à courte distance ou de bord, et les modules ultrasoniques effectuent une détection à longue distance pour confirmer que la voiture ne heurte pas d’obstacle pendant le processus de conduite libre.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d’acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D’ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module Ultrasonique</i>	
<i>Module d’Évitement d’Obstacle</i>	

### Câblage

Connectez le module ultrasonique et les 2 modules d'évitement d'obstacles IR en même temps.

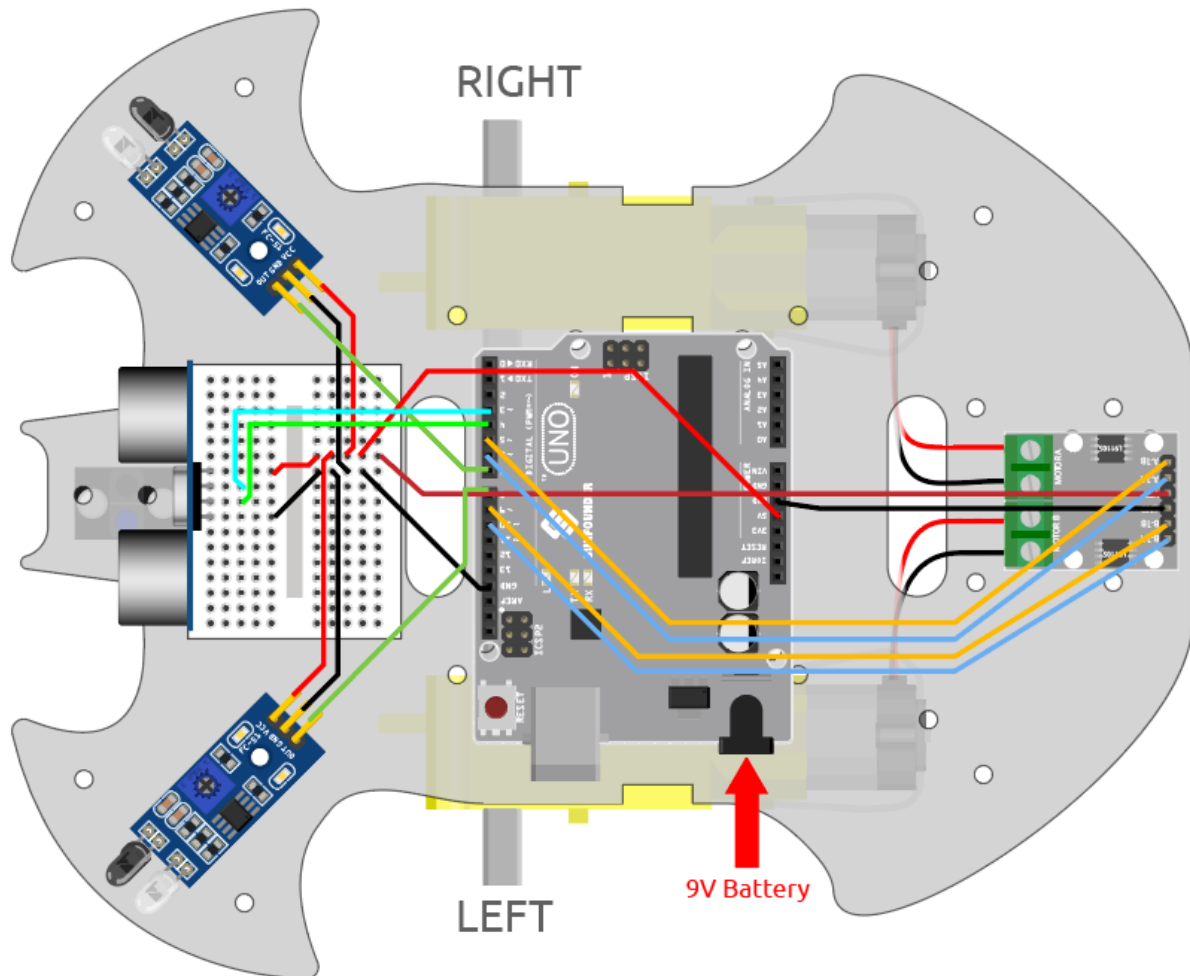
Connectez l'ultrasonique à la carte R3 comme suit.

Module ultrasonique	Carte R3
Vcc	5V
Trig	3
Echo	4
Gnd	GND

Le câblage des 2 modules d'évitement d'obstacles IR à la carte R3 est le suivant.

Left IR Module	R3 Board
OUT	8
GND	GND
VCC	5V

Right IR Module	R3 Board
OUT	7
GND	GND
VCC	5V



## Code

### Note :

- Ouvrez le fichier 8.self\_driving\_car.ino situé dans le dossier 3in1-kit\car\_project\8.self\_driving\_car.
- Ou copiez ce code dans **Arduino IDE**.
- Ou téléchargez le code via [Arduino Web Editor](#).

Une fois le code téléchargé avec succès, la voiture se déplacera librement. Lorsque le module d'obstruction IR des deux côtés détecte un obstacle, elle se déplacera dans la direction opposée pour une évacuation d'urgence ; si un obstacle se trouve à 2~10cm directement devant la voiture, elle reculera vers la gauche, ajustera sa direction, puis avancera.

### Comment ça fonctionne ?

Le déroulement de ce projet est le suivant.

- Lisez d'abord la valeur du module d'évitement d'obstacles IR gauche et droit.
- Si le module IR gauche est à 0 (obstacle détecté), le module IR droit est à 1, laissez la voiture reculer à gauche.
- Si le module IR droit est à 0 (obstacle détecté), laissez la voiture reculer à droite.
- Si les 2 modules IR détectent l'obstacle en même temps, la voiture reculera.
- Sinon, lisez la distance détectée par le module ultrasonique.
- Si la distance est supérieure à 50cm, laissez la voiture avancer.
- Si la distance est entre 2-10cm, laissez la voiture reculer avant de tourner.

— Si la distance est entre 10-50cm, laissez la voiture avancer à faible vitesse.

```
void loop() {

    int left = digitalRead(leftIR);  // 0: Obstructed  1: Empty
    int right = digitalRead(rightIR);

    if (!left && right) {
        backLeft(150);
    } else if (left && !right) {
        backRight(150);
    } else if (!left && !right) {
        moveBackward(150);
    } else {
        float distance = readSensorData();
        Serial.println(distance);
        if (distance > 50) { // Safe
            moveForward(200);
        } else if (distance < 10 && distance > 2) { // Attention
            moveBackward(200);
            delay(1000);
            backLeft(150);
            delay(500);
        } else {
            moveForward(150);
        }
    }
}
```

## 6.10 9. Télécommande

Ce kit est livré avec un récepteur IR, qui vous permet d'utiliser une télécommande IR pour contrôler le mouvement de la voiture.

### Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

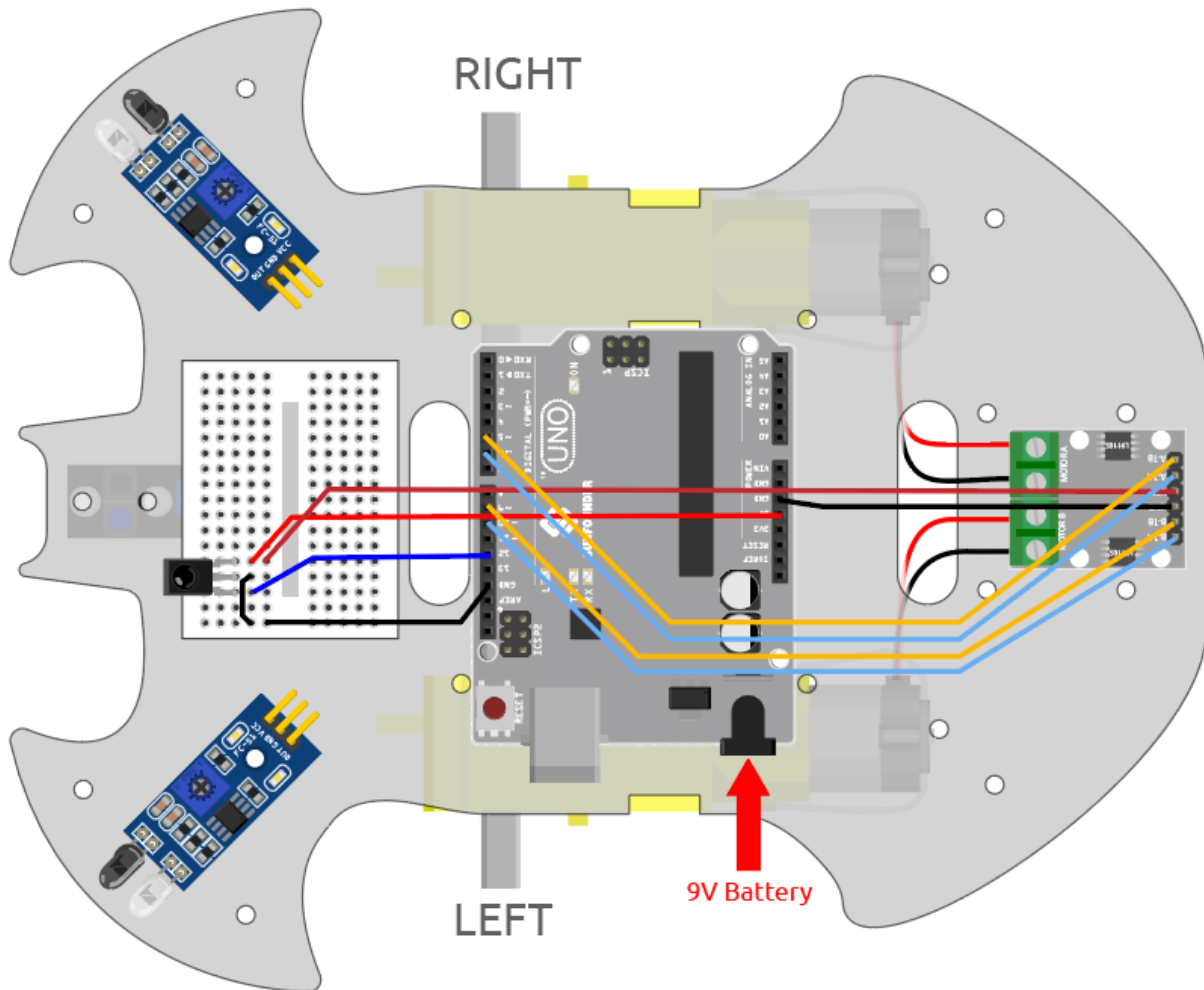
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>LED</i>	
<i>Récepteur IR</i>	-

## Câblage

Construisez maintenant le circuit selon le schéma ci-dessous.

Récepteur IR	Carte R3
OUT	12
GND	GND
VCC	5V

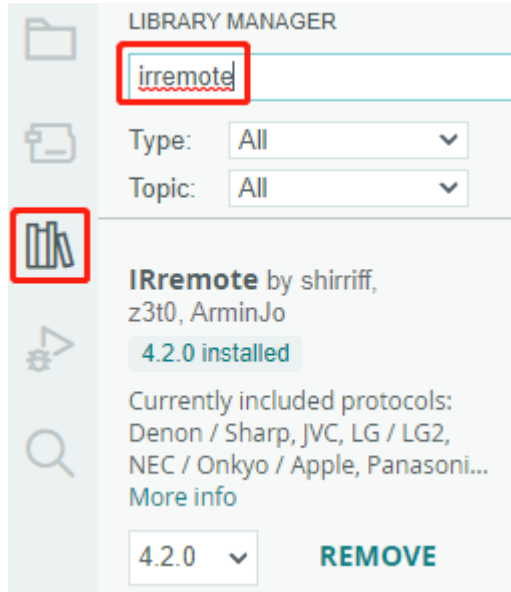


## Code



**Note :**

- Ouvrez le fichier `9.remote_control.ino` situé dans le dossier `3in1-kit\car_project\9.remote_control`.
- Ou copiez ce code dans **Arduino IDE**.
- La bibliothèque **IRremote** est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après le téléchargement réussi du code, appuyez sur le bouton de la télécommande, la LED clignotera une fois pour indiquer que le signal a été reçu, et la voiture se déplacera selon le bouton que vous avez pressé. Vous pouvez appuyer sur les touches suivantes pour contrôler la voiture.

- **+** : Accélérer
- **-** : Ralentir
- **1** : Avancer à gauche
- **2** : Avancer
- **3** : Avancer à droite
- **4** : Tourner à gauche
- **6** : Tourner à droite
- **7** : Reculer à gauche
- **8** : Reculer
- **9** : Reculer à droite

**Comment ça fonctionne ?**

L'effet de ce projet est de faire bouger la voiture en lisant la valeur clé de la télécommande IR. De plus, la LED sur la broche 13 clignotera pour indiquer la réception réussie du signal IR.

1. Importez la bibliothèque **IRremote**, vous pouvez l'installer depuis le **Library Manager**.

```
#include <IRremote.h>

const int IR_RECEIVE_PIN = 12; // Define the pin number for the IR Sensor
```

2. Initialise la communication série à un débit de 9600 bauds. Initialise le récepteur IR sur la broche spécifiée (`IR_RECEIVE_PIN`) et active le retour LED (si applicable).

```

...

void setup() {

    ...
    //IR remote
    IrReceiver.begin(IR_RECEIVE_PIN, ENABLE_LED_FEEDBACK); // Start the
    ↪receiver
    Serial.println("REMOTE CONTROL START");

}

```

3. Lorsque vous appuyez sur les touches de la télécommande, le récepteur infrarouge saura quelle touche est pressée, puis la voiture se déplacera en fonction de la valeur clé correspondante.

```

void loop() {

    if (IrReceiver.decode()) {
        // Serial.println(results.value, HEX);
        String key = decodeKeyValue(IrReceiver.decodedIRData.command);
        if (key != "ERROR") {
            Serial.println(key);

            if (key == "+") {
                speed += 50;
            } else if (key == "-") {
                speed -= 50;
            } else if (key == "2") {
                moveForward(speed);
                delay(1000);
            }
            ...
        }
        IrReceiver.resume(); // Enable receiving of the next value
    }

}

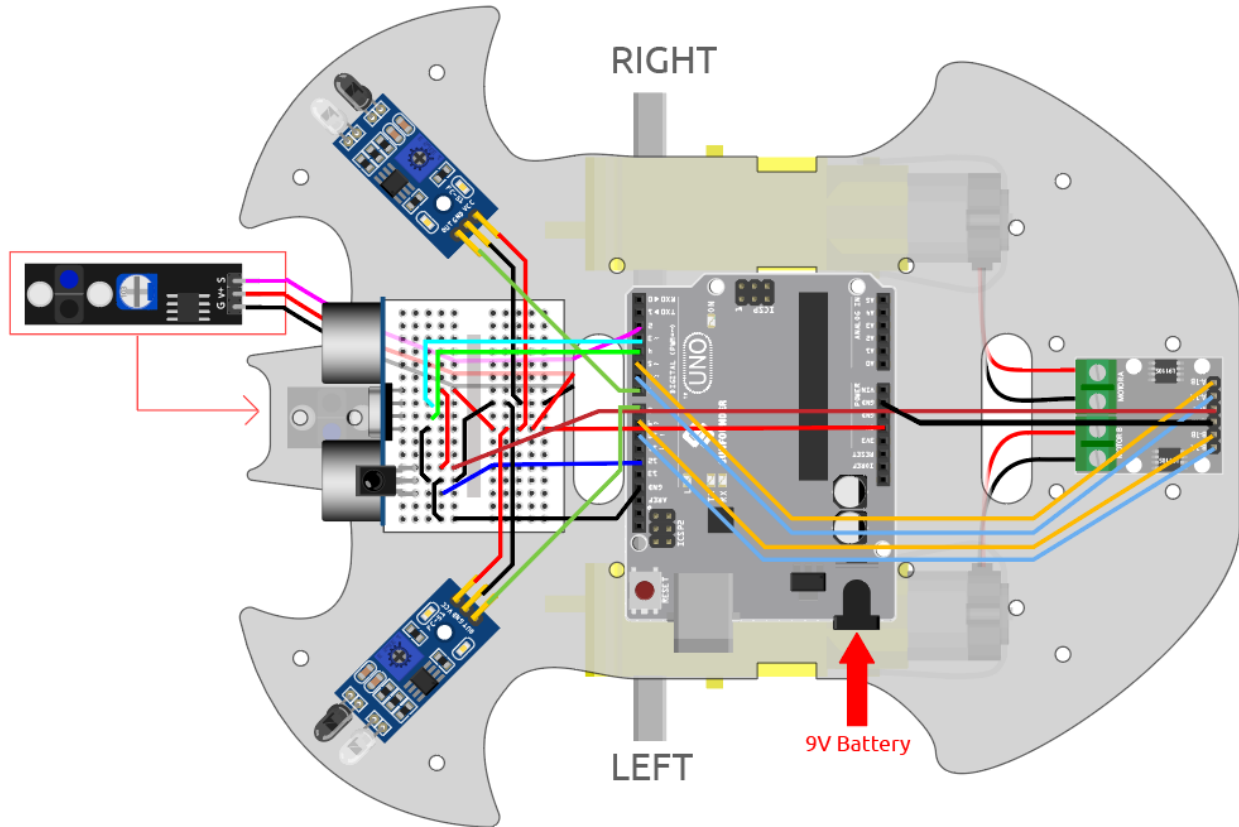
```

- Vérifie si un signal IR est reçu et décodé avec succès.
- Décode la commande IR et la stocke dans key à l'aide d'une fonction decodeKeyValue() personnalisée.
- Vérifie si la valeur décodée n'est pas une erreur.
- Imprime la valeur IR décodée sur le moniteur série.
- Reprend la réception du signal IR pour le signal suivant.

## 6.11 10. Démarrage en Un Clic

Dans ce projet, nous avons intégré les projets précédents - suivi de ligne, suivi, évitement d'obstacles, conduite autonome, etc. Ils peuvent être changés par les boutons de la télécommande, vous permettant ainsi de démarrer la voiture et d'expérimenter toutes les fonctions en une seule fois.

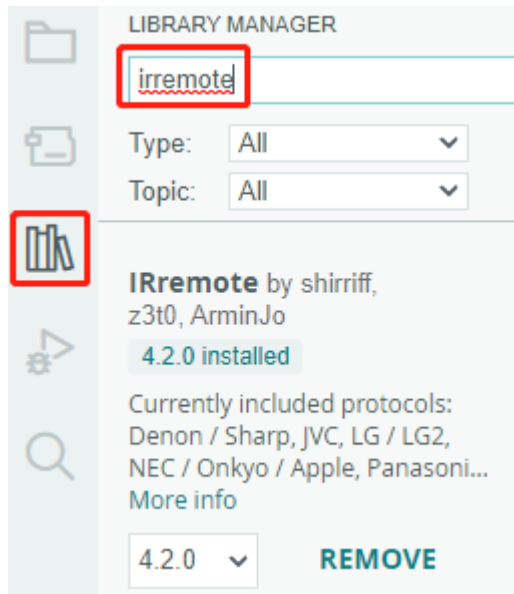
### Câblage



### Code

#### Note :

- Ouvrez le fichier `10.one_touch_start.ino` situé dans le dossier `3in1-kit\car_project\10.one_touch_start`.
- Ou copiez ce code dans **Arduino IDE**.
- La bibliothèque `IRremote` est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



Après le téléchargement réussi du code, la LED clignotera rapidement 3 fois pour chaque signal reçu de la télécommande par le récepteur IR. Vous pouvez appuyer sur les touches suivantes pour faire fonctionner le chariot.

- **+** : Accélérer
- **-** : Ralentir
- **1** : Avancer vers la gauche
- **2** : Avancer
- **3** : Avancer vers la droite.
- **4** : Tourner à gauche
- **6** : Tourner à droite
- **7** : Reculer vers la gauche
- **8** : Reculer.
- **9** : Reculer vers la droite
- **CYCLE** : Suivre la ligne
- **U/SD** : Conduite autonome
- **|** : Évitement d'obstacles avec module ultrasonique
- **|** : Évitement d'obstacles avec module IR Obstacle
- **EQ** : Suivre votre main
- **0** : Arrêter

## 6.12 11. Calibration de la Vitesse

Lors de la mise en mouvement de la voiture, vous pourriez constater que la voiture ne se déplace pas en ligne droite. Cela est dû au fait que les deux moteurs peuvent ne pas avoir la même vitesse en sortie d'usine. Mais nous pouvons écrire un décalage aux deux moteurs pour faire converger leurs vitesses de rotation.

Dans ce projet, nous apprendrons à stocker le décalage dans **EEPROM**. L'intérêt est que, après chaque calibration, tous les projets peuvent obtenir directement la valeur de décalage depuis l'EEPROM, afin que la voiture puisse avancer en ligne droite de manière fluide.

### Câblage

Ce projet a le même câblage que 2. *Mouvement par Code*.

### Comment jouer ?

1. Ouvrez le fichier `11.speed_calibration.ino` situé dans le dossier `3in1-kit\car_project\11.speed_calibration`. Ou copiez ce code dans **Arduino IDE**.
  2. Après le téléchargement réussi du code, connectez la voiture avec une batterie 9V, posez-la sur le sol et laissez-la avancer pour voir de quel côté elle se décale.
- Si la voiture se déplace vers l'avant gauche, cela signifie que la vitesse du moteur droit est trop rapide et doit être réduite.

```
EEPROM.write(1, 100) // 1 means the right motor, 100 means 100% speed, can be
→ set to 90, 95, etc., depending on the actual situation.
```

- Si la voiture se déplace vers la droite, cela signifie que la vitesse du moteur gauche est trop rapide et doit être réduite.

```
EEPROM.write(0, 100) // 0 means the left motor, 100 means the speed is 100%,
→ can be set to 90, 95, etc., depending on the actual situation. 3.
```

3. Après avoir modifié le code, téléchargez le code sur la carte R3 pour voir l'effet. Répétez les étapes ci-dessus jusqu'à ce que la voiture soit presque droite.
4. Ce décalage sera enregistré dans **EEPROM**, vous n'avez besoin de lire ce décalage que lorsque vous l'utilisez dans d'autres projets, prenez *5. Jouer avec le module d'évitement d'obstacles* comme exemple.

```
#include <EEPROM.h>

float leftOffset = 1.0;
float rightOffset = 1.0;

const int A_1B = 5;
const int A_1A = 6;
const int B_1B = 9;
const int B_1A = 10;

const int rightIR = 7;
const int leftIR = 8;

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

  //motor
  pinMode(A_1B, OUTPUT);
  pinMode(A_1A, OUTPUT);
  pinMode(B_1B, OUTPUT);
  pinMode(B_1A, OUTPUT);

  //IR obstacle
  pinMode(leftIR, INPUT);
  pinMode(rightIR, INPUT);

  leftOffset = EEPROM.read(0) * 0.01; //read the offset of the left motor
  rightOffset = EEPROM.read(1) * 0.01; //read the offset of the right motor
}
```

(suite sur la page suivante)

```
void loop() {

    int left = digitalRead(leftIR);  // 0: Obstructed  1: Empty
    int right = digitalRead(rightIR);
    int speed = 150;

    if (!left && right) {
        backLeft(speed);
    } else if (left && !right) {
        backRight(speed);
    } else if (!left && !right) {
        moveBackward(speed);
    } else {
        moveForward(speed);
    }
}

void moveForward(int speed) {
    analogWrite(A_1B, 0);
    analogWrite(A_1A, int(speed * leftOffset));
    analogWrite(B_1B, int(speed * rightOffset));
    analogWrite(B_1A, 0);
}

void moveBackward(int speed) {
    analogWrite(A_1B, speed);
    analogWrite(A_1A, 0);
    analogWrite(B_1B, 0);
    analogWrite(B_1A, speed);
}

void backLeft(int speed) {
    analogWrite(A_1B, speed);
    analogWrite(A_1A, 0);
    analogWrite(B_1B, 0);
    analogWrite(B_1A, 0);
}

void backRight(int speed) {
    analogWrite(A_1B, 0);
    analogWrite(A_1A, 0);
    analogWrite(B_1B, 0);
    analogWrite(B_1A, speed);
}
```

## CHAPITRE 7

---

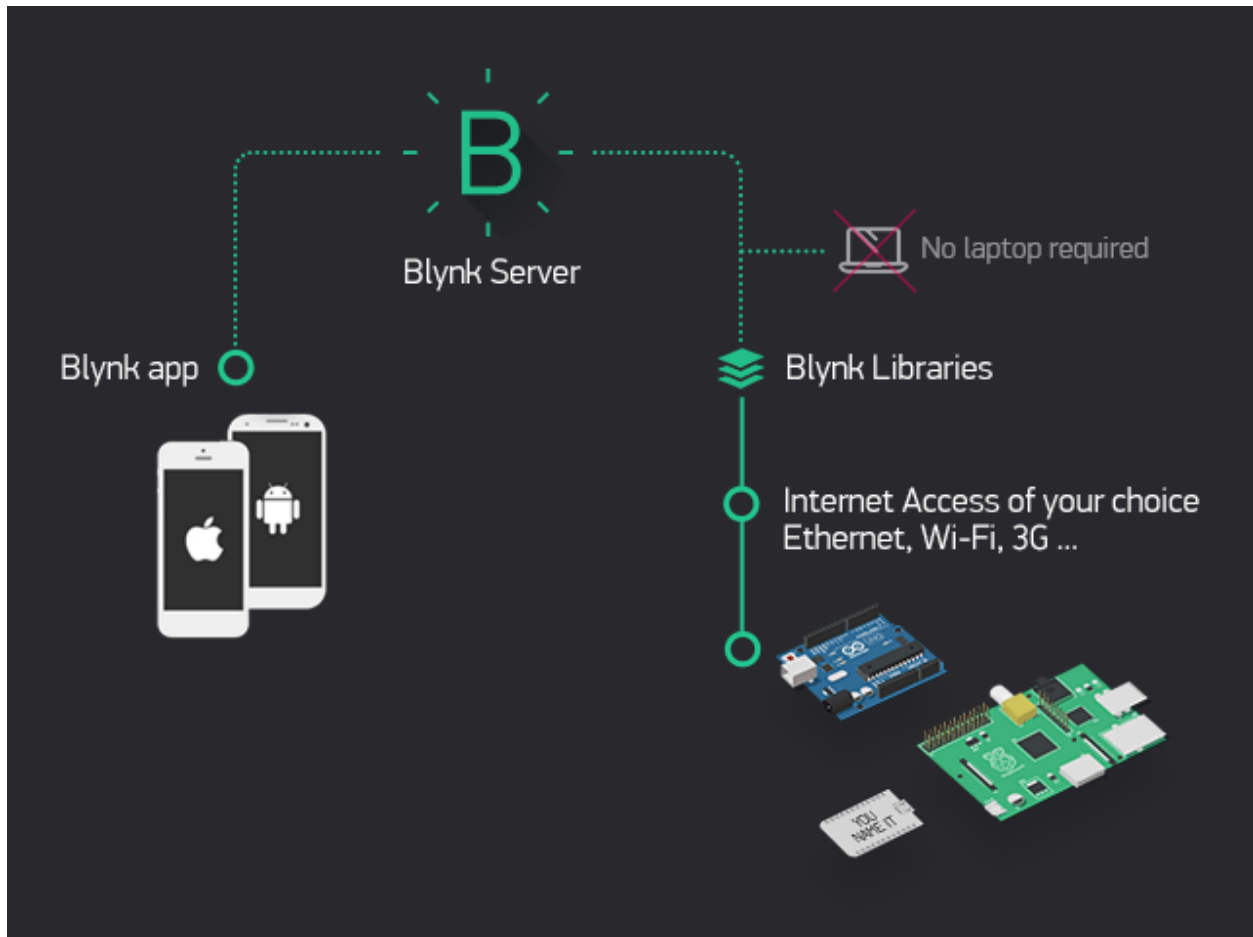
### Projets IoT

---

Ce kit est équipé du module Wifi ESP8266 qui permet à Arduino de se connecter à Internet pour des expériences IoT.

Ici, nous vous guiderons sur comment faire en sorte qu'Arduino se connecte à la plateforme [BLYNK](#) avec l'aide du module Wifi ESP8266 pour réaliser des projets IoT intéressants. Vous pouvez également utiliser l'application Blynk sur votre téléphone portable pour contrôler la voiture intelligente.

Blynk est une suite complète de logiciels nécessaires pour prototyper, déployer et gérer à distance des dispositifs électroniques connectés à n'importe quelle échelle : des projets IoT personnels à des millions de produits connectés commerciaux. Avec Blynk, n'importe qui peut connecter son matériel au cloud et créer des applications iOS, Android et Web sans code pour analyser en temps réel et historiquement les données provenant des dispositifs, les contrôler à distance de n'importe où dans le monde, recevoir des notifications importantes, et bien plus encore...



## 7.1 1. Démarrer avec Blynk

Pour que la carte R3 communique avec Blynk, une certaine configuration est nécessaire lors de la première utilisation de Blynk.

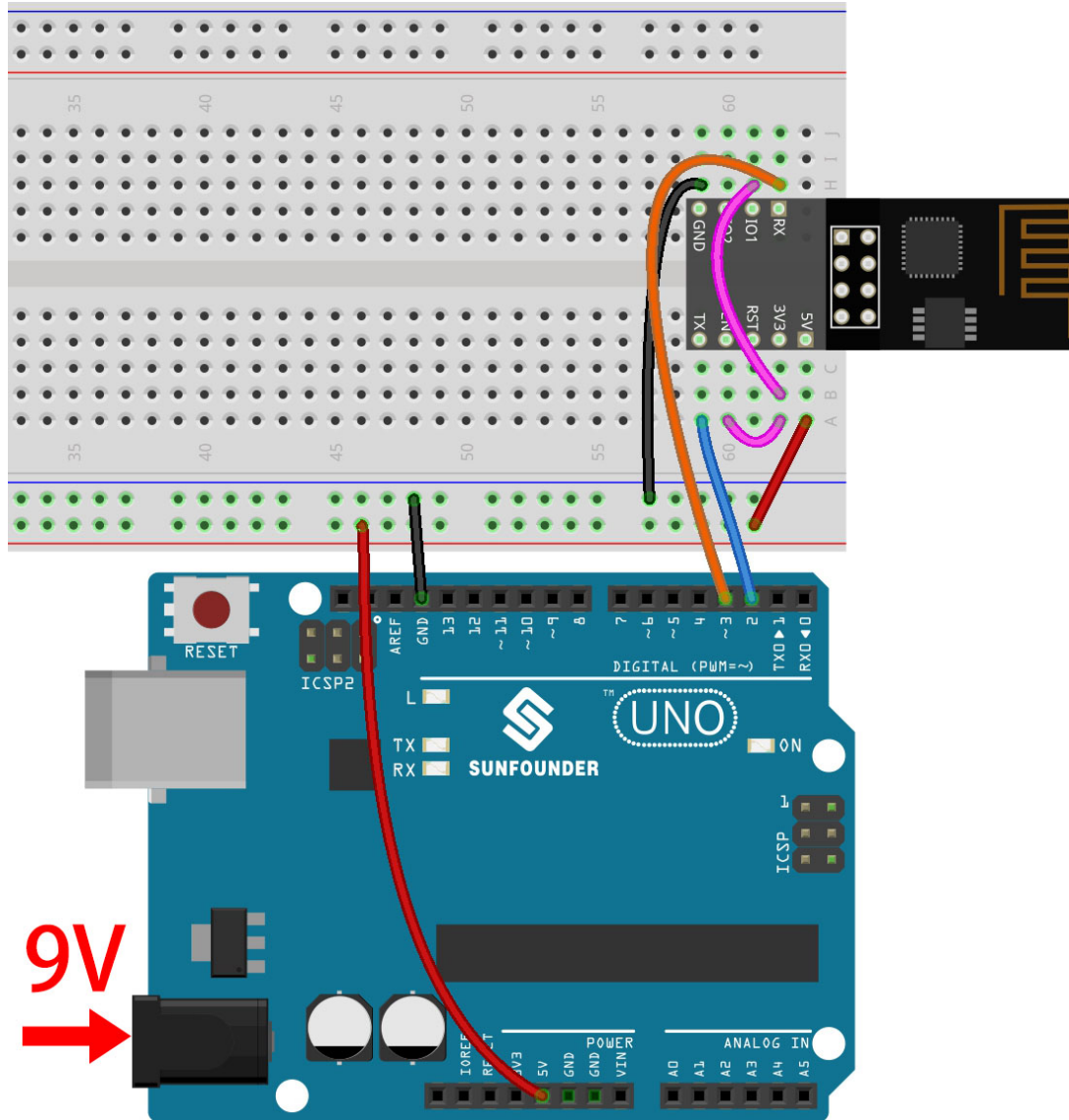
Suivez les étapes ci-dessous, et notez que vous devez les faire dans l'ordre et ne pas sauter de chapitres.



### 7.1.1 1.1 Configuration de l'ESP8266

Le module ESP8266 fourni avec le kit est déjà préchargé avec le firmware AT, mais vous devez tout de même modifier sa configuration en suivant les étapes ci-dessous.

1. Construisez le circuit.



2. Ouvrez le fichier 1.set\_software\_serial.ino situé dans le dossier 3in1-kit\iot\_project\1.set\_software\_serial. Ou copiez ce code dans **Arduino IDE**.

```
#include <SoftwareSerial.h>
SoftwareSerial espSerial(2, 3); //Rx,Tx

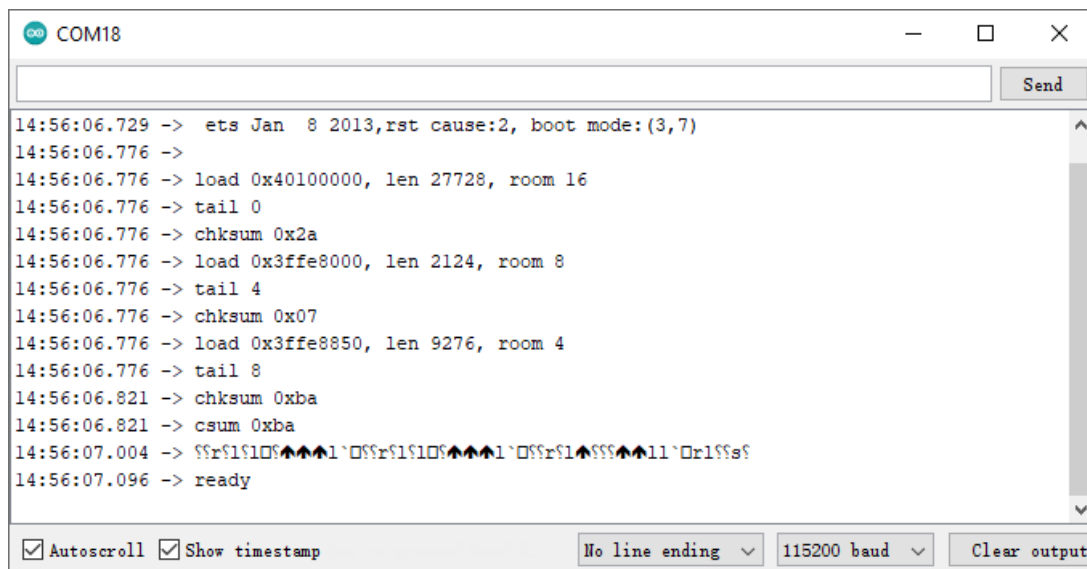
void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  espSerial.begin(115200);
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
void loop() {
  if (espSerial.available()) {
    Serial.write(espSerial.read());
  }
  if (Serial.available()) {
    espSerial.write(Serial.read());
  }
}
```

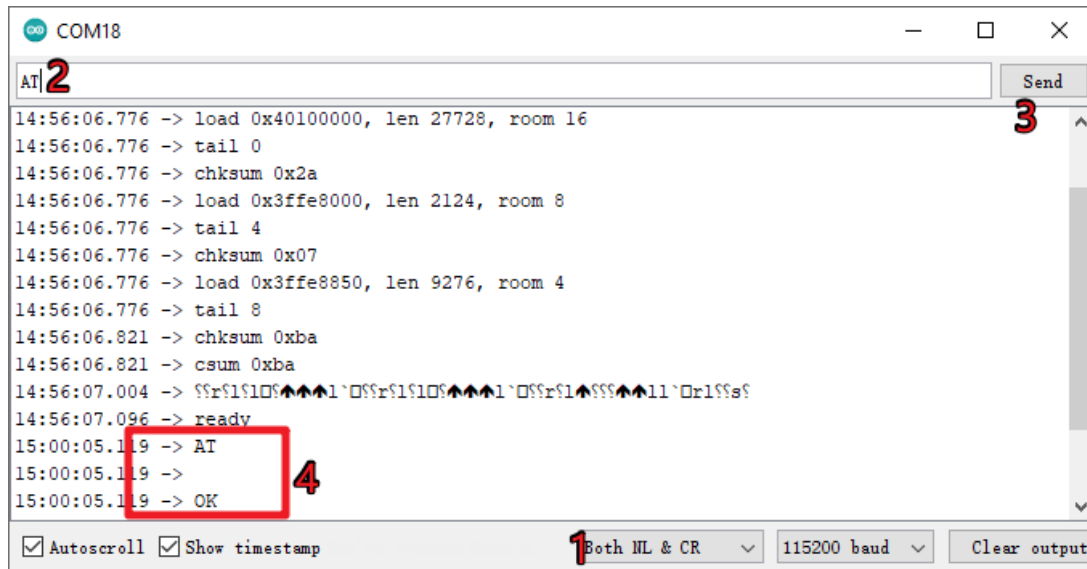
3. Cliquez sur l'icône de la loupe (Moniteur Série) dans le coin supérieur droit et réglez le débit en bauds sur **115200**. (Vous pourriez avoir des informations imprimées comme moi, ou pas, cela n'a pas d'importance, passez simplement à l'étape suivante.)



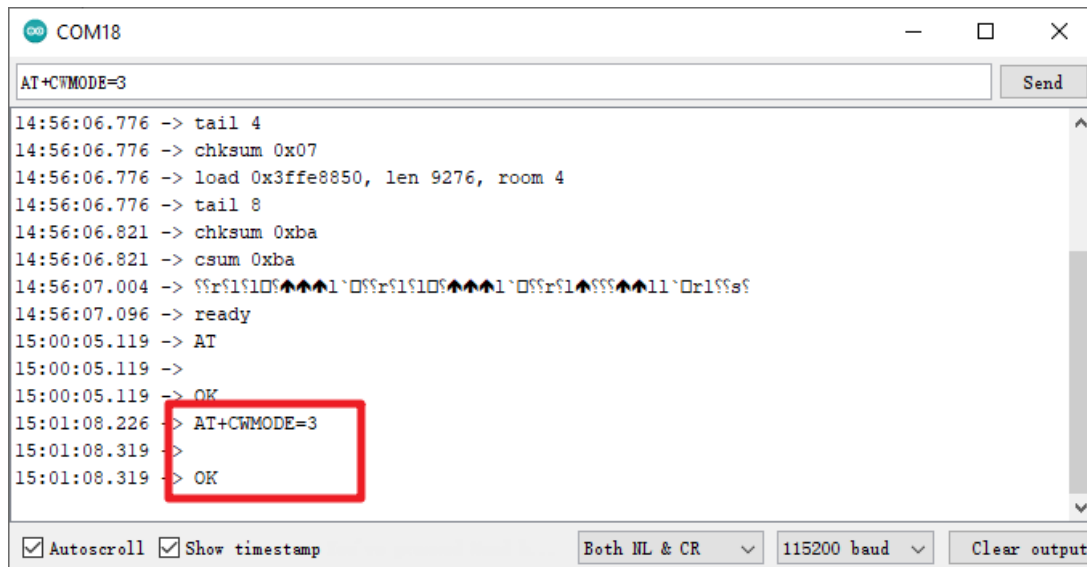
### Avertissement :

- Si ready n'apparaît pas, vous pouvez essayer de réinitialiser le module ESP8266 (connecter RST à GND) et rouvrir le Moniteur Série.
- De plus, si le résultat est OK, vous devrez peut-être reprogrammer le firmware, veuillez vous référer à [Comment re-graver le firmware pour le module ESP8266 ?](#) pour plus de détails. Si vous ne parvenez toujours pas à résoudre le problème, veuillez prendre une capture d'écran du moniteur série et l'envoyer à [sevice@sunfounder.com](mailto:sevice@sunfounder.com), nous vous aiderons à résoudre le problème dès que possible.

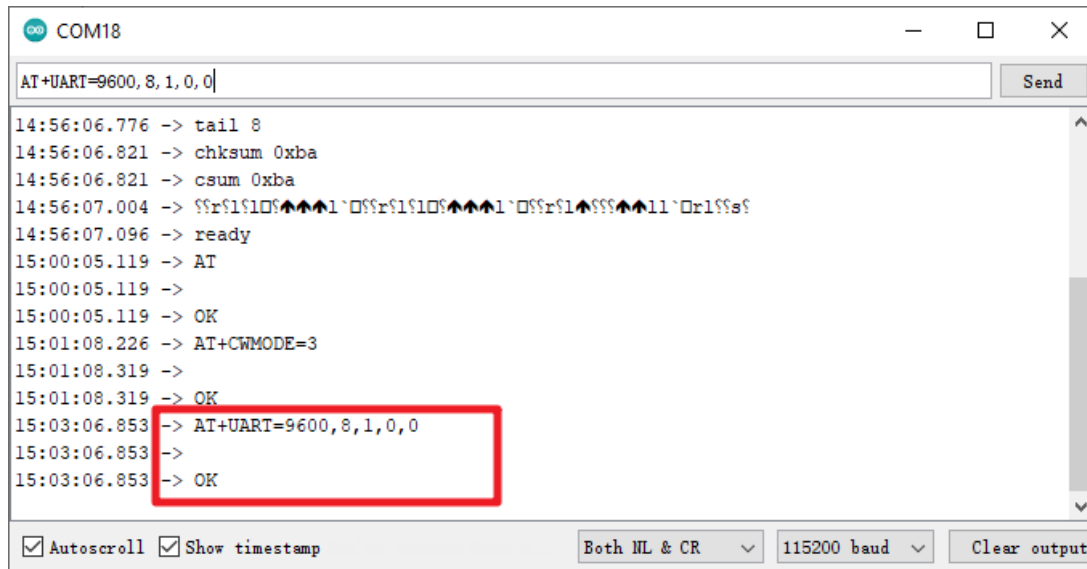
4. Cliquez sur **NEWLINE DROPDOWN BOX**, sélectionnez both NL & CR dans les options déroulantes, entrez AT, si cela retourne OK, cela signifie que l'ESP8266 a établi une connexion réussie avec la carte R3.



5. Entrez AT+CWMODE=3 et le mode géré sera changé en **Station et AP** coexistant.

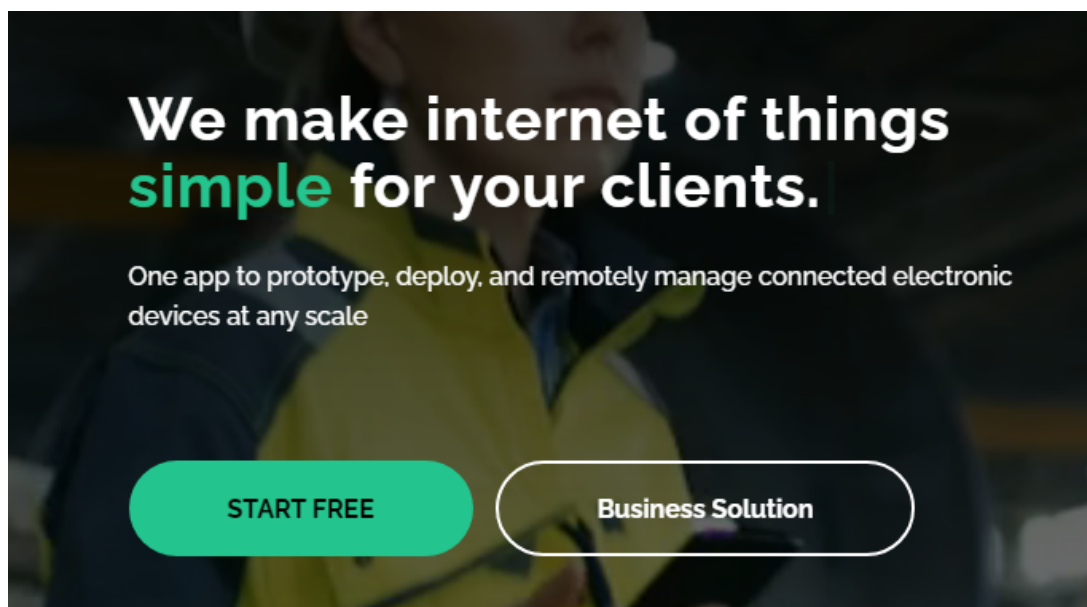


6. Afin d'utiliser le serial logiciel plus tard, vous devez entrer `AT+UART=9600,8,1,0,0` pour modifier le débit en bauds de l'ESP8266 à 9600.




### 7.1.2 1.2 Configuration de Blynk

1. Allez sur [BLYNK](#) et cliquez sur **START FREE**.



2. Remplissez votre adresse e-mail pour enregistrer un compte.



## Sign Up

Welcome! Fill in your email address and we will send an account activation link.

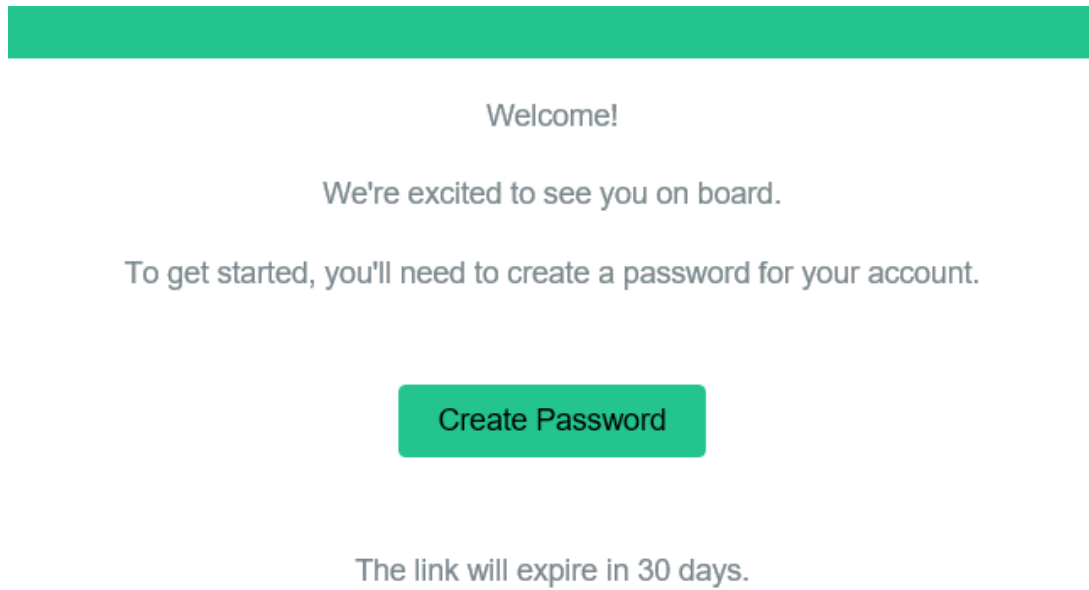
EMAIL

☐ I agree to [Terms and Conditions](#) and accept [Privacy Policy](#)

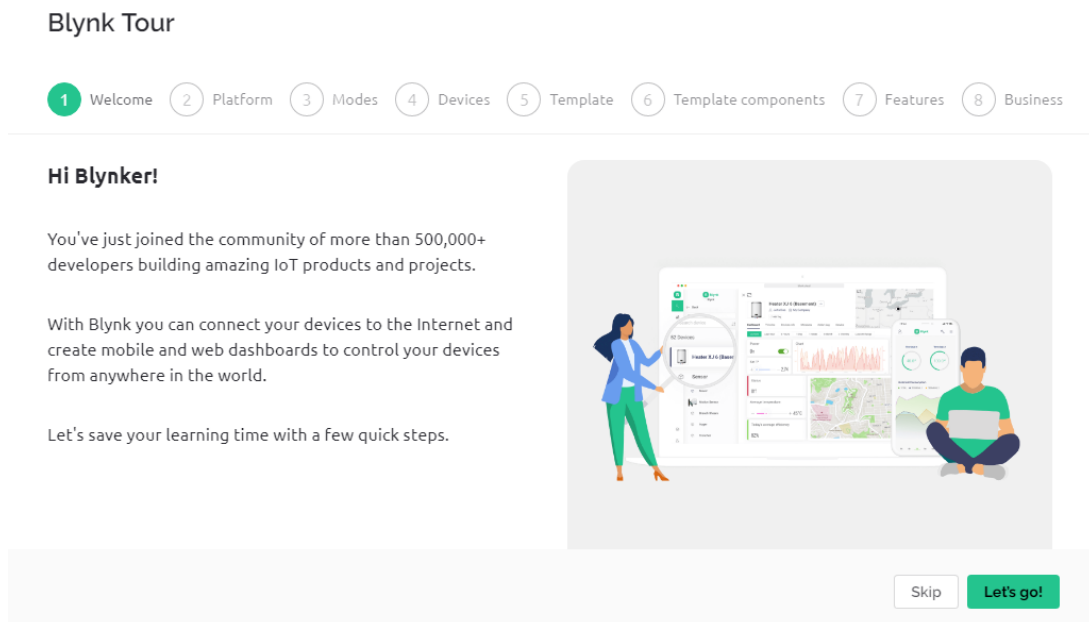
[Sign Up](#)

[Back to Login](#)

3. Rendez-vous sur votre adresse e-mail pour compléter l'enregistrement de votre compte.



4. Par la suite, **Blynk Tour** apparaîtra et vous pourrez le lire pour apprendre les informations de base sur Blynk.



5. Ensuite, nous devons créer un modèle et un appareil avec ce **Quick Start**, cliquez sur **Let's go**.

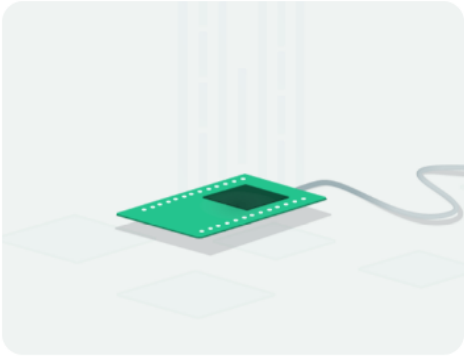
## Quickstart

This is a step by step guide to get your first device online and start controlling it from anywhere in the world in **less than 5 minutes**

**What you will need:**

- Supported hardware. Check the full list of supported hardware [here](#).
- IDE. You can use Arduino IDE or PlatformIO or any other editor.
- Blynk Library
- It will be beneficial if you already know how to upload code to your hardware.

CancelLet's go!



6. Sélectionnez le matériel et le type de connexion.

## Quickstart

1

 Hardware — 

2

 IDE — 

3

 Blynk Library — 

4

 Code — 

5

 Device activation

**Which hardware are you using?**

We will help you prepare the code for you board

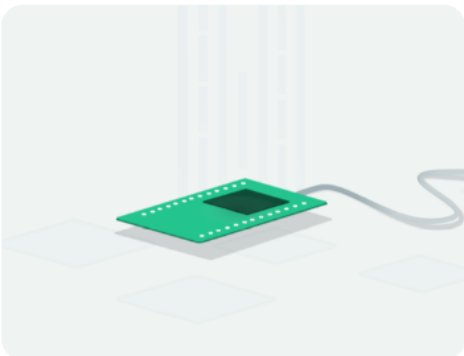
ESP8266

**What is your device connectivity type**

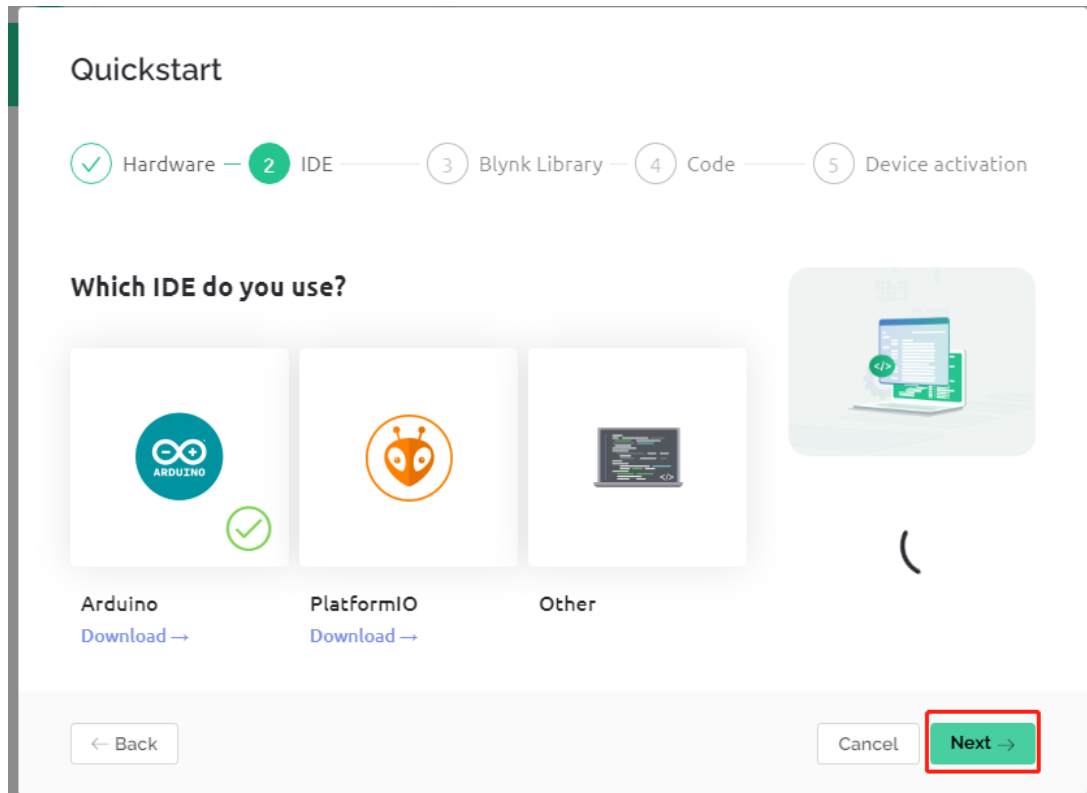
Blynk supports various connection types (BLE is not supported yet).

WiFi

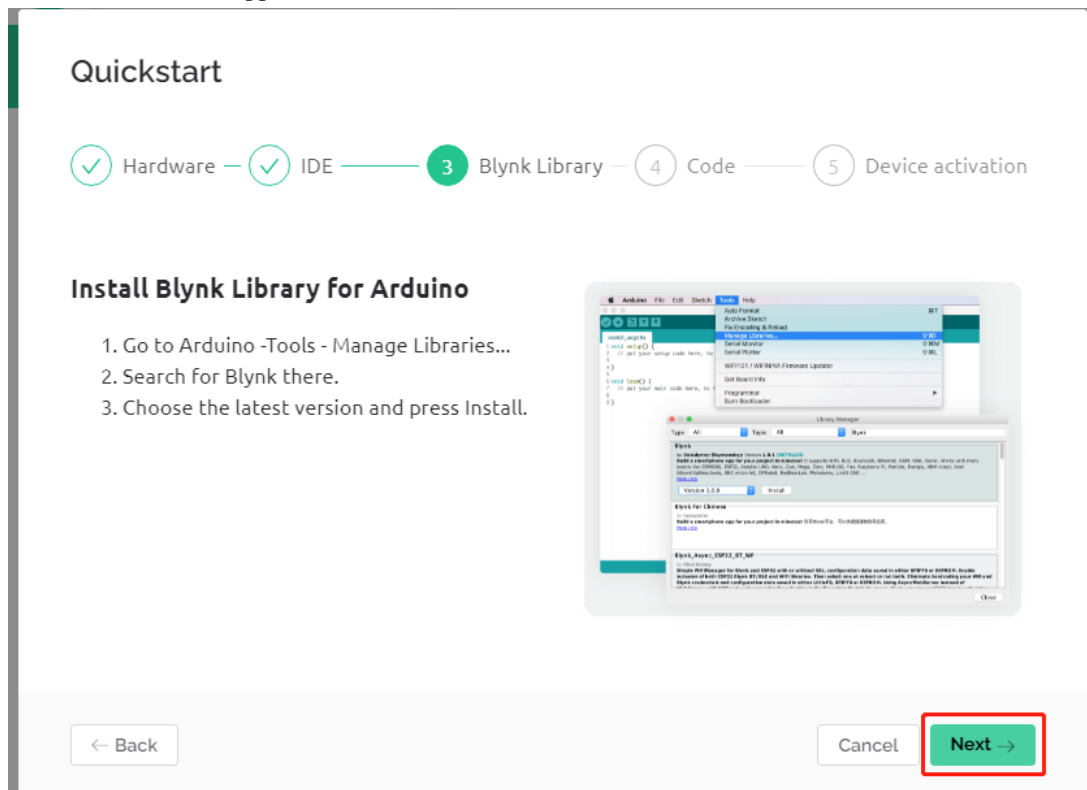
CancelNext →



7. Ici, on vous indique quel IDE vous devez préparer, nous recommandons **Arduino IDE**.



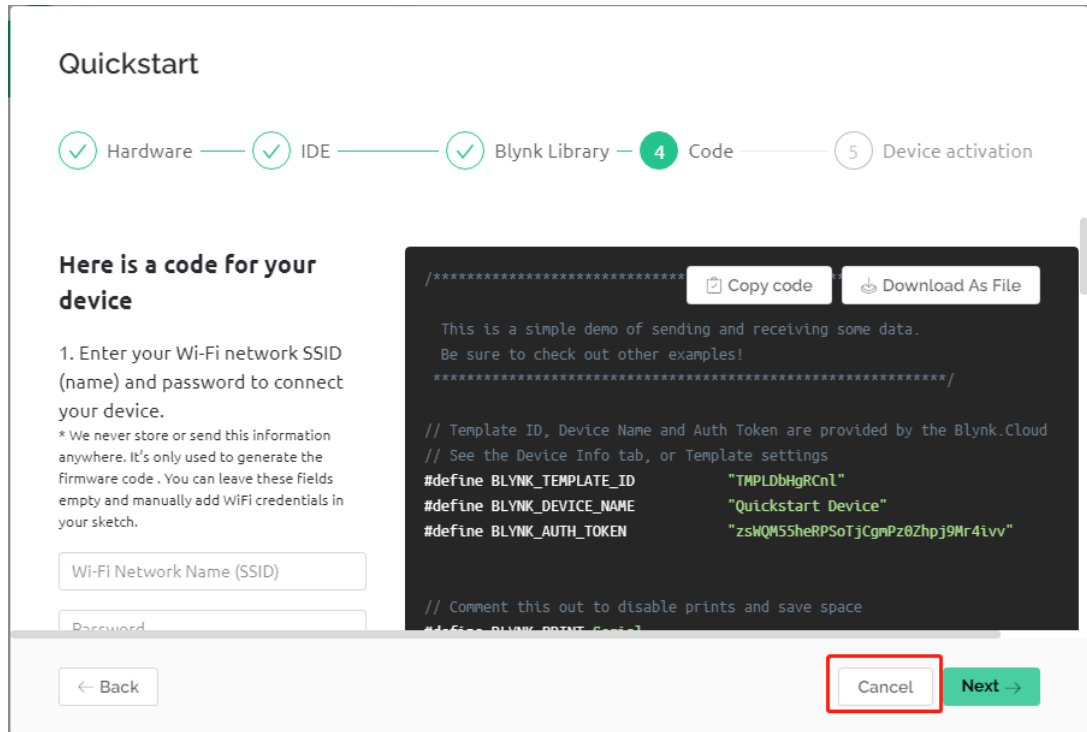
8. Voici la bibliothèque que vous devez ajouter, mais la bibliothèque recommandée ici est un peu problématique, nous devons ajouter d'autres bibliothèques manuellement (nous en parlerons plus tard). Cliquez sur **Next** ici, et un nouveau modèle et appareil seront créés.



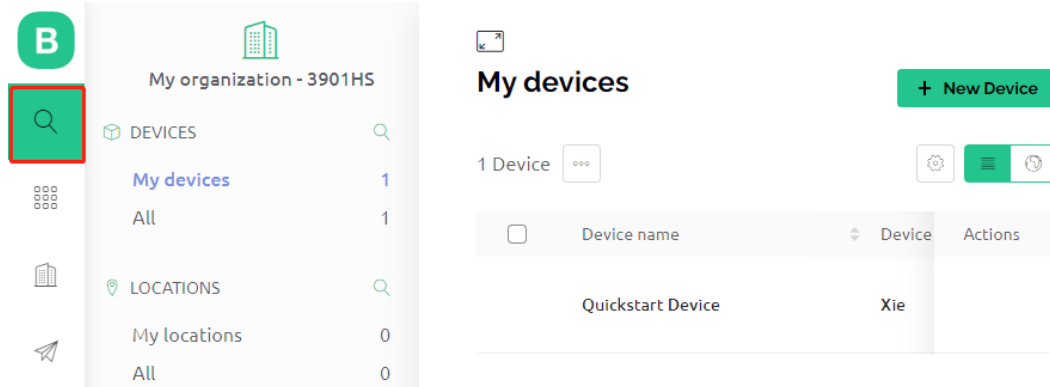
9. Les étapes suivantes consistent à télécharger le code pertinent et à connecter votre carte à Blynk, mais étant



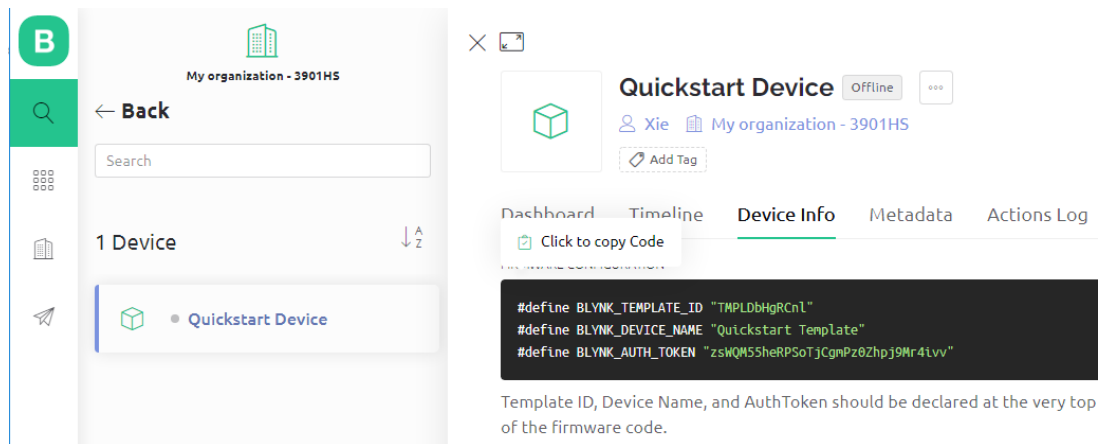
donné qu'il y a un problème avec la bibliothèque fournie plus tôt, vous devez ajouter d'autres bibliothèques à nouveau. Donc, cliquez sur **Cancel** ici pour arrêter **Quick Start**.



10. Cliquez sur le bouton **Search** et vous verrez le nouvel appareil que vous venez de créer.



11. Allez sur ce **Quickstart Device** et vous verrez **TEMPLATE\_ID**, **DEVICE\_NAME** et **AUTH\_TOKEN** sur la page **Device info**, et vous aurez besoin de les copier plus tard.



### 7.1.3 1.3 Ajout des bibliothèques requises

Vous devez ajouter les bonnes bibliothèques pour que l'Arduino IDE puisse utiliser Blynk.

1. Cliquez [ICI](#), faites défiler la page jusqu'en bas et téléchargez le premier fichier .zip.

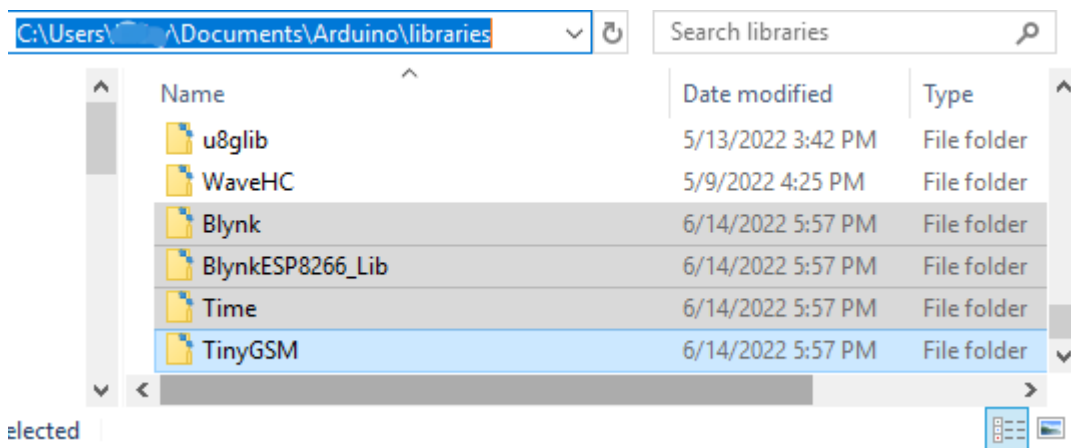
▼ Assets 3

Blynk_Release_v1.1.0.zip	779 KB	22 days ago
Source code (zip)		22 days ago
Source code (tar.gz)		22 days ago

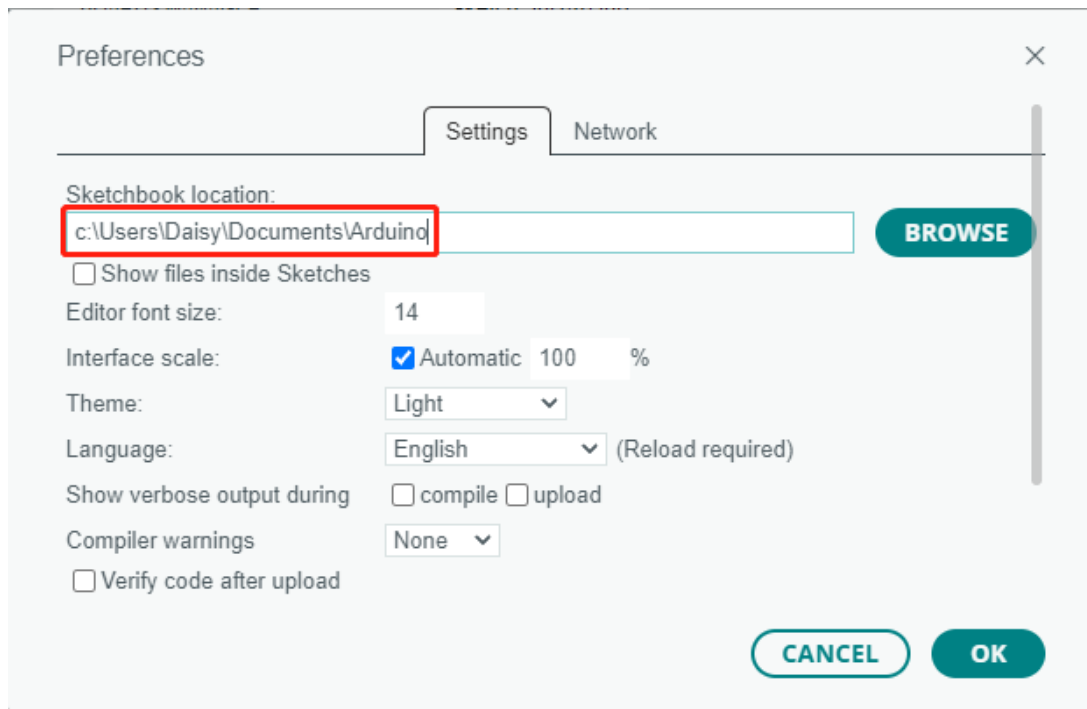
2. Décompressez ce fichier et vous pourrez voir les dossiers suivants.

..			File folder	
Blynk	1,060,482	459,412	File folder	7/16/2021 1:13 ...
BlynkESP8266_Lib	57,090	11,208	File folder	5/25/2021 5:12 ...
Time	63,774	25,512	File folder	5/25/2021 5:12 ...
TinyGSM	602,241	172,967	File folder	5/25/2021 5:12 ...

3. Copiez-les tous et collez-les dans le répertoire de bibliothèques par défaut de l'Arduino IDE, qui se trouve généralement à C:\Users\xxx\Documents\Arduino\libraries.



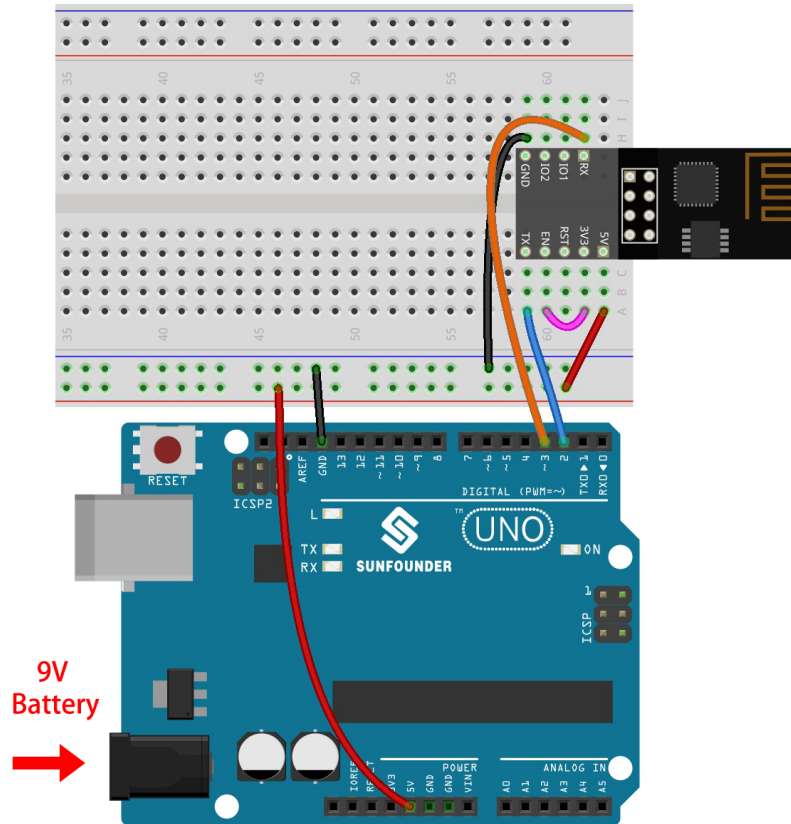
4. Si votre répertoire de bibliothèques est différent, vous pouvez le vérifier en allant dans **File -> Preferences**.



#### 7.1.4 1.4 Connexion de la carte R3 à Blynk

1. Reconnectez le module ESP8266 et la carte R3, ici le serial logiciel est utilisé, donc TX et RX sont connectés respectivement aux broches 2 et 3 de la carte R3.

**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc assurez-vous que la batterie 9V est branchée.



- Ouvrez le fichier `1.connect.ino` situé dans le dossier `3in1-kit\iot_project\1.connect`. Ou copiez ce code dans **Arduino IDE**.
- Remplacez les trois lignes de code suivantes que vous pouvez copier depuis la page **Device info** de votre compte. Ces trois lignes de code permettront à votre carte R3 de trouver votre compte blynk.

```
#define BLYNK_TEMPLATE_ID "TMPLxxxxxx"
#define BLYNK_DEVICE_NAME "Device"
#define BLYNK_AUTH_TOKEN "YourAuthToken"
```

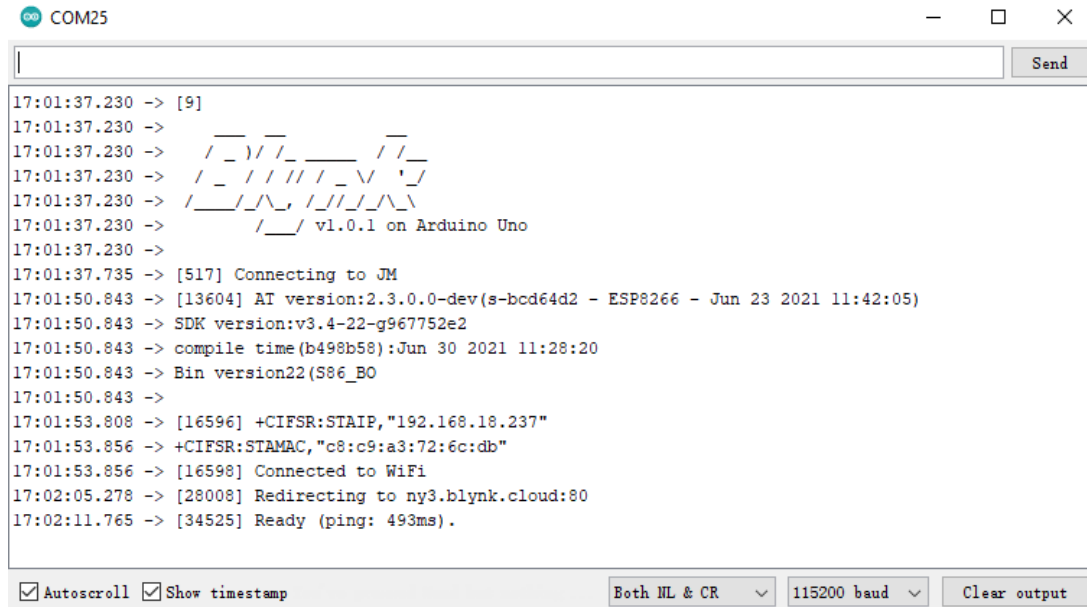
```
#define BLYNK_TEMPLATE_ID "TMPLDbHgRcnL"
#define BLYNK_DEVICE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "zsWQM55heRPSotJcgnPz0Zhpj9Mr4lvv"
```

Template ID, Device Name, and AuthToken should be declared at the very top of the firmware code.

- Remplissez le ssid et le password du WiFi que vous utilisez.

```
char ssid[] = "ssid";  
char pass[] = "password";
```

5. Téléchargez le code sur la carte R3, puis ouvrez le moniteur série et réglez le débit en bauds sur 115200. Lorsque la carte R3 communique avec Blynk avec succès, le moniteur série affichera le caractère `ready`.

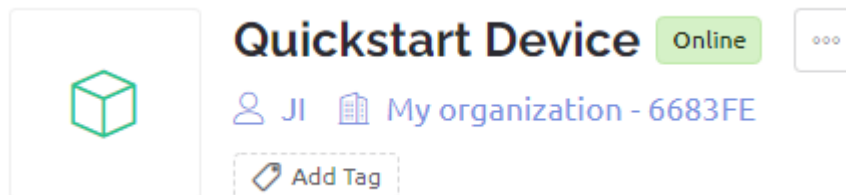


**Note :** Si le message ESP is not responding apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

6. Le statut de Blynk passera de **offline** à **online**.



## 7.2 2. Obtenir des Données depuis Blynk

Dans ce chapitre, vous apprendrez à contrôler le circuit avec Blynk. Allumons les LED via Internet !

### Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

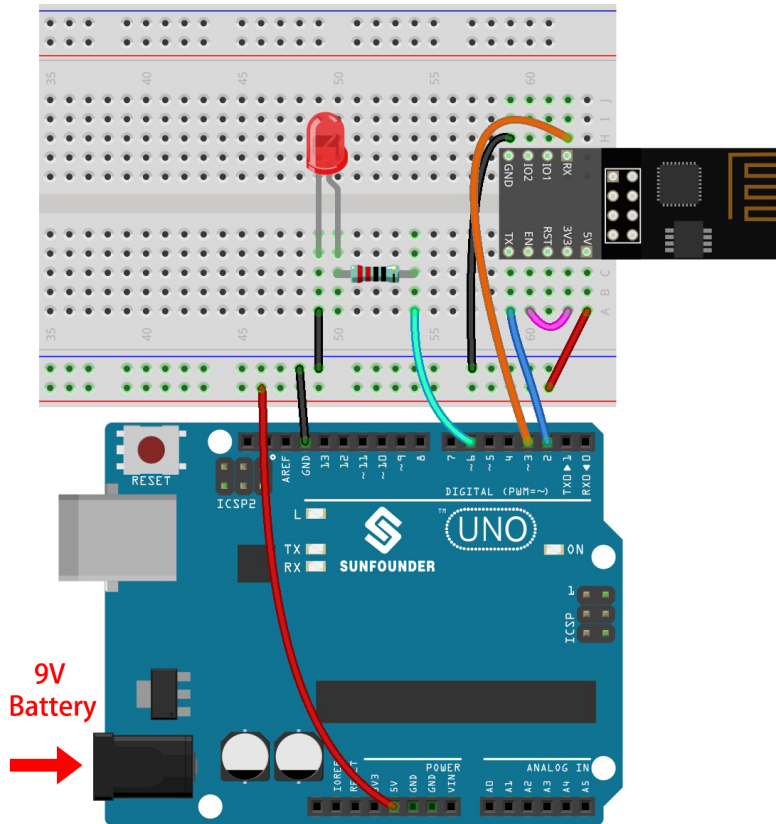
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Module ESP8266</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	

### 1. Construire le Circuit

---

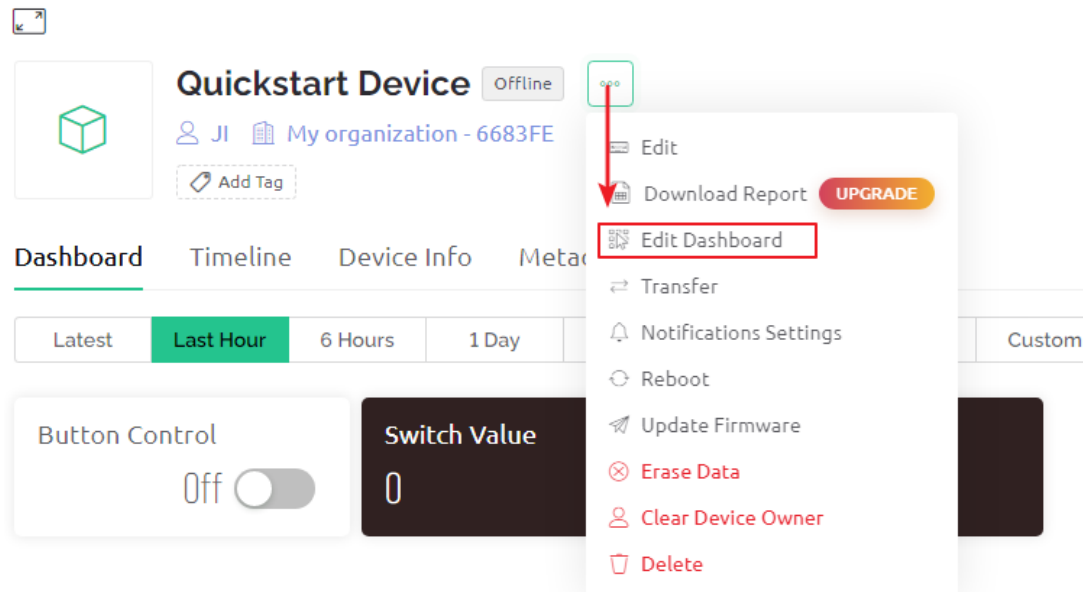
**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc assurez-vous que la batterie 9V est branchée.

---



## 2. Éditer le Tableau de Bord

1. Allez sur le **Quickstart Device** que vous avez créé précédemment, cliquez sur l'icône du menu en haut à droite et sélectionnez **edit dashboard**.







2. Les Datastreams permettent aux widgets sur Blynk et au code sur la carte R3 de se reconnaître mutuellement. Pour expérimenter le processus de configuration complet, supprimez tous les Datastreams de la page Datastreams.

## Quickstart Template

[...](#) [Cancel](#) [Save And Apply](#)
[Info](#) [Metadata](#) [Datastreams](#) [Events](#) [Automations](#) [Web Dashboard](#) [Mobile Dashboard](#)
 Search datastream

[+ New Datastream](#)

 4 Datastreams selected of 4 [Clear selection](#) [🗑](#)

<input checked="" type="checkbox"/>	Id	Name	Alias	Color	Pin	Data Type	Units	Is	Actions
<input checked="" type="checkbox"/>	1	Switch Control	Switch Control		V0	Integer		fa	
<input checked="" type="checkbox"/>	2	Switch Value	Switch Value		V1	Integer		fa	
<input checked="" type="checkbox"/>	3	Seconds	Seconds		V2	Integer		fa	
<input checked="" type="checkbox"/>	4	Button Image	Button Image		V3	String		fa	

3. Veuillez lire attentivement l'avertissement et confirmer qu'il est correct avant de supprimer les Datastreams.

## DANGER ZONE

**i** Deleting Datastream(s) can lead to breaking changes.

- All widgets using this Datastream(s) will stop working
- All Automation scenarios using this Datastream(s) will stop working
- All active rules in Rules Engine using this Datastream(s) will stop working

This can not be undone.

Type DELETE in the field below to proceed with the deletion.

 DELETE

☒ I fully understand that this action is critical and will lead to data loss

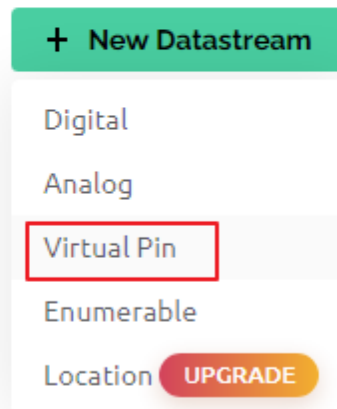
[Cancel](#)
[Delete](#)



4. Créez un Datastream de type **Virtual Pin**, qui sera utilisé pour contrôler la LED en utilisant l'interrupteur Blynk.

## Datastreams

Datastreams is a way to structure data that regularly flows in and out from device. Use it for sensor data, any telemetry, or actuators.



5. Configurez le **Virtual Pin**. Comme le bouton et la LED n'ont besoin que d'être ON et OFF, réglez DATA TYPE sur Integer et MIN et MAX sur 0 et 1.

**Virtual Pin Datastream**

NAME:  ALIAS:

PIN:  DATA TYPE:

UNITS:

MIN:  MAX:  DEFAULT VALUE:

Region: ny3 [Privacy Policy](#)

6. Allez sur la page **Web Dashboard** et supprimez les widgets existants.

**Quickstart Template**

Info Metadata Datastreams Events Automations **Web Dashboard** Mobile Dashboard

**Widget Box**  
3 of 30 widgets

**CONTROL**

Switch ☒

Slider

Number Input

**Device name** Online

Device Owner Company Name

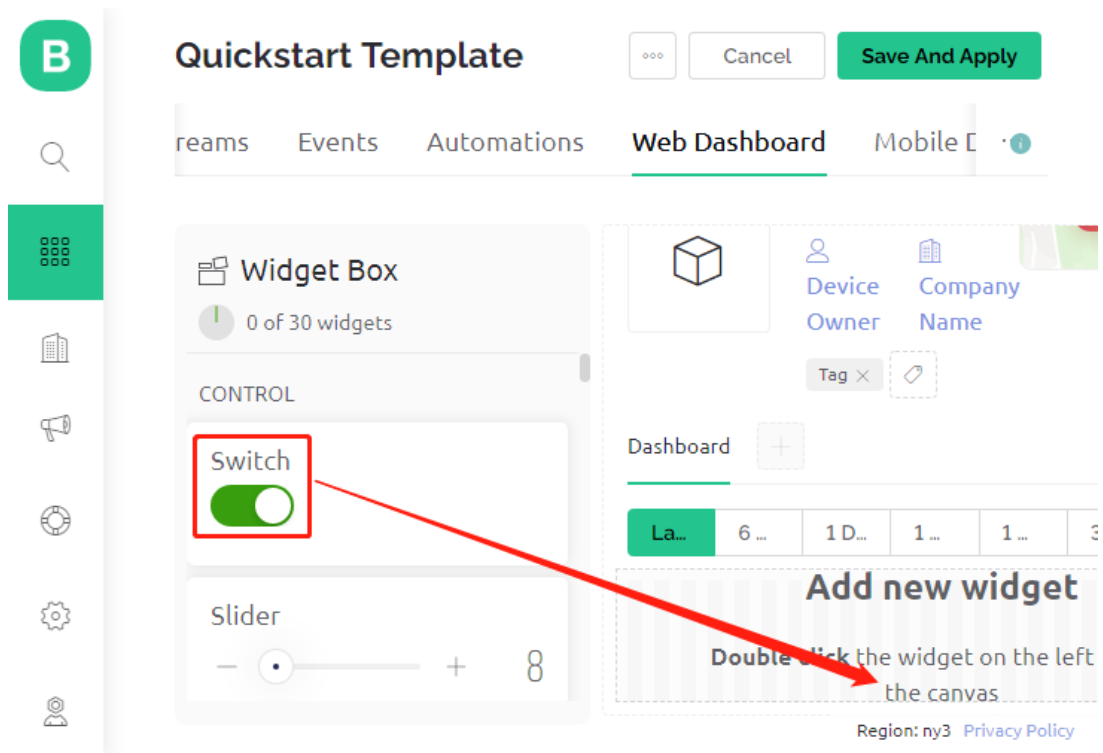
Tag

Dashboard

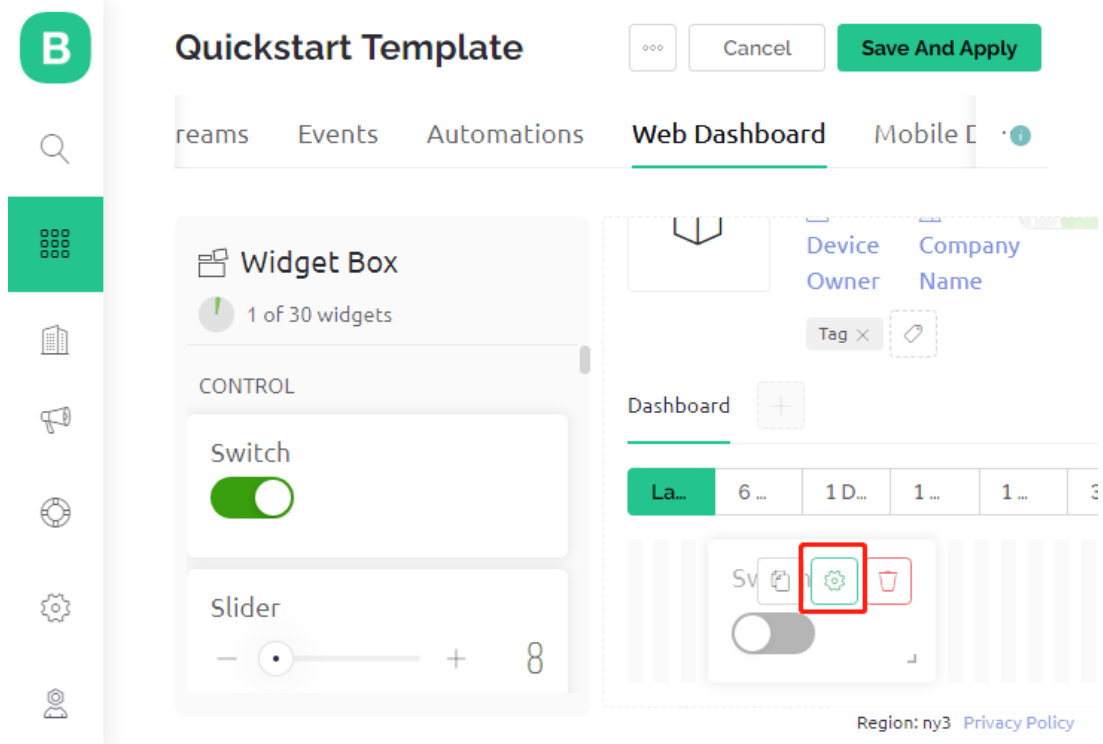
Last H... 6 Hours 1 Day 1 Week 1 Month 3 Mont... Custom

Switc... Uptime

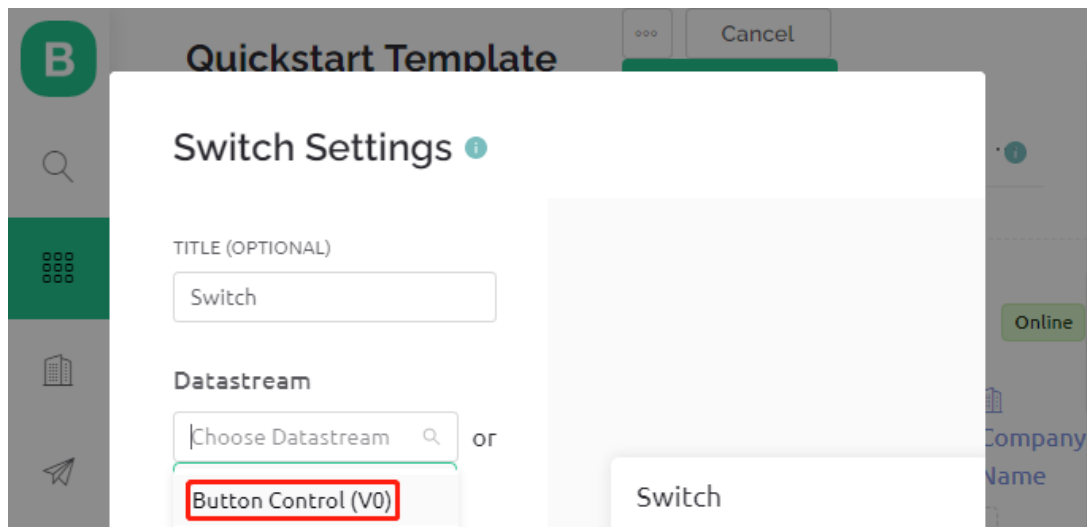
7. Faites glisser et déposez un widget **switch** depuis la **Widget Box** à gauche.



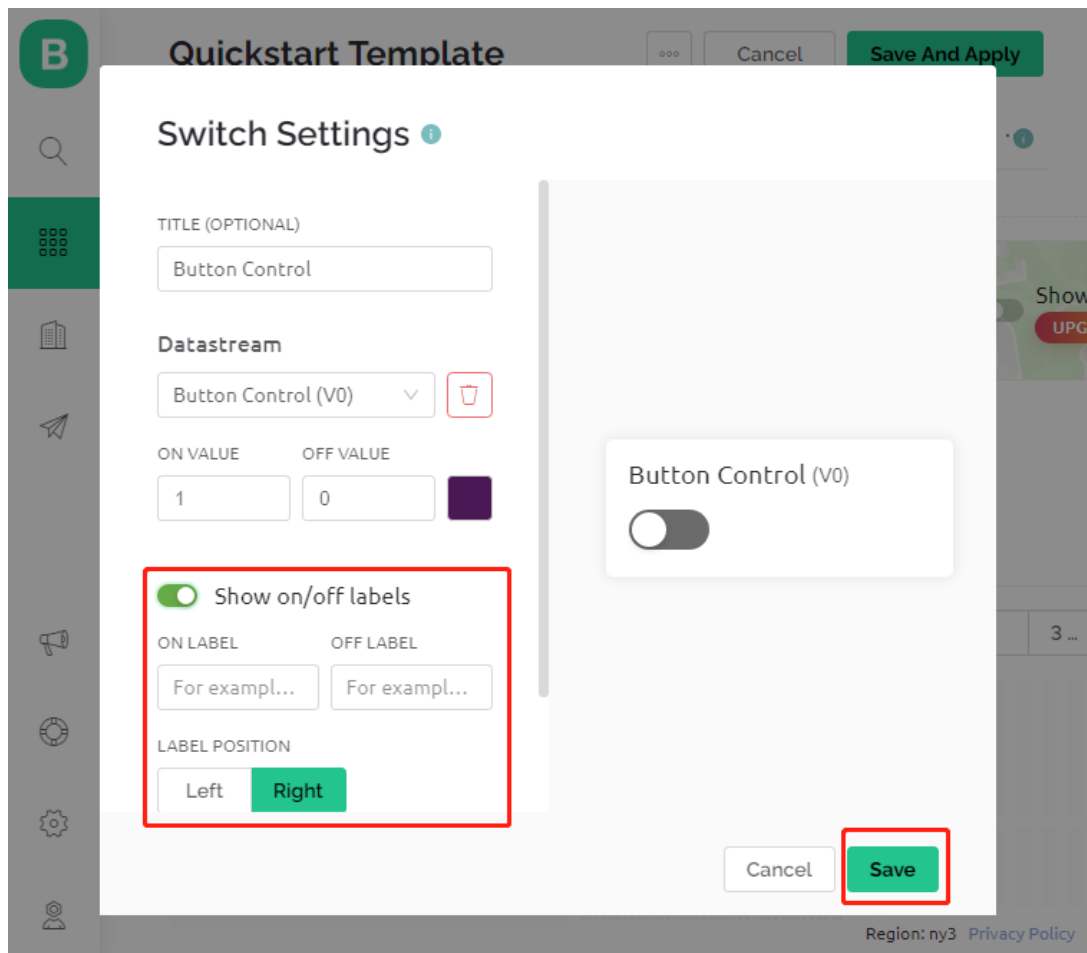
8. Maintenant pour le configurer.



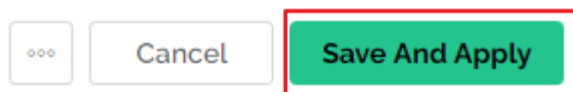
9. Sélectionnez **Datastream** comme celui que vous avez configuré précédemment.



10. Après avoir sélectionné Datastream, vous verrez quelques paramètres personnalisés, puis appuyez sur Enregistrer.

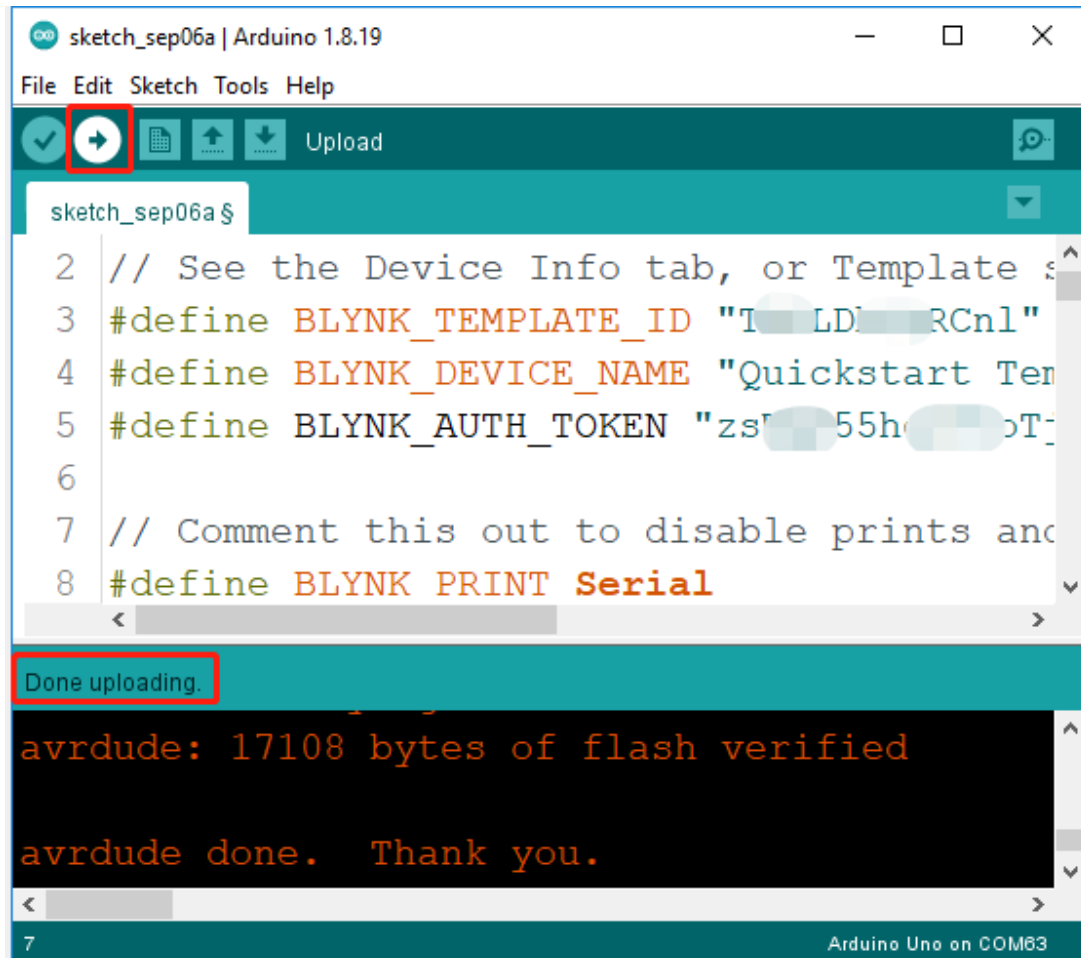


11. Enfin, cliquez sur **Save And Apply**.



### 3. Exécutez le Code

1. Ouvrez le fichier `2.get_data_from_blynk.ino` situé dans le dossier `3in1-kit\iot_project\2.get_data_from_blynk`, ou copiez ce code dans **Arduino IDE**.
2. Remplacez le Template ID, Device Name et Auth Token par les vôtres. Vous devez également entrer le ssid et le mot de passe du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).
3. Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.



4. Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.

```

COM63

#StandWithUkraine https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.75"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).

☒ Autoscroll ☐ Show timestamp
Both NL & CR 115200 baud Clear output

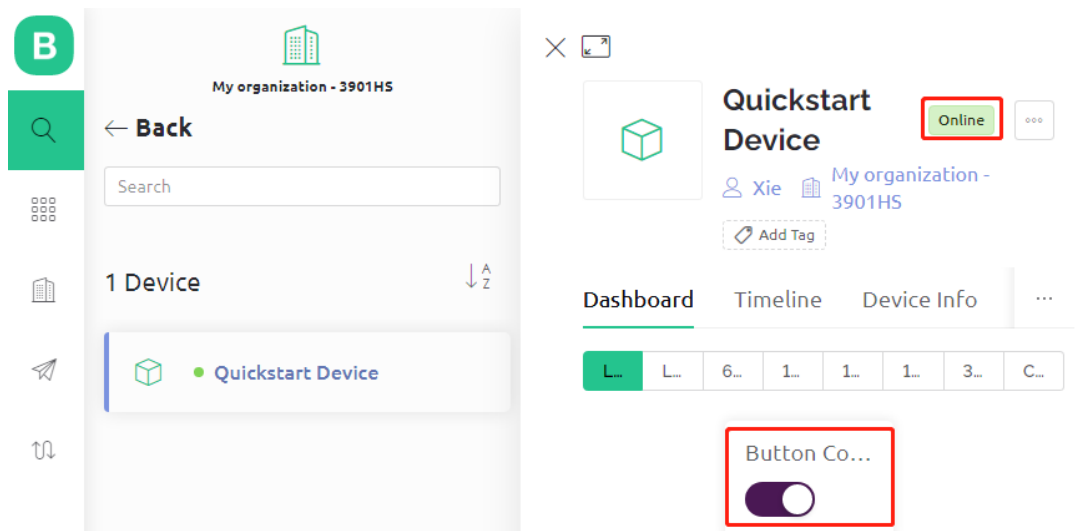
```

**Note :** Si le message ESP is not responding apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

5. De retour sur Blynk, vous pouvez voir que le statut a changé en ligne et vous pouvez maintenant utiliser l'interrupteur widget sur Blynk pour contrôler la LED connectée à la carte R3.



6. Si vous souhaitez utiliser Blynk sur des appareils mobiles, veuillez vous référer à *Comment utiliser Blynk sur un appareil mobile ?*.

### Comment ça fonctionne ?

La différence entre le code de ce projet et le code du chapitre précédent *1.4 Connexion de la carte R3 à Blynk* est les lignes suivantes.

```
const int ledPin=6;

BLYNK_WRITE(V0)
{
    int pinValue = param.asInt(); // assigning incoming value from pin V0 to a variable
    // You can also use:
    // String i = param.asStr();
    // double d = param.asDouble();
    digitalWrite(ledPin,pinValue);
}

void setup()
{
    pinMode(ledPin,OUTPUT);
}
```

En ce qui concerne les fonctions `pinMode` et `digitalWrite` du `ledPin`, je suis sûr que vous les connaissez déjà, donc je ne vais pas les réexpliquer. Ce sur quoi vous devez vous concentrer, c'est la fonction `BLYNK_WRITE(V0)`.

Ce qu'elle va faire, c'est que lorsque la valeur de `V0` sur Blynk change, Blynk.Cloud dira à votre appareil « Je suis en train d'écrire sur le **Virtual Pin** V0 », et votre appareil pourra réaliser quelque chose une fois qu'il aura reçu cette information.

Nous avons créé le Datastream `V0` à l'étape précédente et l'avons appliqué au Widget Interrupteur. Cela signifie que chaque fois que nous utilisons le Widget Interrupteur, `BLYNK_WRITE(V0)` sera déclenché.

Nous écrivons deux instructions dans cette fonction.

```
int pinValue = param.asInt();
```

Obtenez la valeur de `V0` et attribuez-la à la variable `pinValue`.

```
digitalWrite(ledPin,pinValue);
```

Écrivez la valeur de `V0` obtenue sur le `ledPin`, afin que le widget Interrupteur sur Blynk puisse contrôler la LED.

## 7.3 3. Envoyer des Données à Blynk

Ce chapitre vous montrera comment envoyer des données à Blynk.

Nous créons ici un dispositif de détection de portes et fenêtres. Le circuit avec l'interrupteur à lames (reed switch) est placé à côté de la porte et de la fenêtre, et l'aimant est monté sur le bord de la porte et de la fenêtre. Lorsque la porte ou la fenêtre est fermée, l'interrupteur à lames sera activé par la force magnétique et la valeur de pin correspondante sur la carte R3 changera. Blynk.cloud recevra cette valeur afin que vous puissiez voir si les portes et fenêtres de votre maison sont fermées, même lorsque vous n'êtes pas à la maison.

Nous utiliserons maintenant un widget LED dans Blynk pour indiquer si vos fenêtres et portes sont fermées (c'est-à-dire si l'interrupteur à lames est activé ou désactivé).

### Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Module ESP8266</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Interrupteur à Lame Souple</i>	-

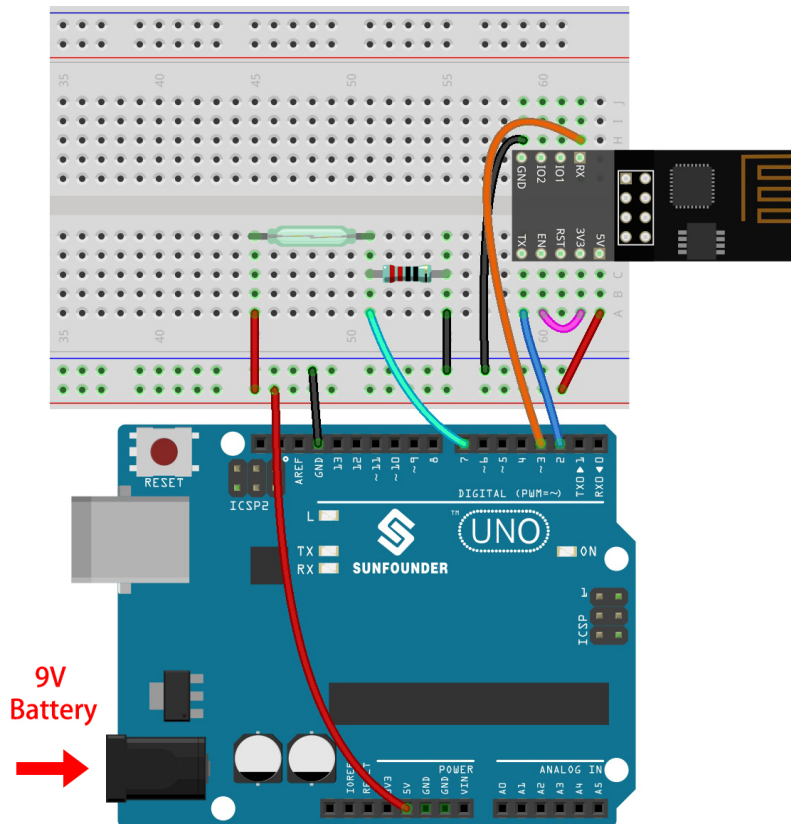
### 1. Construire le Circuit

---

**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc assurez-vous que la batterie 9V est branchée.

---





## 2. Éditer le Tableau de Bord

1. Créez un **Datastream** de type **Virtual Pin** sur la page **Datastream** pour obtenir la valeur de l'Interrupteur à Lames. Réglez le TYPE DE DONNÉES sur **Integer** et MIN et MAX sur **0** et **1**.

Quickstart Template

Virtual Pin Datastream

NAME:  ALIAS:

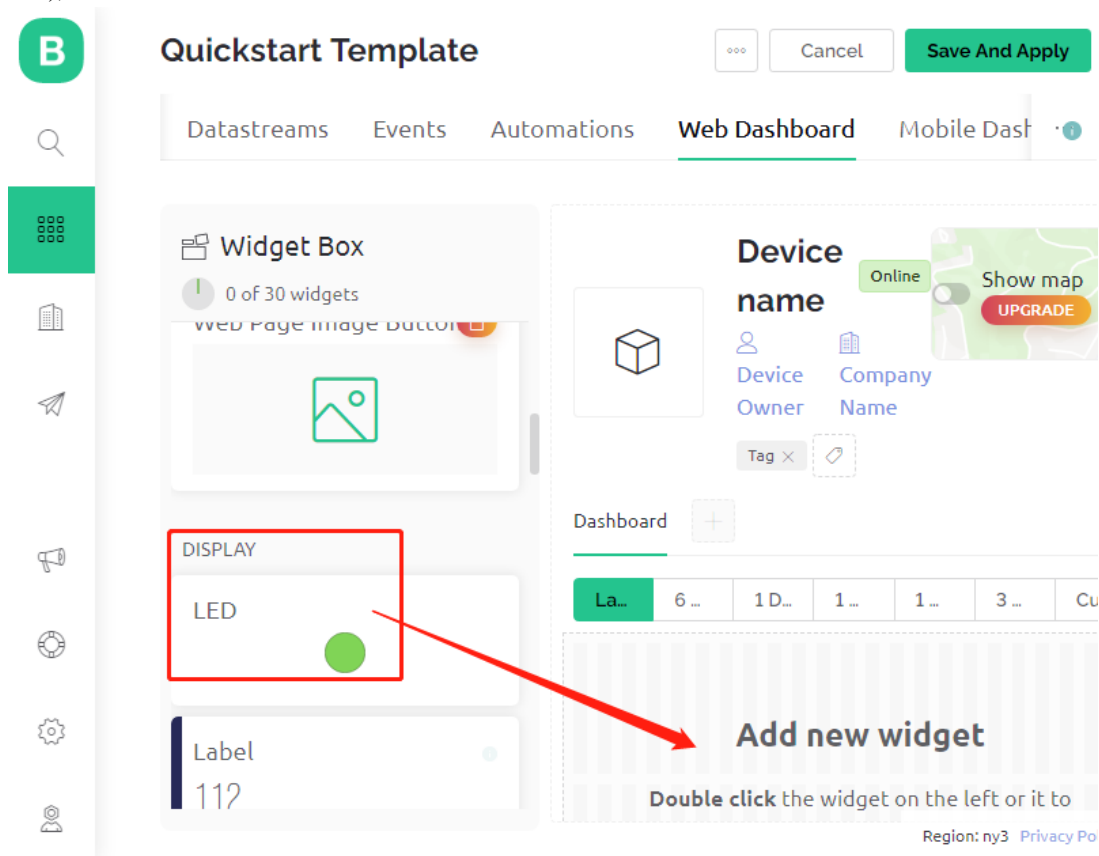
PIN:  DATA TYPE:

UNITS:

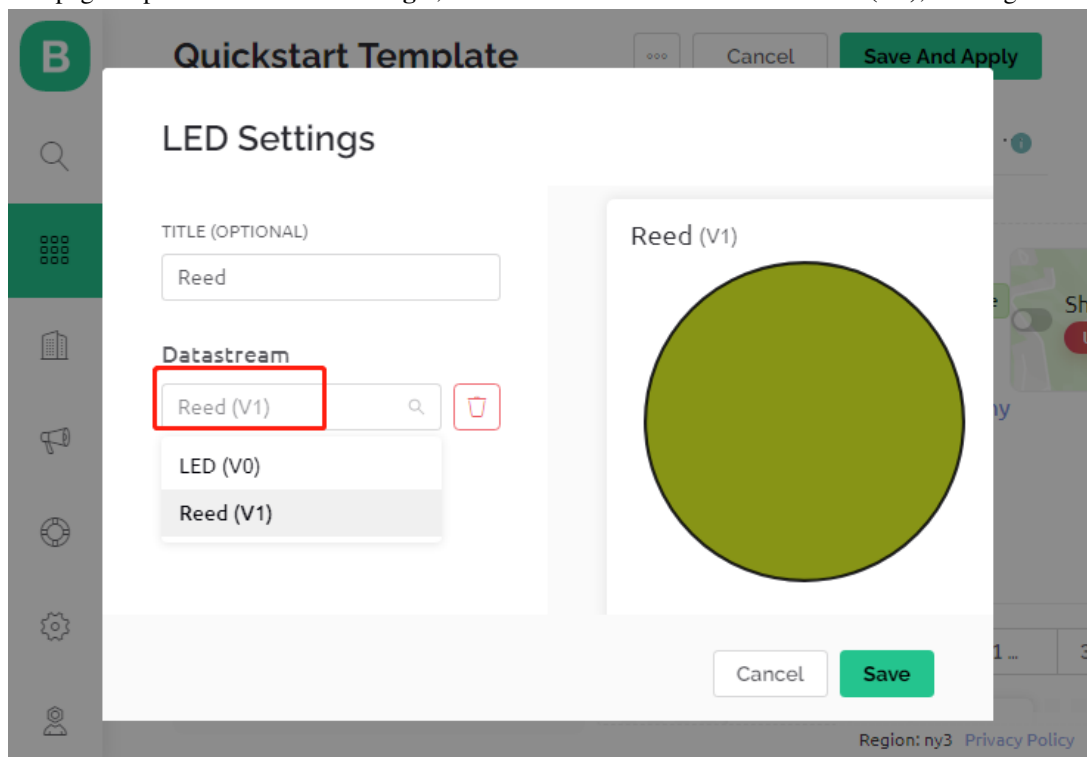
MIN:  MAX:  DEFAULT VALUE:

Region: ny3 Privacy Policy

2. Faites glisser et déposez un **LED widget** sur la page **Web Dashboard**, à une valeur de 1, il s'allumera (avec couleur), sinon il sera blanc.



3. Dans la page de paramètres du **LED widget**, sélectionnez **Datastream** comme **Reed(V1)**, et enregistrez-le.



### 3. Exécutez le Code

1. Ouvrez le fichier `3.push_data_to_blynk.ino` situé dans le dossier `3in1-kit\iot_project\3.push_data_to_blynk`, ou copiez ce code dans **Arduino IDE**.
2. Remplacez le Template ID, Device Name et Auth Token par les vôtres. Vous devez également entrer le ssid et le password du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).
3. Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.
4. Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.

```

COM63

#StandWithUkraine https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.75"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).

☒ Autoscroll ☐ Show timestamp
Both NL & CR 115200 baud Clear output

```

**Note :** Si le message `ESP is not responding` apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

5. Maintenant, Blynk affichera l'état de vos portes et fenêtres. Si vos portes et fenêtres sont fermées, le widget LED sera vert, sinon, il sera gris.
6. Si vous souhaitez utiliser Blynk sur des appareils mobiles, veuillez vous référer à [Comment utiliser Blynk sur un appareil mobile ?](#).

### Comment ça fonctionne ?

Pour cet exemple, vous devriez vous concentrer sur les lignes suivantes. « Écrire des données toutes les secondes dans le Datastream V1 de Blynk Cloud » est défini par ces lignes.

```

BlynkTimer timer;

void myTimerEvent()
{
    Blynk.virtualWrite(V1, pinValue);
}

void setup()
{
    timer.setInterval(1000L, myTimerEvent);
}

void loop()
{
    timer.run(); // Initiates BlynkTimer
}

```

La bibliothèque Blynk fournit un timer intégré, d'abord nous créons un objet timer.

```
BlynkTimer timer;
```

Réglez l'intervalle du timer dans `setup()`, ici nous réglons pour exécuter la fonction `myTimerEvent()` toutes les 1000ms

```
timer.setInterval(1000L, myTimerEvent);
```

Exécutez BlynkTimer dans `loop()`.

```
timer.run();
```

Éditez la fonction personnalisée `myTimerEvent()`, le code `Blynk.virtualWrite(V1, pinValue)` est utilisé pour écrire la donnée `pinValue` pour `V1`.

```

void myTimerEvent()
{
    Blynk.virtualWrite(V1, pinValue);
}

```

## 7.4 4. Lecteur de Musique Cloud

L'objectif de ce projet est de créer un lecteur de musique en utilisant Blynk. La musique est jouée de la même manière que dans *5.7 Tone() ou noTone()*, en écrivant la chanson dans le programme et en la jouant avec un buzzer passif. Cependant, dans cet exemple, nous pouvons cliquer sur l'interrupteur pour jouer/mettre en pause et faire glisser le curseur pour changer le progrès de la lecture.

### Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

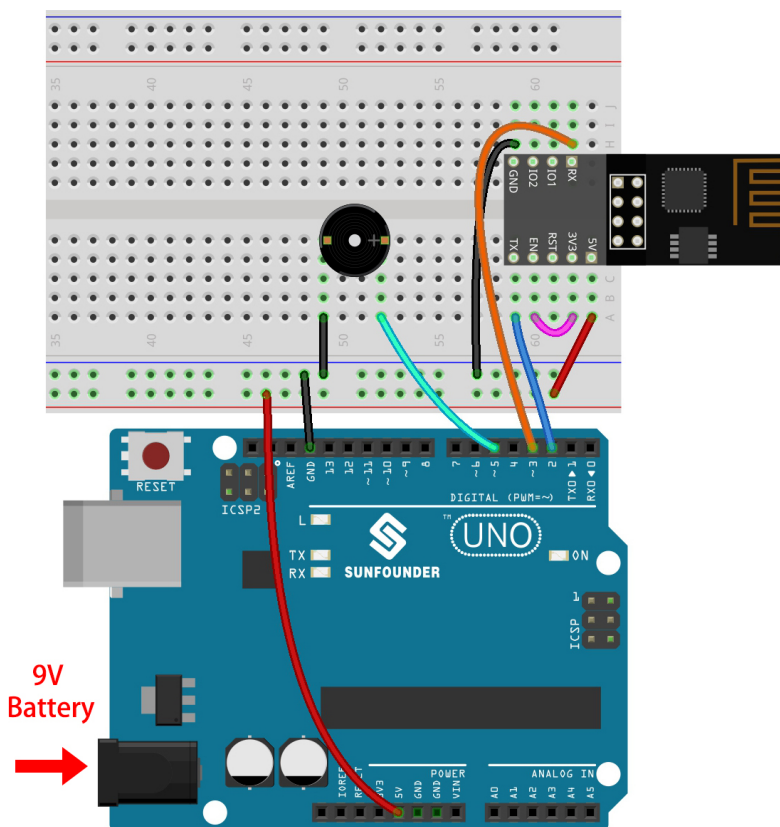
Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Module ESP8266</i>	
<i>Fils de Cavalier</i>	
<i>Buzzer</i>	

## 1. Construire le Circuit

**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc assurez-vous que la batterie 9V est branchée.



## 2. Éditer le Tableau de Bord

1. Créez un **Datastream** de type **Virtual Pin** sur la page **Datastream** comme la valeur modifiée par le widget Curseur ajouté plus tard ou le code. Réglez le TYPE DE DONNÉES sur **Integer** et MIN et MAX sur **0** et **30**.

Quickstart Templa...

## Virtual Pin Datastream

NAME  ALIAS

PIN  DATA TYPE

UNITS

MIN  MAX  DEFAULT VALUE

[+ ADVANCED SETTINGS](#)

[Cancel](#) [Create](#)

Region: ny3 [Privacy Policy](#)

2. Créez également un autre **Datastream** de type **Virtual Pin** pour afficher le nom de la musique, et réglez le TYPE DE DONNÉES sur String.

**Virtual Pin Datastream**

NAME:  ALIAS:

PIN:  DATA TYPE:

DEFAULT VALUE:

Region: ny3 [Privacy Policy](#)

3. Allez sur la page **Wed Dashboard**, faites glisser un widget **Switch** et réglez **Datastream** sur V0 (V0 est déjà réglé dans 2. *Obtenir des Données depuis Blynk*); faites glisser un widget **Label** et réglez-le sur V3; faites glisser un widget **Slider** et réglez-le sur V2.

**Widget Box**  
3 of 30 widgets

**Chart**

**Map**

**Device name** Online  
 Device Owner Company Name  
 Tag

**Dashboard**

Last Hour 6 Hours 1 Day 1 Week 1 Month

Button Co... (V0)

Slider (V2)

Song Name (V3)  
String

**Note :** Vos pins virtuels peuvent être différents des miens, les vôtres prévaudront, mais vous devrez modifier le numéro de pin correspondant dans le code.

### 3. Exécutez le Code

1. Ouvrez le fichier `4.cloud_music_player.ino` situé dans le dossier `3in1-kit\iot_project\4.cloud_music_player`.
2. Remplacez le Template ID, Device Name et Auth Token par les vôtres. Vous devez également entrer le ssid et le password du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).
3. Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.
4. Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.

```
#StandWithUkraine https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.175"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).
```

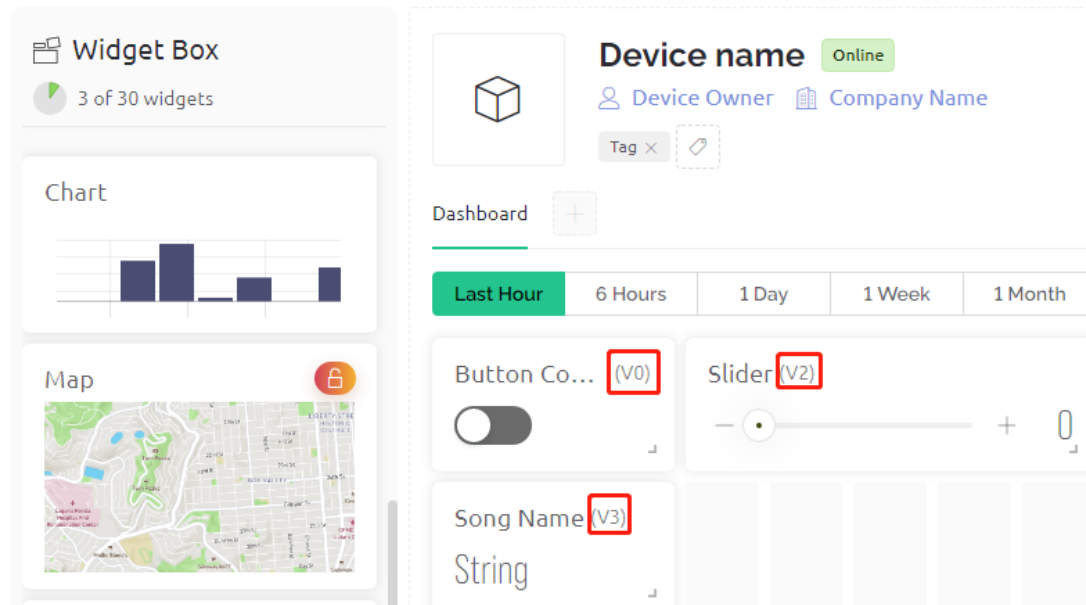
**Note :** Si le message `ESP is not responding` apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

5. Maintenant, vous pouvez utiliser le widget Bouton de Contrôle de Blynk pour démarrer/mettre en pause la musique et le Curseur pour ajuster le progrès de la lecture, et vous verrez également le nom de la musique.





6. Si vous souhaitez utiliser Blynk sur des appareils mobiles, veuillez vous référer à [Comment utiliser Blynk sur un appareil mobile ?](#).

#### Comment ça fonctionne ?

Le datastream **V0** est utilisé pour obtenir l'état du widget Interrupteur et l'assigner à la variable **musicPlayFlag**, qui contrôle la pause et la lecture de la musique.

```
int musicPlayFlag=0;

BLYNK_WRITE(V0)
{
    musicPlayFlag = param.asInt(); // START/PAUSE MUSIC
}
```

Le datastream **V2** est utilisé pour obtenir la valeur du widget Curseur et l'assigner à la variable **scrubBar** lorsque le curseur est déplacé.

```
int scrubBar=0;

BLYNK_WRITE(V2)
{
    scrubBar=param.asInt();
}
```

Lorsque le dispositif est connecté au **Blynk Cloud**, écrivez le nom de la musique pour le datastream **V3** puis affichez-le avec le widget **Label**.

```
BLYNK_CONNECTED() {
    String songName = "Ode to Joy";
    Blynk.virtualWrite(V3, songName);
}
```

**Blynk Timer** s'exécutera toutes les secondes. La musique est jouée si **musicPlayFlag** n'est pas 0, c'est-à-dire que le widget **Switch** est ON. Dès que deux notes sont jouées, la variable de la barre de progression **scrubBar** est incrémentée de 2, et la valeur est ensuite écrite dans le **Blynk Cloud**, ce qui synchronise la valeur du widget **Slider**.

```

void myTimerEvent()
{
    if(musicPlayFlag!=0)
    {
        tone(buzzerPin,melody[scrubBar],250);
        scrubBar=(scrubBar+1)%(sizeof(melody)/sizeof(int));
        delay(500);
        tone(buzzerPin,melody[scrubBar],250);
        scrubBar=(scrubBar+1)%(sizeof(melody)/sizeof(int));
        Serial.println(scrubBar);
        Blynk.virtualWrite(V2, scrubBar);
    }
}

```

## 7.5 5. Surveillance de l'Environnement Domestique

Dans ce chapitre, nous utiliserons Blynk pour créer un moniteur d'environnement domestique. Vous pouvez mesurer la température, l'humidité et l'intensité lumineuse d'une pièce à l'aide du DHT11 et d'une photorésistance. En envoyant ces valeurs à Blynk, vous pourrez connaître l'environnement de votre maison via Internet.

### Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

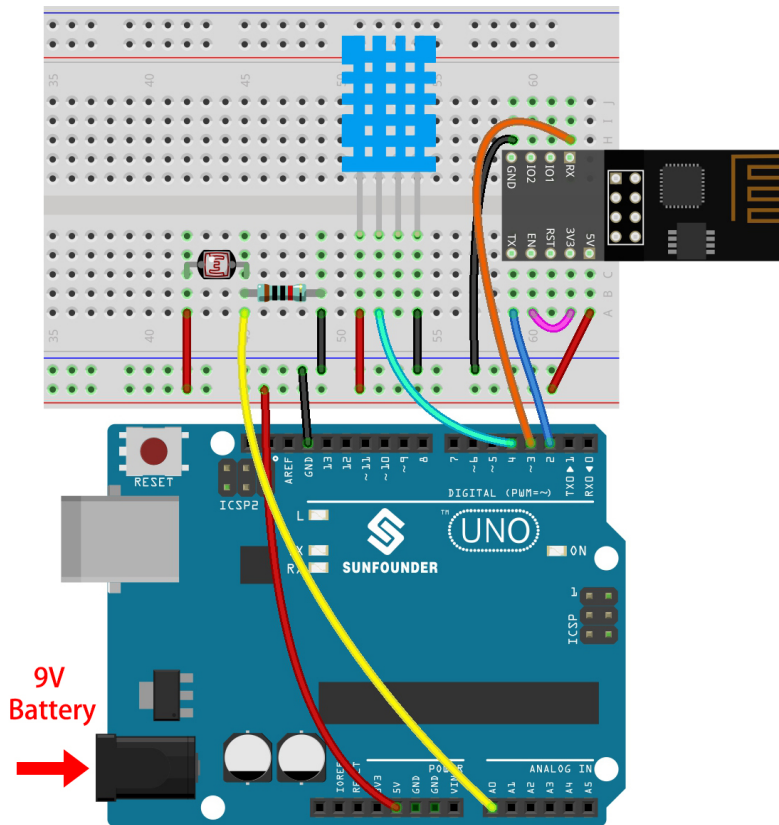
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Module ESP8266</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Photorésistance</i>	
<i>Capteur d'Humidité et de Température DHT11</i>	-

### 1. Construire le Circuit

**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc

assurez-vous que la batterie 9V est branchée.



## 2. Éditer le Tableau de Bord

1. Pour enregistrer les valeurs d'humidité, créez un **DataStream** de type **Virtual Pin** sur la page **DataStream**. Réglez le TYPE DE DONNÉES sur **Double** et MIN et MAX sur **0** et **100**. Réglez également les unités sur **Percentage, %**.

**Quickstart Template**

### Virtual Pin Datastream

NAME	ALIAS
<input type="text" value="Humidity"/>	<input type="text" value="Humidity"/>

PIN	DATA TYPE
<input type="text" value="V4"/>	<input type="text" value="Double"/>

UNITS
<input type="text" value="Percentage, %"/>

MIN	MAX	DECIMALS	DEFAULT VALUE
<input type="text" value="0"/>	<input type="text" value="100"/>	<input type="text" value="###"/>	<input type="text" value="Default Value"/>

☐ ADVANCED SETTINGS

Region: ny3 [Privacy Policy](#)

2. Ensuite, créez un **Datastream** de type **Virtual Pin** pour enregistrer la température. Réglez le TYPE DE DONNÉES sur **Double**, MIN et MAX sur **-30** et **50**, et les unités sur **Celsius, °C**.

**Quickstart Template**

**Virtual Pin Datastream**

NAME:  ALIAS:

PIN:  DATA TYPE:

UNITS:

MIN:  MAX:  DECIMALS:  DEFAULT VALUE:

Region: ny3 [Privacy Policy](#)

3. Créez également un **Datastream** de type **Virtual Pin** pour enregistrer l'intensité lumineuse. Utilisez le type de données par défaut - **Integer**, avec MIN et MAX réglés sur 0 et 1024.

Quickstart Templa...

### Virtual Pin Datastream

NAME  ALIAS

PIN  DATA TYPE

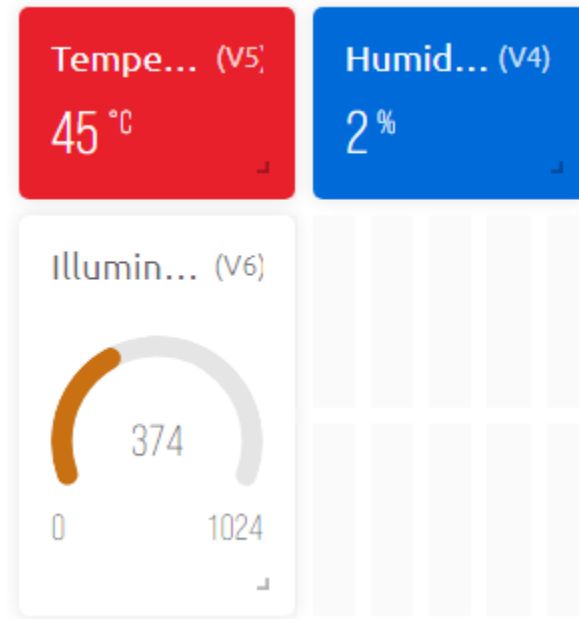
UNITS

MIN  MAX  DEFAULT VALUE

[+ ADVANCED SETTINGS](#)

Region: ny3 [Privacy Policy](#)

- Allez sur la page **Wed Dashboard**, faites glisser deux widgets **Label** et réglez leurs data streams respectivement sur **V4** et **V5**, et faites glisser un widget **Gauge** et réglez le data stream sur **V6**. De plus, dans le réglage du widget, vous pouvez activer **Changer de couleur selon la valeur** et sélectionner la couleur appropriée pour rendre le widget plus attrayant et intuitif.

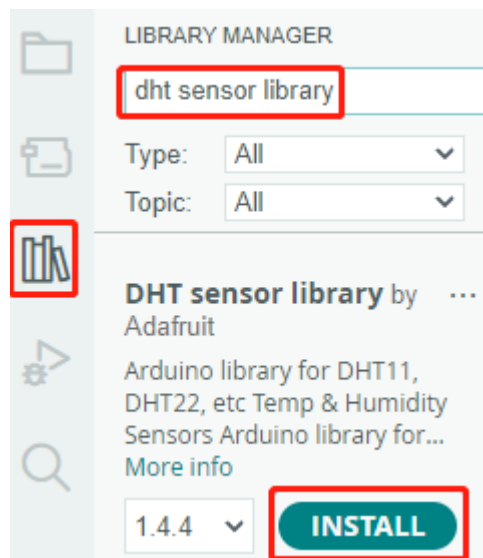


### 3. Exécutez le Code

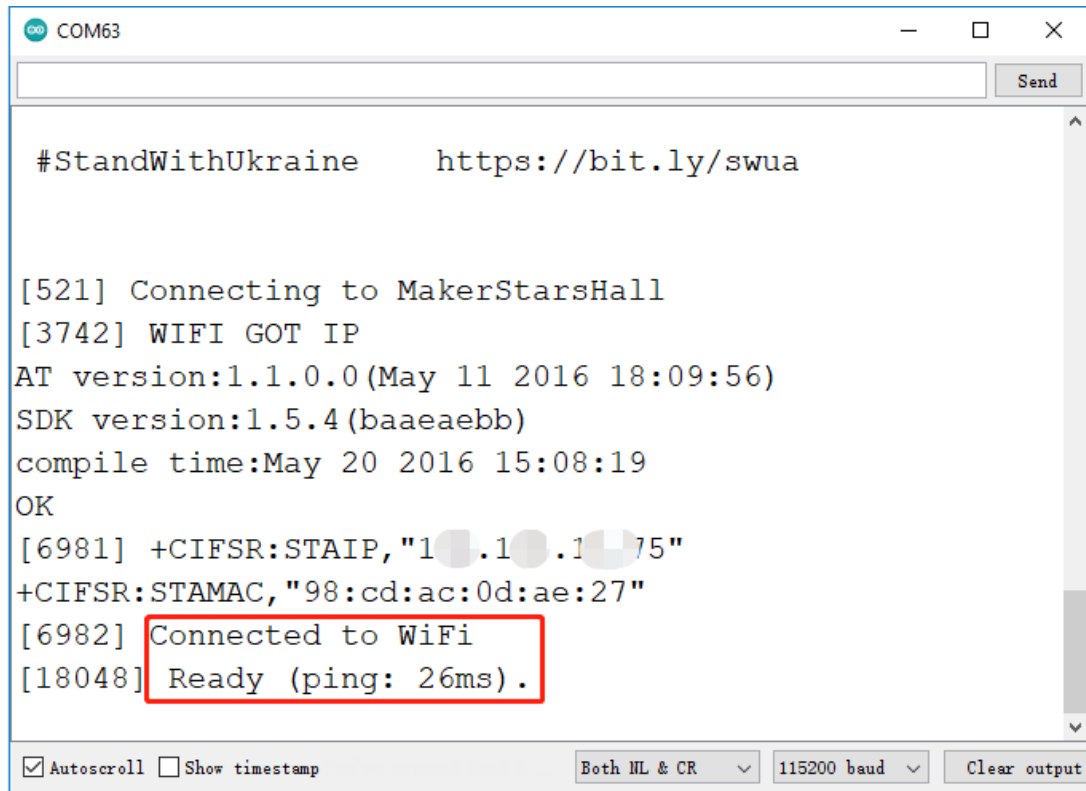
1. Ouvrez le fichier `5.home_environment_monitoring.ino` situé dans le dossier `3in1-kit\iot_project\5.home_environment_monitoring`, ou copiez ce code dans **Arduino IDE**.

#### Note :

— La `DHT sensor library` est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



2. Remplacez le Template ID, Device Name et Auth Token par les vôtres. Vous devez également entrer le ssid et le password du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).
3. Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.
4. Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.



```
COM63

#StandWithUkraine https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.175"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).
```

☒ Autoscroll ☐ Show timestamp Both NL & CR 115200 baud Clear output

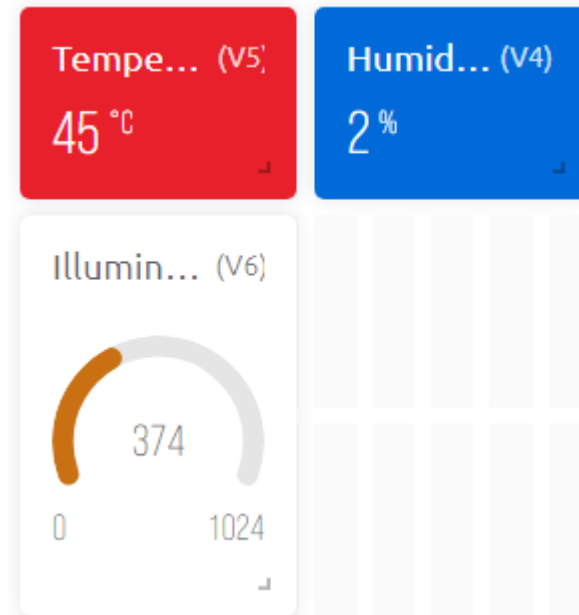
**Note :** Si le message ESP is not responding apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

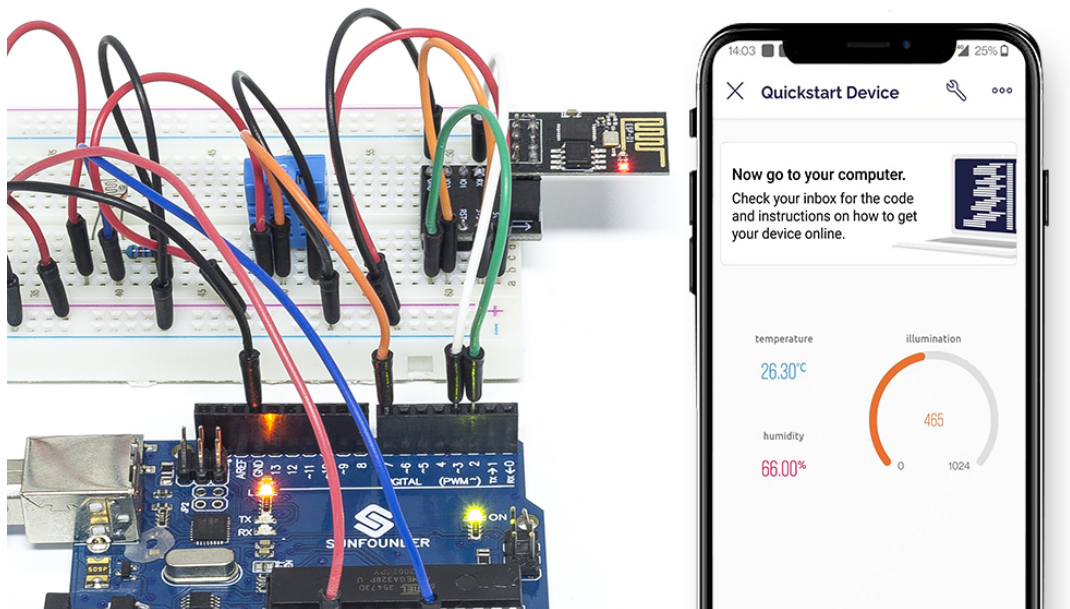
Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

5. Maintenant, vous verrez la température ambiante actuelle, l'humidité et l'intensité lumineuse sur Blynk.





6. Si vous souhaitez utiliser Blynk sur des appareils mobiles, veuillez vous référer à [Comment utiliser Blynk sur un appareil mobile ?](#).



### Comment ça fonctionne ?

Ces deux fonctions sont utilisées pour obtenir la température, l'humidité et l'intensité lumineuse de la pièce.

```
int readLight(){
    return analogRead(lightPin);
}

bool readDHT() {

    // Reading temperature or humidity takes about 250 milliseconds!
```

(suite sur la page suivante)

(suite de la page précédente)

```
// Sensor readings may also be up to 2 seconds 'old' (it's a very slow sensor)
humidity = dht.readHumidity();
// Read temperature as Celsius (the default)
temperature = dht.readTemperature();

// Check if any reads failed and exit early (to try again).
if (isnan(humidity) || isnan(temperature)) {
    Serial.println("Failed to read from DHT sensor!");
    return false;
}
return true;
}
```

Avec le Timer de Blynk, la température ambiante, l'humidité et l'intensité lumineuse sont obtenues chaque seconde et envoyées au data stream sur le Blynk Cloud, à partir duquel les widgets affichent les données.

```
void myTimerEvent()
{
    bool chk = readDHT();
    int light = readLight();
    if(chk){
        Blynk.virtualWrite(V4,humidity);
        Blynk.virtualWrite(V5,temperature);
    }
    Blynk.virtualWrite(V6,light);
}
```

## 7.6 6. Moniteur de Plantes

L'objectif de ce projet est de créer un système d'arrosage intelligent qui détecte la température actuelle, l'humidité, l'intensité de la lumière et l'humidité du sol et les affiche sur Blynk.

Dès que vous activez le basculeur Switch dans Blynk Cloud, la pompe commencera à fonctionner et les plantes seront hydratées.

### Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

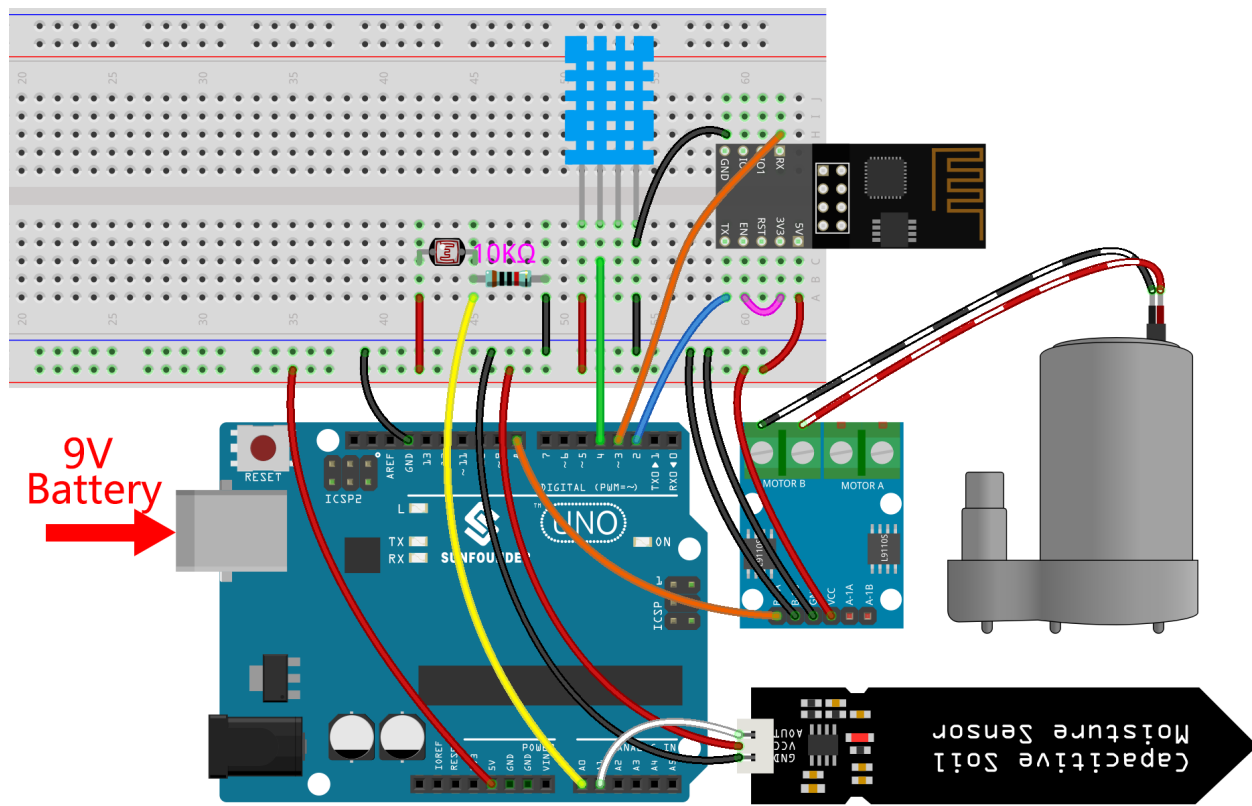
Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Module ESP8266</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Photorésistance</i>	
<i>Capteur d'Humidité et de Température DHT11</i>	-
<i>Module d'Humidité du Sol</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Pompe Centrifuge</i>	-

## 1. Construire le Circuit

**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc assurez-vous que la batterie 9V est branchée.



## 2. Éditer le Tableau de Bord

1. Les data streams créés dans les projets précédents doivent être conservés, et ils seront utilisés dans ce projet également.
2. Pour enregistrer l'humidité du sol, créez un autre **Datastream** de type **Virtual Pin** sur la page **Datastream**. Réglez le TYPE DE DONNÉES sur Integer et MIN et MAX sur 0 et 1024.

**Virtual Pin Datastream**

NAME:  ALIAS:

PIN:  DATA TYPE:

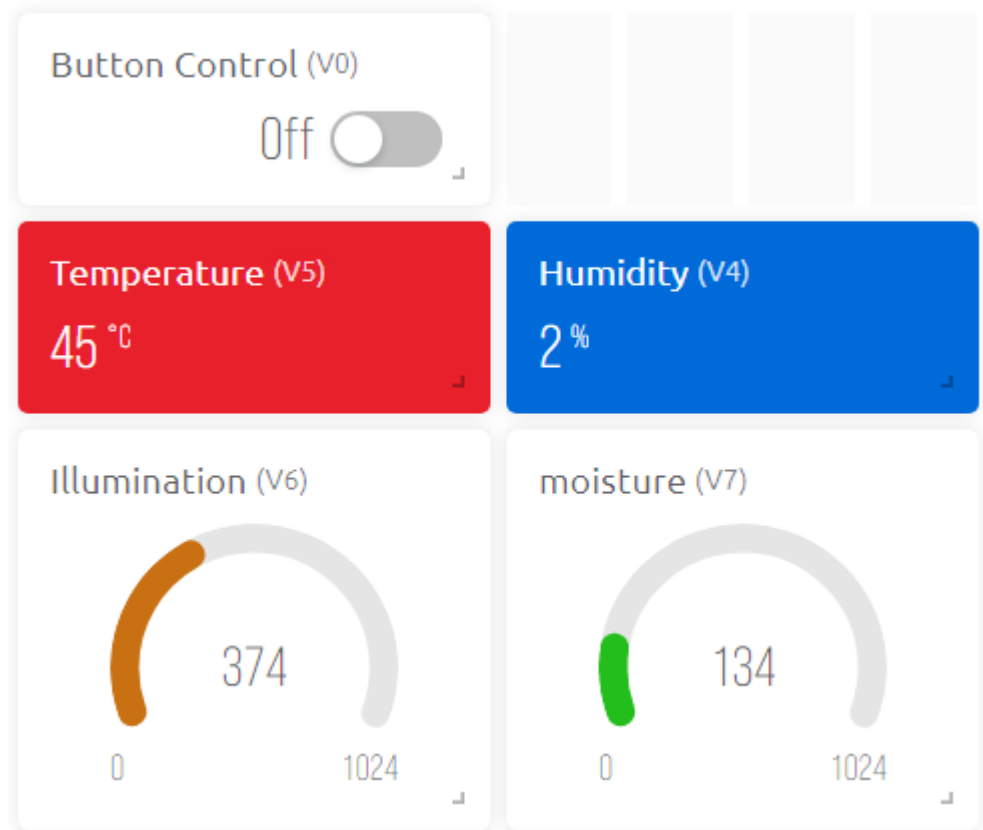
UNITS:

MIN:  MAX:  DEFAULT VALUE:

[+ ADVANCED SETTINGS](#) [Cancel](#) [Create](#)

Region: ny3 [Privacy Policy](#)

3. Maintenant, allez sur la page **Web Dashboard**, faites glisser 2 widgets **Label** et réglez leurs data streams respectivement sur **V4** et **V5**; faites glisser 2 widgets **Gauge** et réglez leurs data streams pour afficher **V6** et **V7** respectivement; et enfin faites glisser un widget **Switch** et réglez son data stream sur **V0**.

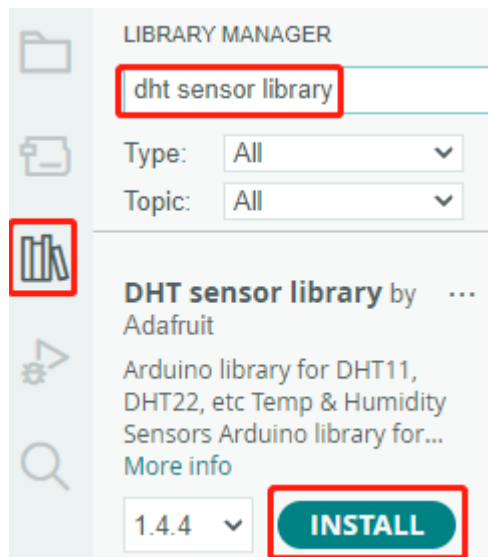


### 3. Exécutez le Code

Ouvrez le fichier `6.plant_monitoring.ino` situé dans le dossier `3in1-kit\iot_project\6.plant_monitoring`, ou copiez ce code dans **Arduino IDE**.

#### Note :

— La DHT sensor library est utilisée ici, vous pouvez l'installer depuis le **Library Manager**.



1. Remplacez le Template ID, Device Name et Auth Token par les vôtres. Vous devez également entrer le

ssid et le mot de passe du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).

- Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.
- Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.

```

COM63

#StandWithUkraine https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.75"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).

☒ Autoscroll ☐ Show timestamp
Both NL & CR 115200 baud Clear output

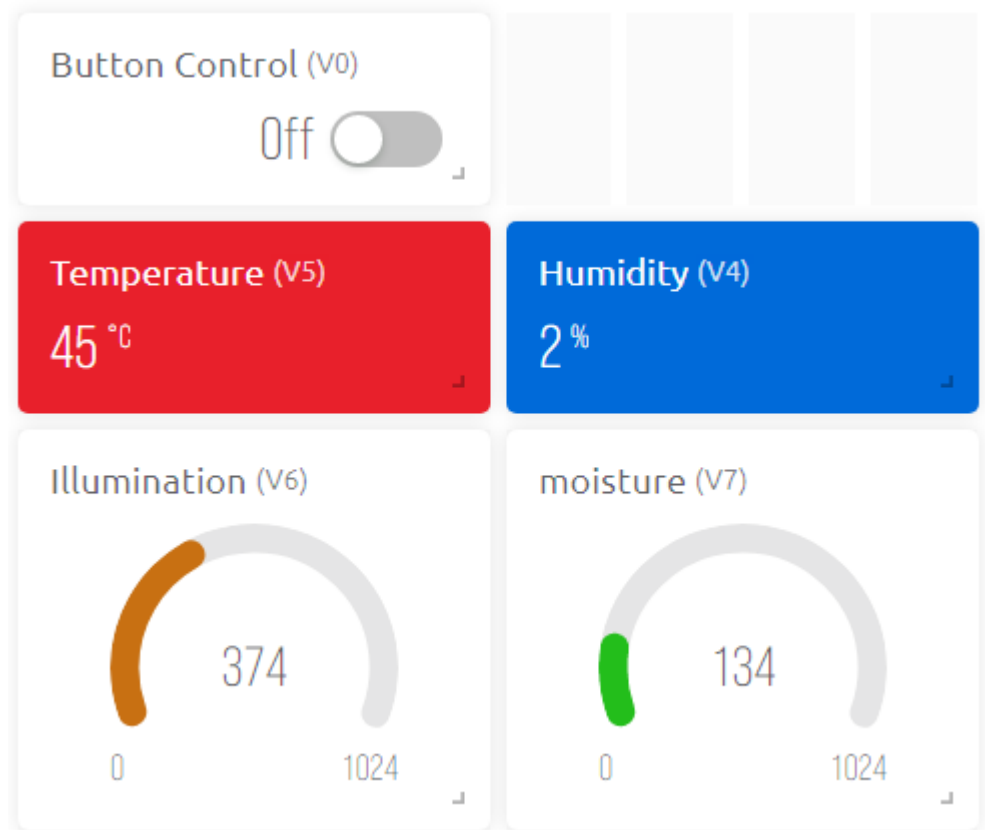
```

**Note :** Si le message ESP is not responding apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

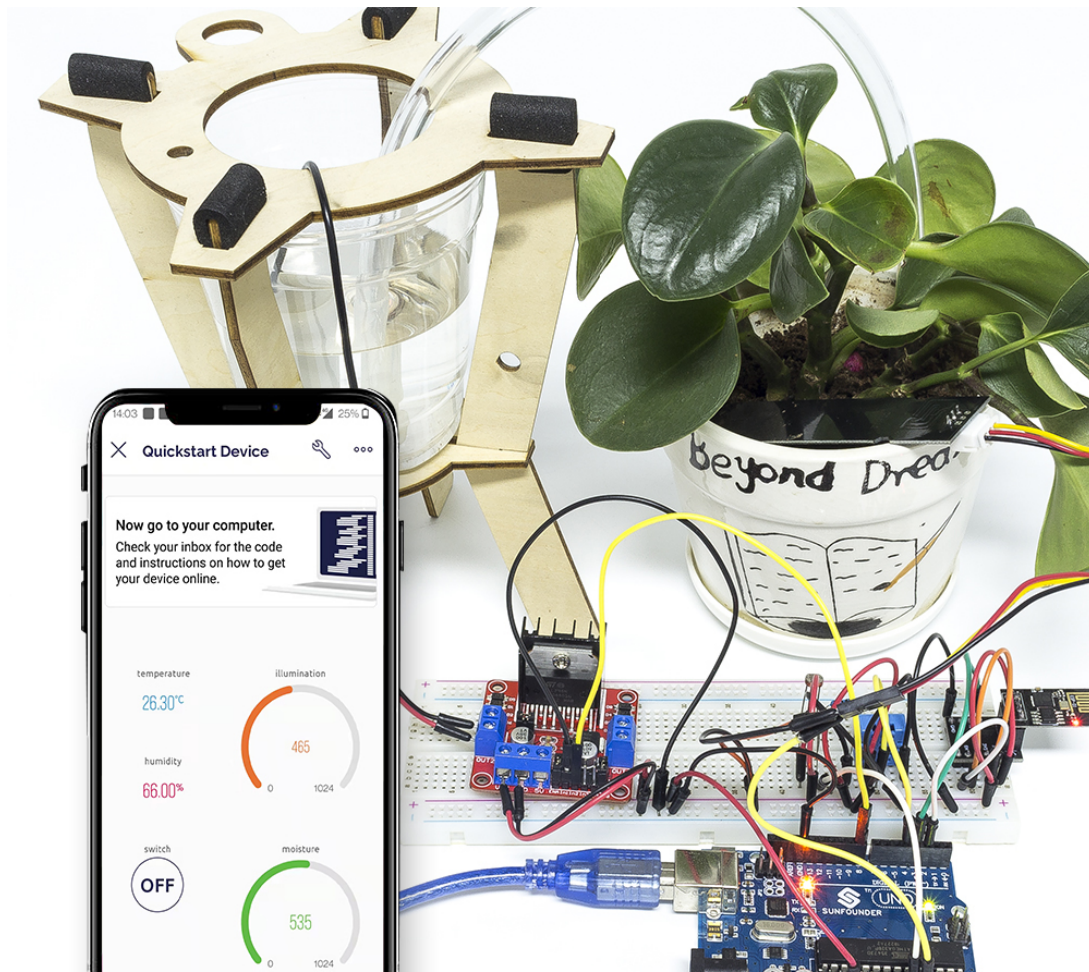
- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

- De retour sur Blynk, vous verrez la température actuelle, l'humidité, l'intensité lumineuse et l'humidité du sol. Vous pouvez laisser la pompe arroser les plantes si nécessaire en cliquant sur le widget Bouton de Contrôle.



5. Si vous souhaitez utiliser Blynk sur des appareils mobiles, veuillez vous référer à *Comment utiliser Blynk sur un appareil mobile ?*.



### Comment ça fonctionne ?

Ce BLYNK\_WRITE permet au widget **Switch** de Blynk de démarrer la pompe lorsqu'il est ON et de l'éteindre lorsqu'il est OFF.

```
BLYNK_WRITE(V0)
{
  if(param.asInt()==1){
    digitalWrite(pumpA,HIGH);
  }else{
    digitalWrite(pumpA,LOW);
  }
}
```

Ces trois fonctions sont utilisées pour obtenir la température ambiante actuelle, l'humidité, l'intensité lumineuse et l'humidité du sol.

```
int readMoisture(){
  return analogRead(moisturePin);
}

int readLight(){
  return analogRead(lightPin);
}
```

(suite sur la page suivante)



(suite de la page précédente)

```

}

bool readDHT() {

    // Reading temperature or humidity takes about 250 milliseconds!
    // Sensor readings may also be up to 2 seconds 'old' (it's a very slow sensor)
    humidity = dht.readHumidity();
    // Read temperature as Celsius (the default)
    temperature = dht.readTemperature();

    // Check if any reads failed and exit early (to try again).
    if (isnan(humidity) || isnan(temperature)) {
        Serial.println("Failed to read from DHT sensor!");
        return false;
    }
    return true;
}

```

Avec le Timer de Blynk, la température ambiante, l'humidité, l'intensité lumineuse et l'humidité du sol sont obtenues chaque seconde et envoyées au data stream sur le **Blynk Cloud**, à partir duquel les widgets affichent les données.

```

void myTimerEvent()
{
    bool chk = readDHT();
    int light = readLight();
    int moisture = readMoisture();
    if(chk){
        Blynk.virtualWrite(V4,humidity);
        Blynk.virtualWrite(V5,temperature);
    }
    Blynk.virtualWrite(V6,light);
    Blynk.virtualWrite(V7,moisture);
}

```

## 7.7 7. Portail à Limite de Courant

Certaines situations, comme les parkings, nécessitent une gestion de quantité.

Ici, nous créons un portail intelligent : un servo est utilisé comme portail, et un détecteur d'obstacles IR est placé devant ; si un objet (comme une voiture) est détecté, le portail s'ouvre et le nombre est augmenté de 1. Le compte est affiché avec un afficheur à 7 segments et est également téléchargé sur le Cloud Blynk pour que vous puissiez le visualiser à distance. Enfin, Blynk possède un widget Interrupteur pour activer ou désactiver ce système de portail intelligent.

### Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

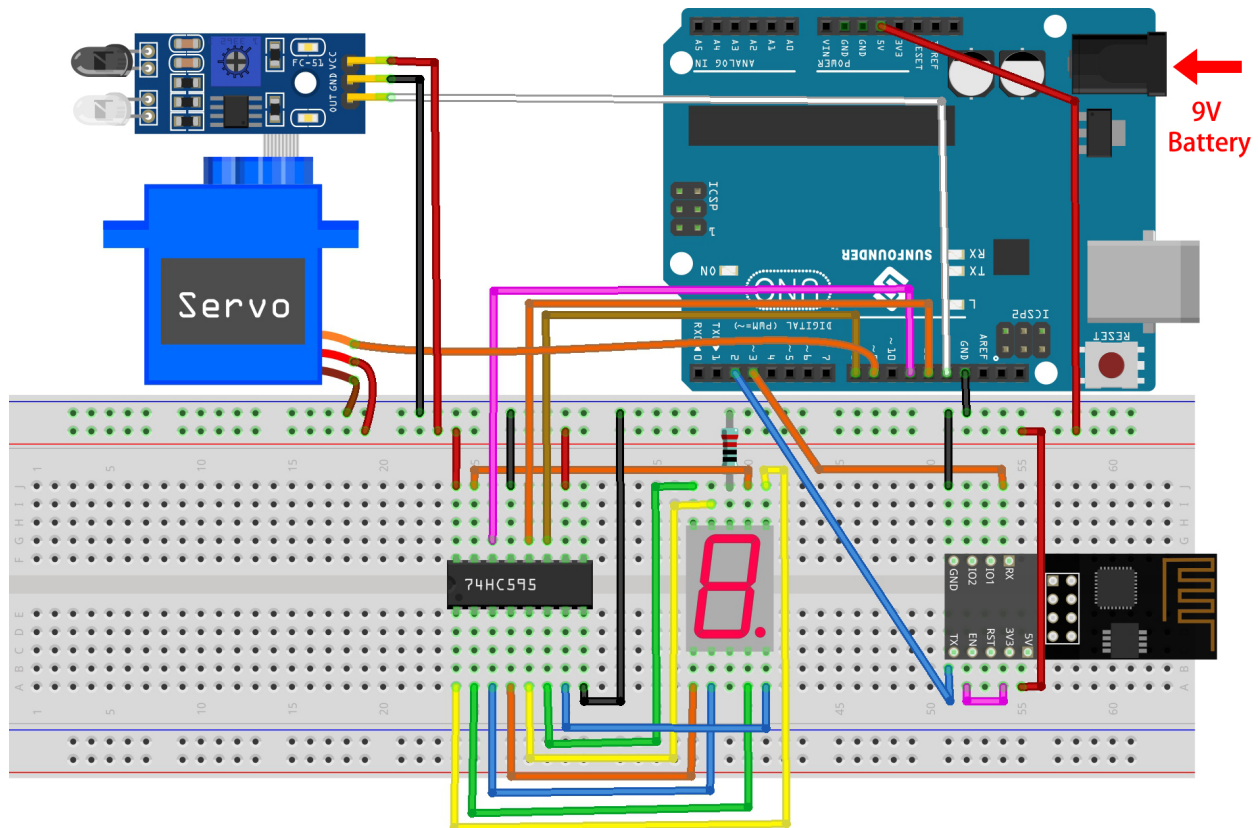
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Module ESP8266</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Servomoteur</i>	
<i>Module d'Évitement d'Obstacle</i>	
<i>Affichage 7 segments</i>	
<i>74HC595</i>	

### 1. Construire le Circuit

---

**Note :** Le module ESP8266 nécessite un courant élevé pour fournir un environnement de fonctionnement stable, donc assurez-vous que la batterie 9V est branchée.

---



## 2. Éditer le Tableau de Bord

1. Pour enregistrer le nombre, créez un **Datastream** de type **Virtual Pin** sur la page **Datastream**. Réglez le TYPE DE DONNÉES sur Integer et MIN et MAX sur 0 et 10.

**Virtual Pin Datastream**

NAME:  ALIAS:

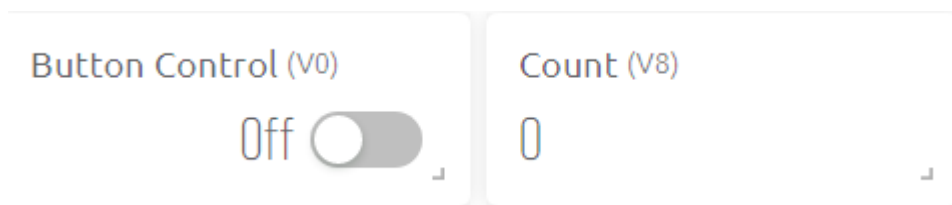
PIN:  DATA TYPE:

UNITS:

MIN:  MAX:  DEFAULT VALUE:

Region: ny3 [Privacy Policy](#)

- Maintenant, allez sur la page **Web Dashboard**, faites glisser un widget **Switch** pour régler son data stream sur **V0** et un widget **Label** pour régler son data stream sur **V8**.



### 3. Exécutez le Code

- Ouvrez le fichier `7.current_limiting_gate.ino` situé dans le dossier `3in1-kit\iot_project\7.current_limiting_gate`, ou copiez ce code dans **Arduino IDE**.
- Remplacez le Template ID, Device Name et Auth Token par les vôtres. Vous devez également entrer le ssid et le password du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).
- Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.
- Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.

```

COM63

#StandWithUkraine https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.75"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).

☒ Autoscroll ☐ Show timestamp
Both NL & CR 115200 baud Clear output

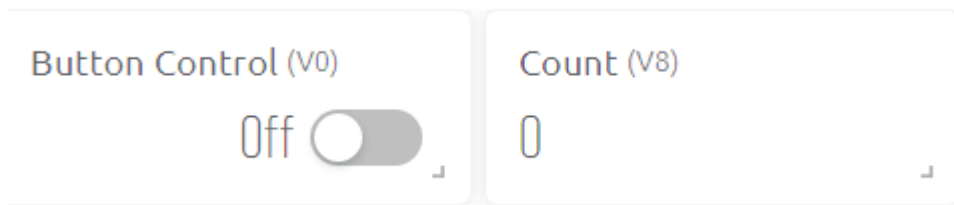
```

**Note :** Si le message ESP is not responding apparaît lorsque vous vous connectez, veuillez suivre ces étapes.

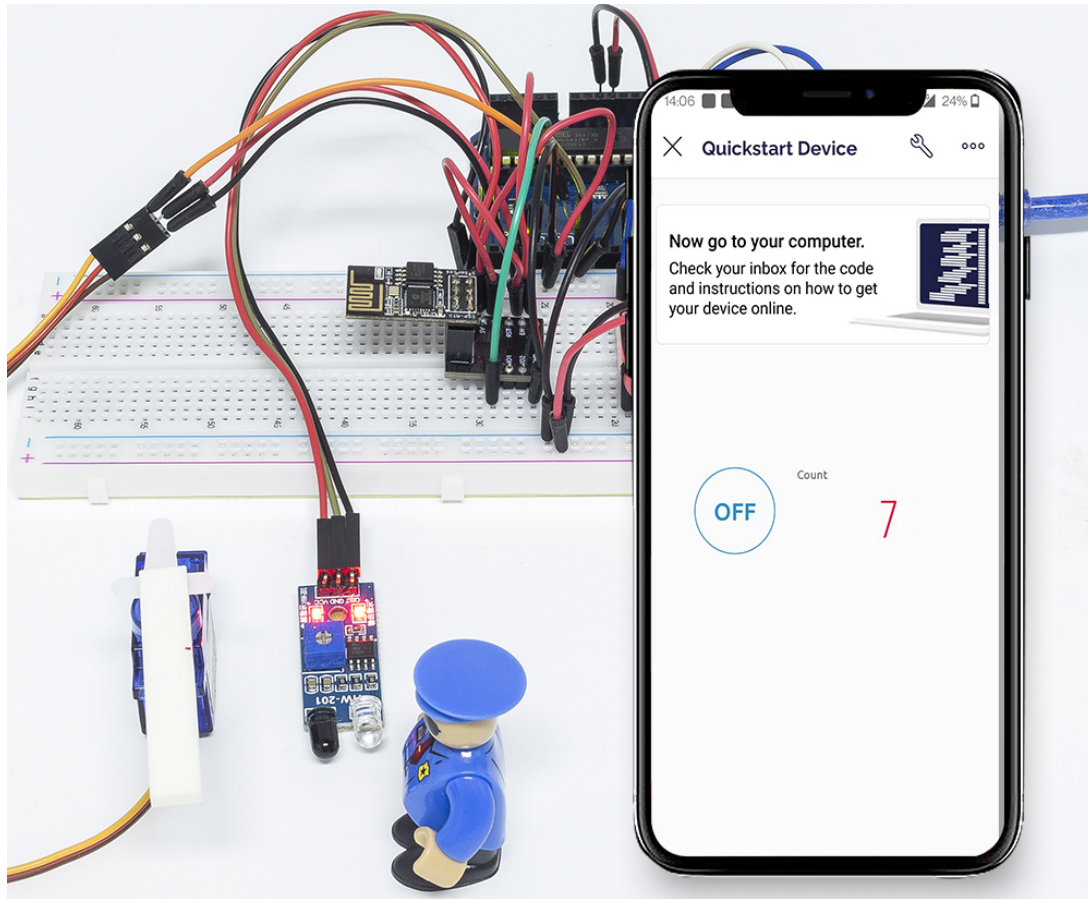
- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

5. Maintenant, cliquez sur le widget Bouton de Contrôle sur Blynk pour activer le système de porte intelligent. Si le module d'évitement d'obstacles IR détecte un obstacle, le portail s'ouvrira et l'afficheur à 7 segments et le widget Compteur sur Blynk ajouteront 1.



6. Si vous souhaitez utiliser Blynk sur des appareils mobiles, veuillez vous référer à *Comment utiliser Blynk sur un appareil mobile ?*.



### Comment ça fonctionne ?

La fonction `BLYNK_WRITE(V0)` obtient l'état du widget **Switch** et l'assigne à la variable `doorFlag`, qui sera utilisée pour déterminer si le système de portail intelligent est activé ou non.

```
BLYNK_WRITE(V0)
{
    doorFlag = param.asInt(); // Enable Gate
}
```

Dans le Timer Blynk, `doorFlag` est évalué chaque seconde et si activé, la fonction principale du portail est exécutée.

```
void myTimerEvent()
{
    if (doorFlag)
    {
        channelEntrance();
    }
}
```

La fonction principale du portail est `channelEntrance()`. Lorsqu'un objet s'approche du portail (le capteur détecte un obstacle), le count est augmenté de 1. Écrivez count dans le datastream V8 du Cloud Blynk et sur l'afficheur à 7 segments du circuit, et ouvrez le portail. Si l'objet passe de présent à absent, ce qui signifie que l'objet est entré par le portail, fermez le portail.

```

void channelEntrance()
{
    int currentState = digitalRead(irPin); // 0:obstacle 1:no-obstacle
    if (currentState == 0 && lastState == 1) {
        count=(count+1)%10;
        Blynk.virtualWrite(V8, count);
        showNumber(count);
        operateGate(true);
    } else if ((currentState == 1 && lastState == 0)) {
        operateGate(false);
    }
    lastState = currentState;
}

```

La fonction showNumber(int num) est utilisée pour faire afficher la valeur par l'afficheur à 7 segments.

```

void showNumber(int num)
{
    digitalWrite(STcp, LOW); //ground ST_CP and hold low for as long as you are_
    ↪transmitting
    shiftOut(DS, SHcp, MSBFIRST, dataArray[num]);
    digitalWrite(STcp, HIGH); //pull the ST_CPST_CP to save the data
}

```

La fonction operateGate(bool openGate) ouvre lentement la porte lorsque la référence est True, et ferme lentement la porte lorsque la référence est False.

```

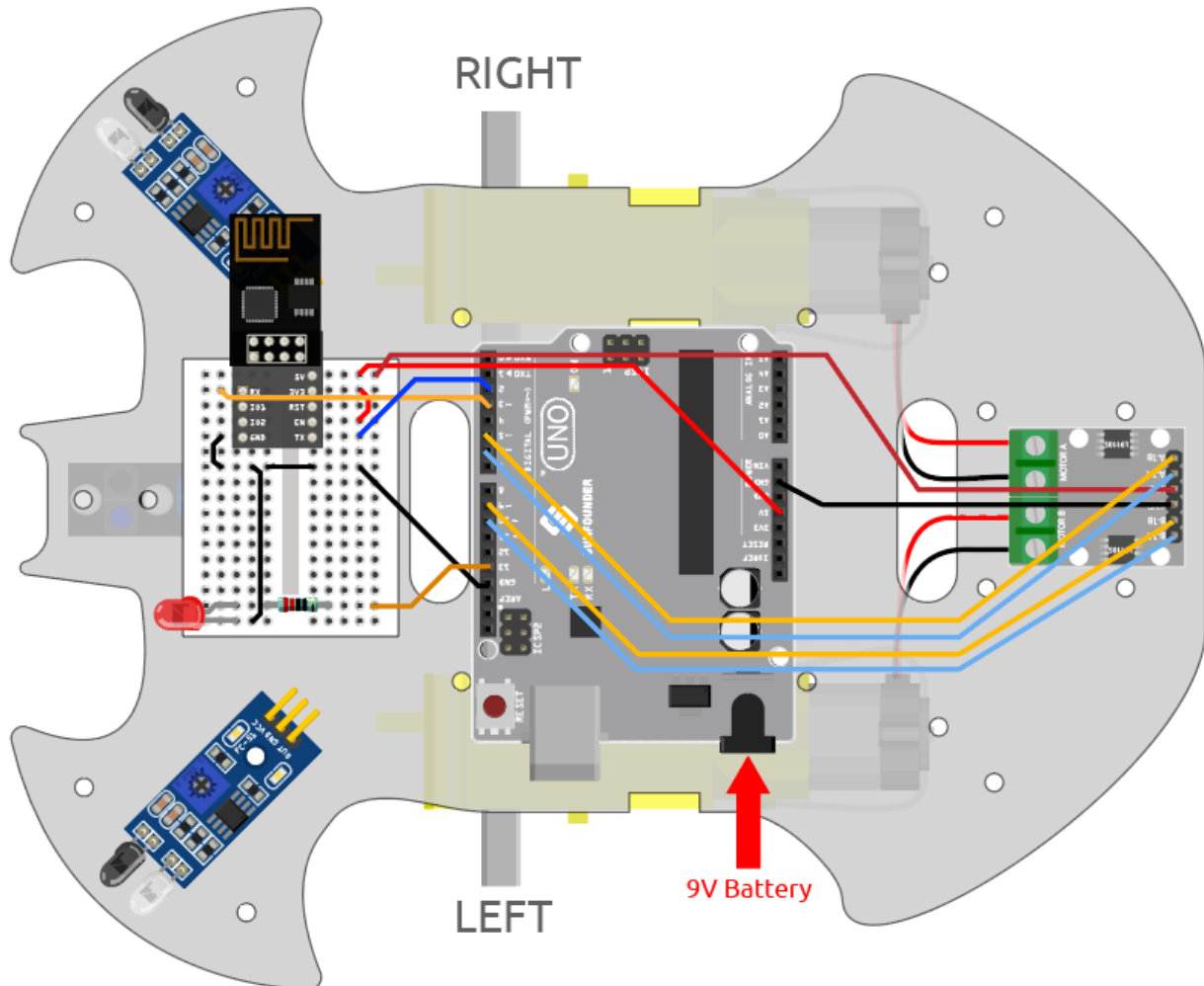
void operateGate(bool openGate) {
    if (openGate == true)
    {
        // open gate
        while (angle <= 90) {
            angle++;
            myservo.write(angle);
            delay(5);
        }
    } else {
        // close gate
        while (angle >= 0){
            angle--;
            myservo.write(angle);
            delay(5);
        }
    }
}

```

## 7.8 8. Voiture IoT

Pour ce projet, nous avons utilisé l'application Blynk sur le téléphone portable pour contrôler la voiture. Mais vous devez vous référer à [Projets de Voiture](#) pour assembler la voiture et pour en comprendre les bases. À l'ère de la popularité du réseau 5G, ce mode pourrait devenir l'une des principales méthodes de production dans de nombreuses industries, alors expérimentons ce jeu en avance.

### 1. Construire le Circuit



### 2. Éditer le Tableau de Bord

Blynk sur mobile ne peut pas éditer les Datastream, nous devons donc toujours effectuer ces étapes sur le côté web.

1. Créez un **Datastream** de type **Virtual Pin** sur la page **Datastream**, pour enregistrer la valeur de l'axe X du joystick. Nommez-le **Xvalue**, réglez le **TYPE DE DONNÉES** sur **Integer**, et **MIN** et **MAX** sur **-10** et **10**.



**Virtual Pin Datastream**

NAME: Xvalue ALIAS: Xvalue

PIN: V9 DATA TYPE: Integer

UNITS: None

MIN: -10 MAX: 10 DEFAULT VALUE: 0

+ ADVANCED SETTINGS Cancel Create

2. Créez un **Datastream** de type **Virtual Pin** pour enregistrer la valeur de l'axe Y du joystick. Nommez-le Yvalue, réglez le TYPE DE DONNÉES sur Integer, MIN et MAX sur -10 et 10.

**Virtual Pin Datastream**

NAME: Yvalue ALIAS: Yvalue

PIN: V10 DATA TYPE: Integer

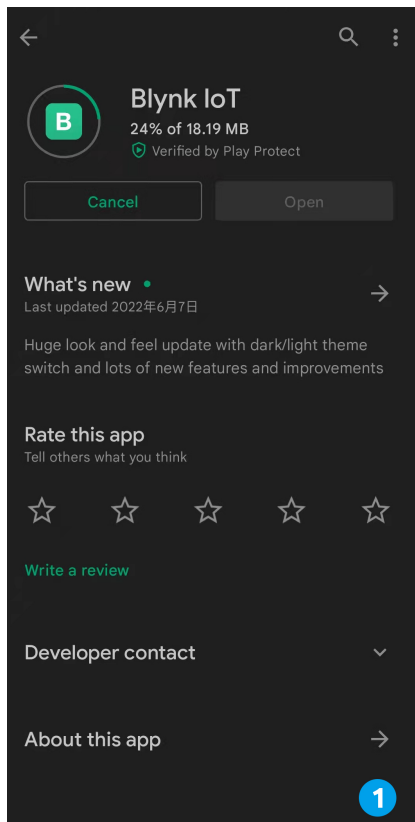
UNITS: None

MIN: -10 MAX: 10 DEFAULT VALUE: 0

[+ ADVANCED SETTINGS](#) [Cancel](#) [Create](#)

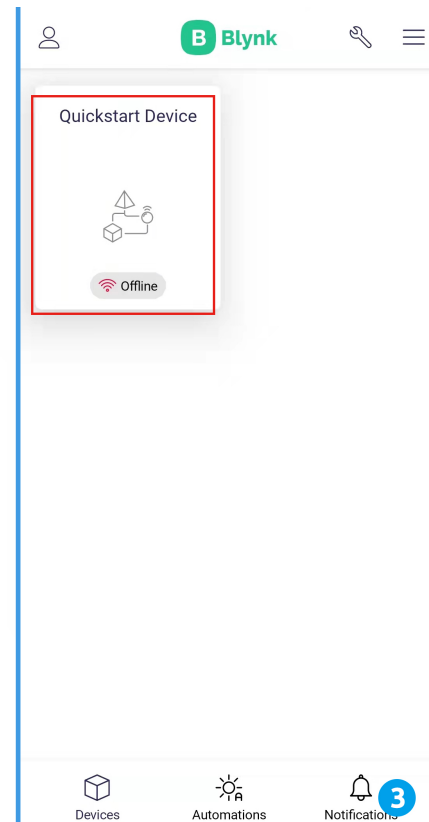
Ensuite, vous devez faire ce qui suit sur votre téléphone.

1. Recherchez « Blynk IoT » (pas Blynk(legacy)) sur GOOGLE Play ou l'APP Store pour le télécharger.
2. Après avoir ouvert l'APP, connectez-vous, ce compte doit être le même que celui utilisé sur le client web.
3. Allez ensuite sur le tableau de bord (si vous n'en avez pas, créez-en un) et vous verrez que les tableaux de bord mobiles et web sont indépendants l'un de l'autre.

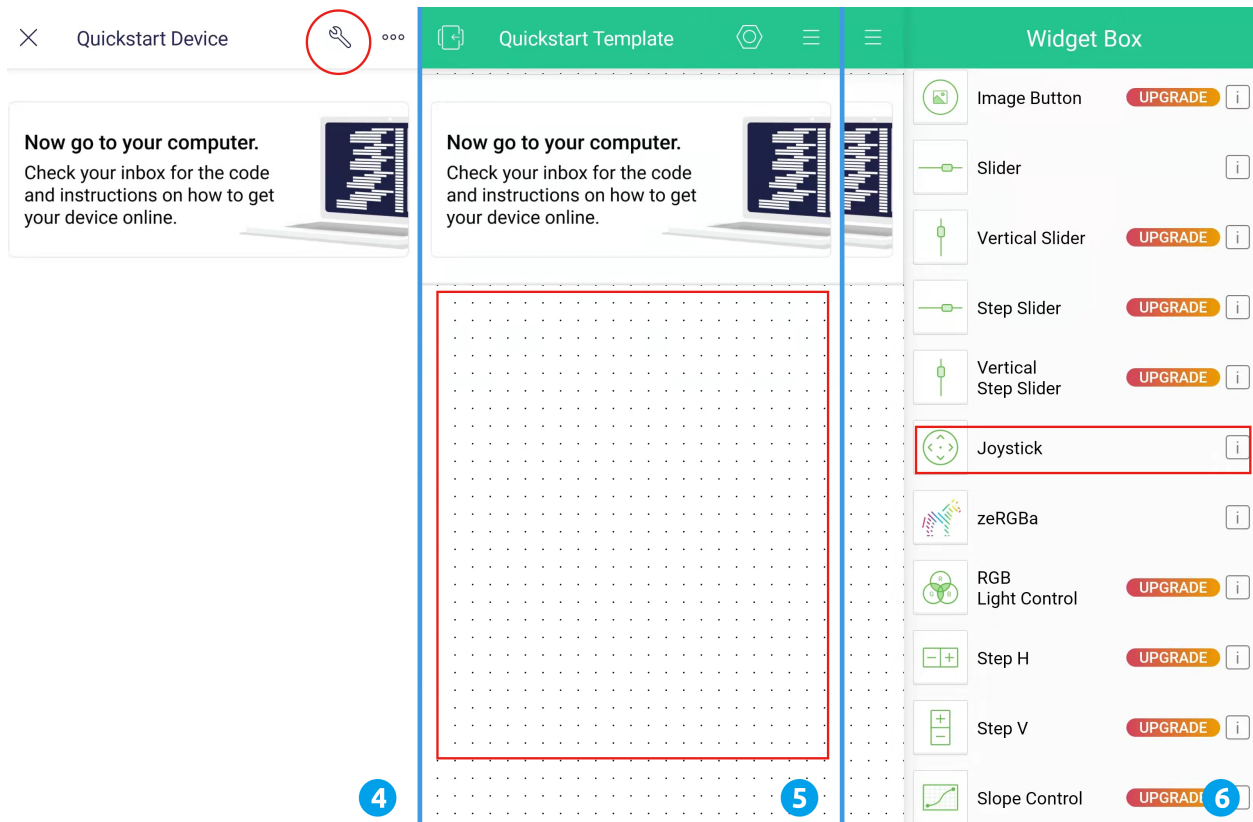


Sign Up

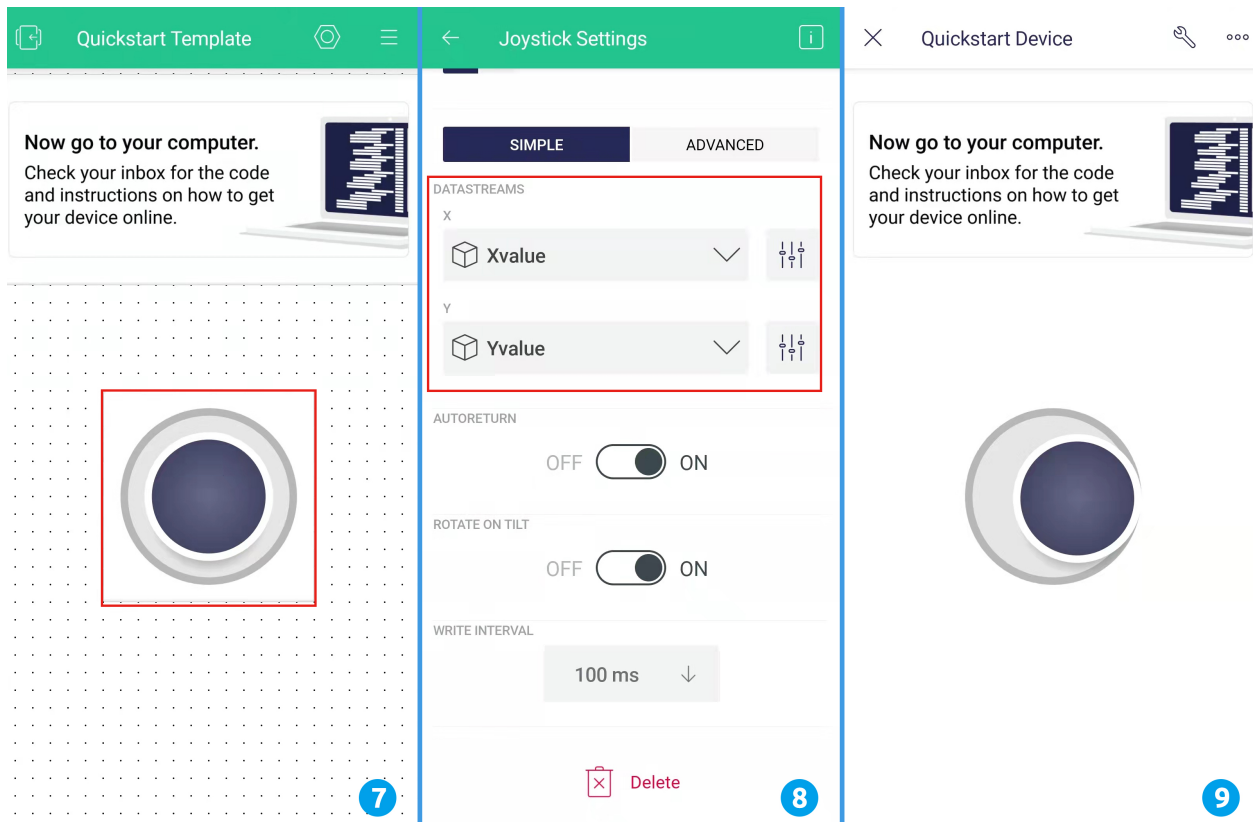
Log In



4. Cliquez sur l'icône Éditer.
5. Cliquez sur la zone vide.
6. Sélectionnez un widget Joystick.

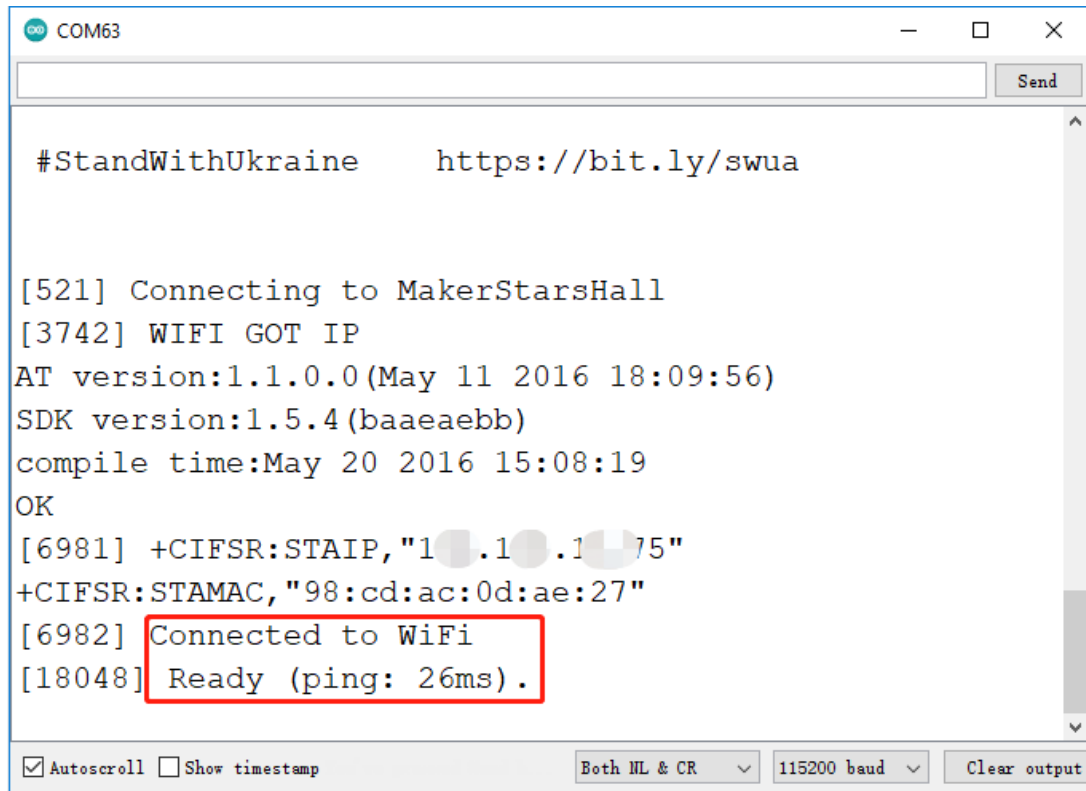


7. Maintenant, vous verrez un widget Joystick apparaître dans la zone vide, cliquez dessus.
8. Les paramètres du Joystick apparaîtront, sélectionnez les Xvalue et Yvalue que vous venez de régler dans les datastreams.
9. Retournez sur la page du tableau de bord et vous pourrez utiliser le Joystick quand vous le souhaitez.



### 3. Exécutez le Code

1. Ouvrez le fichier `8.iot_car.ino` situé dans le dossier `3in1-kit\iot_project\8.iot_car`, ou copiez ce code dans **Arduino IDE**.
2. Remplacez le **Template ID**, **Device Name** et **Auth Token** par les vôtres. Vous devez également entrer le **ssid** et le **password** du WiFi que vous utilisez. Pour des tutoriels détaillés, veuillez vous référer à [1.4 Connexion de la carte R3 à Blynk](#).
3. Après avoir sélectionné la bonne carte et le bon port, cliquez sur le bouton **Upload**.
4. Ouvrez le moniteur série (réglez le débit en bauds sur 115200) et attendez qu'un message tel qu'une connexion réussie apparaisse.



```
COM63

#StandWithUkraine    https://bit.ly/swua

[521] Connecting to MakerStarsHall
[3742] WIFI GOT IP
AT version:1.1.0.0(May 11 2016 18:09:56)
SDK version:1.5.4(baaeaebb)
compile time:May 20 2016 15:08:19
OK
[6981] +CIFSR:STAIP,"192.168.1.175"
+CIFSR:STAMAC,"98:cd:ac:0d:ae:27"
[6982] Connected to WiFi
[18048] Ready (ping: 26ms).
```

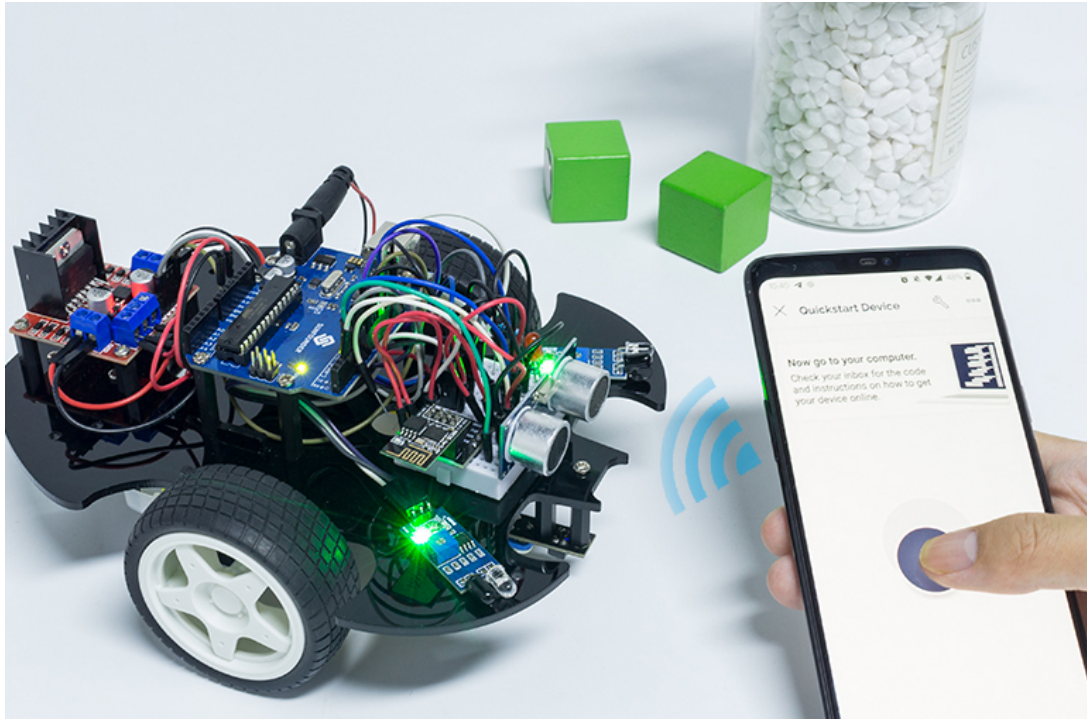
☒ Autoscroll ☐ Show timestamp    Both NL & CR    115200 baud    Clear output

**Note :** Si le message ESP is not responding apparaît lors de la connexion, veuillez suivre ces étapes.

- Assurez-vous que la batterie 9V est branchée.
- Réinitialisez le module ESP8266 en connectant la broche RST à GND pendant 1 seconde, puis débranchez-la.
- Appuyez sur le bouton de réinitialisation de la carte R3.

Parfois, il peut être nécessaire de répéter l'opération ci-dessus 3 à 5 fois, veuillez être patient.

5. Maintenant, débranchez le câble USB et alimentez la voiture avec une batterie 9V uniquement, puis attendez que la LED s'allume, indiquant que la voiture est connectée à Blynk.
6. Ouvrez Blynk sur votre téléphone et vous pouvez utiliser le widget Joystick pour contrôler le mouvement de la voiture.



### Comment ça fonctionne ?

Ces fonctions sont utilisées pour contrôler le mouvement de la voiture.

```
void moveForward(int speed) {...}
void moveBackward(int speed) {...}
void turnRight(int speed) {...}
void turnLeft(int speed) {...}
void stopMove() {...}
```

La section IoT lit les valeurs du widget Joystick et les assigne aux variables Xvalue et Yvalue.

```
int Xvalue = 0;
int Yvalue = 0;

BLYNK_WRITE(V9)
{
    Xvalue = param.asInt();
}

BLYNK_WRITE(V10)
{
    Yvalue = param.asInt();
}
```

Dans loop(), faites exécuter différentes actions à la voiture en fonction des Xvalue et Yvalue.

```
if (Yvalue >= 5) {
    moveForward(255);
} else if (Yvalue <= -5) {
    moveBackward(255);
}
```

(suite sur la page suivante)

(suite de la page précédente)

```
} else if (Xvalue >= 5) {  
    turnRight(150);  
}  
else if (Xvalue <= -5) {  
    turnLeft(150);  
}  
else {  
    stopMove();  
}
```

Ajoutez également une détermination de l'état du réseau à `loop()` pour allumer une LED si elle est connectée au Cloud Blynk.

```
if (!Blynk.connected()) {  
    digitalWrite(ledPin, LOW);  
    Serial.print("offline!");  
    bool result = Blynk.connect();  
    Serial.println(result);  
}  
else {  
    digitalWrite(ledPin, HIGH);  
}
```

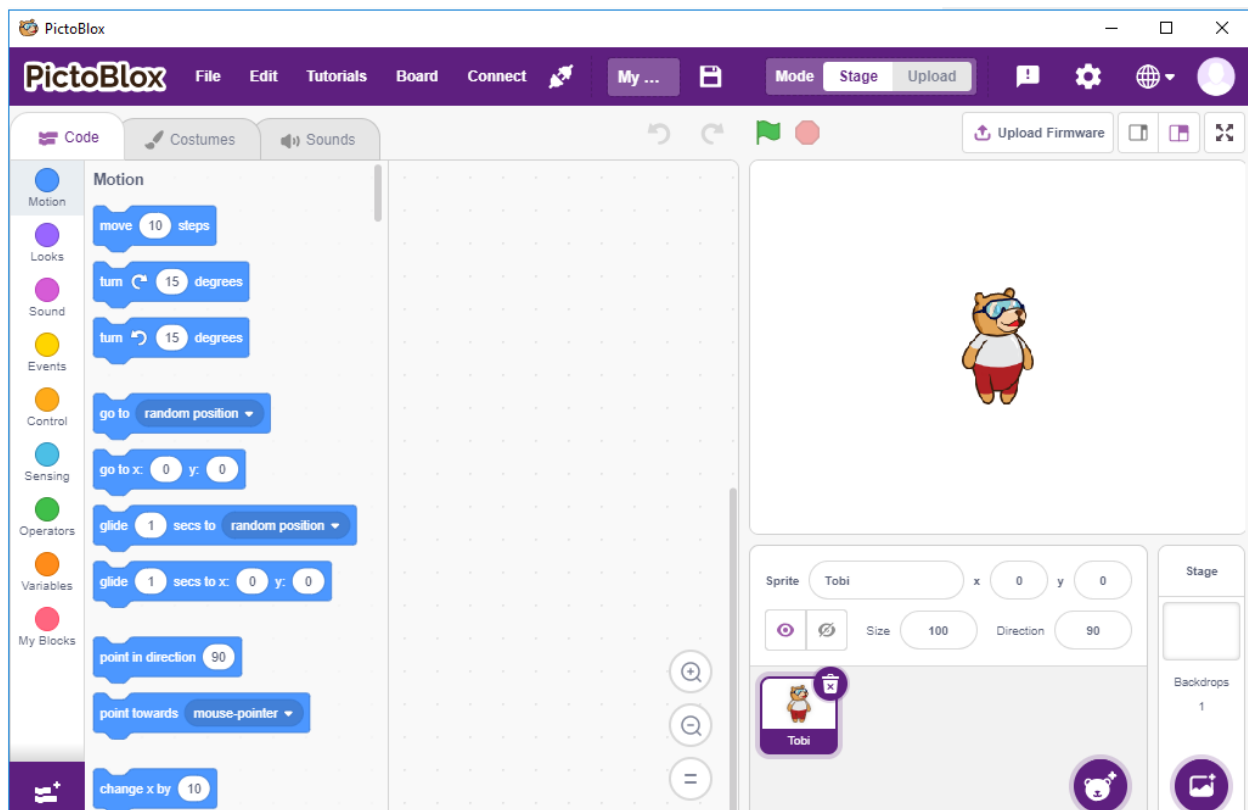


## CHAPITRE 8

### Jouez avec Scratch

Outre la programmation sur Arduino IDE, nous pouvons également utiliser la programmation graphique.

Ici, nous recommandons la programmation avec Scratch, mais le Scratch officiel est actuellement seulement compatible avec Raspberry Pi. Pour cela, nous nous sommes associés à une entreprise, STEMPedia, qui a développé un logiciel de programmation graphique basé sur Scratch 3 pour les cartes Arduino (Uno, Mega2560 et Nano) - [PictoBlox](#).



Il conserve les fonctions de base de Scratch 3, mais ajoute également des cartes de contrôle, telles qu'Arduino Uno,

Mega, Nano, ESP32, Microbit et les cartes principales faites maison de STEAMPedia, qui peuvent utiliser des capteurs externes, des robots pour contrôler les sprites sur la scène, avec de fortes capacités d'interaction matérielle.

De plus, il intègre l'IA et l'apprentissage automatique, même si vous n'avez pas beaucoup de bases en programmation, vous pouvez apprendre et utiliser ces technologies populaires et de pointe.

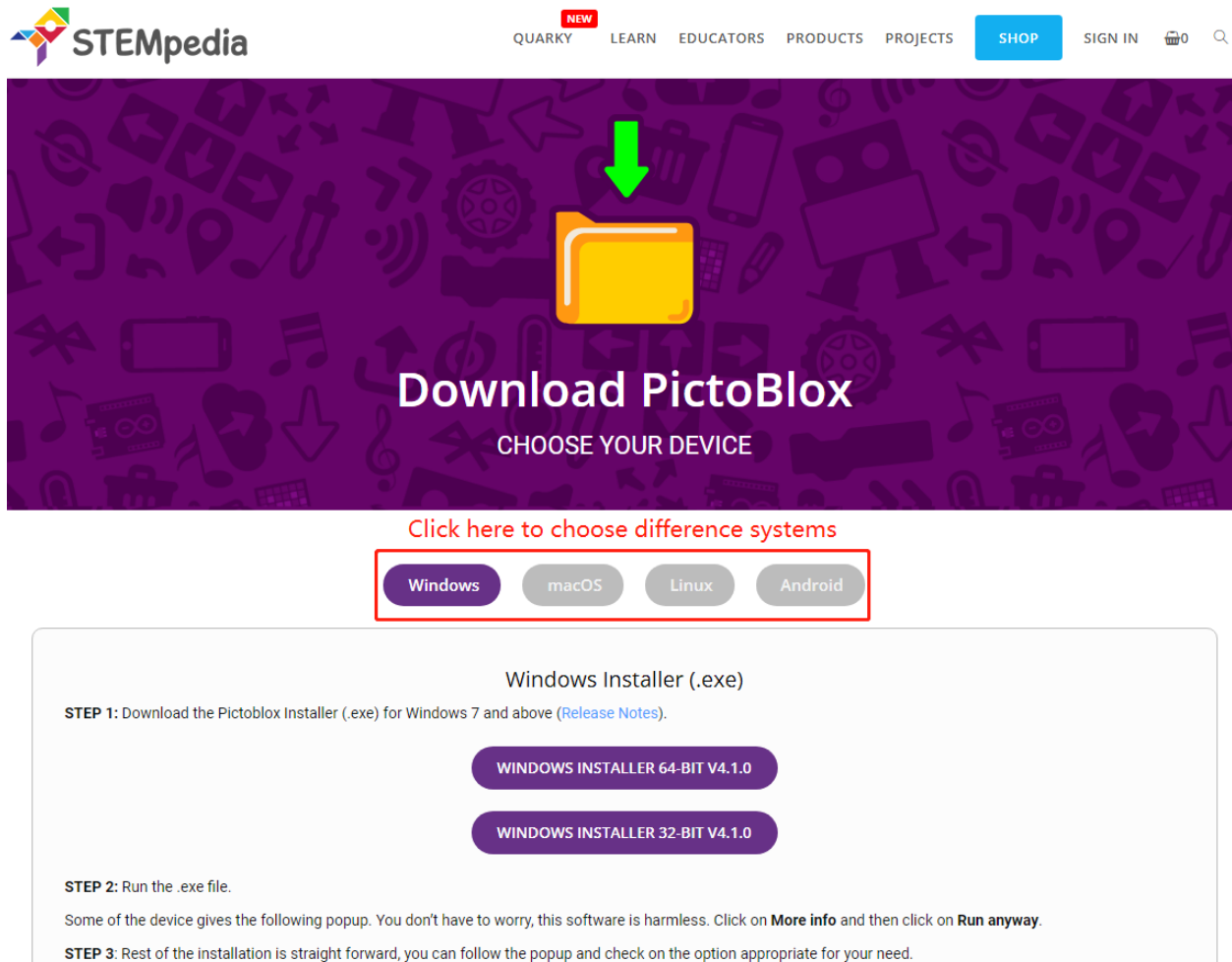
Glissez-déposez simplement les blocs de codage Scratch et créez des jeux sympas, des animations, des projets interactifs, et même contrôlez des robots comme vous le souhaitez !

Maintenant, commençons le voyage de découverte !

### 1. Commencer

## 8.1 1.1 Installer PictoBlox

Cliquez sur ce lien : <https://thestempedia.com/product/pictoblox/download-pictoblox/>, choisissez le système d'exploitation approprié (Windows, macOS, Linux) et suivez les étapes pour installer.



**STEMpedia** NEW QUARKY LEARN EDUCATORS PRODUCTS PROJECTS SHOP SIGN IN

**Download PictoBlox**  
CHOOSE YOUR DEVICE

Click here to choose difference systems

Windows macOS Linux Android

Windows Installer (.exe)

**STEP 1:** Download the Pictoblox Installer (.exe) for Windows 7 and above ([Release Notes](#)).

WINDOWS INSTALLER 64-BIT V4.1.0

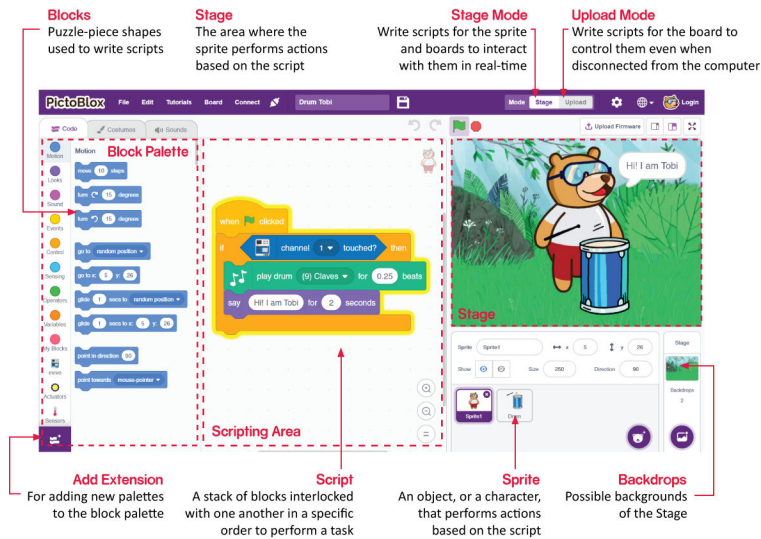
WINDOWS INSTALLER 32-BIT V4.1.0

**STEP 2:** Run the .exe file.

Some of the device gives the following popup. You don't have to worry, this software is harmless. Click on **More info** and then click on **Run anyway**.

**STEP 3:** Rest of the installation is straight forward, you can follow the popup and check on the option appropriate for your need.

## 8.2 1.2 Présentation de l'interface



### Sprites

Un sprite est un objet ou un personnage qui exécute différentes actions dans un projet. Il comprend et obéit aux commandes qui lui sont données. Chaque sprite possède des costumes et des sons spécifiques que vous pouvez également personnaliser.

### Stage

La scène est l'espace où le sprite effectue des actions sur des décors selon votre programme.

### Backdrops

Les décors sont utilisés pour décorer la scène. Vous pouvez choisir un décor dans PictoBlox, le dessiner vous-même ou télécharger une image depuis votre ordinateur.

### Script Area

Un script est un programme ou un code dans le langage PictoBlox/Scratch. C'est un ensemble de « blocs » disposés dans un ordre spécifique pour effectuer une tâche ou une série de tâches. Vous pouvez écrire plusieurs scripts, qui peuvent tous s'exécuter simultanément. Vous ne pouvez écrire des scripts que dans la zone de script au centre de l'écran.

### Blocks

Les blocs sont comme des pièces d'un puzzle utilisées pour écrire des programmes en les empilant simplement dans la zone de script. Utiliser des blocs pour écrire du code peut faciliter la programmation et réduire la probabilité d'erreurs.

### Block Palette

Les palettes de blocs sont situées dans la zone de gauche et sont nommées selon leurs fonctions, telles que mouvement, son et contrôle. Chaque palette a des blocs différents, par exemple, les blocs dans la palette Mouvement contrôleront le mouvement des sprites, et les blocs dans la palette Contrôle régiront le fonctionnement du script en fonction de conditions spécifiques.

Il existe d'autres types de palettes de blocs qui peuvent être chargées à partir du bouton **Add Extension** situé en bas à gauche.

### Modes

Contrairement à Scratch, PictoBlox dispose de deux modes :

- *Mode Scène* : Dans ce mode, vous pouvez écrire des scripts pour que le sprite et les cartes interagissent avec les sprites en temps réel. Si vous déconnectez la carte de Pictoblox, vous ne pouvez plus interagir.

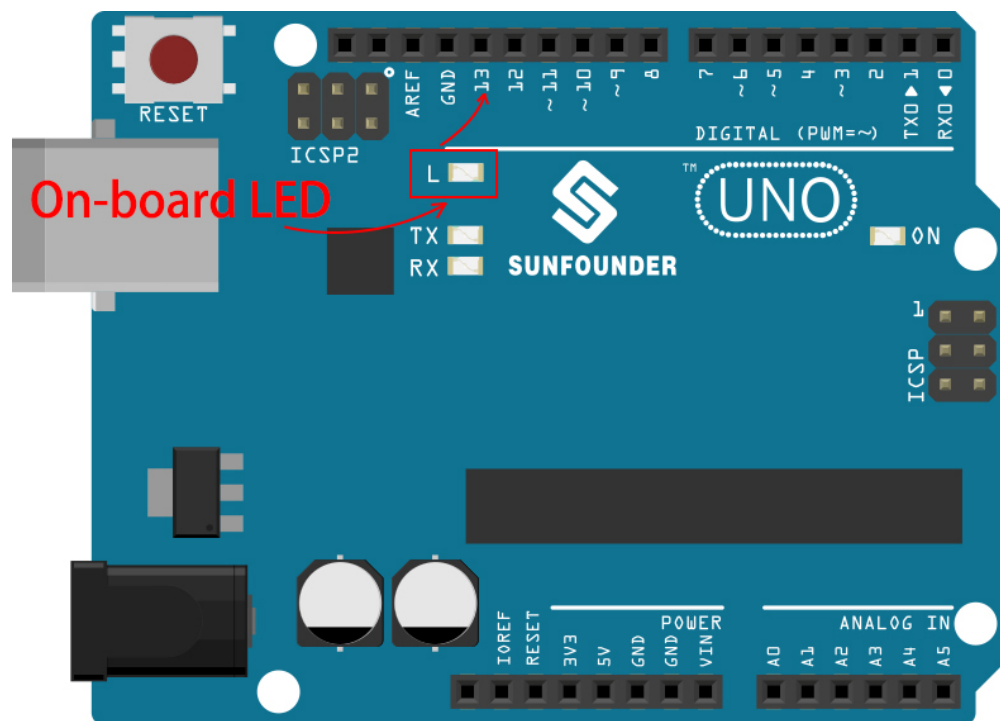
- *Mode Téléchargement* : Ce mode vous permet d'écrire des scripts et de les télécharger sur la carte pour que vous puissiez les utiliser même lorsqu'elle n'est pas connectée à votre ordinateur, par exemple, vous devez télécharger un script pour fabriquer des robots mobiles.

Pour plus d'informations, veuillez consulter : <https://thetempedia.com/tutorials/getting-started-pictoblox>

### 8.3 1.3 Guide rapide sur PictoBlox

Apprenons maintenant comment utiliser PictoBlox dans deux modes.

De plus, il y a une LED intégrée connectée à la broche 13 sur la carte R3, nous allons apprendre à faire clignoter cette LED dans 2 modes différents.

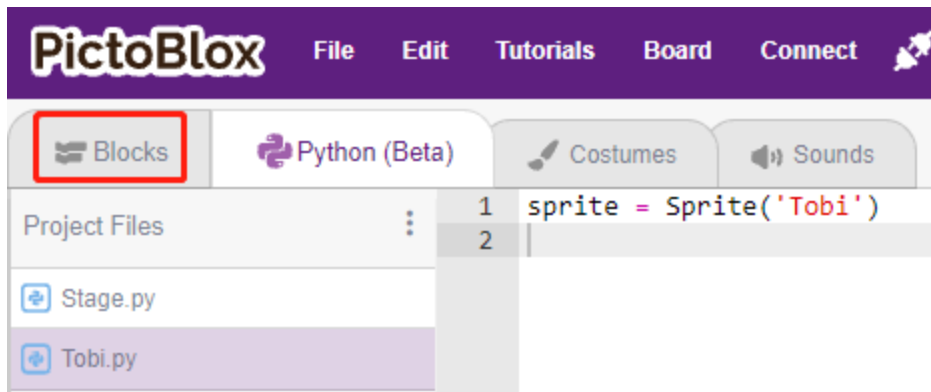


#### 8.3.1 Mode Scène

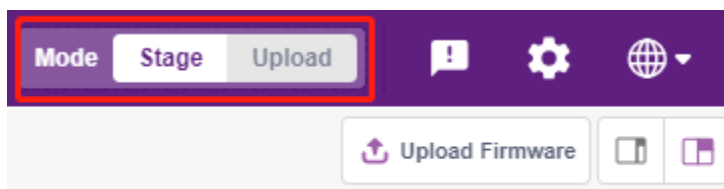
##### 1. Connecter à la carte Arduino

Connectez votre carte Arduino à l'ordinateur avec un câble USB, généralement l'ordinateur reconnaîtra automatiquement votre carte et finalement attribuera un port COM.

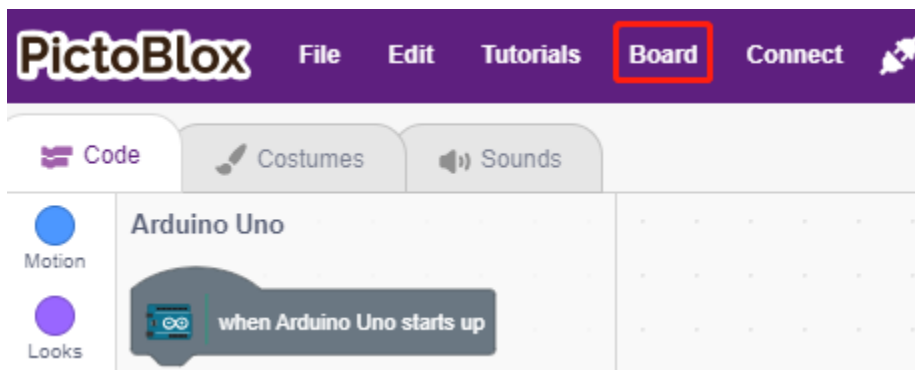
Ouvrez PictoBlox, l'interface de programmation Python s'ouvrira par défaut. Et nous devons passer à l'interface des Blocs.



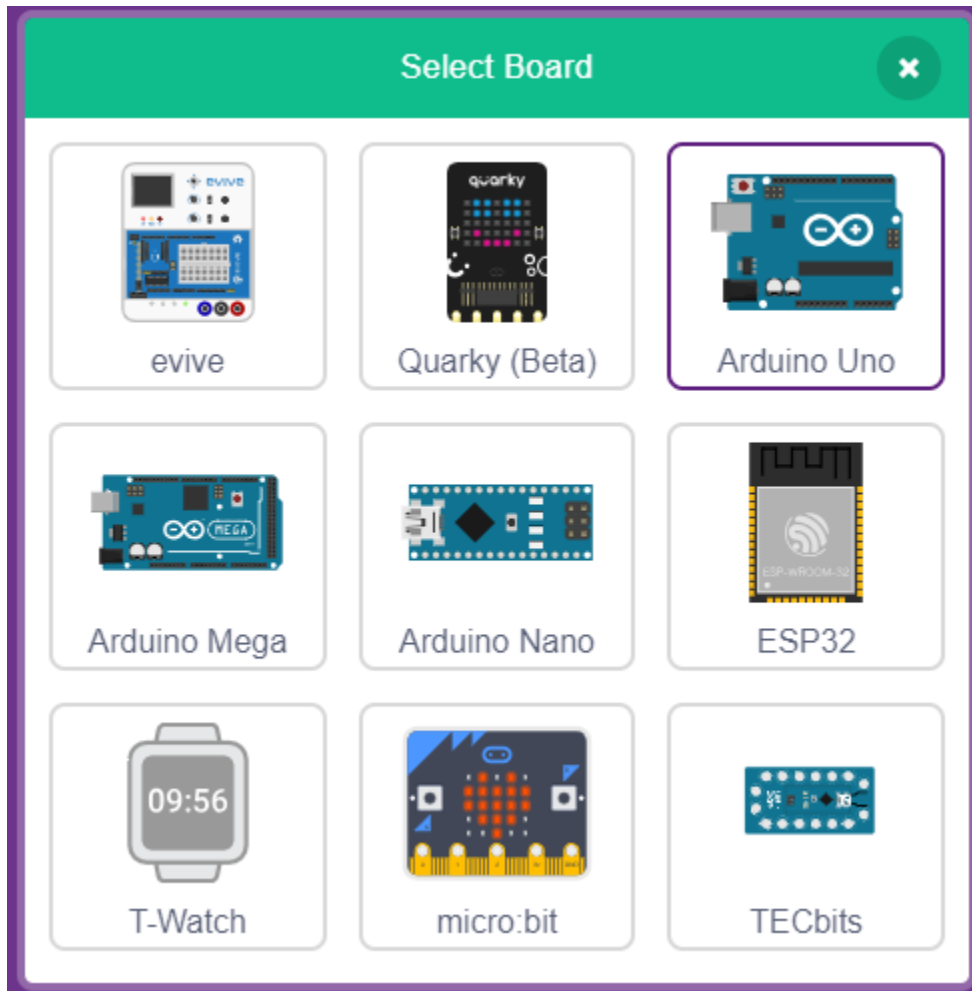
Vous verrez alors le coin supérieur droit pour le changement de mode. Le mode par défaut est le mode Scène, où Tobi se tient sur la scène.



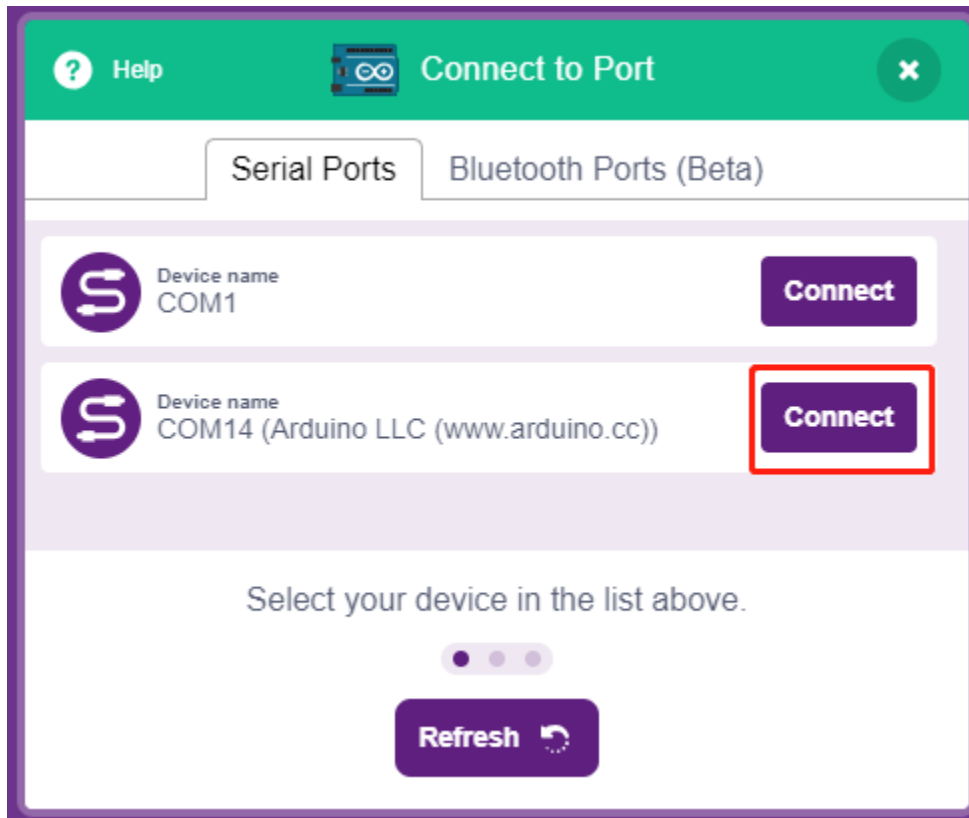
Cliquez sur **Board** dans la barre de navigation en haut à droite pour sélectionner la carte.



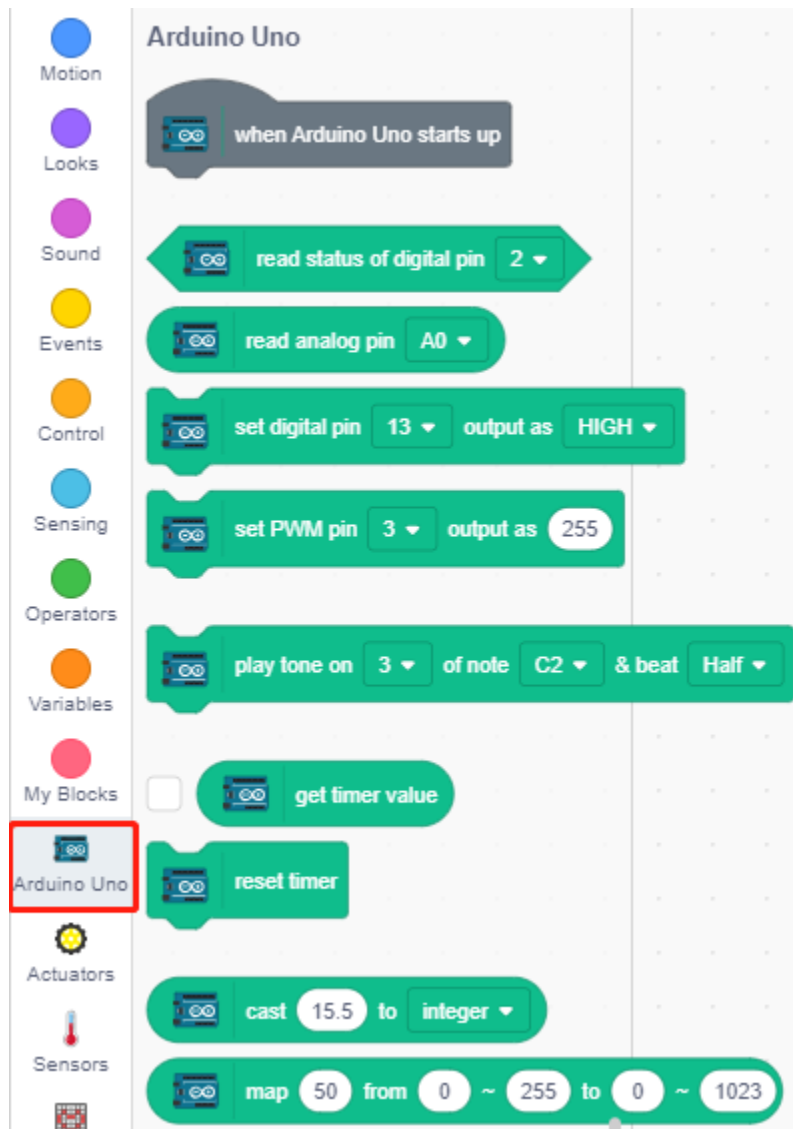
Par exemple, choisissez **Arduino Uno**.



Une fenêtre de connexion s'ouvrira alors pour que vous sélectionniez le port à connecter, et retournez à la page d'accueil lorsque la connexion est complète. Si vous interrompez la connexion pendant l'utilisation, vous pouvez également cliquer sur **Connect** pour vous reconnecter.



En même temps, les palettes liées à l'Arduino Uno, telles que Arduino Uno, Actuators, etc., apparaîtront dans la **Block Palette**.



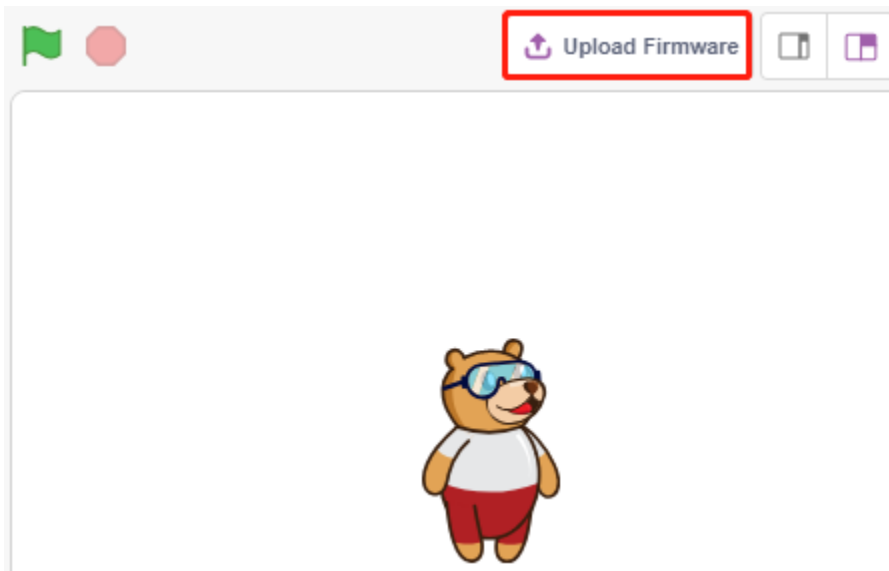
## 2. Télécharger le Firmware

Comme nous allons travailler dans le mode Scène, nous devons télécharger le firmware sur la carte. Cela garantira une communication en temps réel entre la carte et l'ordinateur. Télécharger le firmware est un processus ponctuel. Pour ce faire, cliquez sur le bouton Télécharger le Firmware.

Après avoir attendu un moment, le message de réussite de téléchargement apparaîtra.

**Note :** Si vous utilisez cette carte Arduino dans PictoBlox pour la première fois, ou si cet Arduino a été précédemment téléchargé avec l'IDE Arduino. Alors vous devez appuyer sur **Upload Firmware** avant de pouvoir l'utiliser.



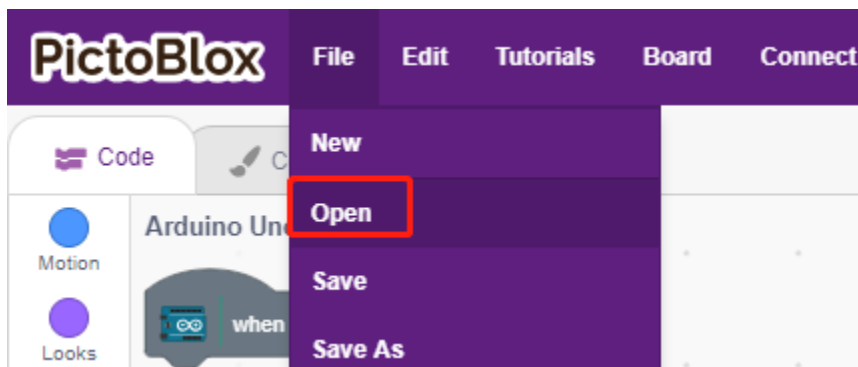


### 3. Programmation

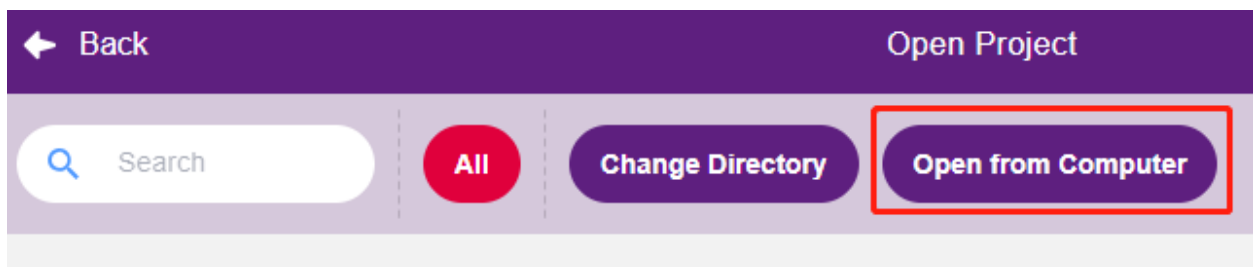
— Ouvrir et exécuter le script directement

Bien sûr, vous pouvez ouvrir directement les scripts pour les exécuter, mais veuillez d'abord les télécharger depuis [github](#).

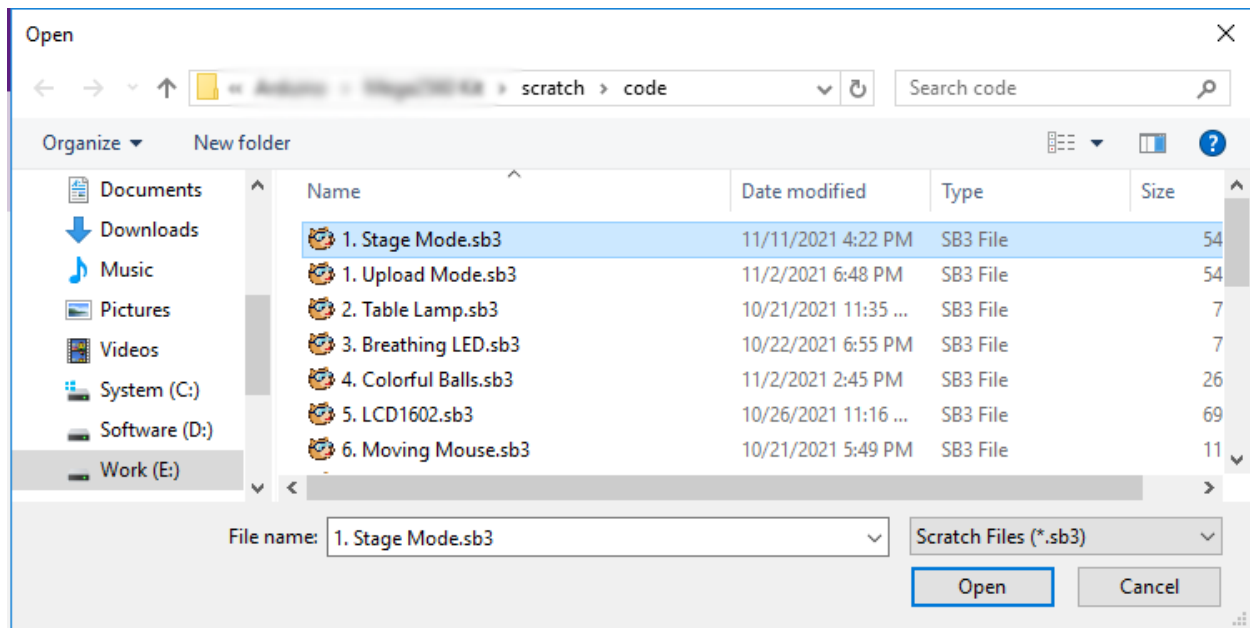
Vous pouvez cliquer sur **File** dans le coin supérieur droit, puis choisir **Open**.



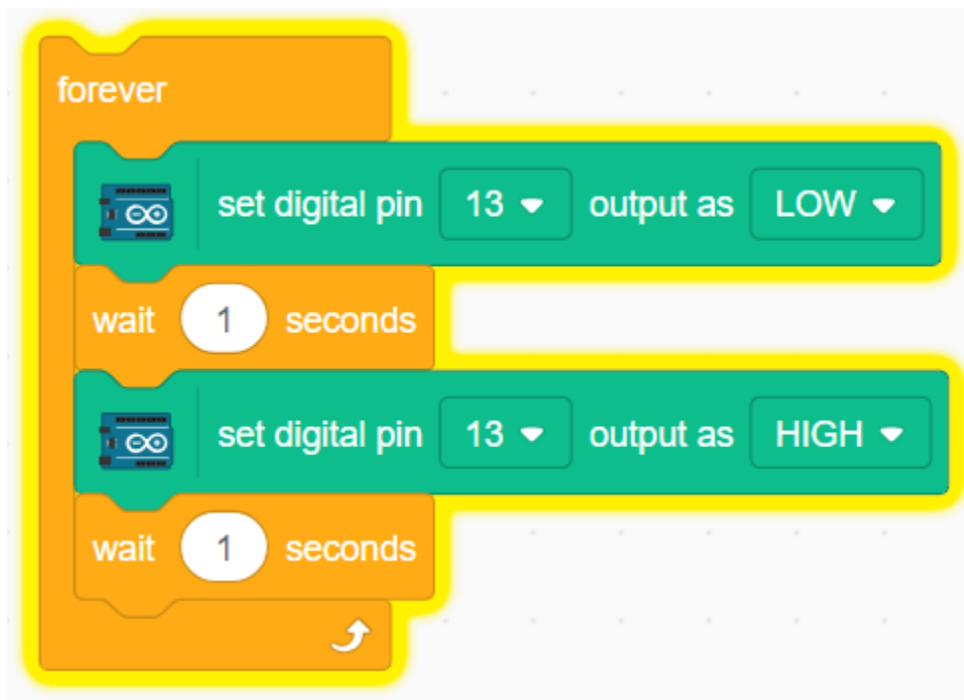
Choisissez **Open from Computer**.



Allez ensuite dans le chemin de 3in1-kit\scratch\_project\code, et ouvrez **1. Stage Mode.sb3**. Veuillez vous assurer d'avoir téléchargé le code requis depuis [github](#).



Cliquez directement sur le script pour l'exécuter, certains projets nécessitent de cliquer sur le drapeau vert ou sur le sprite.



— Programmer étape par étape

Vous pouvez également écrire le script étape par étape en suivant ces étapes.

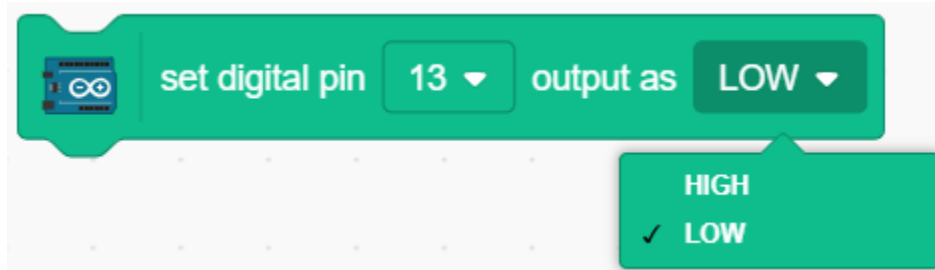
Cliquez sur la palette **Arduino Uno**.



La LED sur la carte Arduino est contrôlée par la broche numérique 13 (seulement 2 états, HIGH ou LOW), donc glissez le bloc [set digital pin out as] dans la zone de script.

Puisque l'état par défaut de la LED est allumé, réglez maintenant la broche 13 sur LOW et cliquez sur ce bloc et vous verrez la LED s'éteindre.

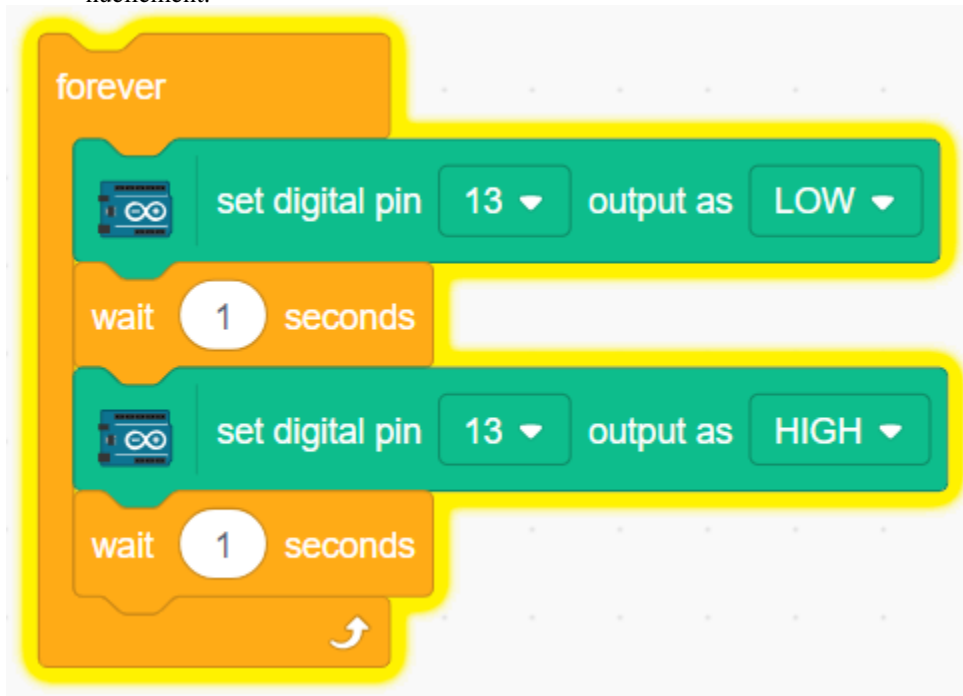
— [set digital pin out as] : Régler les broches numériques (2~13) au niveau (HIGH/LOW).



Pour voir l'effet d'une LED clignotante continue, vous devez utiliser les blocs [Wait 1 seconds] et [forever] dans la palette **Contrôle**. Cliquez sur ces blocs après les avoir écrits, un halo jaune signifie qu'ils sont en cours d'exécution.

— [Wait 1 seconde] : de la palette **Contrôle**, utilisé pour définir l'intervalle de temps entre 2 blocs.

- [forever] : de la palette **Contrôle**, permet au script de continuer à s'exécuter à moins d'être mis en pause manuellement.

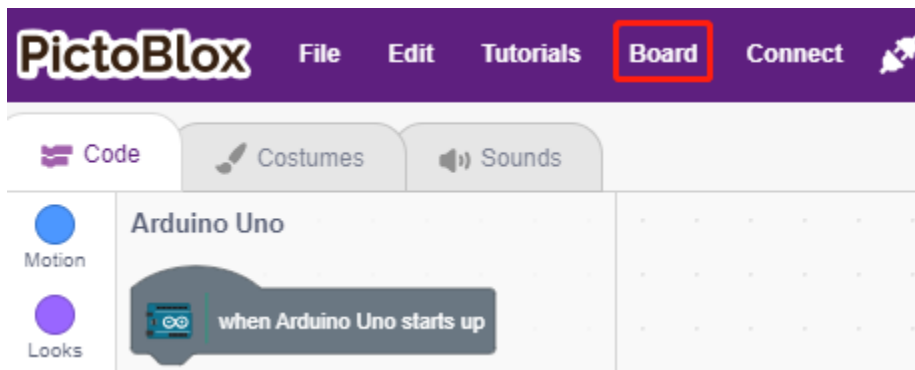


### 8.3.2 Mode Téléchargement

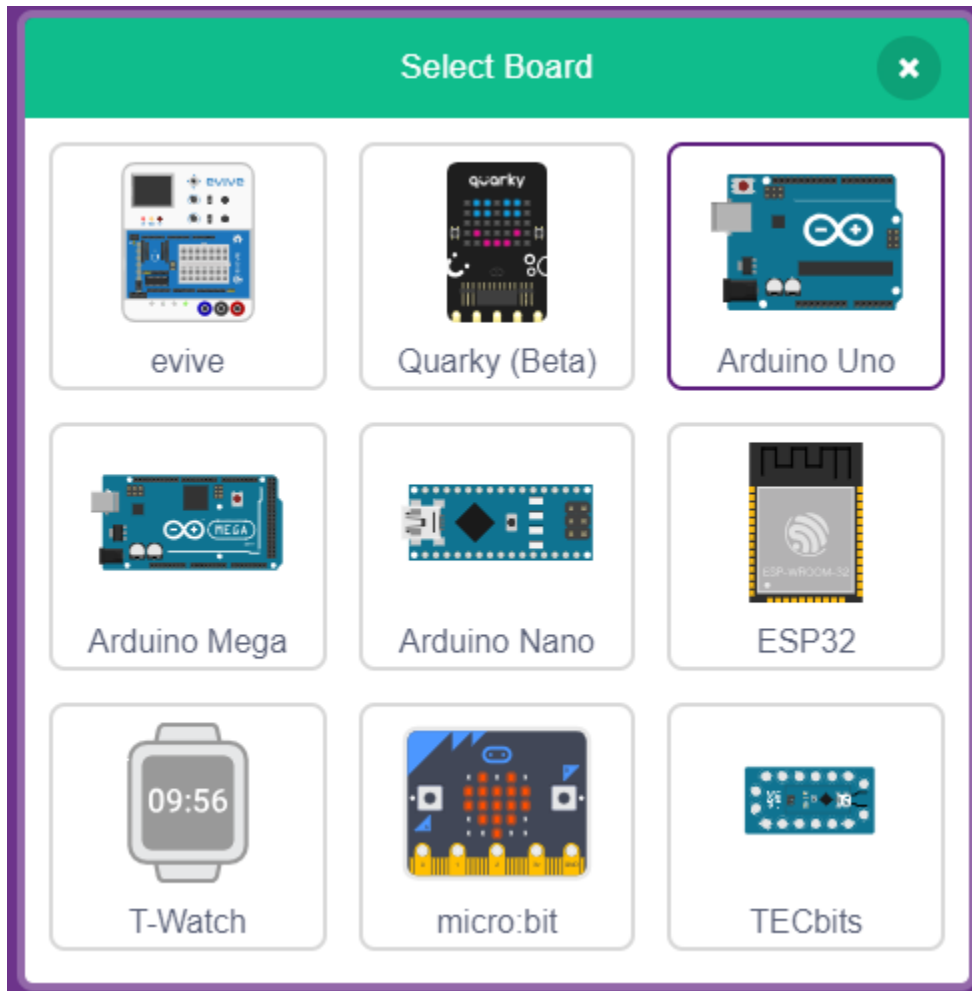
#### 1. Connecter à la carte Arduino

Connectez votre carte Arduino à l'ordinateur avec un câble USB, généralement l'ordinateur reconnaîtra automatiquement votre carte et finalement attribuera un port COM.

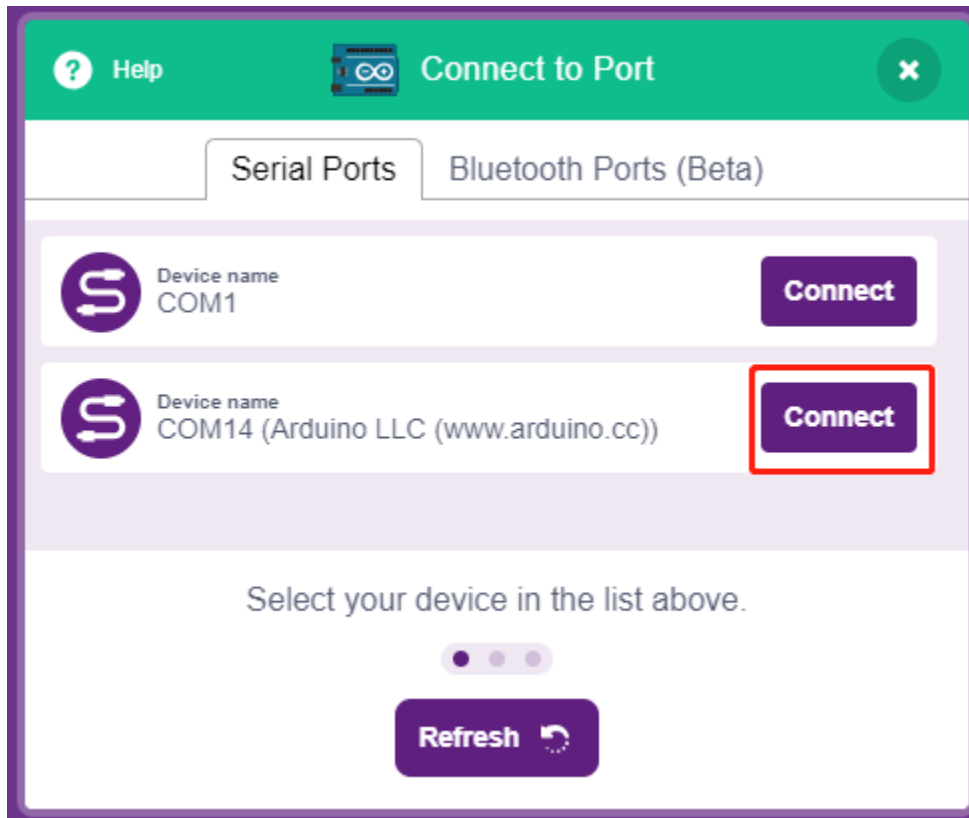
Ouvrez PictoBlox et cliquez sur **Board** dans la barre de navigation en haut à droite pour sélectionner la carte.



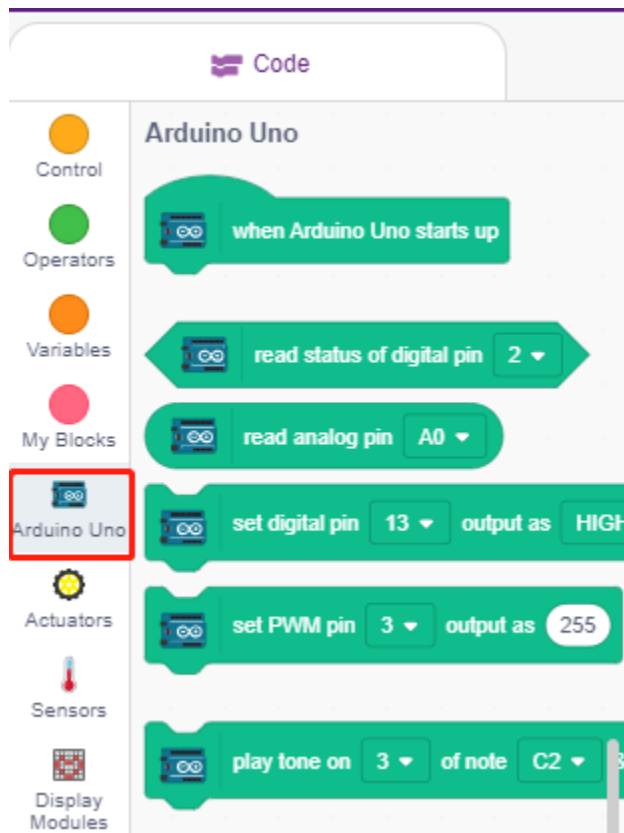
Par exemple, choisissez **Arduino Uno**.



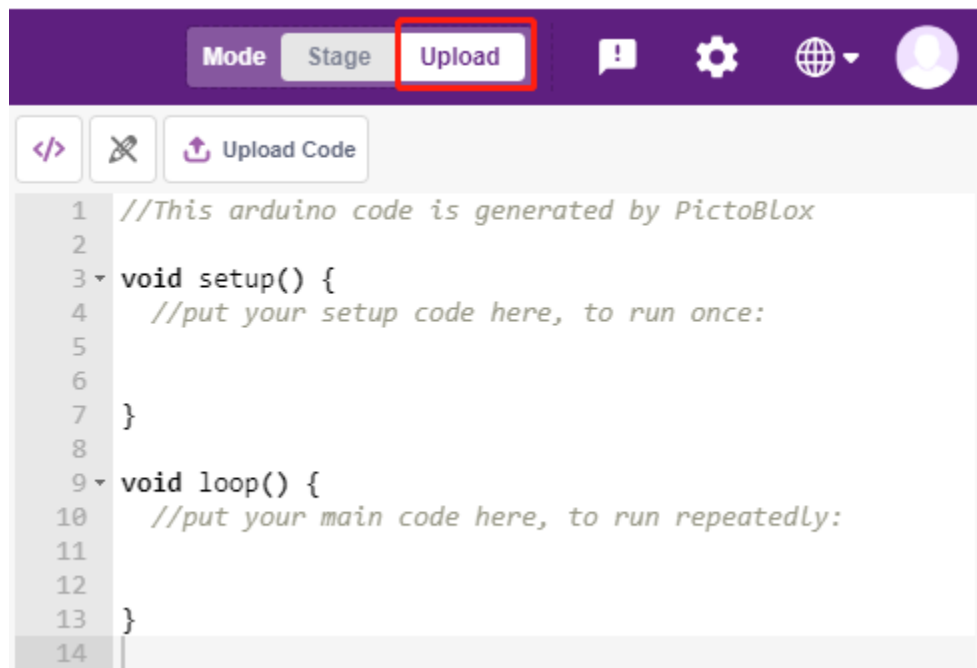
Une fenêtre de connexion s’ouvrira alors pour que vous sélectionniez le port à connecter, et retournez à la page d’accueil lorsque la connexion est complète. Si vous interrompez la connexion pendant l’utilisation, vous pouvez également cliquer sur **Connect** pour vous reconnecter.



En même temps, les palettes liées à l'Arduino Uno, telles que Arduino Uno, Actuators, etc., apparaîtront dans la **Block Palette**.



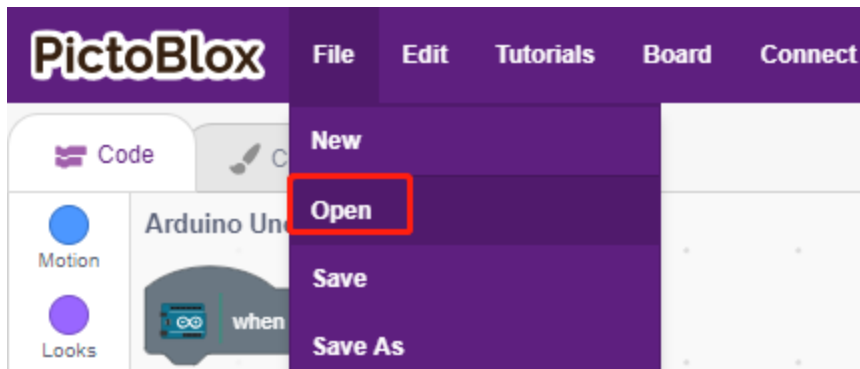
Après avoir sélectionné le mode Téléchargement, la scène passera à la zone de code Arduino originale.



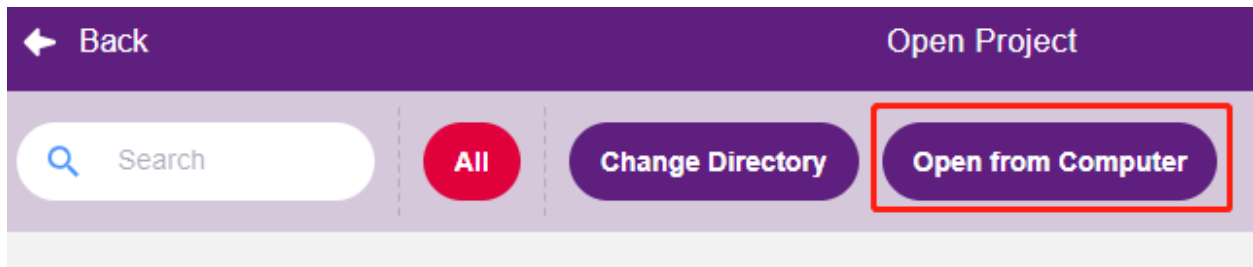
## 2. Programmation

— Ouvrir et exécuter le script directement

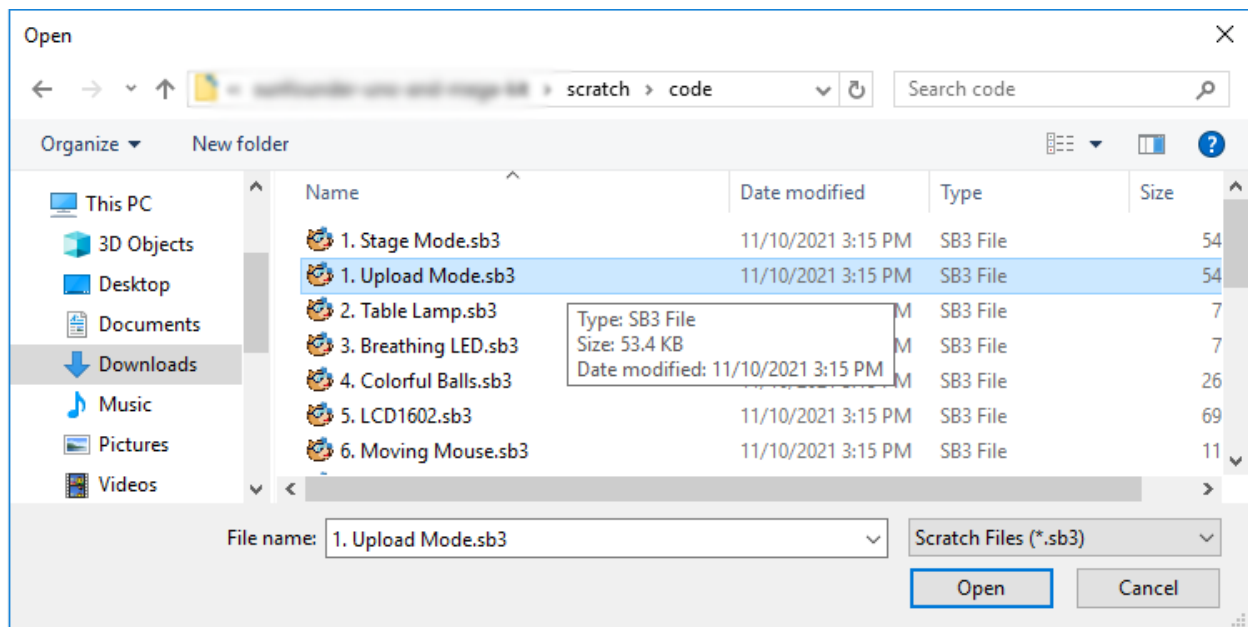
Vous pouvez cliquer sur **File** dans le coin supérieur droit.



Choisissez **Open from Computer**.



Ensuite, allez au chemin 3in1-kit\scratch\_project\code, et ouvrez **1. Upload Mode.sb3**. Veuillez vous assurer d'avoir téléchargé le code requis depuis [github](#).



Enfin, cliquez sur le bouton **Upload Code**.



The screenshot displays the PictoBlox IDE interface. The top menu bar includes 'Connect', '1. Upload Mode', 'Mode', 'Stage', and 'Upload'. The 'Upload Code' button is highlighted with a red rectangle. The main workspace shows a Scratch-style script for an Arduino Uno. The script begins with a 'when Arduino Uno starts up' block, followed by a 'forever' loop. Inside the loop, the sequence of blocks is: 'set digital pin 13 output as LOW', 'wait 1 seconds', 'set digital pin 13 output as HIGH', and 'wait 1 seconds'. On the right side, the corresponding C++ code is visible, with line numbers 1 through 19. The code includes comments and functions for setting up the pin and writing to it. The 'Upload Code' button is highlighted with a red rectangle.

```

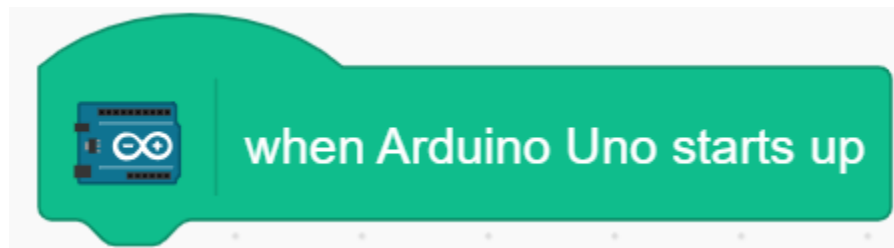
1 //This arduino code is generated by PictoBlox
2
3 void setup() {
4   //put your setup code here, to initialize your variables
5   pinMode(13, OUTPUT);
6
7 }
8
9
10 void loop() {
11   //put your main code here, to run repeatedly
12
13
14   digitalWrite(13, false);
15   delay(1 * 1000);
16   digitalWrite(13, true);
17   delay(1 * 1000);
18 }
19

```

— Programmer étape par étape  
 Vous pouvez également écrire le script étape par étape en suivant ces étapes.  
 Cliquez sur la palette **Arduino Uno**.



Glissez [when Arduino Uno starts up] dans la zone de script, ce qui est requis pour chaque script.



La LED sur la carte Arduino est contrôlée par la broche numérique 13 (seulement 2 états HIGH ou LOW), donc glissez le bloc [set digital pin out as] dans la zone de script.

Puisque l'état par défaut de la LED est allumé, réglez maintenant la broche 13 sur LOW et cliquez sur ce bloc et vous verrez la LED s'éteindre.

— [set digital pin out as] : Régler la broche numérique (2~13) au niveau (HIGH/LOW).

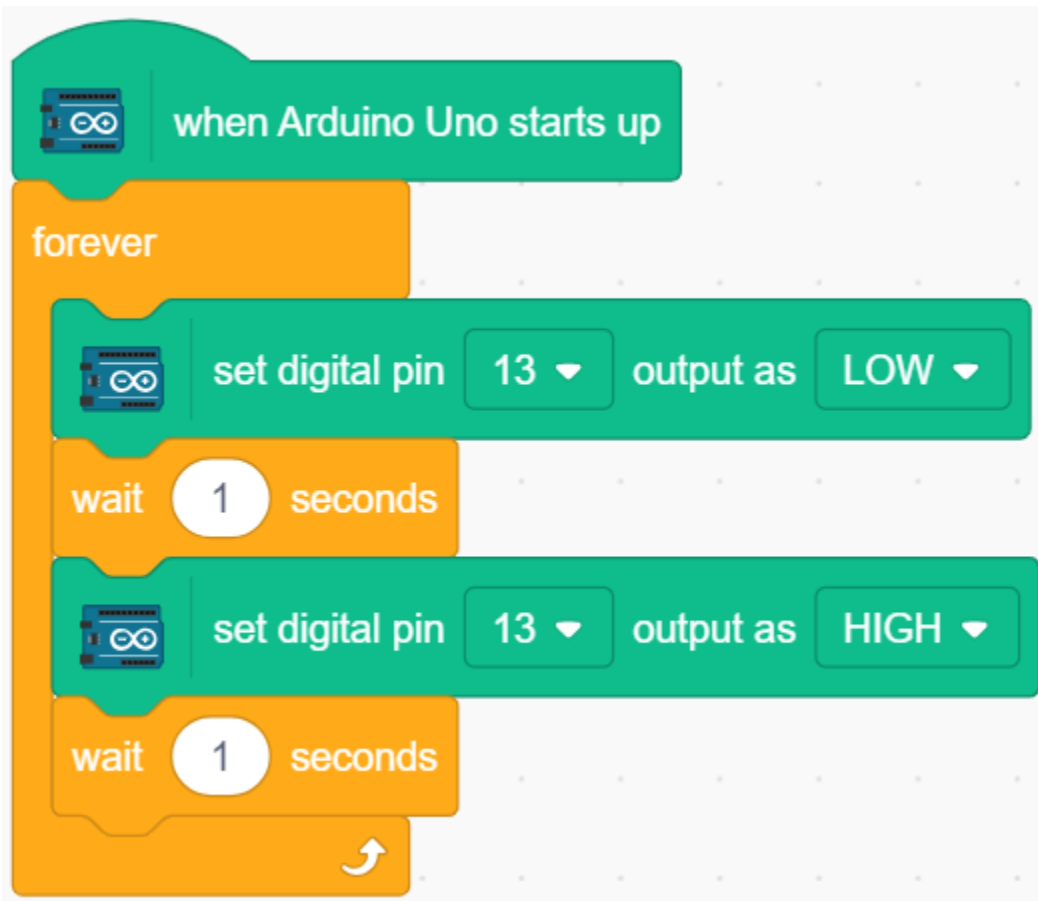


À ce stade, vous verrez le code Arduino apparaître sur le côté droit, si vous souhaitez éditer ce code, vous pouvez activer le mode Édition.

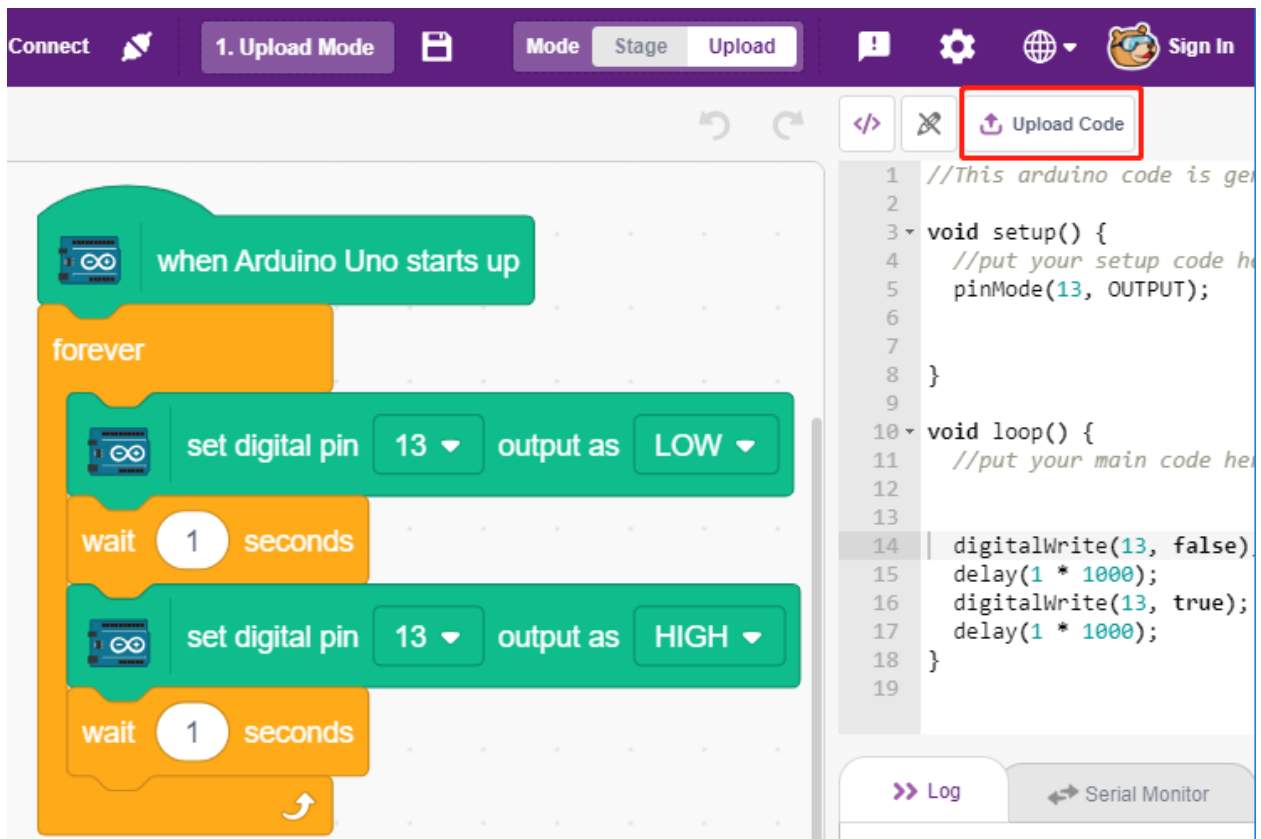


Pour voir l'effet d'une LED clignotante continue, vous devez utiliser les blocs [Wait 1 seconds] et [forever] dans la palette **Contrôle**. Cliquez sur ces blocs après les avoir écrits, un halo jaune signifie qu'ils sont en cours d'exécution.

- [Wait 1 seconds] : de la palette **Contrôle**, utilisé pour définir l'intervalle de temps entre 2 blocs.
- [forever] : de la palette **Contrôle**, permet au script de continuer à s'exécuter à moins que l'alimentation soit coupée.



Enfin, cliquez sur le bouton **Upload Code**.



## 2. Projets

Les projets suivants sont écrits dans l'ordre de difficulté de programmation, il est recommandé de les lire dans l'ordre.

Dans chaque projet, il y a des étapes très détaillées pour vous apprendre à construire le circuit et comment le programmer étape par étape pour obtenir le résultat final.

Bien sûr, vous pouvez également ouvrir le script directement pour l'exécuter, mais vous devez vous assurer d'avoir téléchargé le matériel pertinent de [github](#).

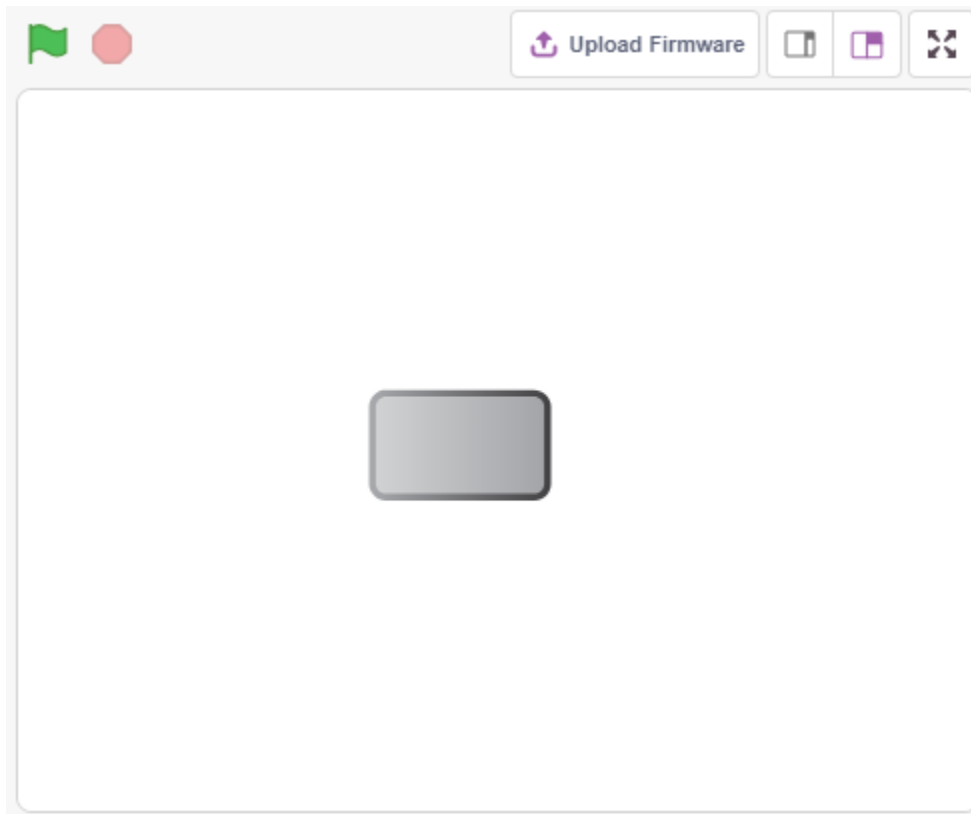
Une fois le téléchargement terminé, décompressez-le. Référez-vous à *Mode Scène* pour exécuter individuellement les scripts directement.

Mais le *2.9 Lecture de la Température et de l'Humidité* utilise le *Mode Téléchargement*.

## 8.4 2.1 Lampe de Table

Ici, nous connectons une LED sur la plaque d'essai et faisons en sorte que le sprite contrôle le clignotement de cette LED.

Lorsque le sprite Bouton sur la scène est cliqué, la LED clignotera 5 fois puis s'arrêtera.



### 8.4.1 Vous Apprendrez

- Plaque d'essai, LEDs et résistances
- Construction d'un circuit sur une plaque d'essai
- Supprimer et sélectionner des sprites
- Changer de costumes
- Définir un nombre limité de boucles répétitives

### 8.4.2 Composants requis

Dans ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

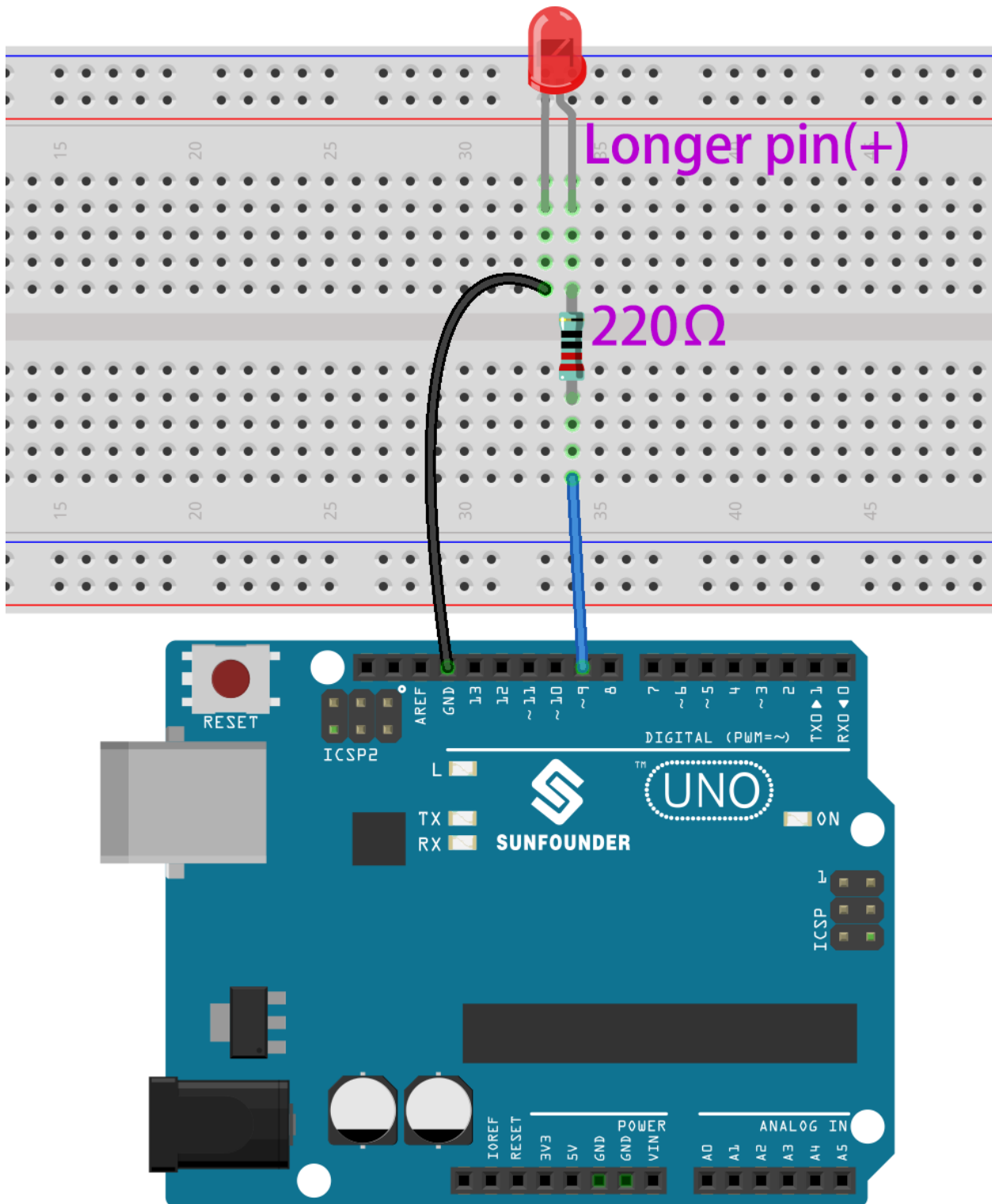
Vous pouvez également les acheter séparément à partir des liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	

### 8.4.3 Construisez le Circuit

Suivez le schéma ci-dessous pour construire le circuit sur la plaque d'essai.

Puisque l'anode de la LED (la broche la plus longue) est connectée à la broche 9 via une résistance de 220, et la cathode de la LED est connectée à GND, vous pouvez allumer la LED en donnant un niveau haut à la broche 9.



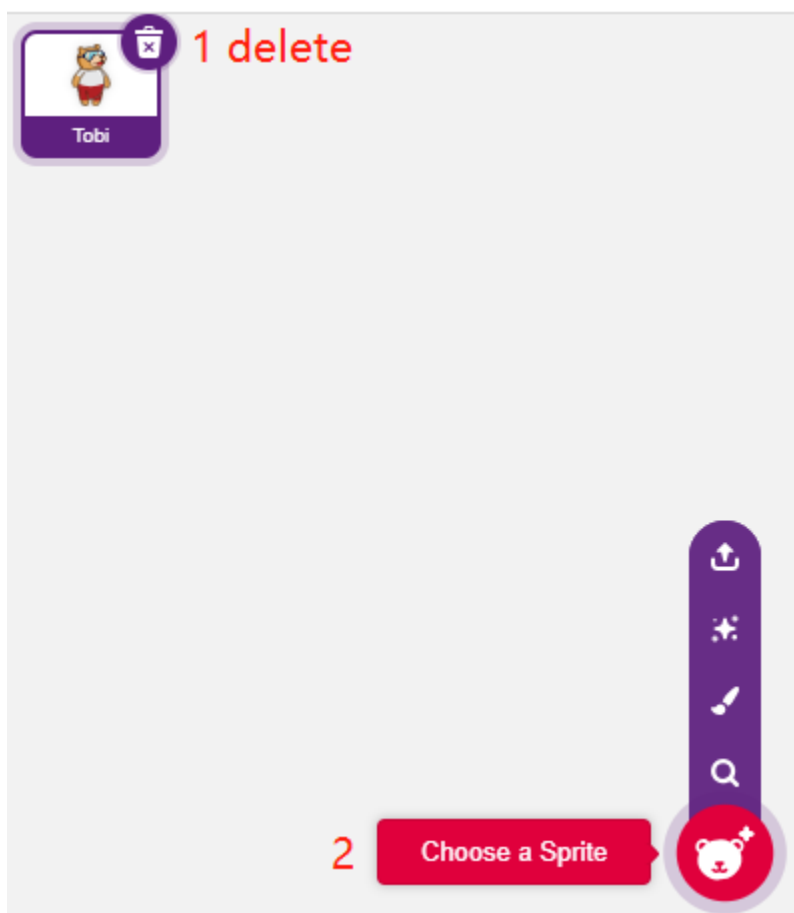


### 8.4.4 Programmation

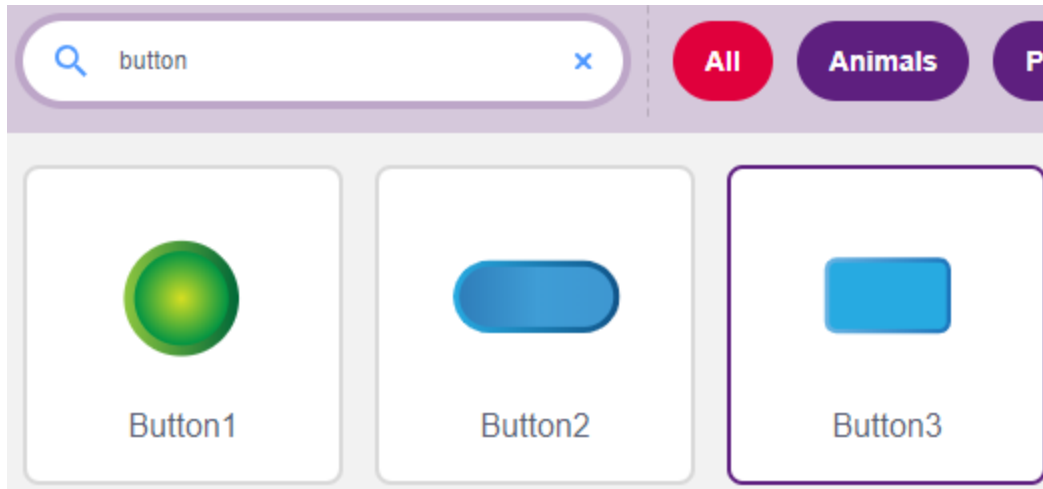
La programmation complète est divisée en 3 parties, la première partie consiste à sélectionner le sprite désiré, la deuxième partie consiste à changer le costume du sprite pour le rendre cliquable, et la troisième partie consiste à faire clignoter la LED.

#### 1. Sélectionnez le sprite **Button3**

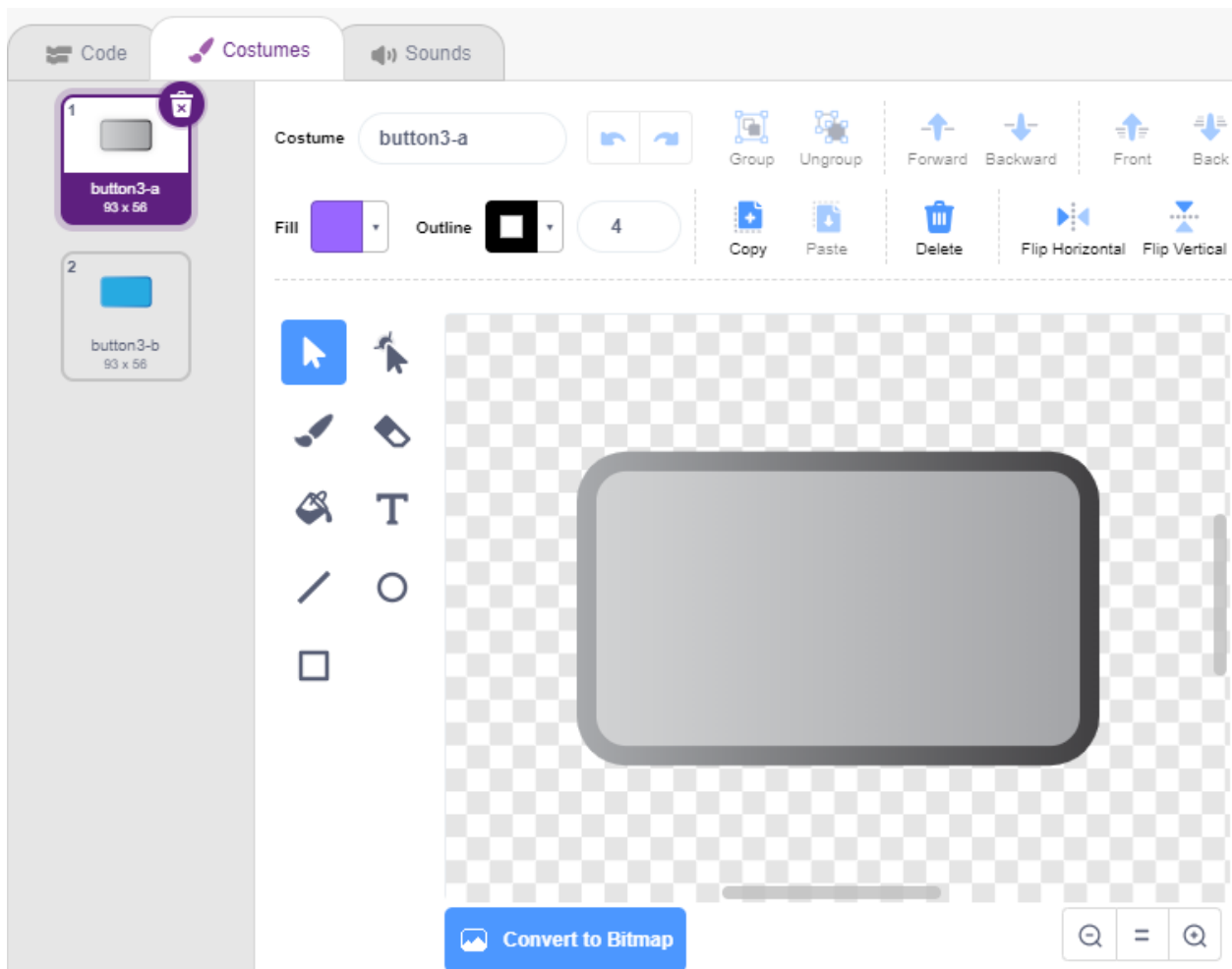
Supprimez le sprite Tobi existant en utilisant le bouton Supprimer dans le coin supérieur droit, et sélectionnez à nouveau un sprite.



Ici, nous sélectionnons le sprite **Button3**.

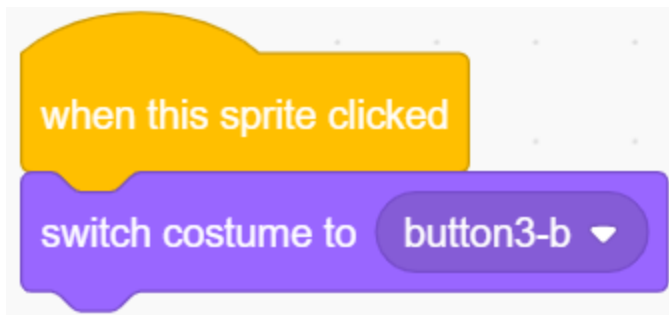


Cliquez sur Costumes dans le coin supérieur droit et vous verrez que le sprite Button3 a 2 costumes, nous définissons **button3-a** comme relâché et **button3-b** comme pressé.



## 2. Changer de costumes.

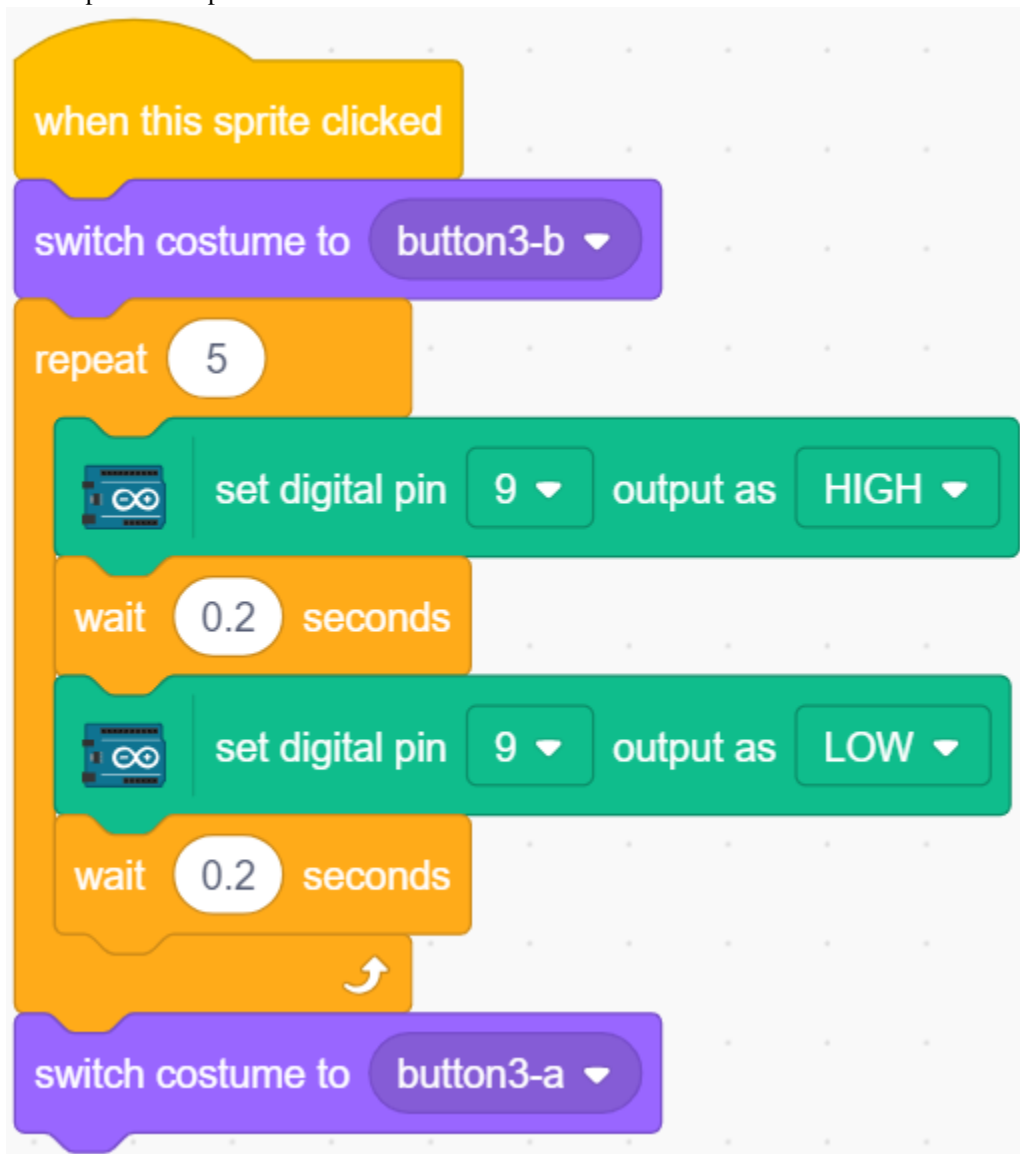
Lorsque le sprite est cliqué (**palette Événements**), il passe au costume pour **button3-b** (**palette apparence**).



### 3. Faire clignoter la LED 5 fois

Utilisez le bloc [Repeat] pour faire clignoter la LED 5 fois (cycle Haut-> BAS), n'oubliez pas de changer la broche 13 en broche 9, et enfin repassez le costume à **button3-a**.

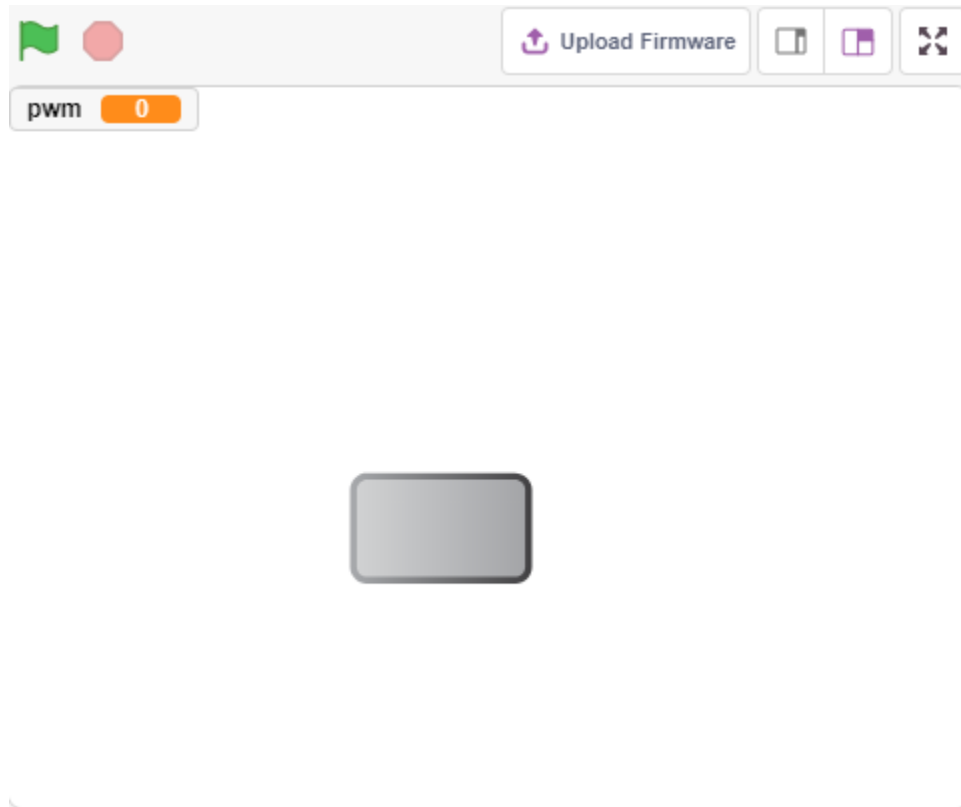
- [Repeat 10] : nombre limité de boucles répétitives, vous pouvez définir le nombre de répétitions vous-même, à partir de la palette **Contrôle**.



## 8.5 2.2 LED Respiration

Nous utilisons maintenant une autre méthode pour contrôler la luminosité de la LED. Contrairement au projet précédent, ici la luminosité de la LED diminue lentement jusqu'à disparaître.

Lorsque le sprite sur la scène est cliqué, la luminosité de la LED augmente lentement puis s'éteint instantanément.



### 8.5.1 Vous Apprendrez

- Régler la valeur de sortie de la broche PWM
- Créer des variables
- Changer la luminosité du sprite

### 8.5.2 Composants requis

Dans ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

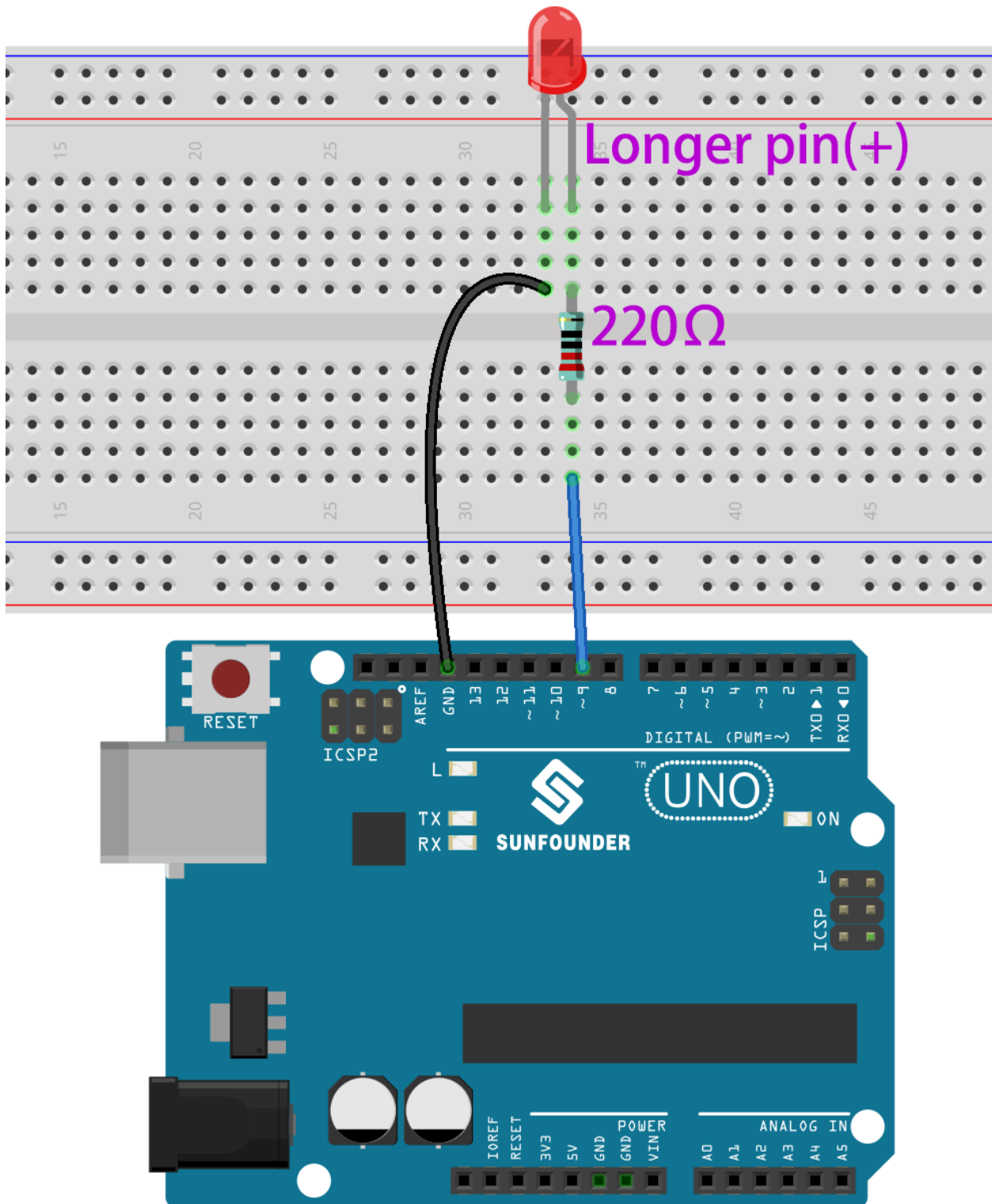
Vous pouvez également les acheter séparément à partir des liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED</i>	

### 8.5.3 Construisez le Circuit

Ce projet utilise le même circuit que le projet précédent *2.1 Lampe de Table*, mais au lieu d'utiliser HAUT/BAS pour allumer ou éteindre les LED, ce projet utilise le signal **PWM - Wikipedia** pour allumer ou atténuer lentement la LED.

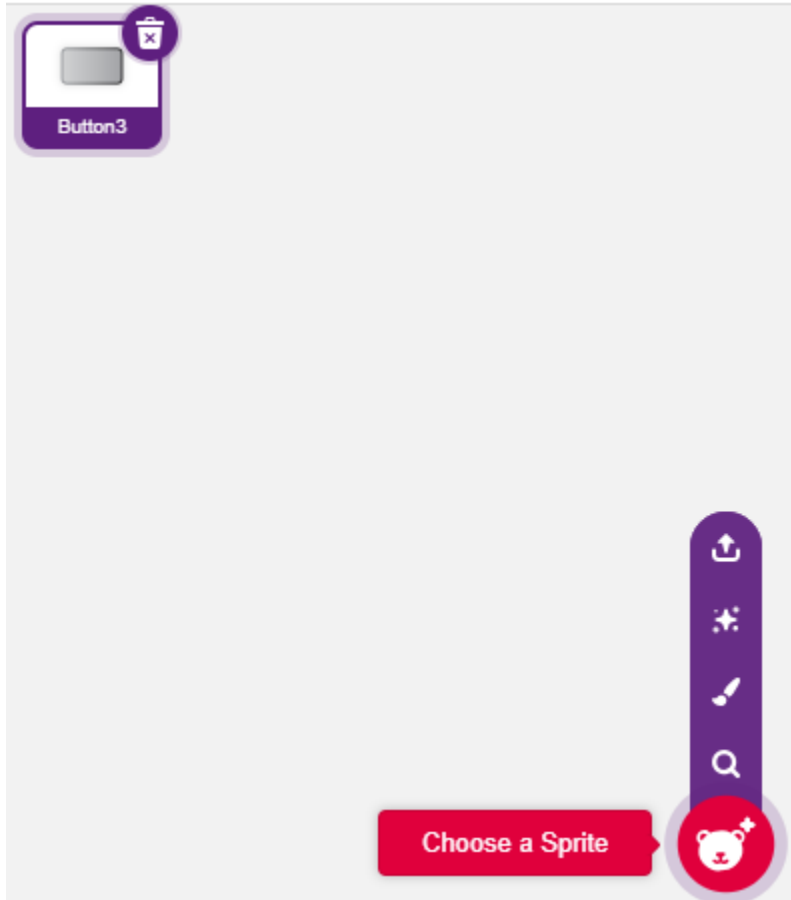
La plage du signal PWM est de 0 à 255, sur la carte Arduino Uno, les broches 3, 5, 6, 9, 10, 11 peuvent émettre un signal PWM ; sur le Mega2560, les broches 2 à 13, 44 à 46 peuvent émettre un signal PWM.



## 8.5.4 Programmation

### 1. Sélectionner un sprite

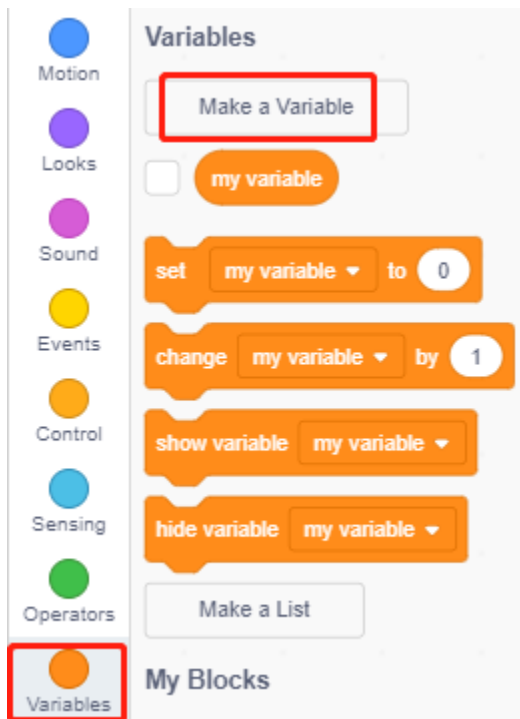
Supprimez le sprite par défaut, cliquez sur le bouton **Choose a Sprite** dans le coin inférieur droit de la zone de sprite, entrez **button3** dans la boîte de recherche, puis cliquez pour l'ajouter.



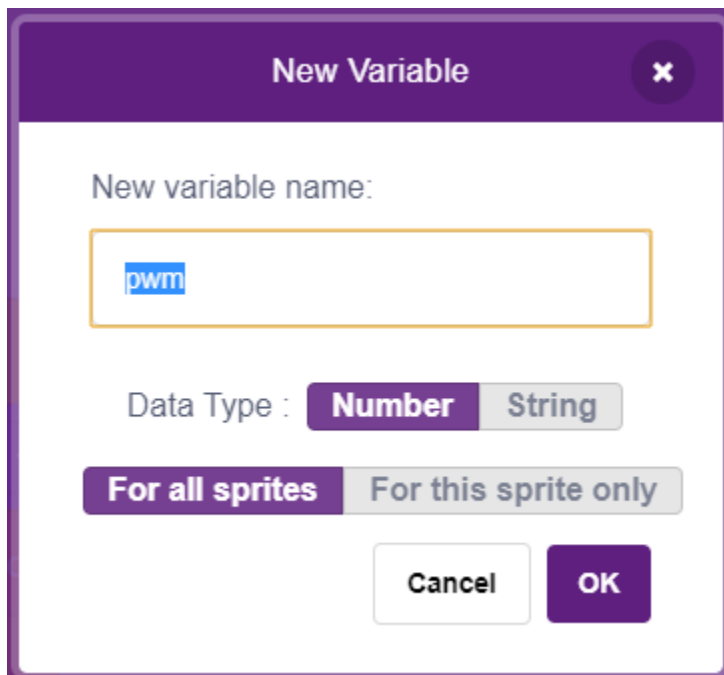
### 2. Création d'une variable.

Créez une variable appelée **pwm** pour stocker la valeur du changement de pwm.

Cliquez sur la palette **Variables** et sélectionnez **Make a Variable**.

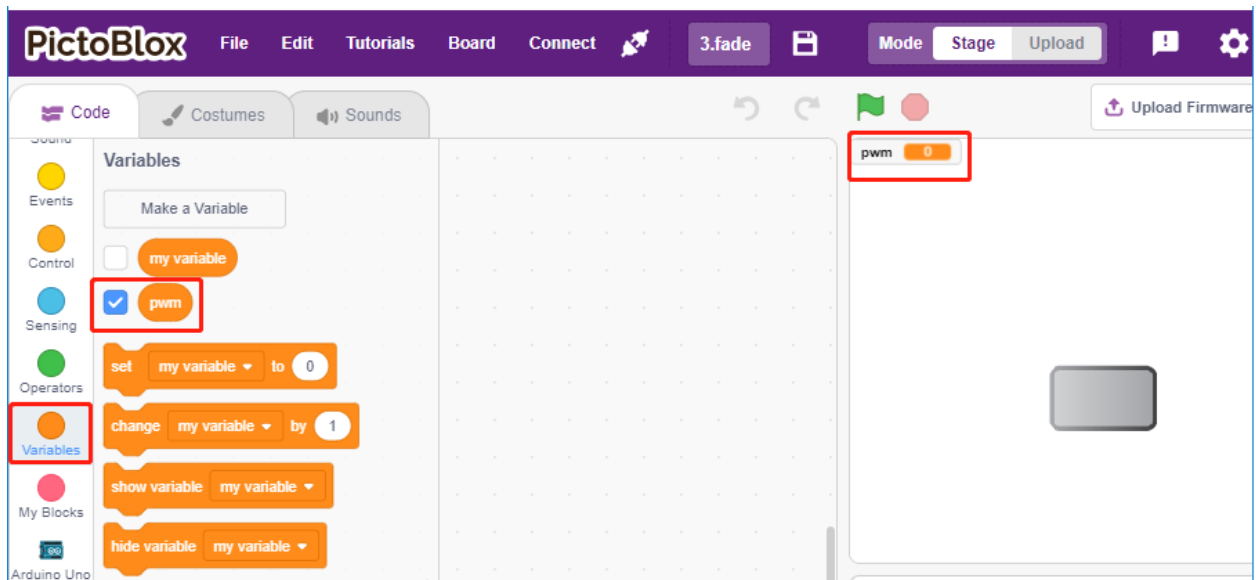


Entrez le nom de la variable, cela peut être n'importe quel nom, mais il est recommandé de décrire sa fonction. Le type de données est un nombre et pour tous les sprites.



Une fois créée, vous verrez **pwm** dans la palette **Variables** et dans l'état coché, ce qui signifie que cette variable apparaîtra sur la scène. Vous pouvez essayer de la décocher pour voir si pwm est toujours présent sur la scène.

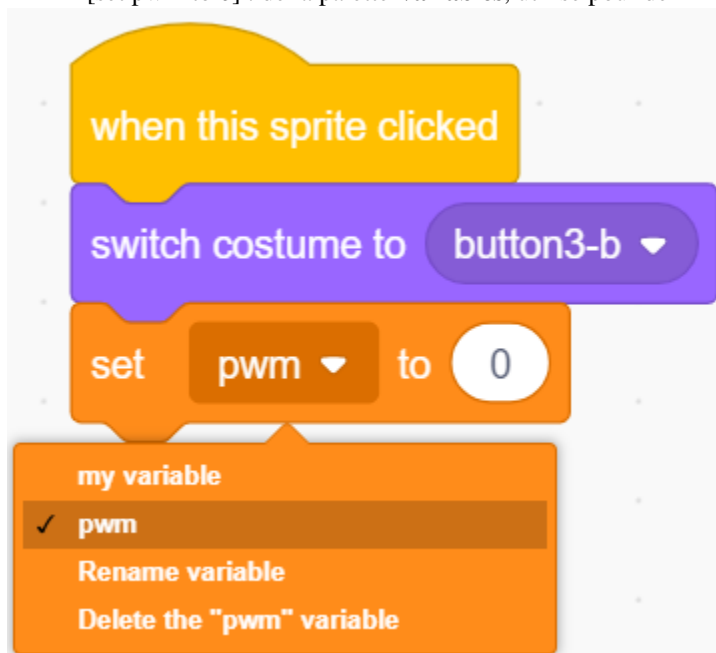




### 3. Définir l'état initial

Lorsque le sprite **button3** est cliqué, changez le costume en **button-b** (état cliqué), et définissez la valeur initiale de la variable **pwm** à 0.

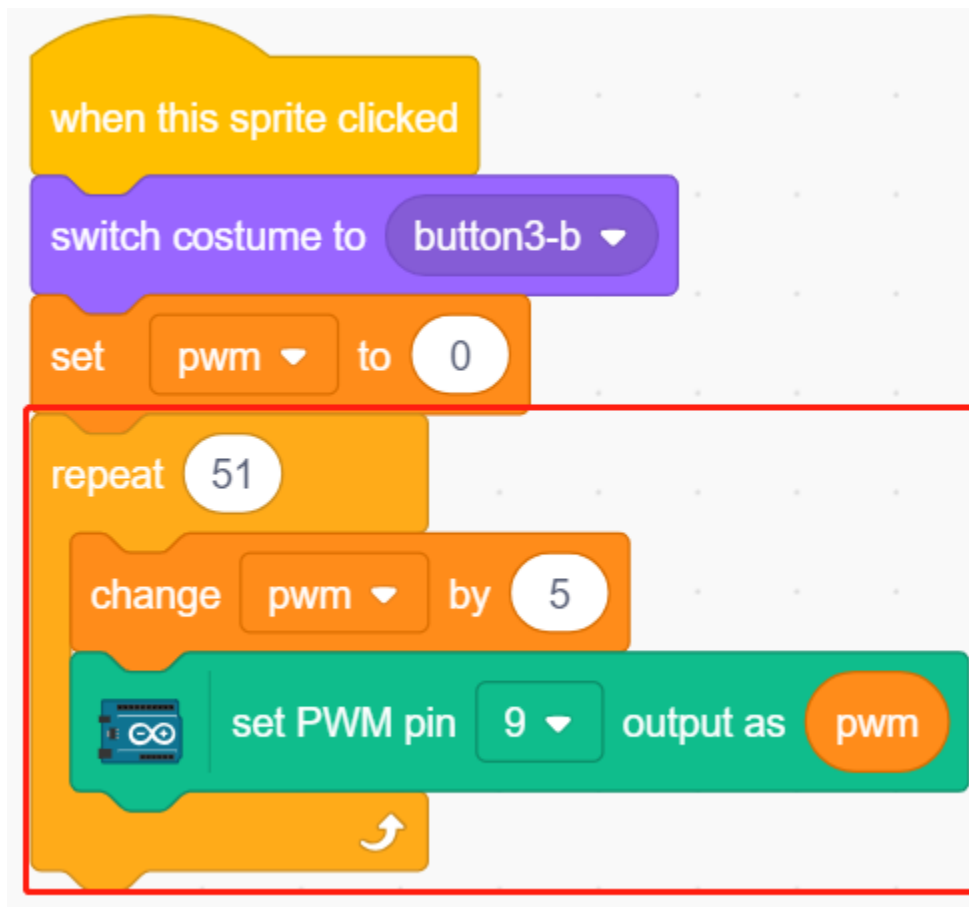
- [set pwm to 0] : de la palette **Variables**, utilisé pour définir la valeur de la variable.



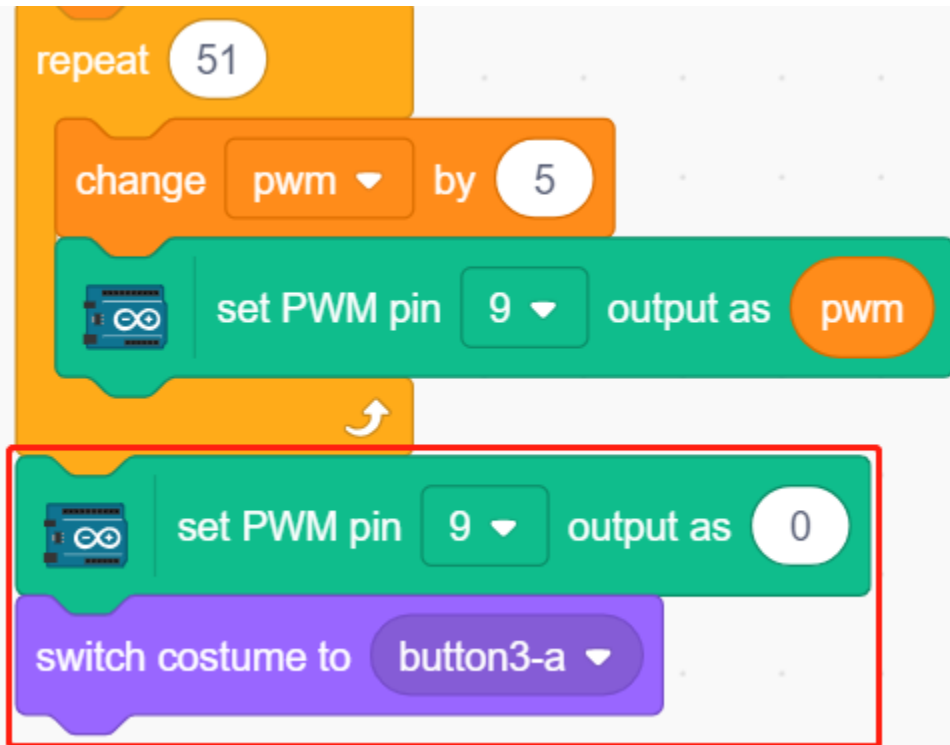
### 4. Rendre la LED de plus en plus lumineuse

Comme la plage de pwm est de 255, alors par le bloc [repeat], la variable **pwm** est accumulée à 255 par 5, puis mise dans le bloc [set PWM pin], donc vous pouvez voir la LED s'allumer lentement.

- [change pwm by 5] : de la palette **Variables**, laissez la variable changer un nombre spécifique à chaque fois. Cela peut être un nombre positif ou négatif, positif signifie qu'il augmente chaque fois, négatif signifie qu'il diminue chaque fois, par exemple, ici la variable pwm est augmentée de 5 chaque fois.
- [set PWM pin] : de la palette **Arduino Uno**, utilisé pour régler la valeur de sortie de la broche pwm.



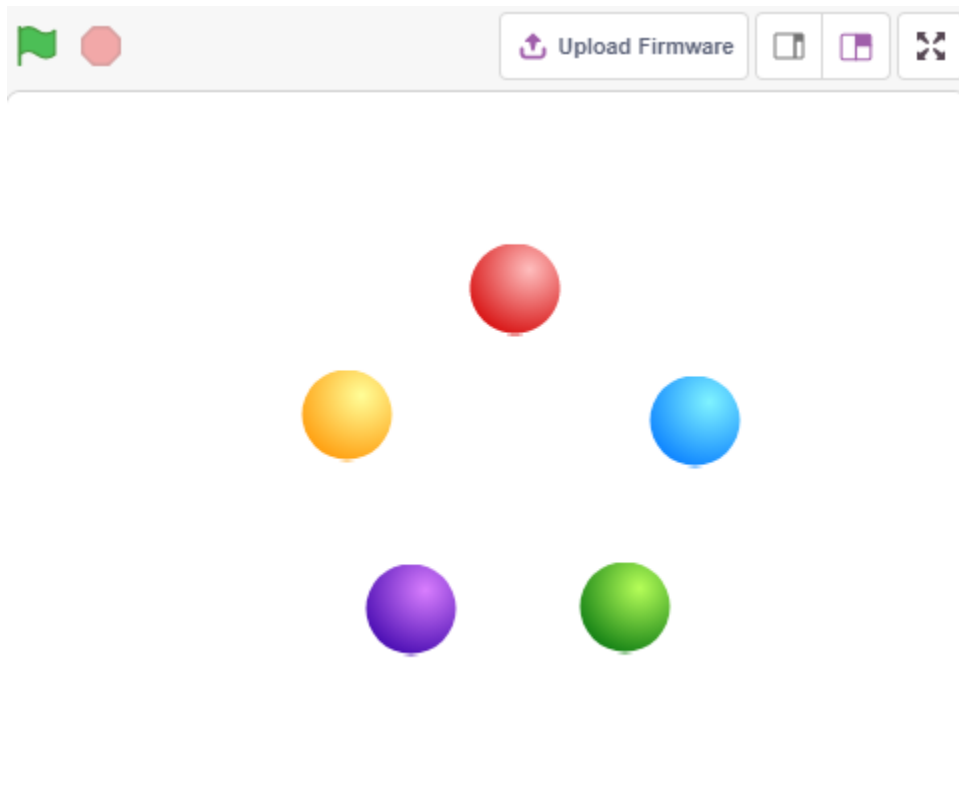
Enfin, changez le costume de button3 en **button-a** et mettez la valeur de la broche PWM à 0, afin que la LED s'allume lentement puis s'éteigne à nouveau.



## 8.6 2.3 Balles Colorées

Dans ce projet, nous allons faire en sorte que les LED RVB affichent différentes couleurs.

Cliquer sur des balles de différentes couleurs dans l'espace scénique entraînera l'illumination de la LED RVB dans différentes couleurs.



### 8.6.1 Vous Apprendrez

- Le principe de la LED RVB
- Copier des sprites et sélectionner différents costumes
- Superposition des trois couleurs primaires

### 8.6.2 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

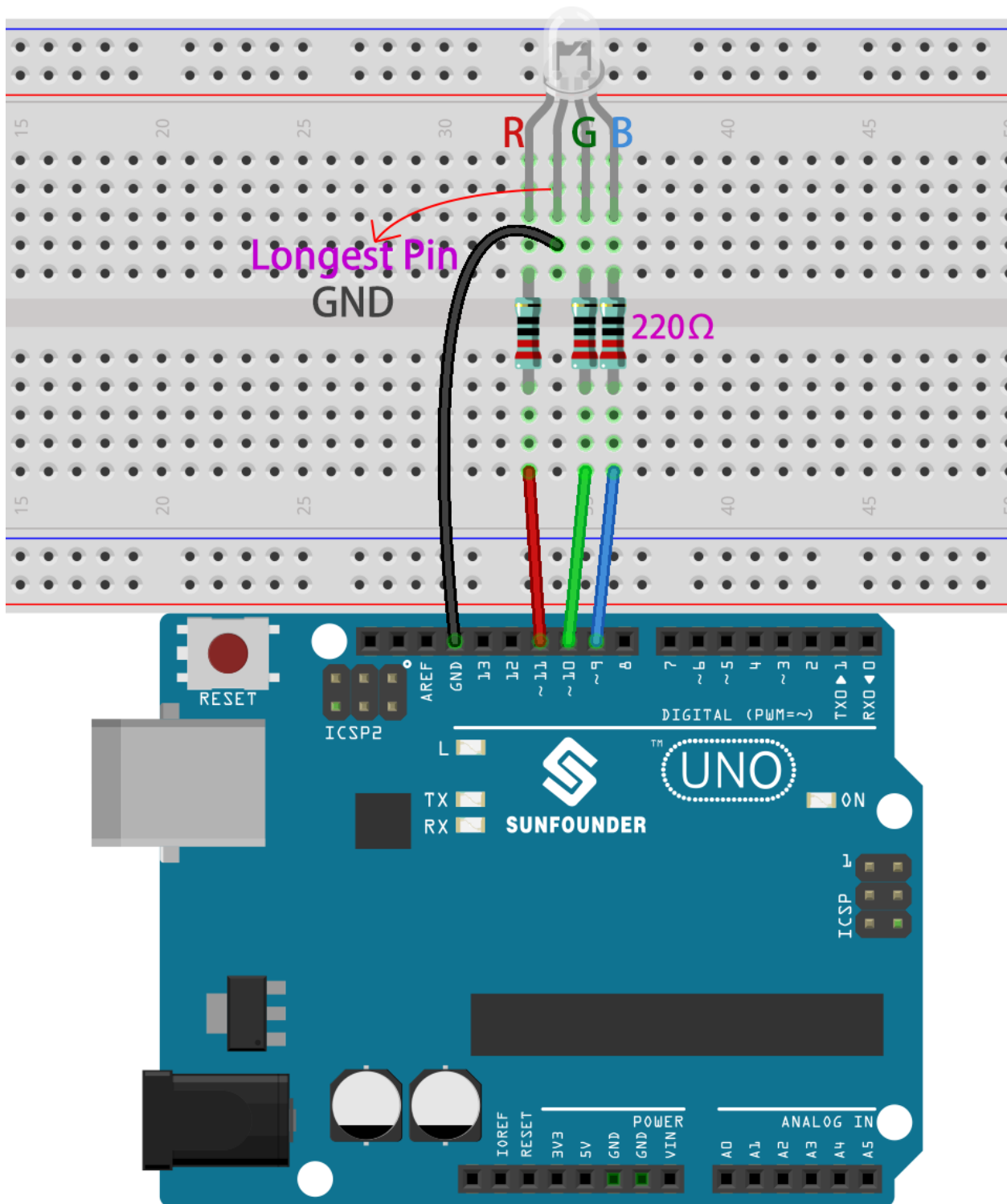
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>LED RVB</i>	

### 8.6.3 Construire le Circuit

Une LED RVB combine trois LED de rouge, vert et bleu dans une coque en plastique transparent ou semi-transparent. Elle peut afficher diverses couleurs en changeant la tension d'entrée des trois broches et en les superposant, ce qui, selon les statistiques, peut créer 16 777 216 couleurs différentes.

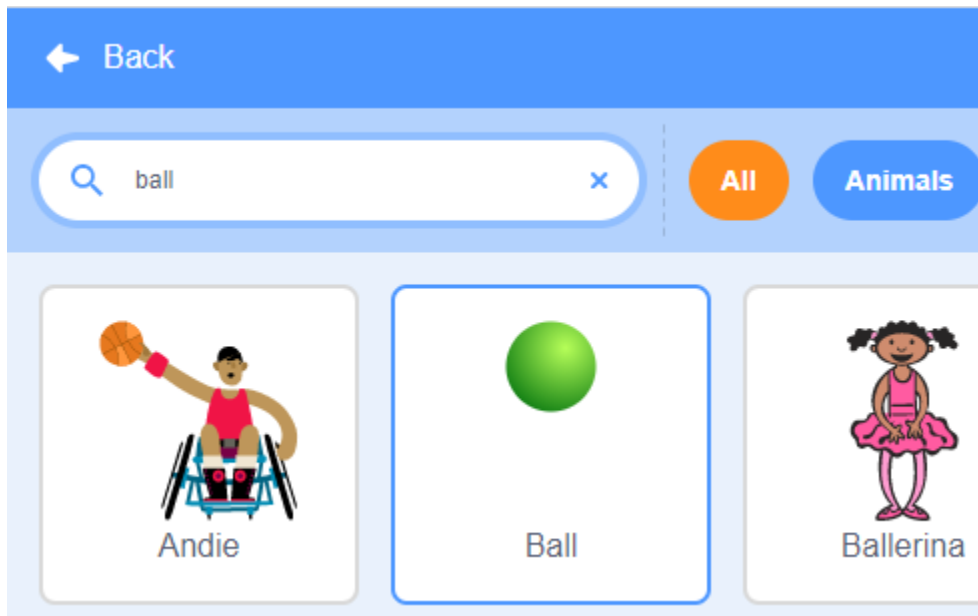




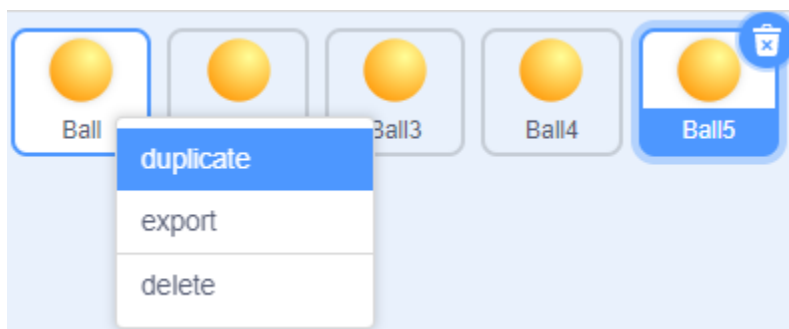
## 8.6.4 Programmation

### 1. Sélectionner un sprite

Supprimez le sprite par défaut, puis choisissez le sprite **Ball**.

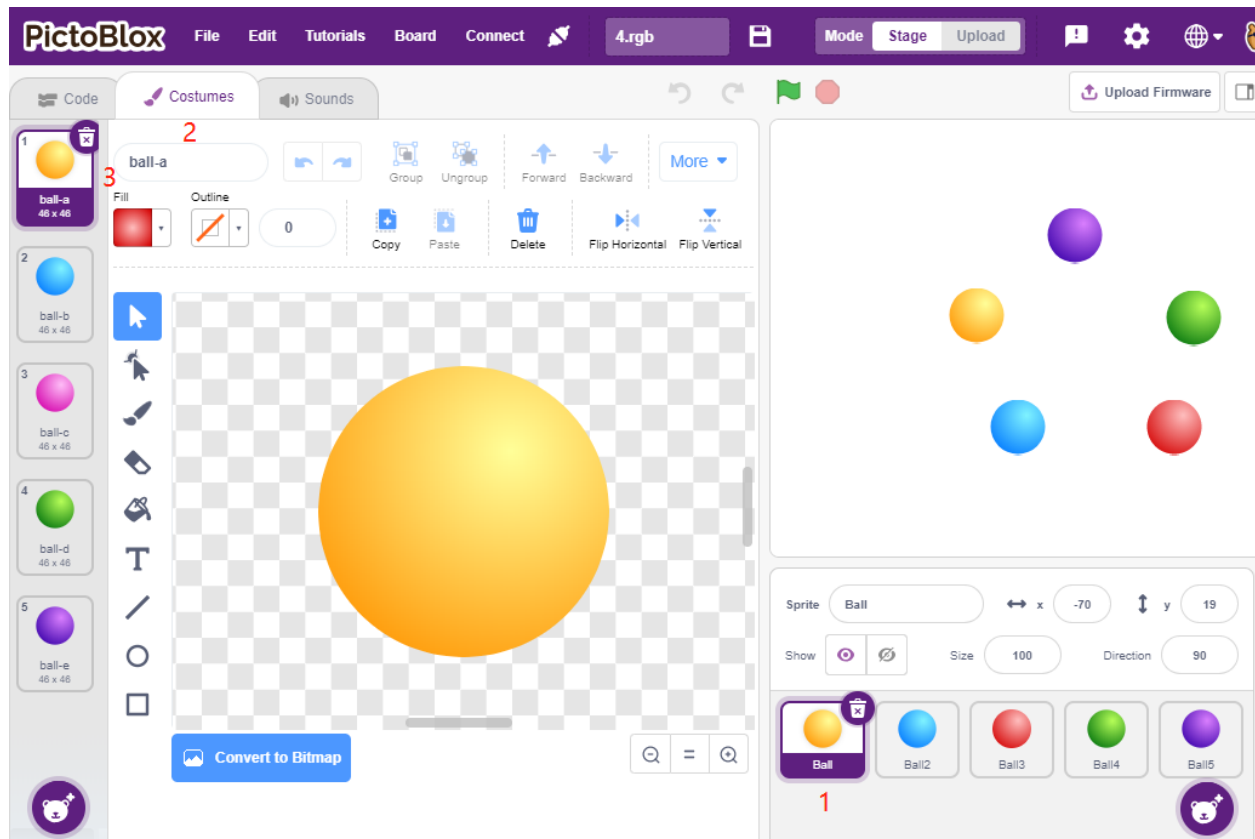


Et dupliquez-le 5 fois.



Choisissez différents costumes pour ces 5 sprites **Ball** et placez-les aux positions correspondantes.

**Note :** Le costume du sprite **Ball3** doit être manuellement changé en rouge.



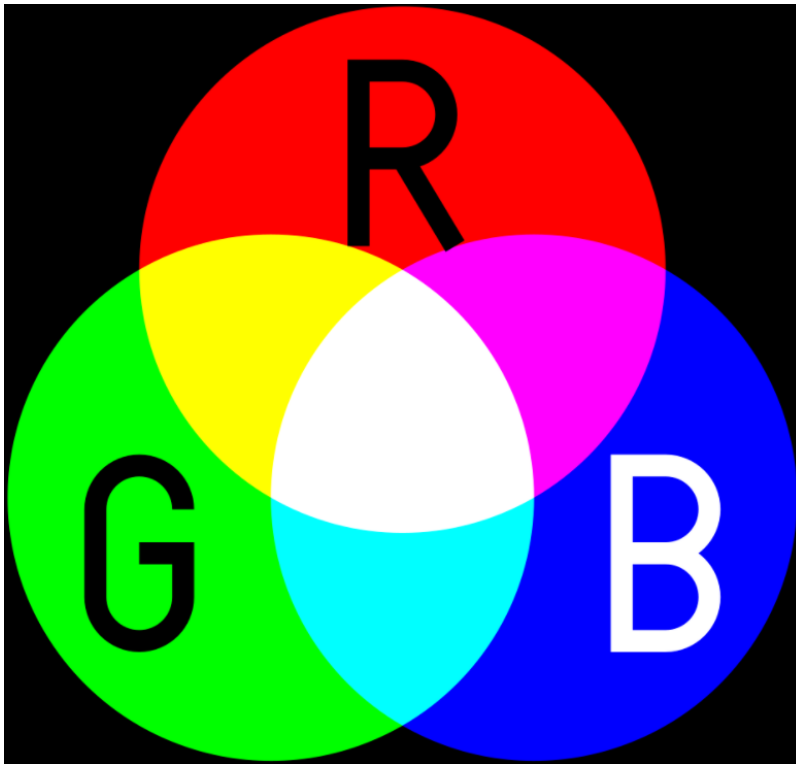
## 2. Faire s'illuminer les LED RVB dans la couleur appropriée

Avant de comprendre le code, nous devons comprendre le [modèle de couleur RVB](#).

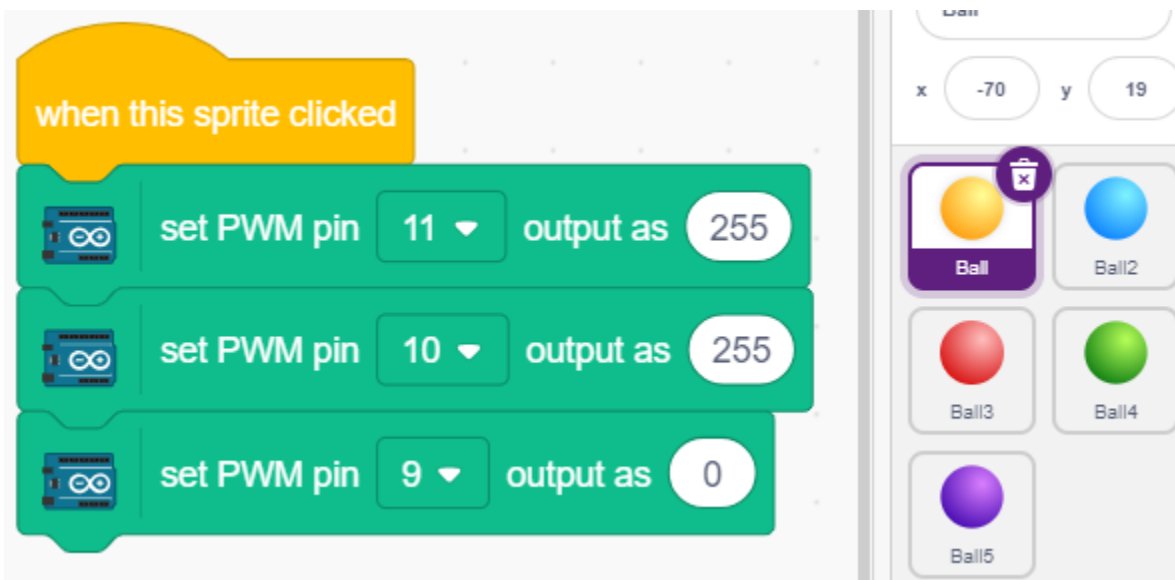
Le modèle de couleur RVB est un modèle de couleur additive dans lequel les lumières rouge, verte et bleue sont combinées de diverses manières pour reproduire un large éventail de couleurs.

Mélange de couleurs additives : ajouter du rouge au vert donne du jaune ; ajouter du vert au bleu donne du cyan ; ajouter du bleu au rouge donne du magenta ; ajouter les trois couleurs primaires ensemble donne du blanc.





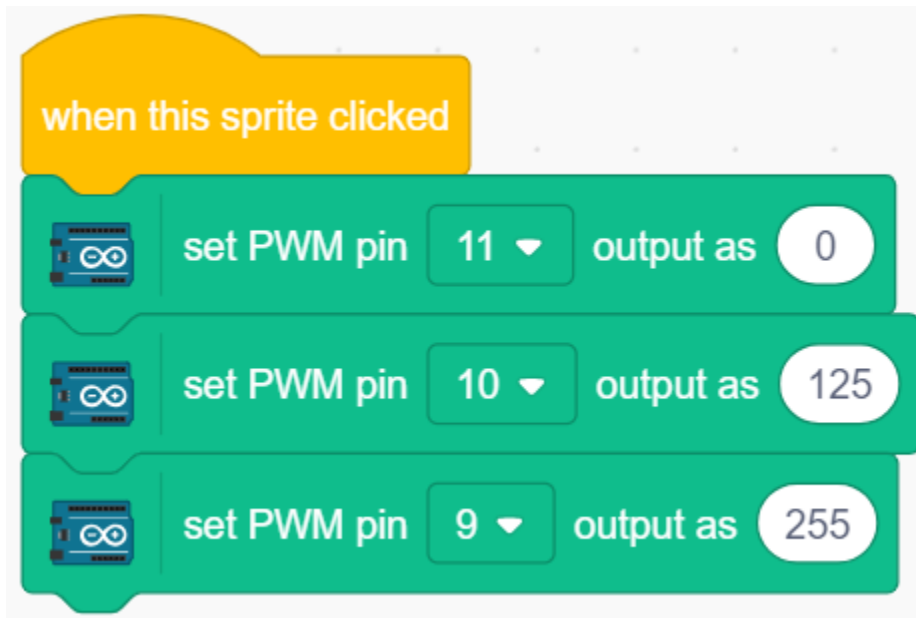
Ainsi, le code pour faire s'illuminer la LED RVB en jaune est le suivant.



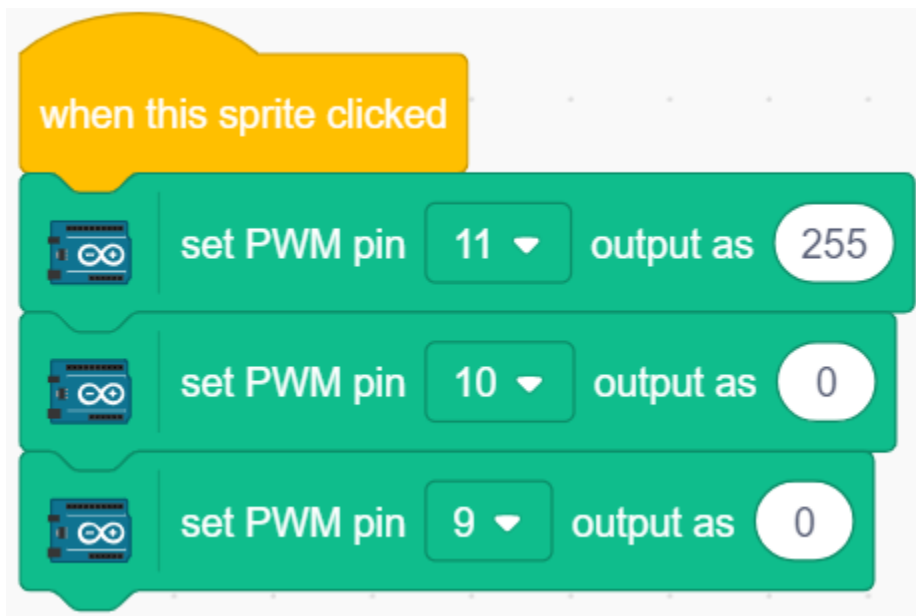
Lorsque le sprite Ballon (balle jaune) est cliqué, nous réglons la broche 11 en mode haut (LED rouge allumée), la broche 10 en mode haut (LED verte allumée) et la broche 9 en mode bas (LED bleue éteinte) afin que la LED RVB s'illumine en jaune.

Vous pouvez écrire des codes pour les autres sprites de la même manière pour faire s'illuminer les LED RVB dans les couleurs correspondantes.

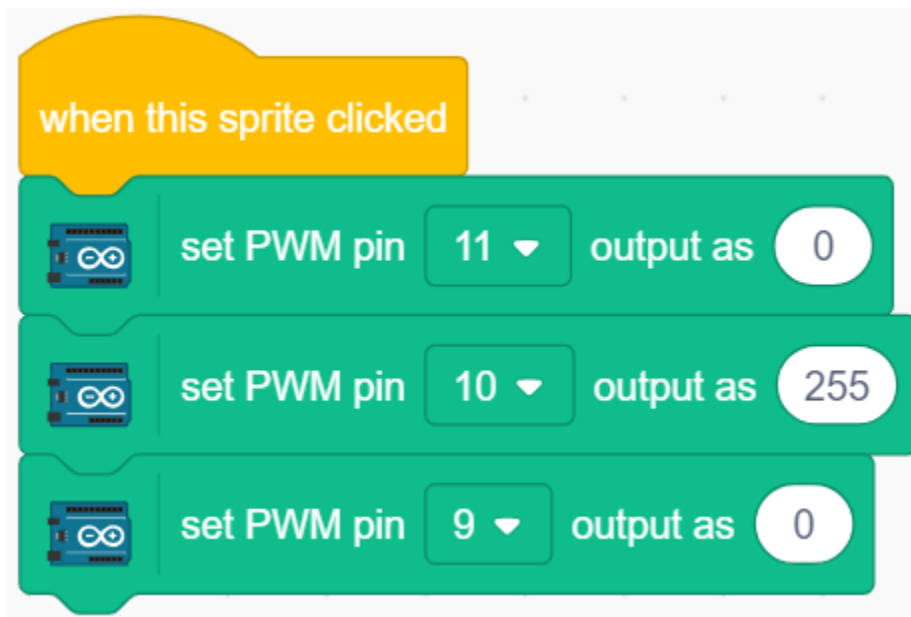
### 3. Sprite Ballon2 (bleu clair)



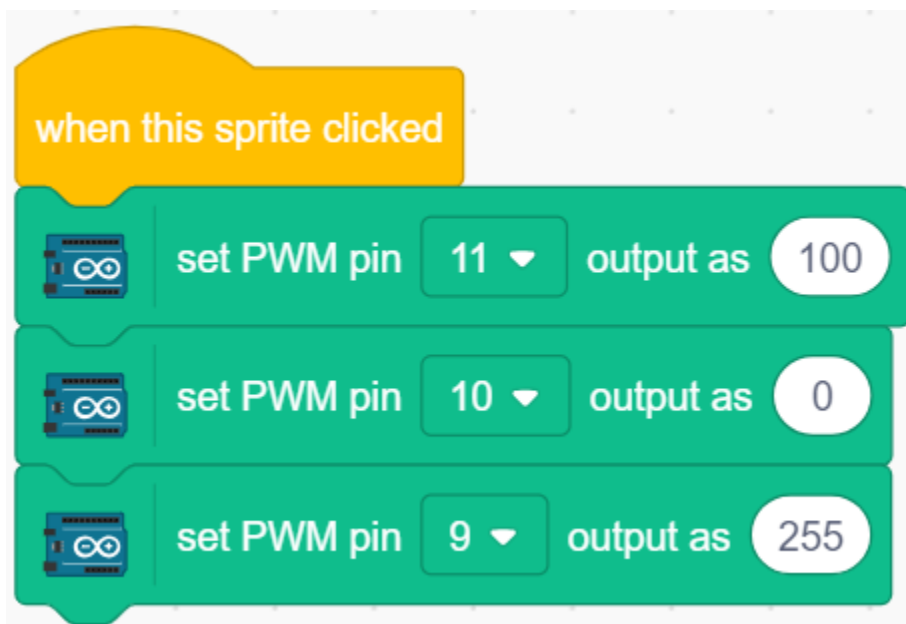
#### 4. Sprite Ballon3 (rouge)



#### 5. Sprite Ballon4 (vert)



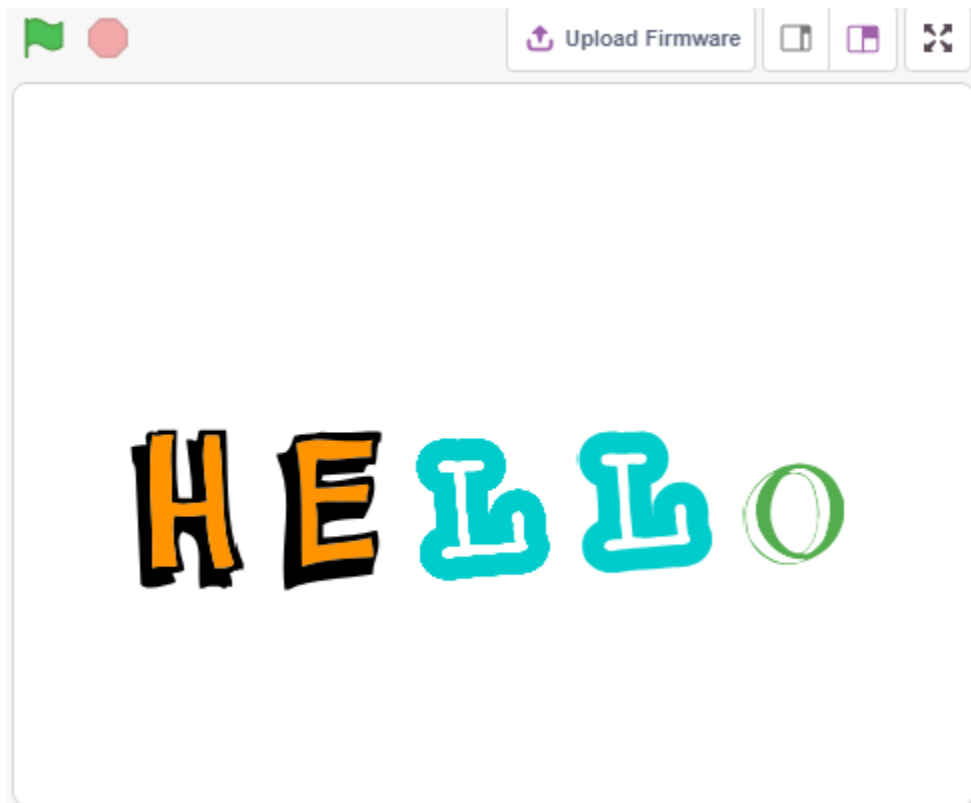
#### 6. Sprite Ballon5 (violet)



## 8.7 2.4 LCD1602

Le LCD1602 peut afficher 2x16 caractères. Nous allons le faire afficher les caractères correspondants aux sprites de caractères sur la scène.

Lorsque vous cliquez sur les « Hello » sur la scène un par un, ils auront différents effets d'animation et les caractères seront affichés simultanément sur le LCD1602.



### 8.7.1 Vous Apprendrez

- Utiliser le LCD1602
- Sélectionner plusieurs sprites différents
- Modifier la taille, l'angle de rotation, la couleur des sprites et les afficher ou les masquer.

### 8.7.2 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

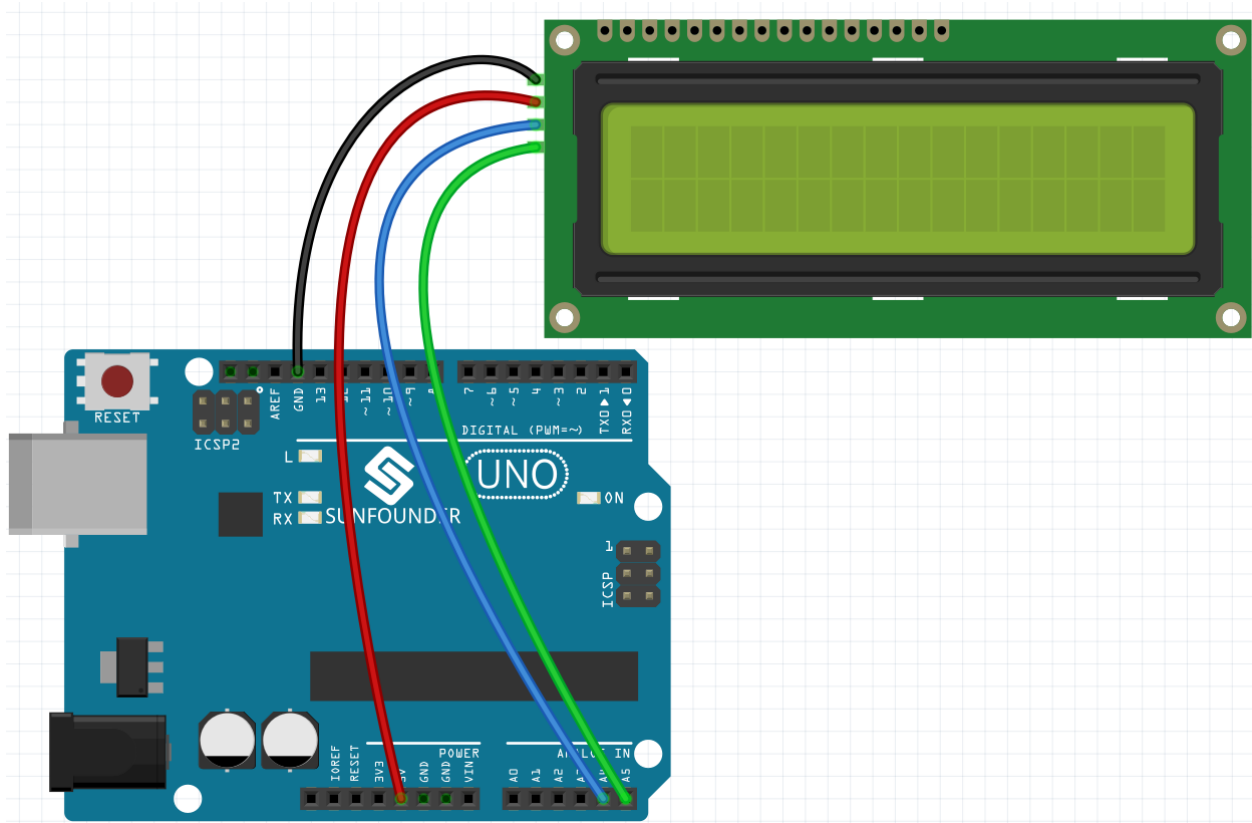
Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>I2C LCD1602</i>	

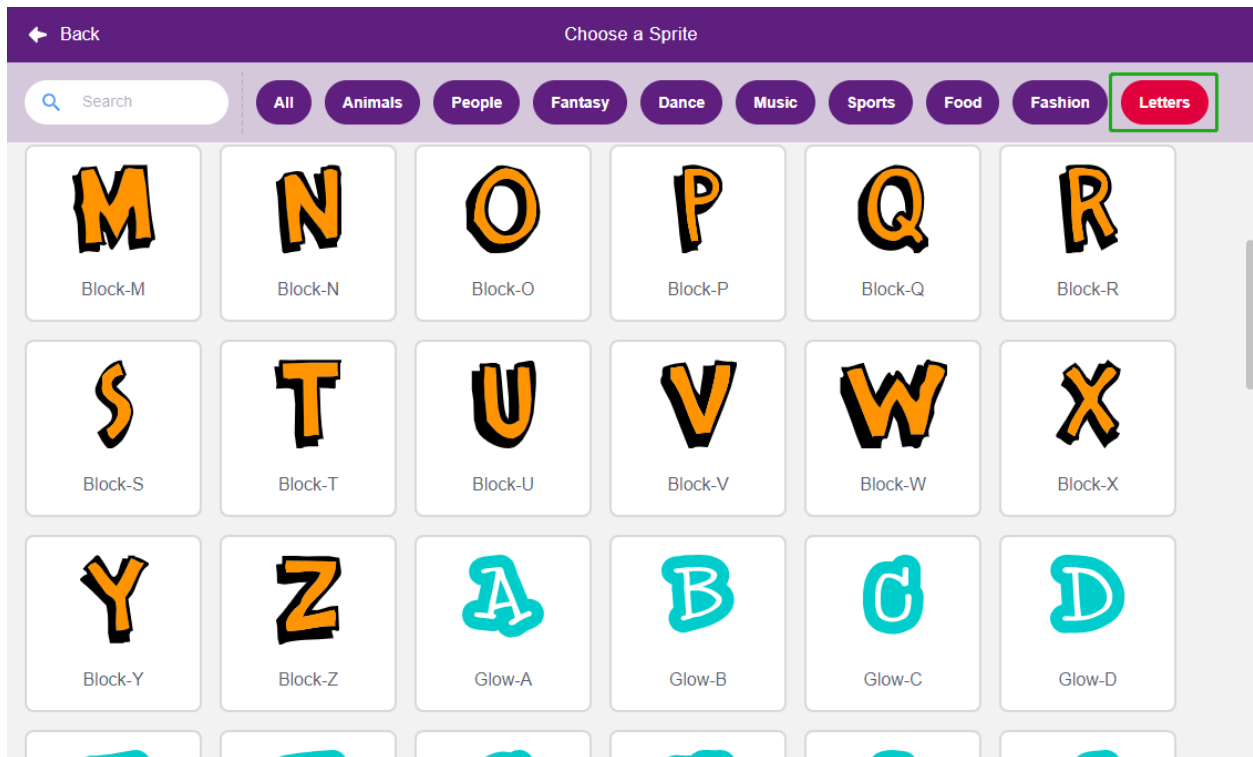
### 8.7.3 Construire le Circuit



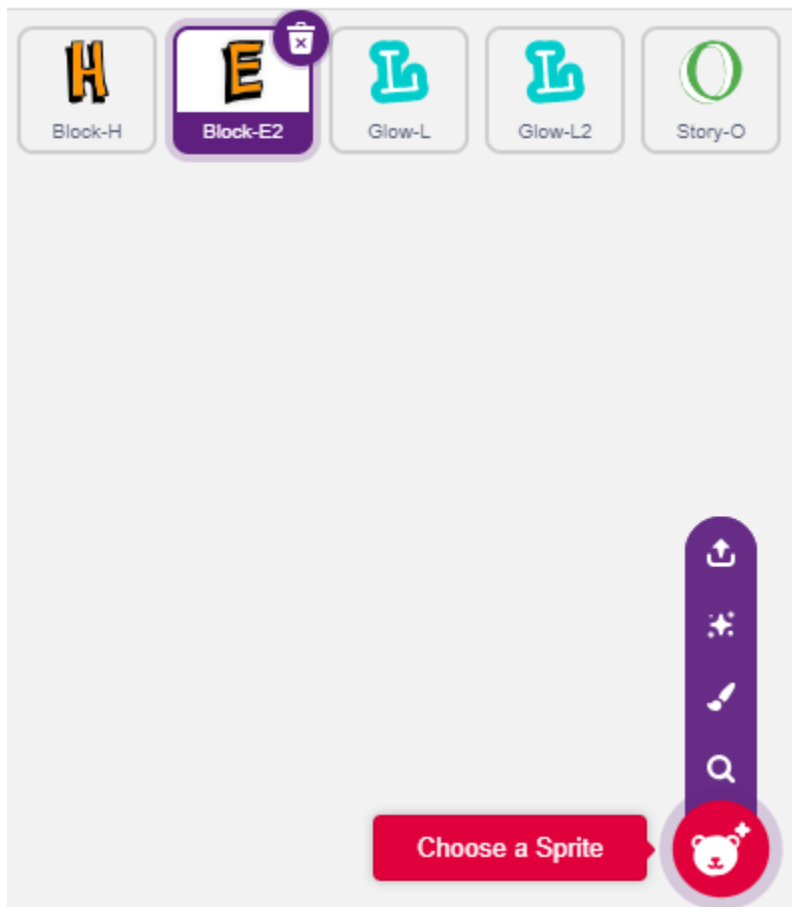
### 8.7.4 Programmation

#### 1. Sélectionner un sprite

Supprimez le sprite par défaut, cliquez sur **Choose a Sprite**, puis cliquez sur **letters** et sélectionnez le sprite désiré.



Par exemple, j'ai choisi Hello, comme montré ci-dessous.



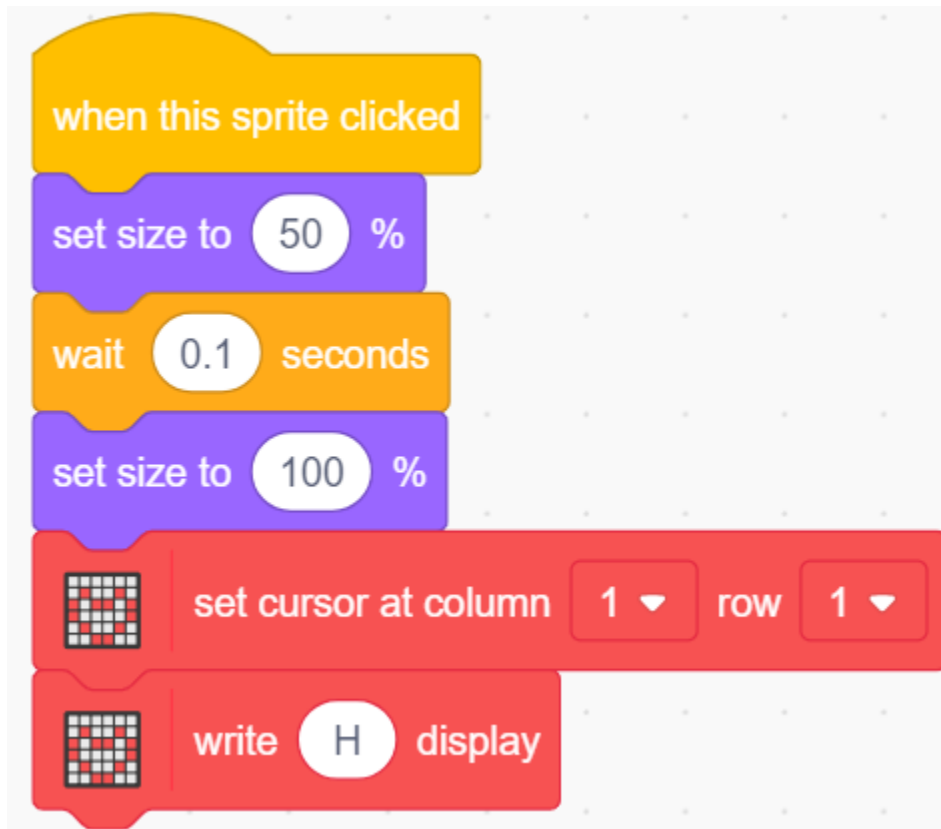
Maintenant, définissons différents effets pour ces sprites et affichons-les sur le LCD1602 en cliquant.

## 2. H est agrandissement et réduction

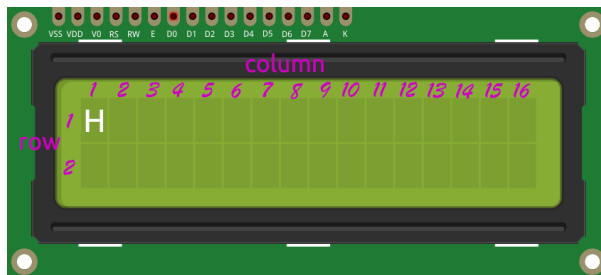
Cliquez sur le sprite **H**, et écrivez maintenant un script pour lui.

Lorsque le sprite **H** est cliqué, réduisez sa taille à 50 %, puis restaurez-la ; tout en affichant H sur la première ligne et la première colonne du LCD1602.

- [set size to] : De la palette **Looks**, utilisé pour régler la taille du sprite, de 0 % à 100 %.
- [set cursor at column row] : De la palette **Display Modules**, utilisé pour positionner le curseur à une ligne spécifique du LCD1602 pour commencer à afficher des caractères.
- [write display] : De la palette **Display Modules**, utilisé pour afficher des caractères ou des chaînes sur le LCD1602.



La distribution des lignes et des colonnes sur le LCD1602 est illustrée dans la figure.



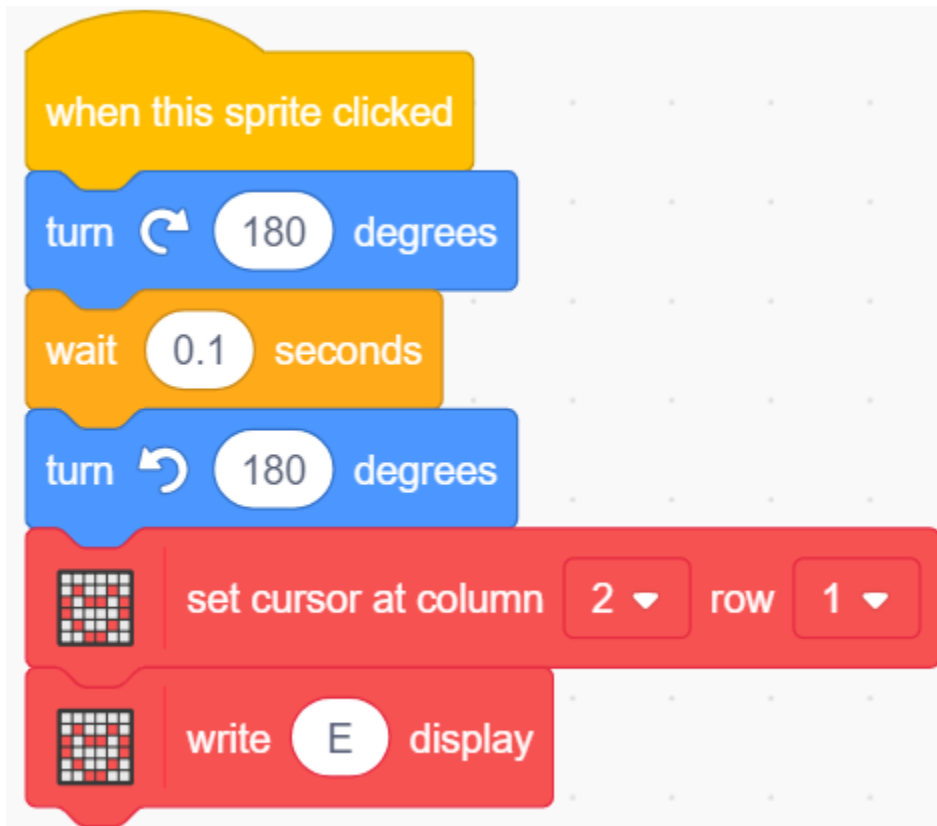
### 3. E est basculement gauche et droite

Cliquez sur le sprite **E**, et écrivez maintenant un script pour lui.

Lorsque le sprite **E** est cliqué, faites-le tourner de 180 degrés dans le sens des aiguilles d'une montre, puis de 180 degrés dans le sens contraire, afin de le voir basculer de gauche à droite ; et affichez H dans la première ligne et la colonne 2 du LCD1602.

- [turn degrees] : De la palette **Motions**, utilisé pour faire tourner le sprite dans le sens horaire ou antihoraire, la plage est de 0 à 360 degrés.



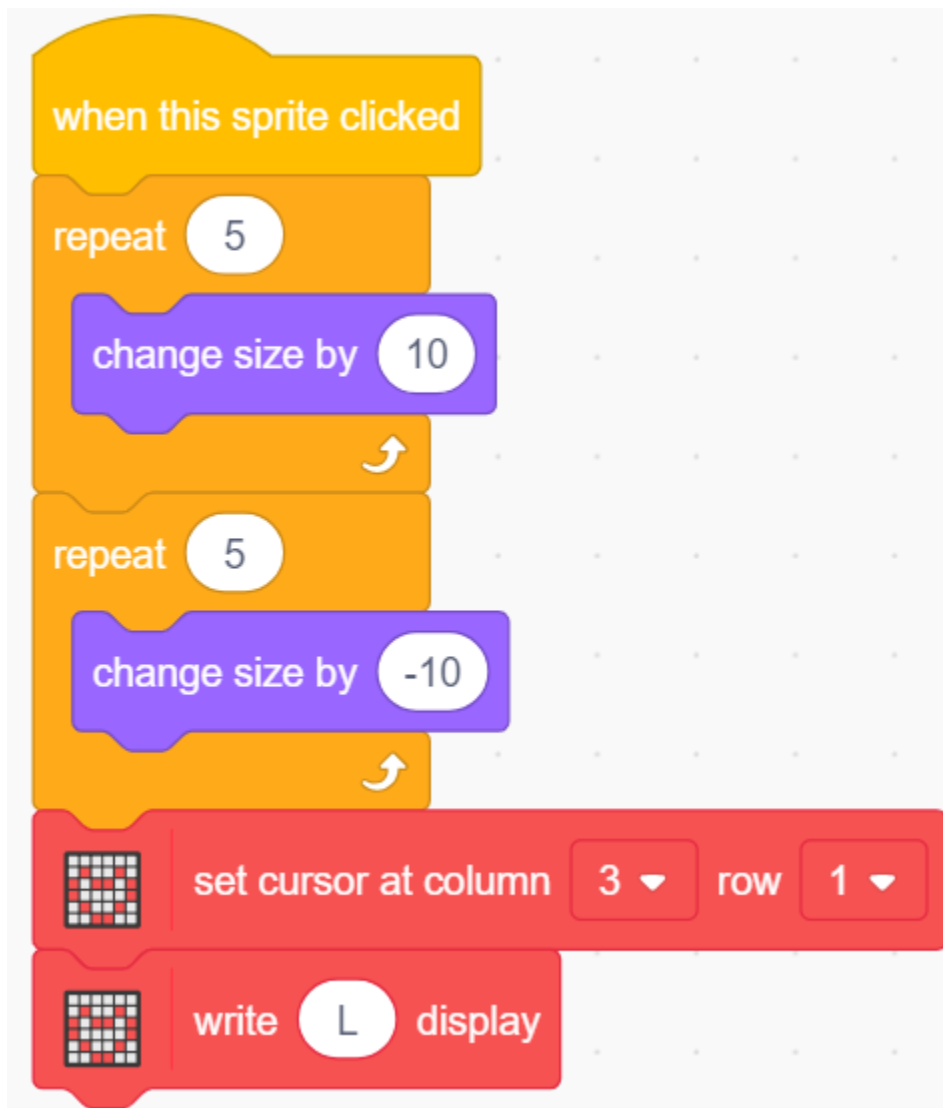


#### 4. L se rétrécit lentement et s'agrandit

Cliquez sur le sprite **first L** et écrivez maintenant un script pour lui.

Lorsque le sprite **L** est cliqué, utilisez le bloc [repeat] pour augmenter sa taille de 50 % (5 fois, 10 % à chaque fois), puis réduisez-la à sa taille originale de la même manière, tout en affichant L dans la première ligne et la colonne 3 du LCD1602.

- [change size by] : De la palette Mouvements, utilisé pour changer la taille du sprite.

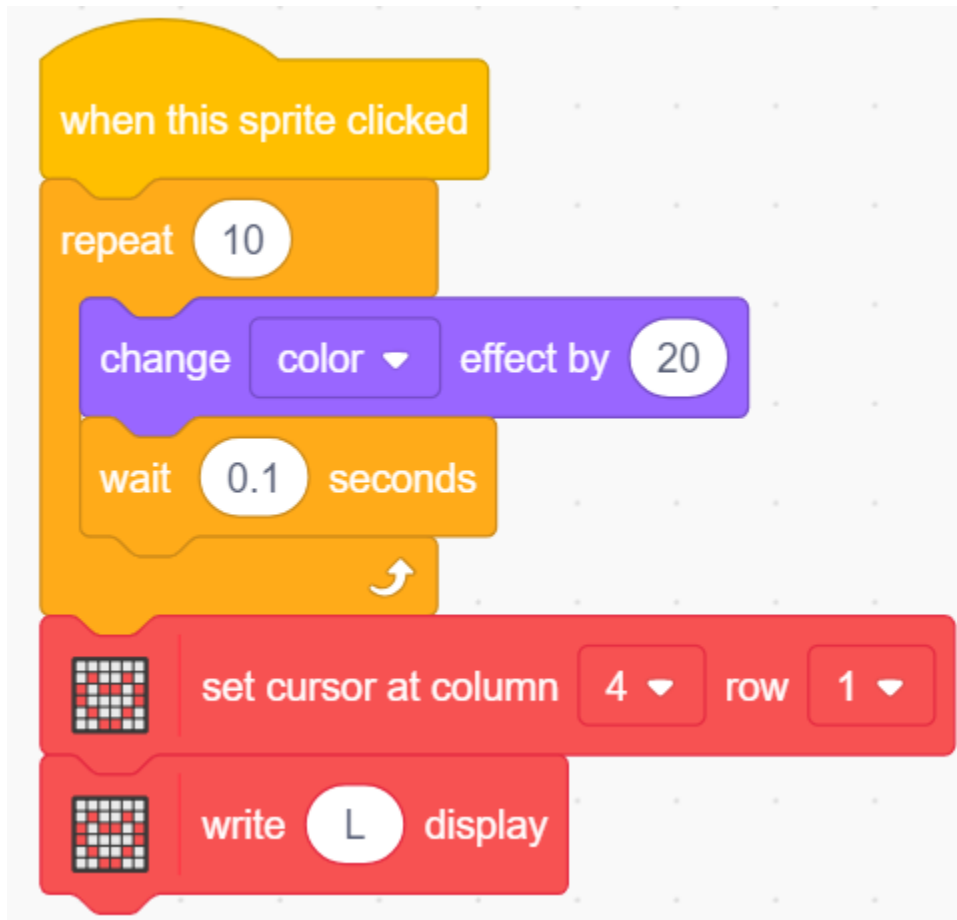


### 5. Le second L change de couleur

Cliquez sur le sprite **second L** et écrivez maintenant un script pour lui.

Lorsque le sprite **L** est cliqué, utilisez le bloc [repeat] pour répéter 10 fois à un rythme de 20 incréments pour changer de couleurs et revenir à la couleur originale. Affichez également L dans la première ligne et la colonne 4 du LCD1602.

- [change color effect by] : Utilisé pour changer l'Effet de couleur, un costume peut prendre 200 schémas de couleurs différents avec l'effet de couleur, 0 et 200 sont la même couleur.

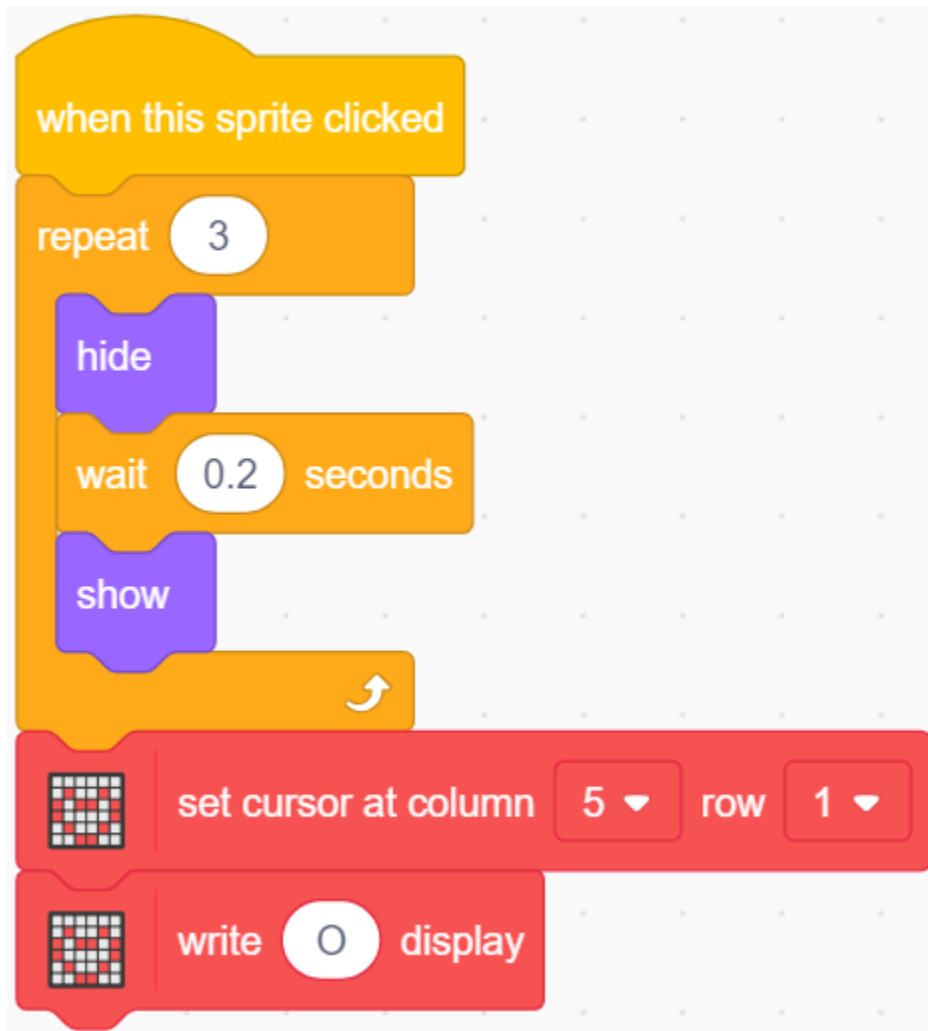


#### 6. O est caché et affiché

Cliquez sur le sprite **O** et écrivez maintenant un script pour lui.

Lorsque le sprite **O** est cliqué, il répète le processus de caché et affiché 3 fois, tout en affichant O dans la première ligne et la colonne 5 du LCD1602.

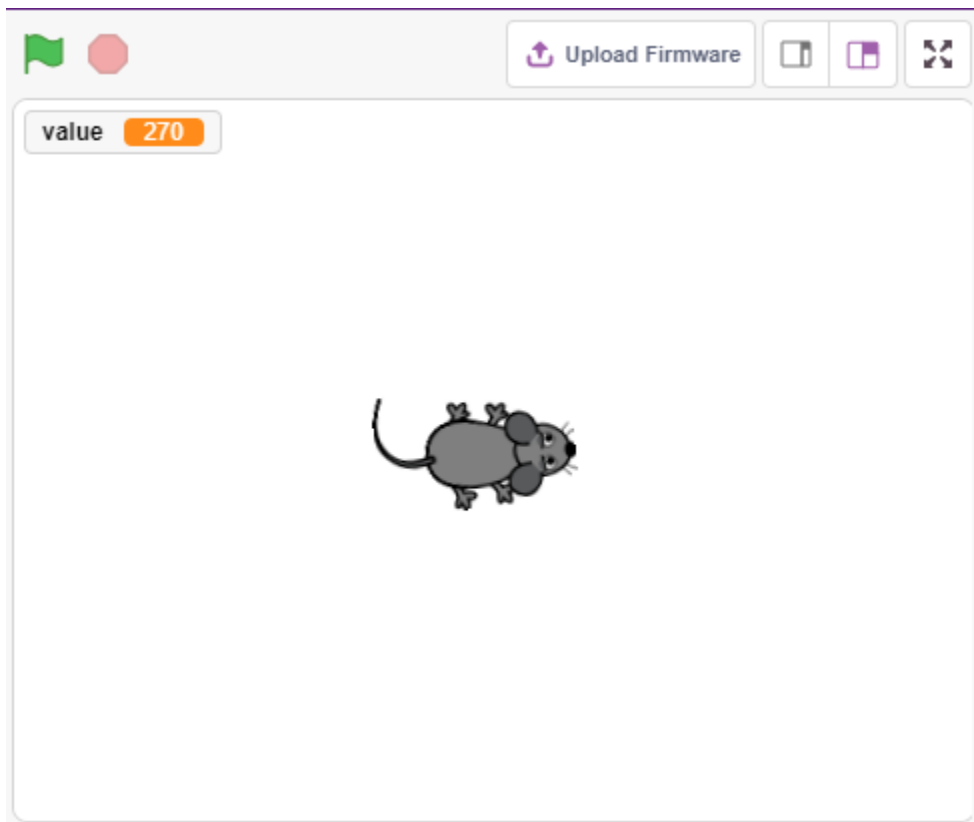
— [Hide] & [Show] : pour rendre le sprite caché et visible.



## 8.8 2.5 Souris Mobile

Aujourd'hui, nous allons créer un jouet en forme de souris contrôlé par un potentiomètre.

Lorsque le drapeau vert est cliqué, la souris sur la scène avance, et lorsque vous tournez le potentiomètre, la souris change de direction de mouvement.



### 8.8.1 Vous Apprendrez

- Principe du potentiomètre
- Lire une broche analogique et ses plages de valeurs
- Mapper une plage de valeurs à une autre
- Déplacer et changer la direction d'un sprite

### 8.8.2 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

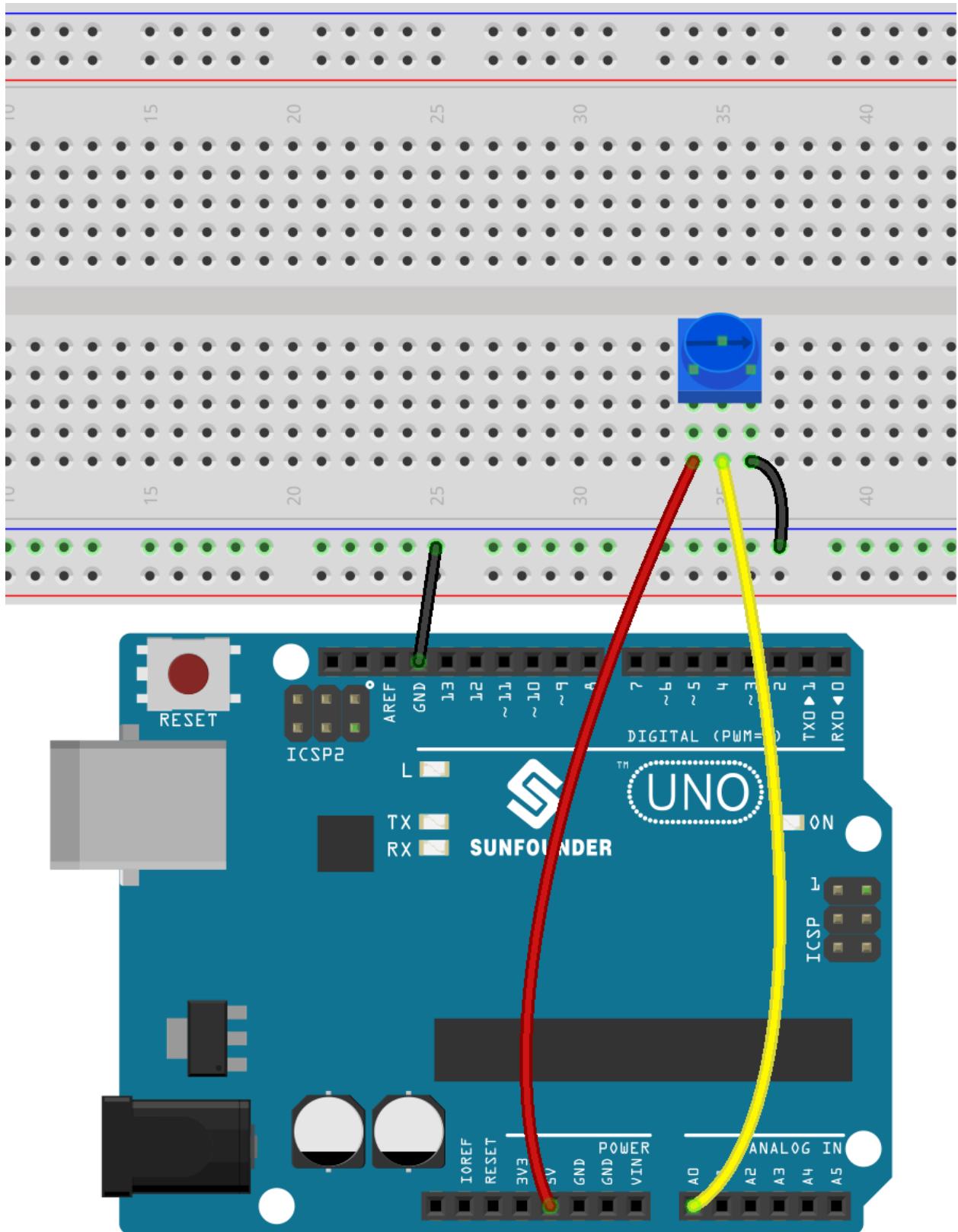
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Potentiomètre</i>	

### 8.8.3 Construire le Circuit

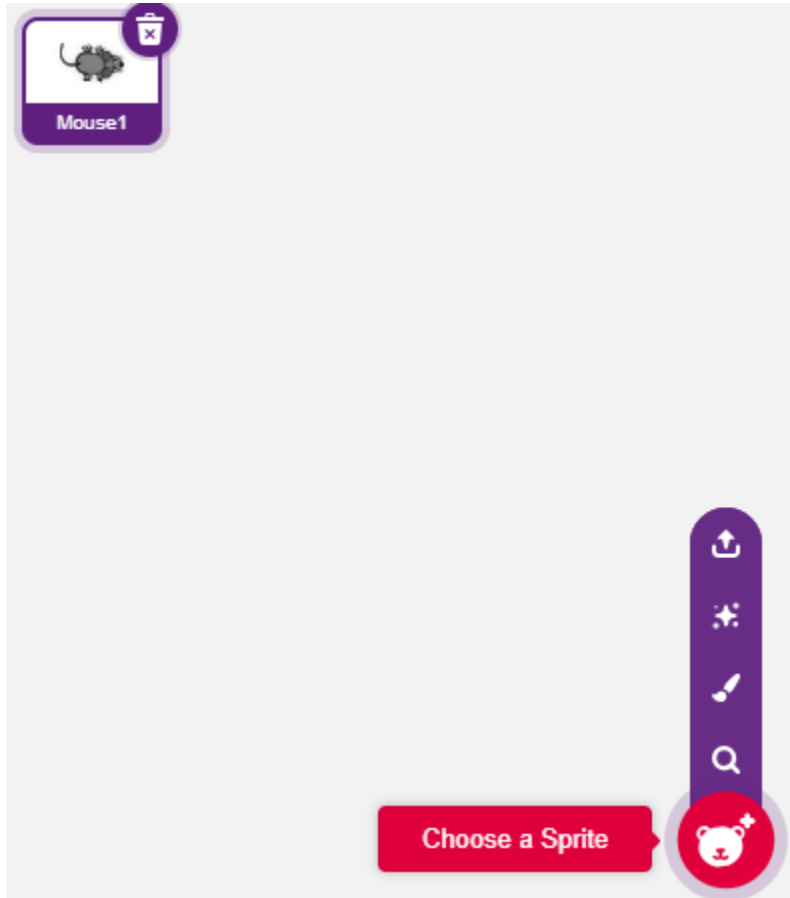
Le potentiomètre est un élément résistif avec 3 bornes, les 2 broches latérales sont connectées à 5V et GND, et la broche centrale est connectée à A0. Après conversion par le convertisseur ADC de la carte Arduino, la plage de valeurs est de 0-1023.



## 8.8.4 Programmation

### 1. Choisir un sprite

Supprimez le sprite par défaut, cliquez sur le bouton **Choose a Sprite** dans le coin inférieur droit de la zone de sprite, entrez **mouse** dans la boîte de recherche, puis cliquez pour l'ajouter.

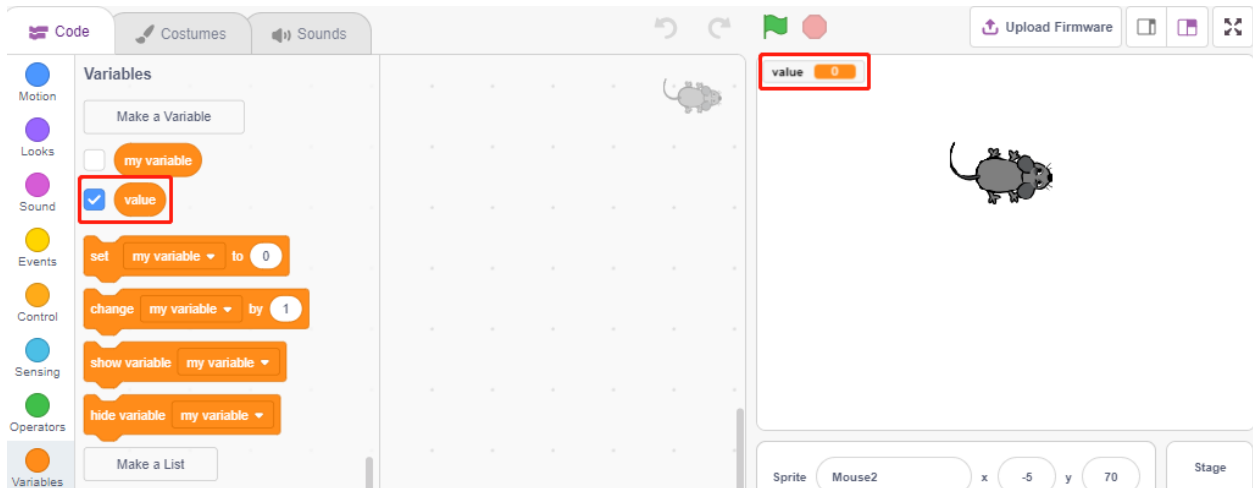


### 2. Création d'une variable.

Créez une variable nommée **value** pour stocker la valeur lue du potentiomètre.

Une fois créée, vous verrez **value** apparaître dans la palette **Variables** et dans l'état coché, ce qui signifie que cette variable apparaîtra sur la scène.

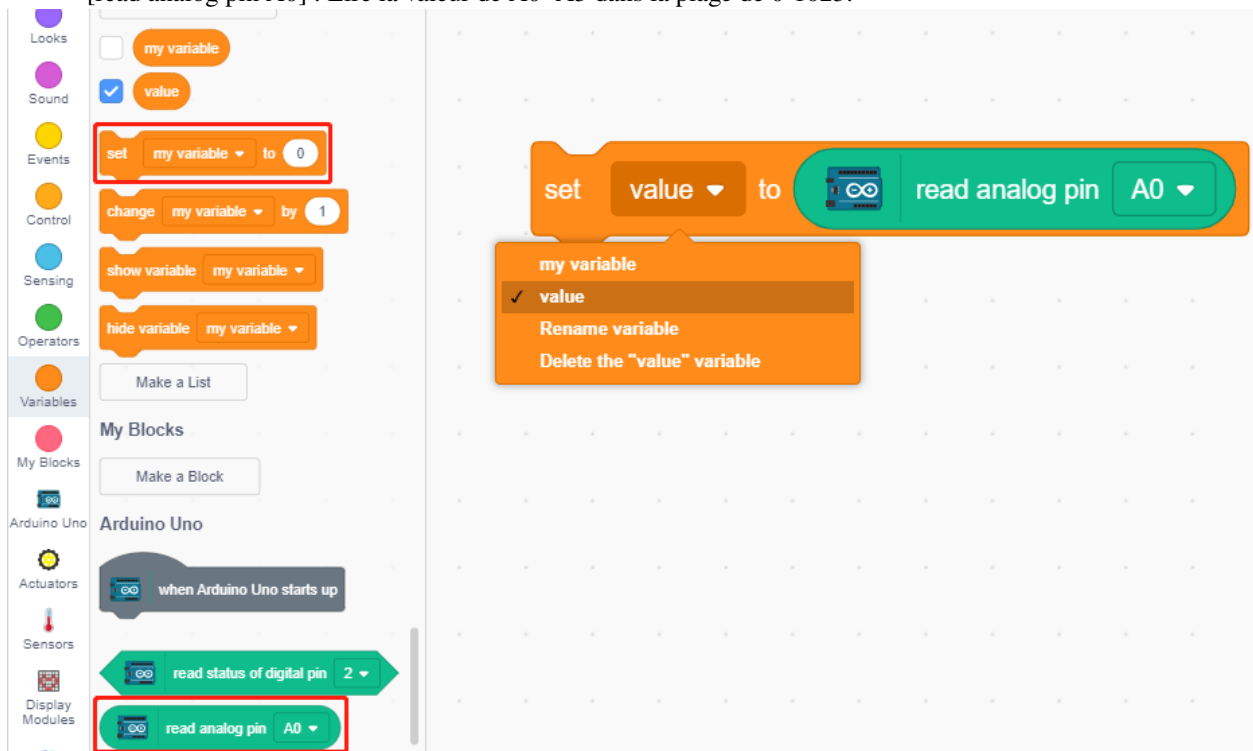




### 3. Lire la valeur de A0

Stockez la valeur lue de A0 dans la variable **value**.

- [set my variable to 0] : Définir la valeur de la variable.
- [read analog pin A0] : Lire la valeur de A0~A5 dans la plage de 0-1023.

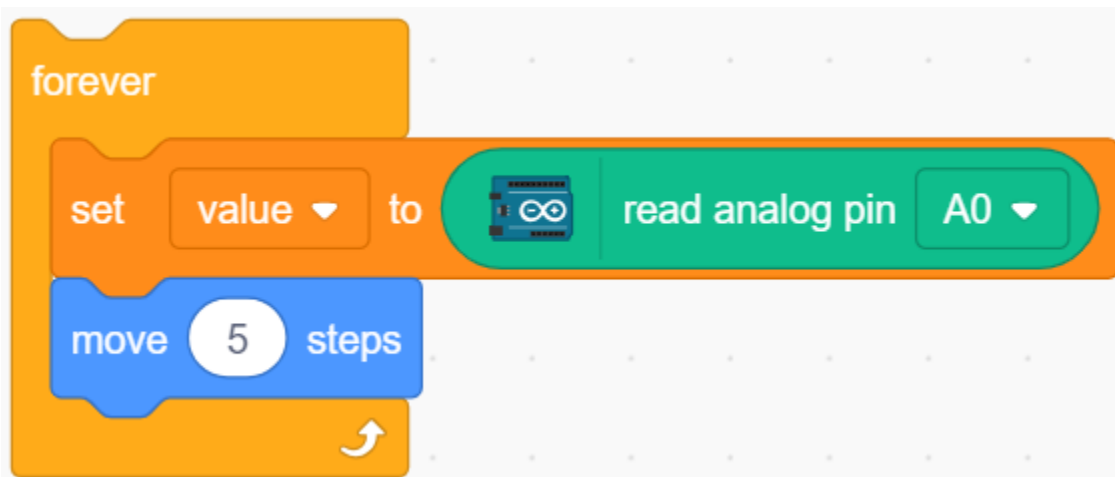


Pour pouvoir lire en continu, vous devez utiliser le bloc [forever]. Cliquez sur ce script pour l'exécuter, tournez le potentiomètre dans les deux sens, et vous verrez que la plage de valeurs est de 0-1023.



#### 4. Déplacer le sprite

Utilisez le bloc [move steps] pour déplacer le sprite, exécutez le script et vous verrez le sprite se déplacer du milieu vers la droite.

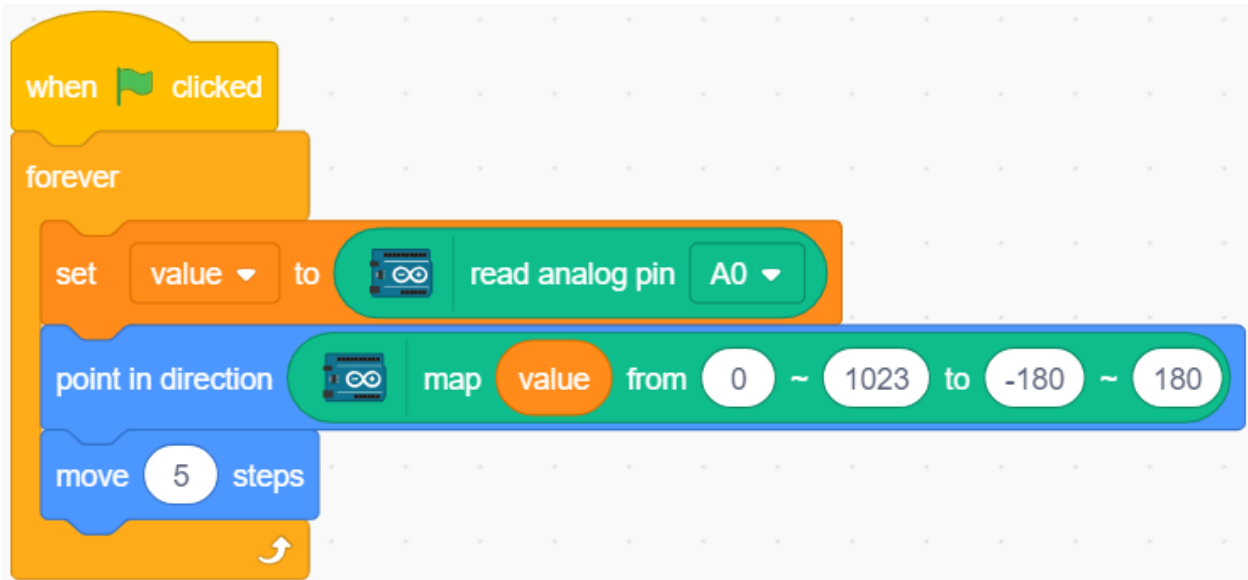


#### 5. Changer la direction du sprite

Changez maintenant la direction du mouvement du sprite par la valeur de A0. Comme la valeur de A0 varie de 0-1023, mais la direction de rotation du sprite est de -180~180, un bloc [map] doit être utilisé.

Ajoutez également [when green flag clicked] au début pour démarrer le script.

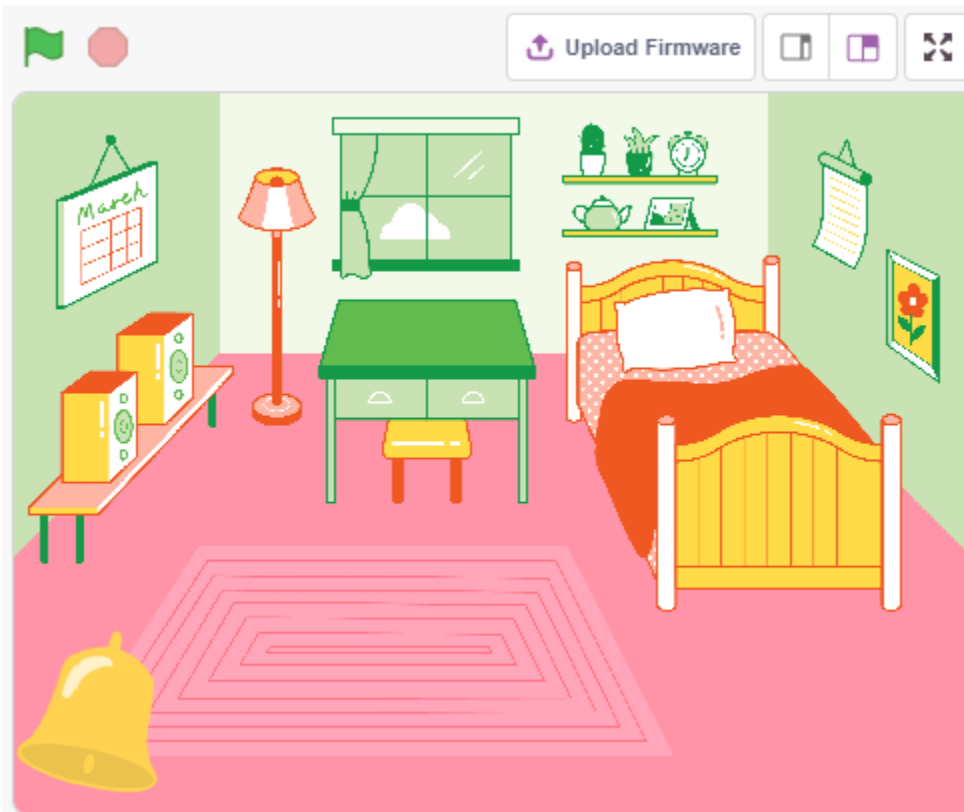
- [point in direction] : Définir l'angle de direction du sprite, de la palette **Motion**.
- [map from to] : Mapper une plage à une autre plage.



## 8.9 2.6 Sonnette

Ici, nous utiliserons le bouton et la cloche sur la scène pour créer une sonnette.

Après avoir cliqué sur le drapeau vert, vous pouvez appuyer sur le bouton et la cloche sur la scène émettra un son.



### 8.9.1 Vous Apprendrez

- Fonctionnement du bouton
- Lecture d'une broche numérique et ses plages de valeurs
- Création d'une boucle conditionnelle
- Ajout d'un arrière-plan
- Jouer un son

### 8.9.2 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

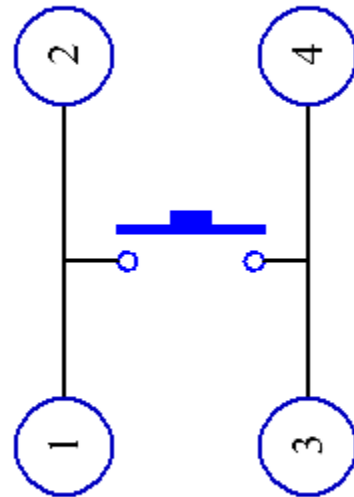
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Bouton</i>	
<i>Condensateur</i>	

### 8.9.3 Construire le Circuit

Le bouton est un dispositif à 4 broches, puisque la broche 1 est connectée à la broche 2, et la broche 3 à la broche 4, lorsque le bouton est pressé, les 4 broches sont connectées, fermant ainsi le circuit.



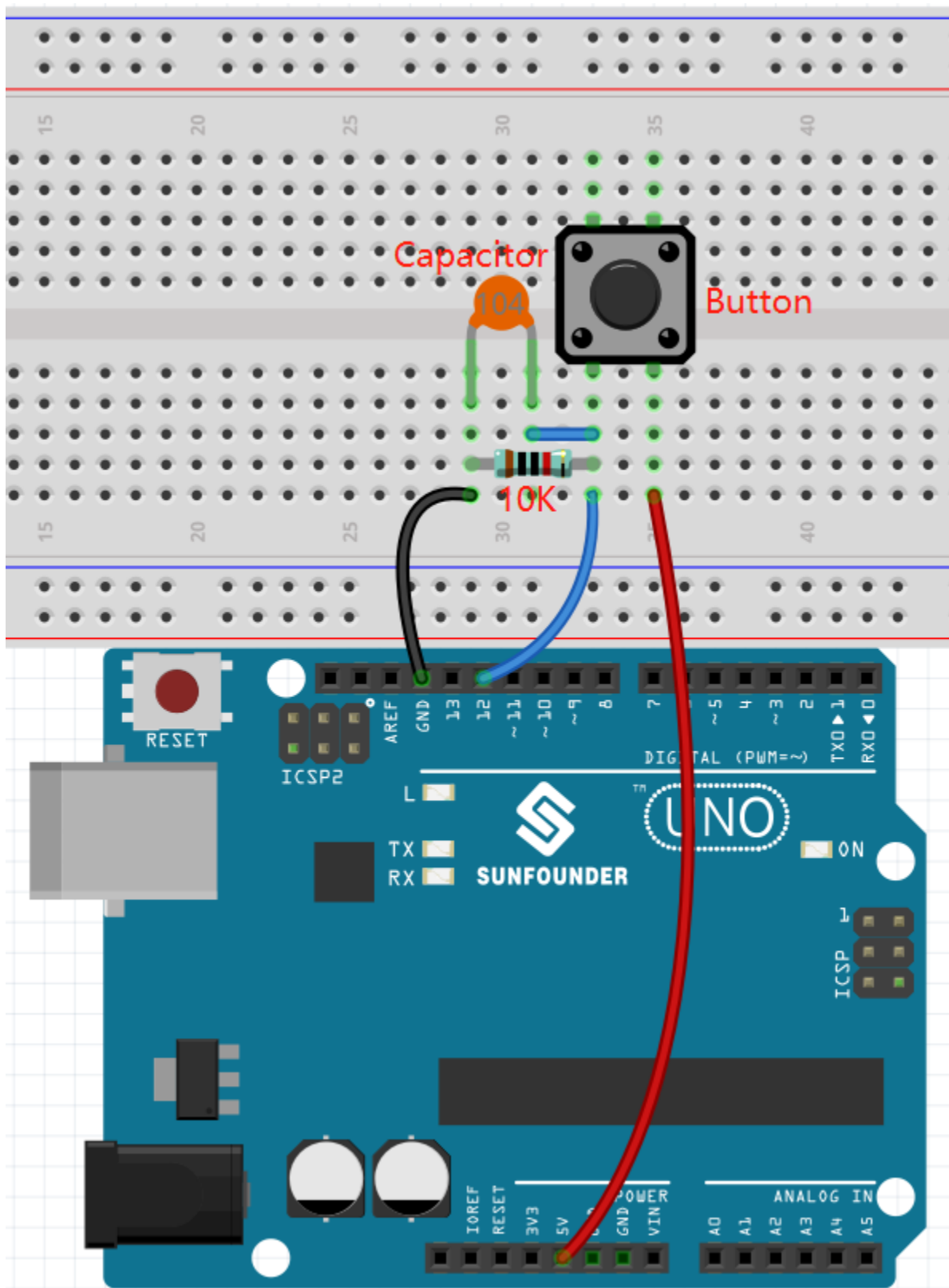
Button



Internal Structure

Construisez le circuit selon le schéma suivant.

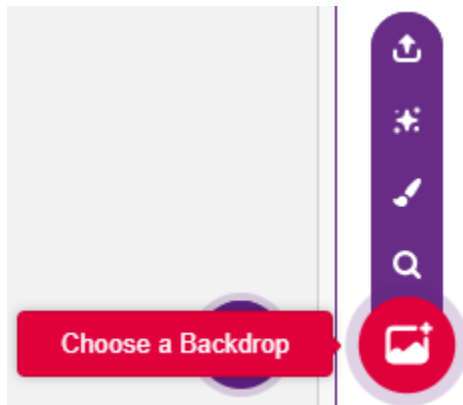
- Connectez l'une des broches du côté gauche du bouton à la broche 12, qui est connectée à une résistance de tirage vers le bas et un condensateur de 0.1uF (104) (pour éliminer le jitter et produire un niveau stable lorsque le bouton fonctionne).
- Connectez l'autre extrémité de la résistance et du condensateur à GND, et l'une des broches du côté droit du bouton à 5V.



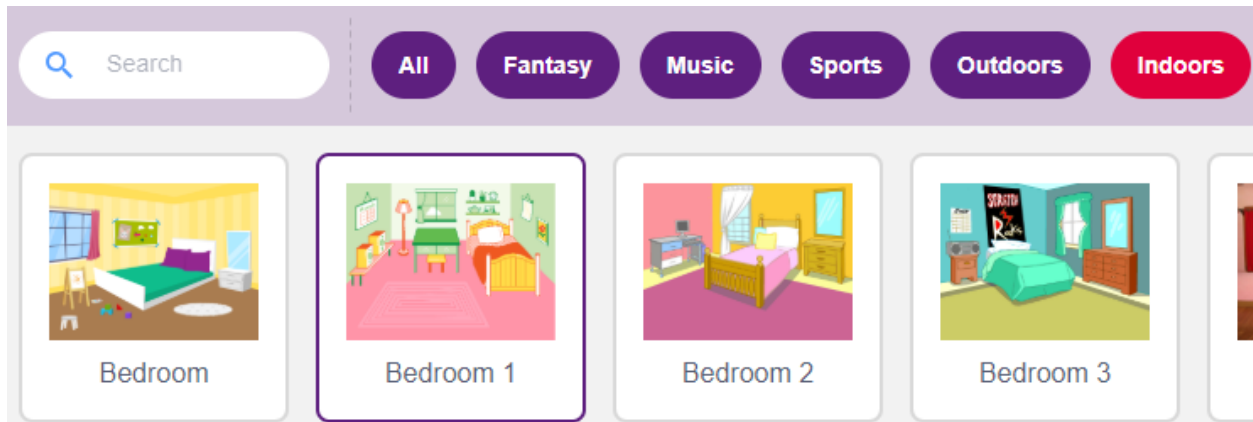
## 8.9.4 Programmation

### 1. Ajouter un arrière-plan

Cliquez sur le bouton **Choose a Backdrop** dans le coin inférieur droit.

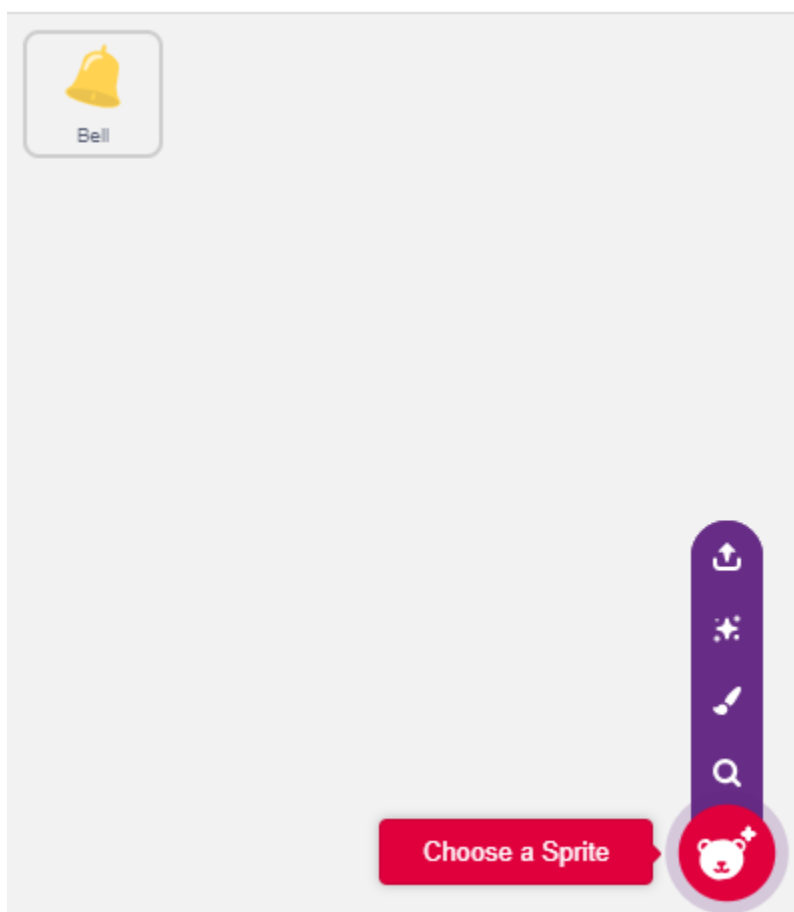


Choisissez **Bedroom 1**.



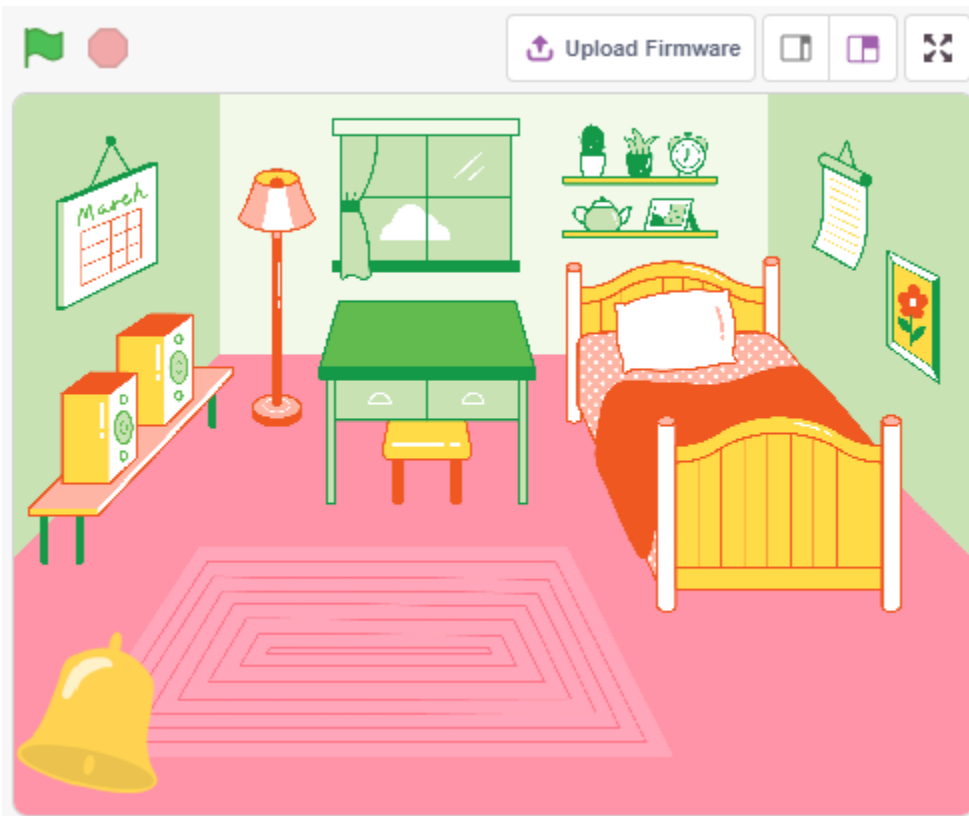
### 2. Sélectionner le sprite

Supprimez le sprite par défaut, cliquez sur le bouton **Choose a Sprite** dans le coin inférieur droit de la zone des sprites, entrez **bell** dans la boîte de recherche, puis cliquez pour l'ajouter.



Ensuite, sélectionnez le sprite **bell** sur la scène et placez-le à la position appropriée.

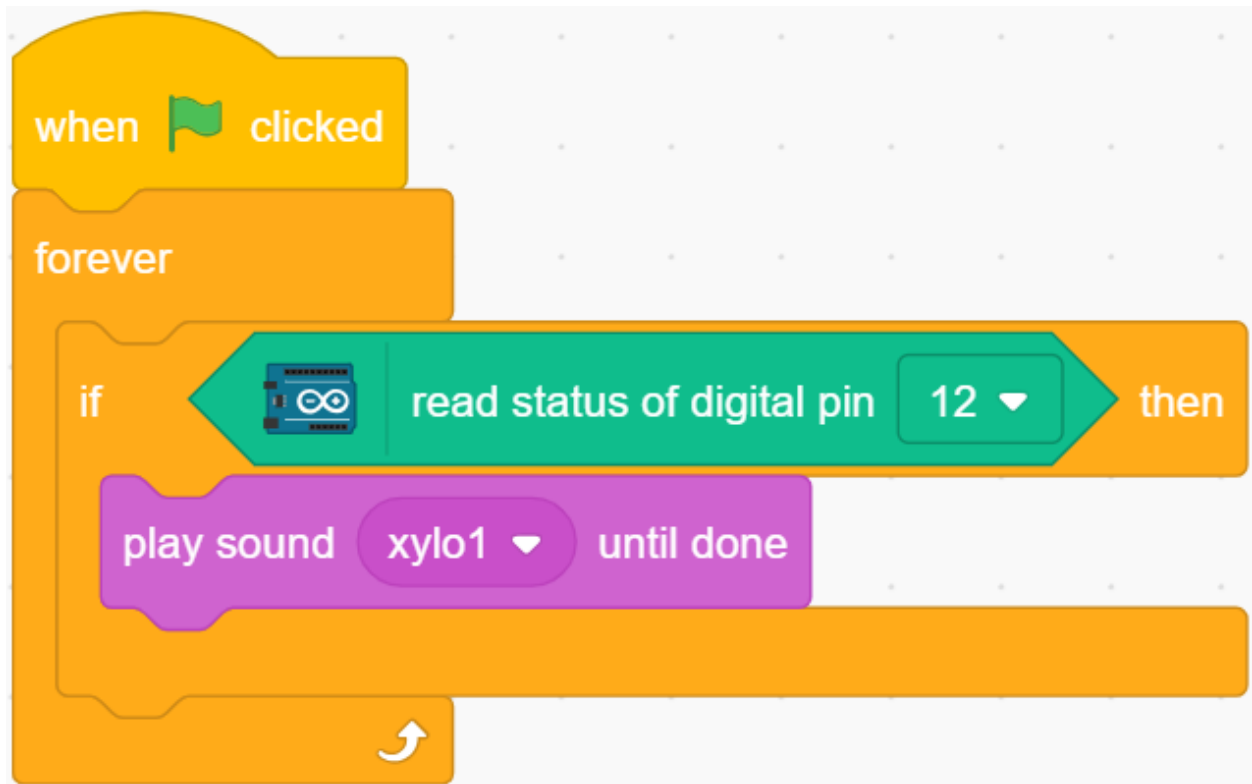




### 3. Appuyez sur le bouton et la cloche émet un son

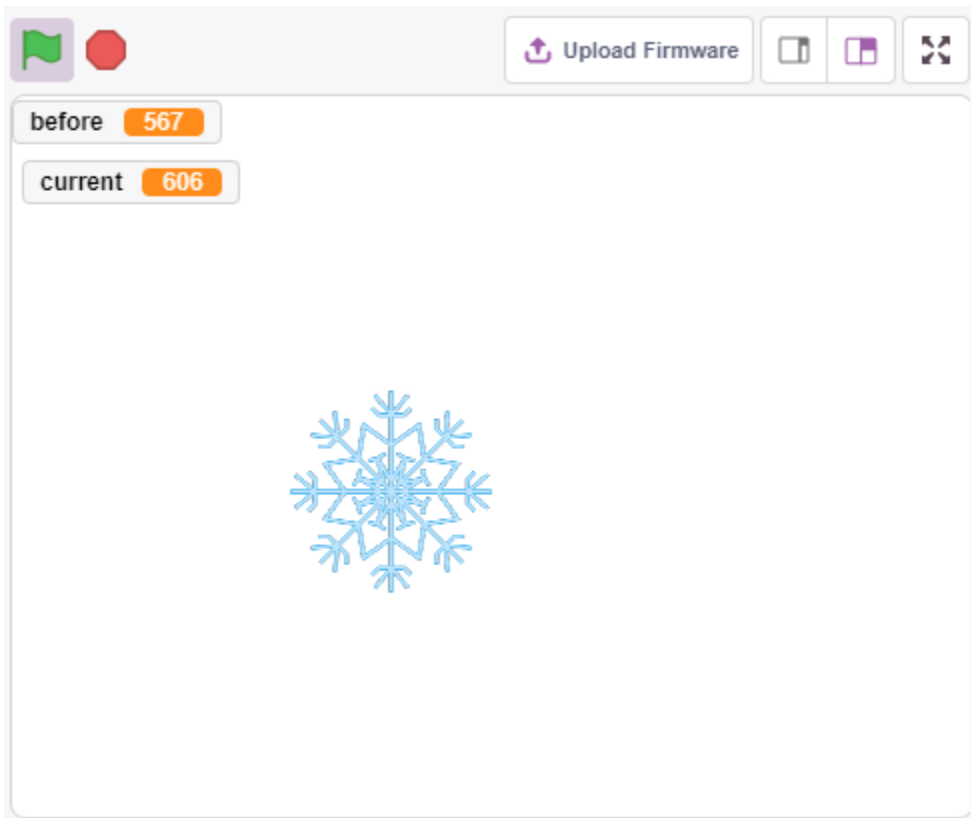
Utilisez [if then] pour faire une déclaration conditionnelle qui, lorsque la valeur de la broche 12 lue est égale à 1 (la touche est pressée), le son **xylo1** sera joué.

- [read status of digital pin] : Ce bloc vient de la palette **Arduino Uno** et sert à lire la valeur d'une broche numérique, le résultat est 0 ou 1.
- [if then] : Ce bloc est un bloc de contrôle et provient de la palette **Contrôle**. Si sa condition booléenne est vraie, les blocs qu'il contient seront exécutés, puis le script concerné continuera. Si la condition est fausse, les scripts à l'intérieur du bloc seront ignorés. La condition n'est vérifiée qu'une seule fois ; si la condition devient fausse pendant que le script à l'intérieur du bloc est en cours d'exécution, il continuera jusqu'à sa fin.
- [play sound until done] : de la palette **Son**, utilisé pour jouer des sons spécifiques.



## 8.10 2.7 Alarme de Température Basse

Dans ce projet, nous allons créer un système d'alarme de basse température. Lorsque la température descend sous un seuil défini, le sprite **Flocon de Neige** apparaîtra sur la scène.



### 8.10.1 Vous Apprendrez

- Principe de fonctionnement d'une thermistance
- Opérations multivariables et soustractives

### 8.10.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Thermistance</i>	

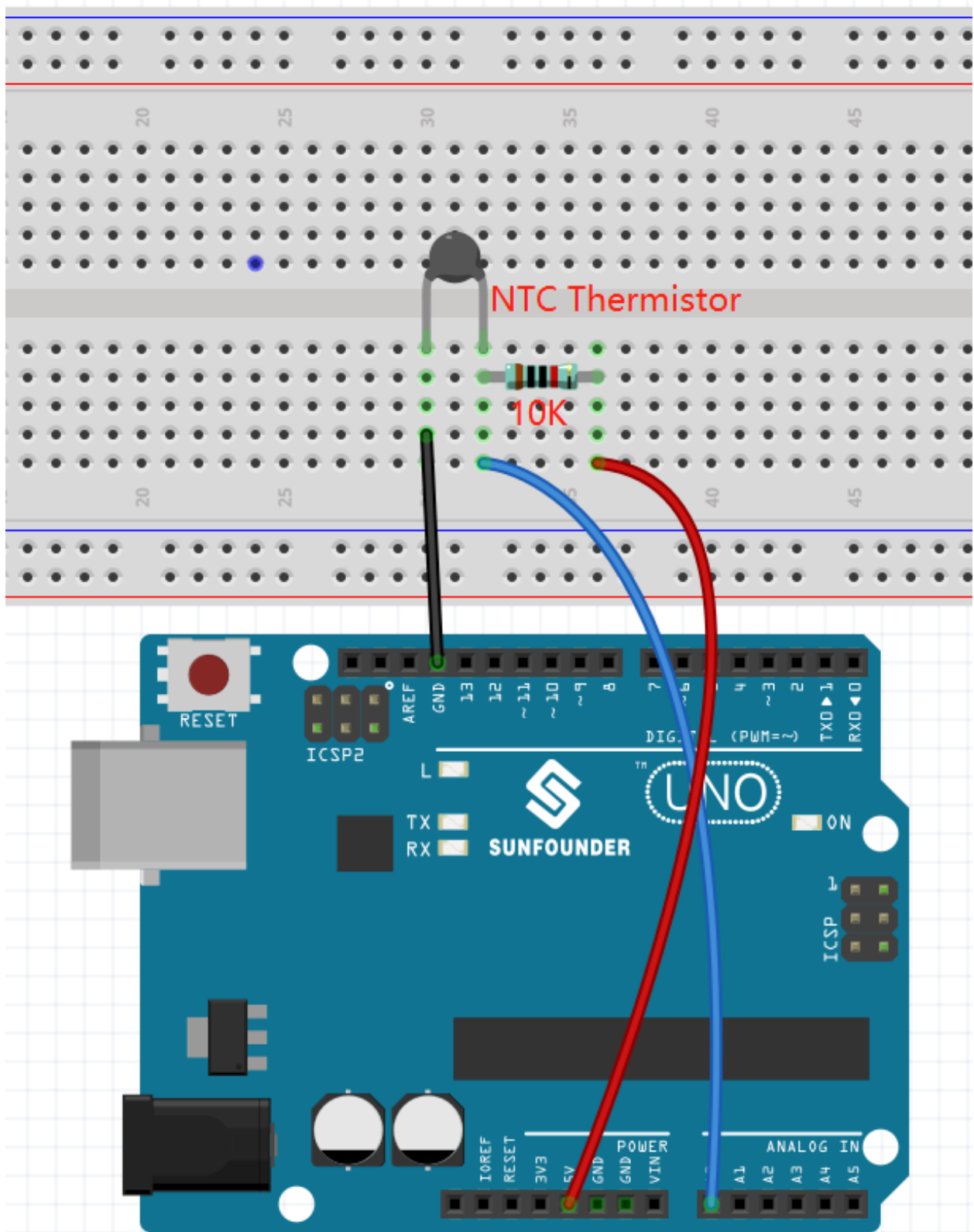
### 8.10.3 Construisez le Circuit

Une thermistance est un type de résistance dont la résistance dépend fortement de la température, bien plus que dans les résistances standards. Il existe deux types de résistances, PTC (la résistance augmente avec la température) et NTC (la résistance diminue avec la température).

Construisez le circuit selon le schéma suivant.

Une extrémité de la thermistance est connectée à GND, l'autre à A0, et une résistance de 10K est connectée en série à 5V.

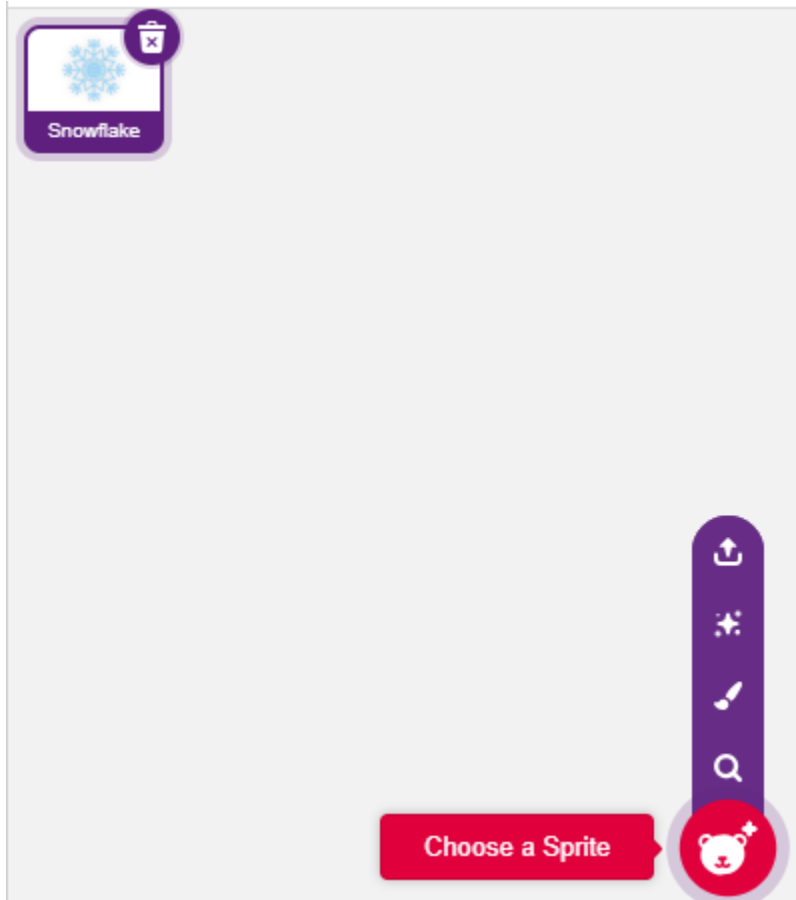
La thermistance NTC est utilisée ici, donc lorsque la température augmente, la résistance de la thermistance diminue, la division de tension de A0 diminue, et la valeur obtenue de A0 diminue, et inversement elle augmente.



## 8.10.4 Programmation

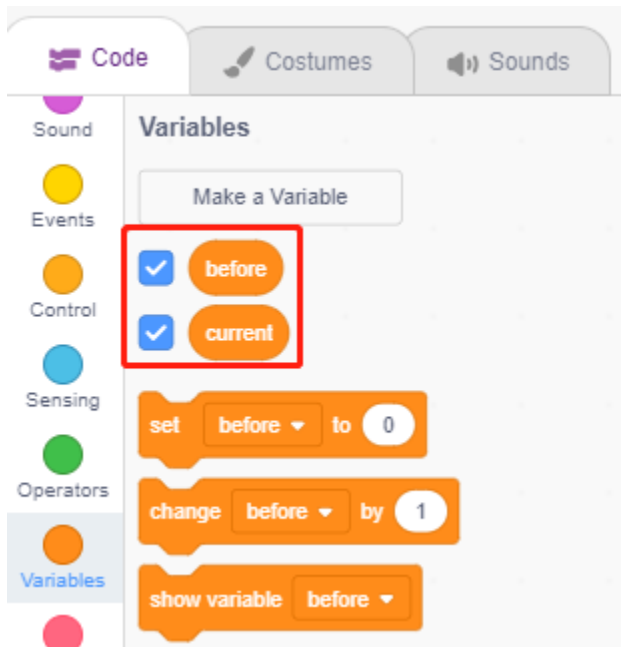
### 1. Sélectionnez un sprite

Supprimez le sprite par défaut, cliquez sur le bouton **Choose a Sprite** dans le coin inférieur droit de la zone des sprites, entrez **Snowflake** dans la barre de recherche, puis cliquez pour l'ajouter.



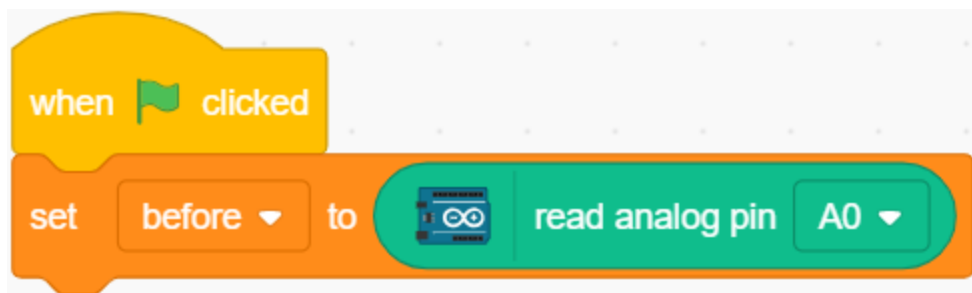
### 2. Créez 2 variables

Créez deux variables, **before** et **current**, pour stocker la valeur de A0 dans différents cas.



### 3. Lisez la valeur de A0

Lorsque le drapeau vert est cliqué, la valeur de A0 est lue et stockée dans la variable **before**.



### 4. Lisez à nouveau la valeur de A0

Dans [forever], lisez à nouveau la valeur de A0 et stockez-la dans la variable **current**.

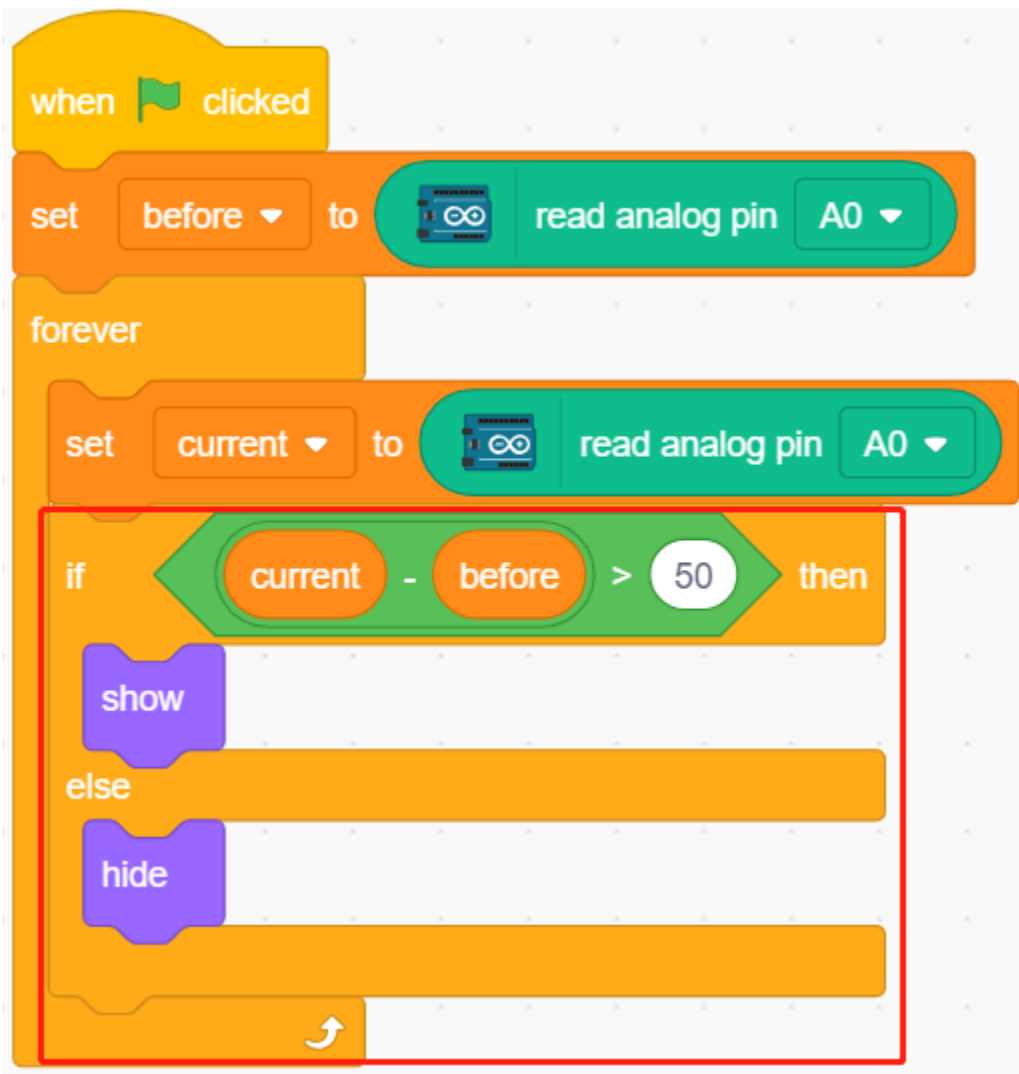


### 5. Détermination des changements de température

Utilisez le bloc [if else] pour déterminer si la valeur actuelle de A0 est supérieure de 50 à celle d'avant, ce qui représente une baisse de température. Dans ce cas, laissez le sprite **Snowflake** apparaître, sinon cachez-le.

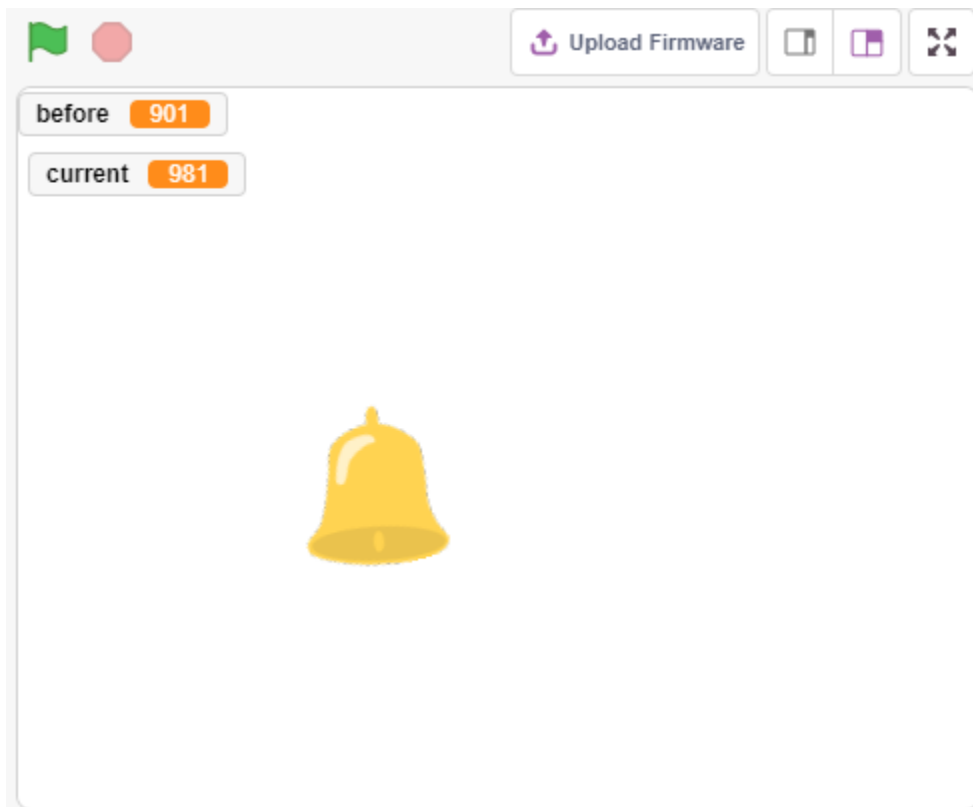
— [-] & [>] : opérateurs de soustraction et de comparaison de la palette **Operators**.





## 8.11 2.8 Réveil Lumineux

Dans la vie, il existe divers types de réveils. Créons maintenant un réveil commandé par la lumière. Quand le matin arrive, la luminosité augmente et ce réveil lumineux vous rappellera qu'il est temps de se lever.



### 8.11.1 Vous Apprendrez

- Principe de fonctionnement d'une photorésistance
- Arrêter la lecture de son et stopper l'exécution des scripts

### 8.11.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Photorésistance</i>	

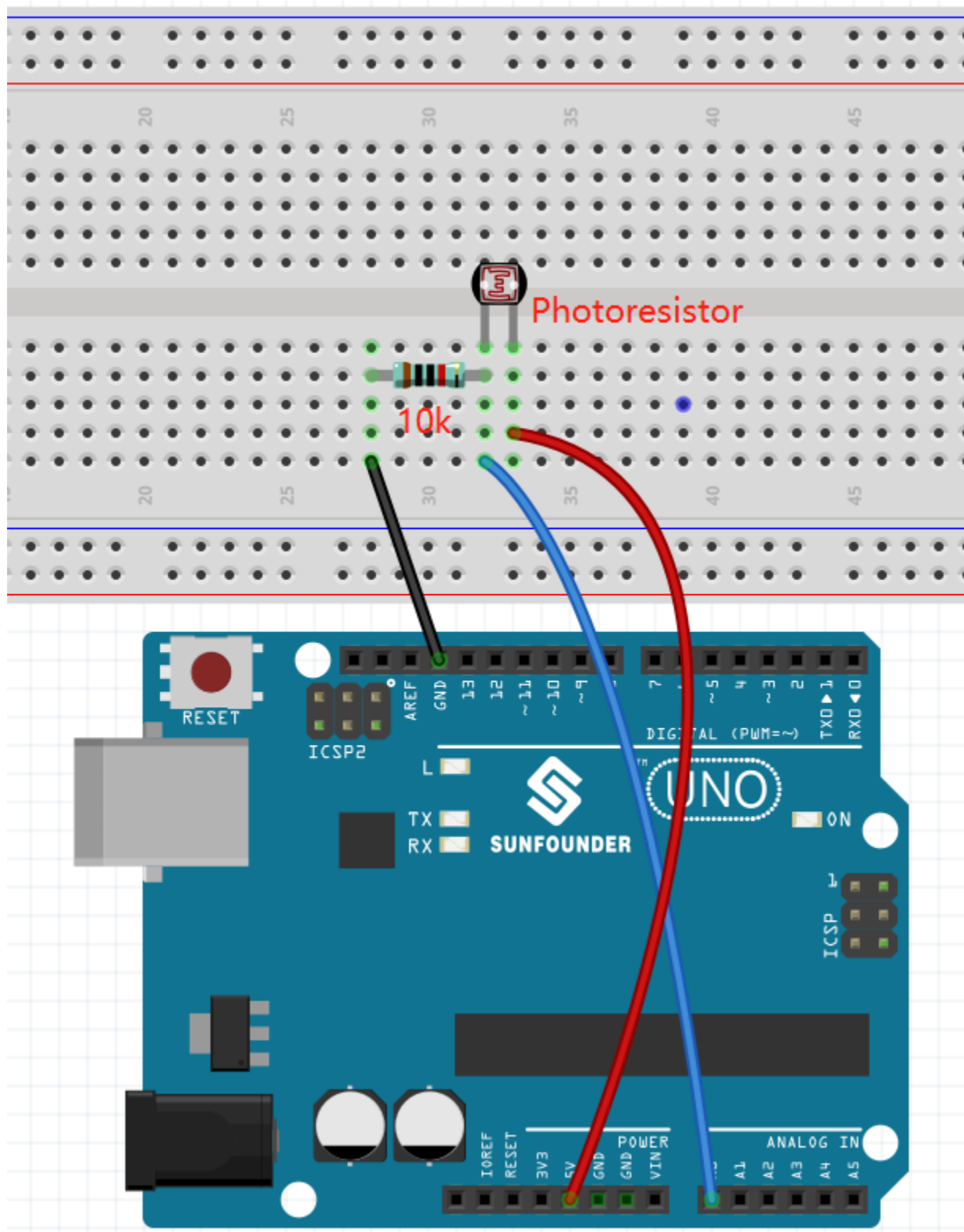
### 8.11.3 Construisez le Circuit

Une photorésistance ou cellule photoélectrique est une résistance variable contrôlée par la lumière. La résistance d'une photorésistance diminue avec l'augmentation de l'intensité lumineuse incidente.

Construisez le circuit selon le schéma suivant.

Connectez une extrémité de la photorésistance à 5V, l'autre à A0, et connectez une résistance de 10K en série avec GND à cette extrémité.

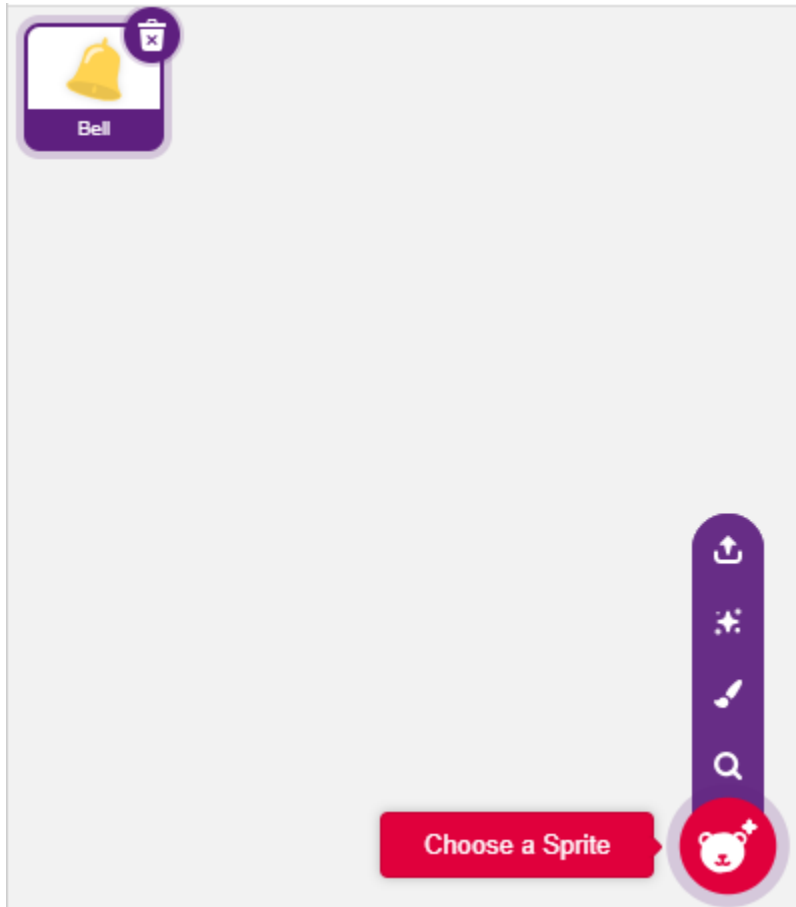
Ainsi, lorsque l'intensité lumineuse augmente, la résistance de la photorésistance diminue, la division de tension de la résistance de 10K augmente, et la valeur obtenue par A0 devient plus grande.



### 8.11.4 Programmation

#### 1. Sélectionnez un sprite

Supprimez le sprite par défaut, cliquez sur le bouton **Choose a Sprite** dans le coin inférieur droit de la zone des sprites, entrez **bell** dans la barre de recherche, puis cliquez pour l'ajouter.



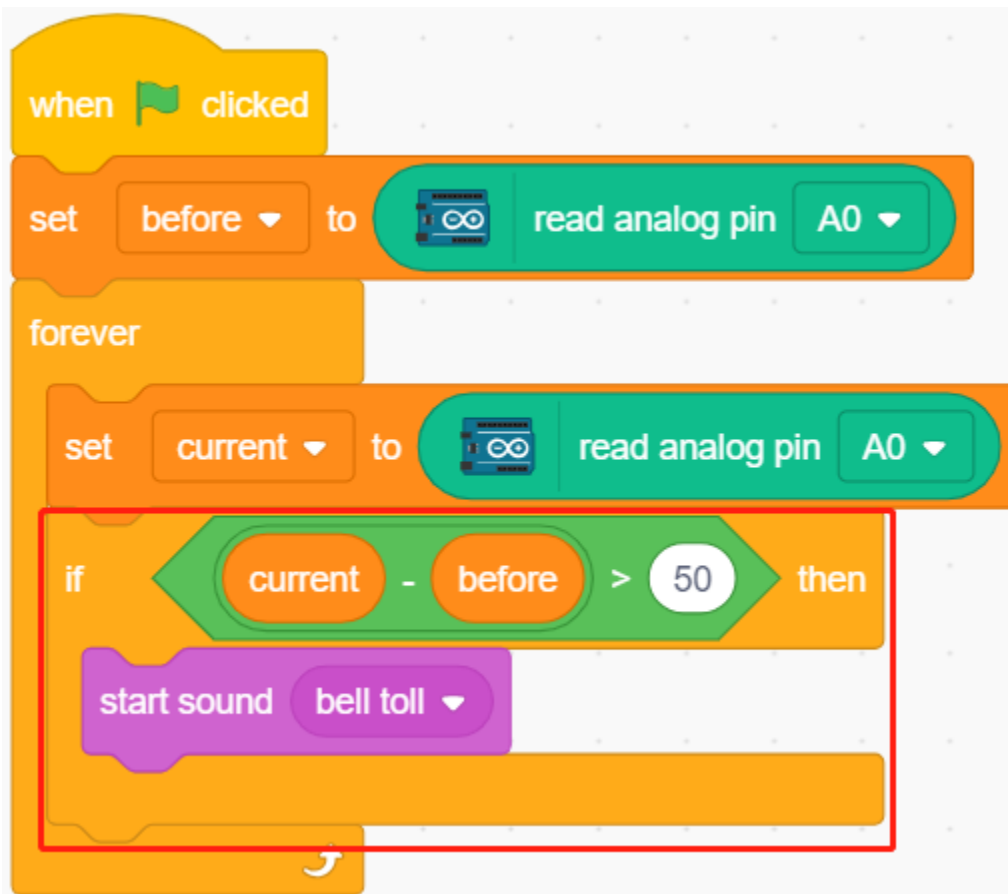
#### 2. Lisez la valeur de A0

Créez deux variables **before** et **current**. Lorsque le drapeau vert est cliqué, lisez la valeur de A0 et stockez-la dans la variable **before** comme valeur de référence. Dans [forever], lisez à nouveau la valeur de A0 et stockez-la dans la variable **current**.



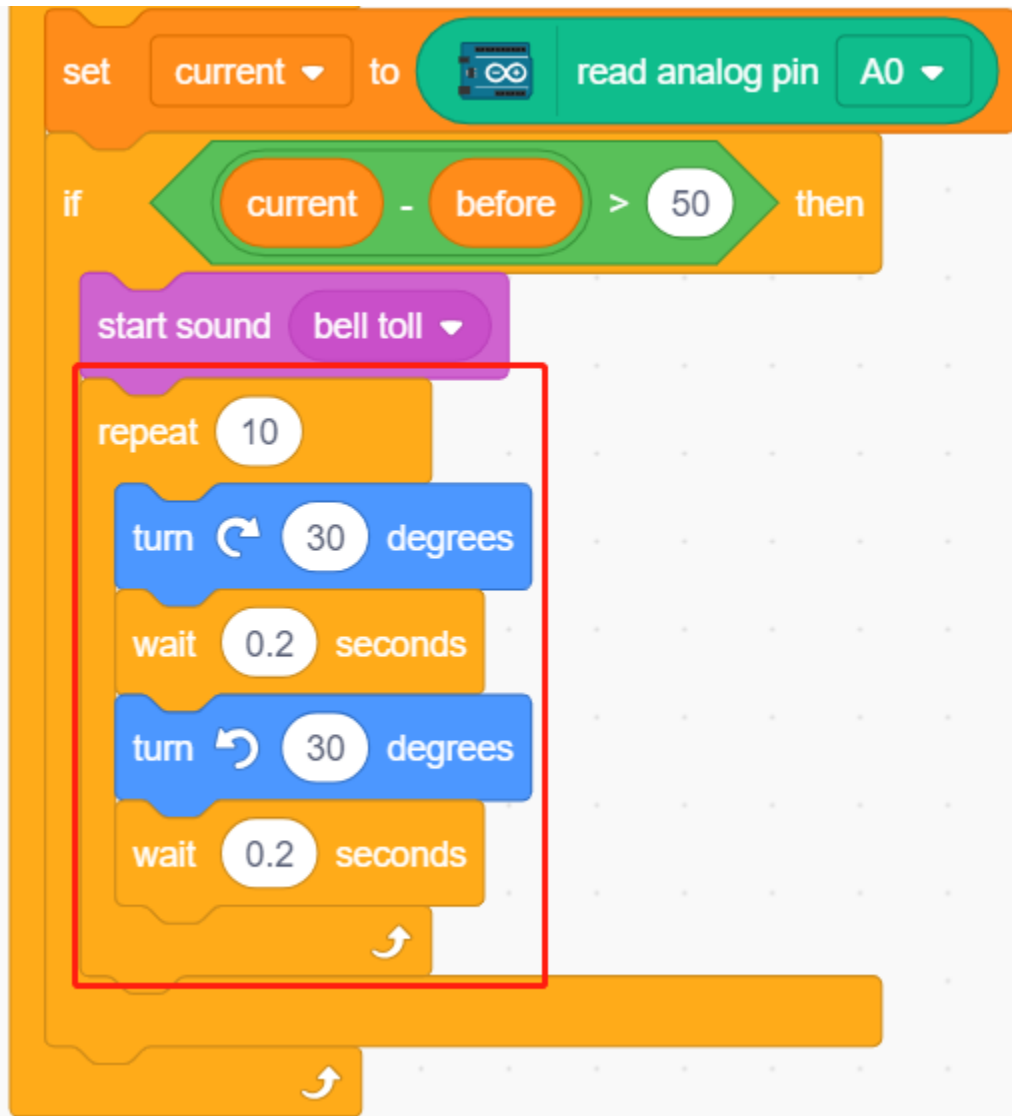
### 3. Produire un son

Lorsque la valeur actuelle de A0 est supérieure de 50 à la précédente, ce qui représente une intensité lumineuse supérieure au seuil, alors faites sonner le sprite.



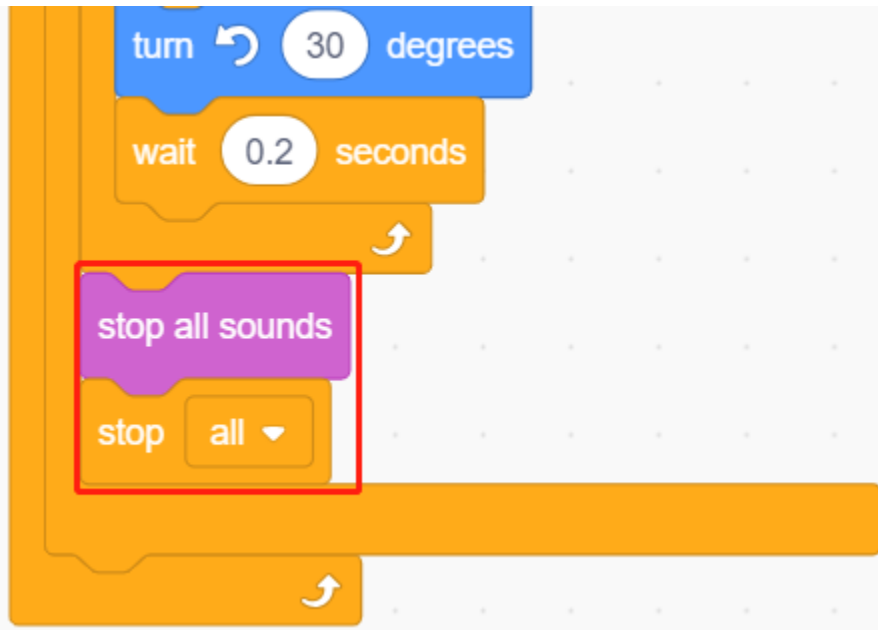
### 4. Faire tourner le sprite

Utilisez [turn block] pour faire tourner le sprite **bell** à gauche et à droite pour obtenir l'effet de réveil.



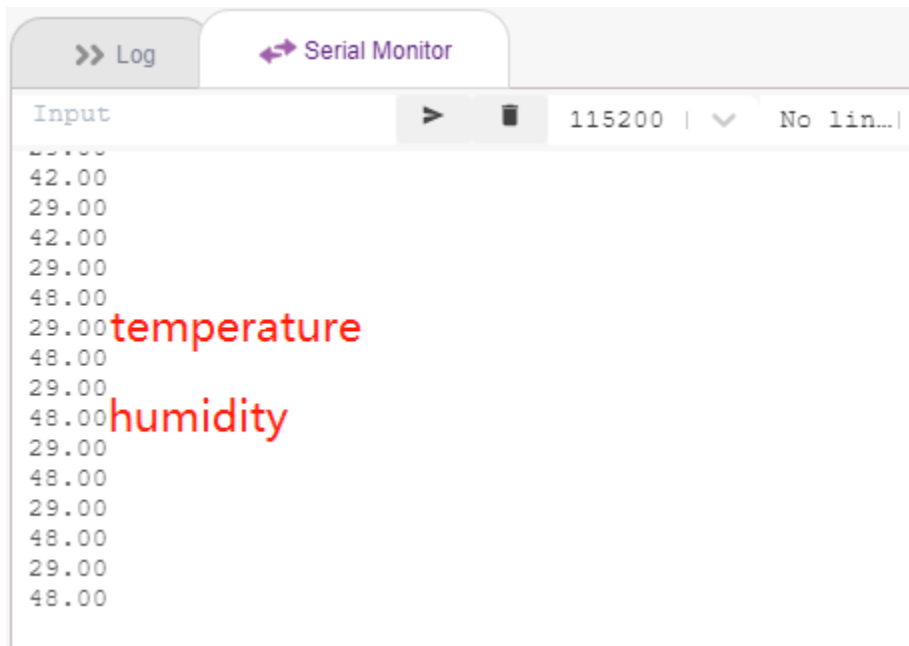
### 5. Arrêter tout

Arrêtez l'alarme après qu'elle ait sonné pendant un moment.



## 8.12 2.9 Lecture de la Température et de l'Humidité

Dans les projets précédents, nous avons utilisé le mode scène, mais certaines fonctions ne sont disponibles qu'en mode de téléversement, comme la fonction de communication série. Dans ce projet, nous imprimerons la température et l'humidité du DHT11 en utilisant le Moniteur Série dans le *Mode Téléchargement*.





### 8.12.1 Vous Apprendrez

- Obtenir la température et l'humidité du module DHT11
- Moniteur Série pour le *Mode Téléchargement*
- Ajouter une extension

### 8.12.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

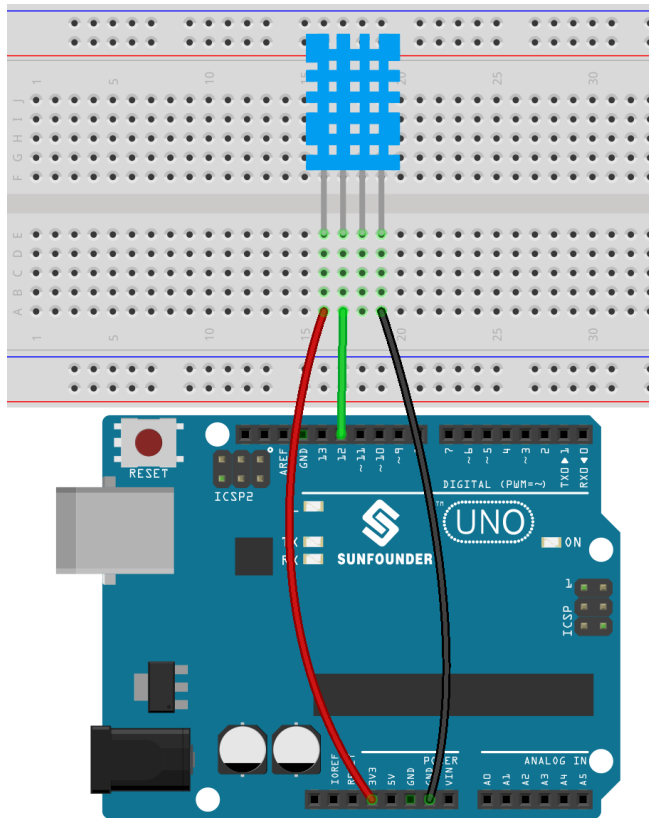
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Capteur d'Humidité et de Température DHT11</i>	-

### 8.12.3 Construisez le Circuit

Le capteur numérique de température et d'humidité DHT11 est un capteur composite qui contient une sortie de signal numérique calibrée de température et d'humidité.

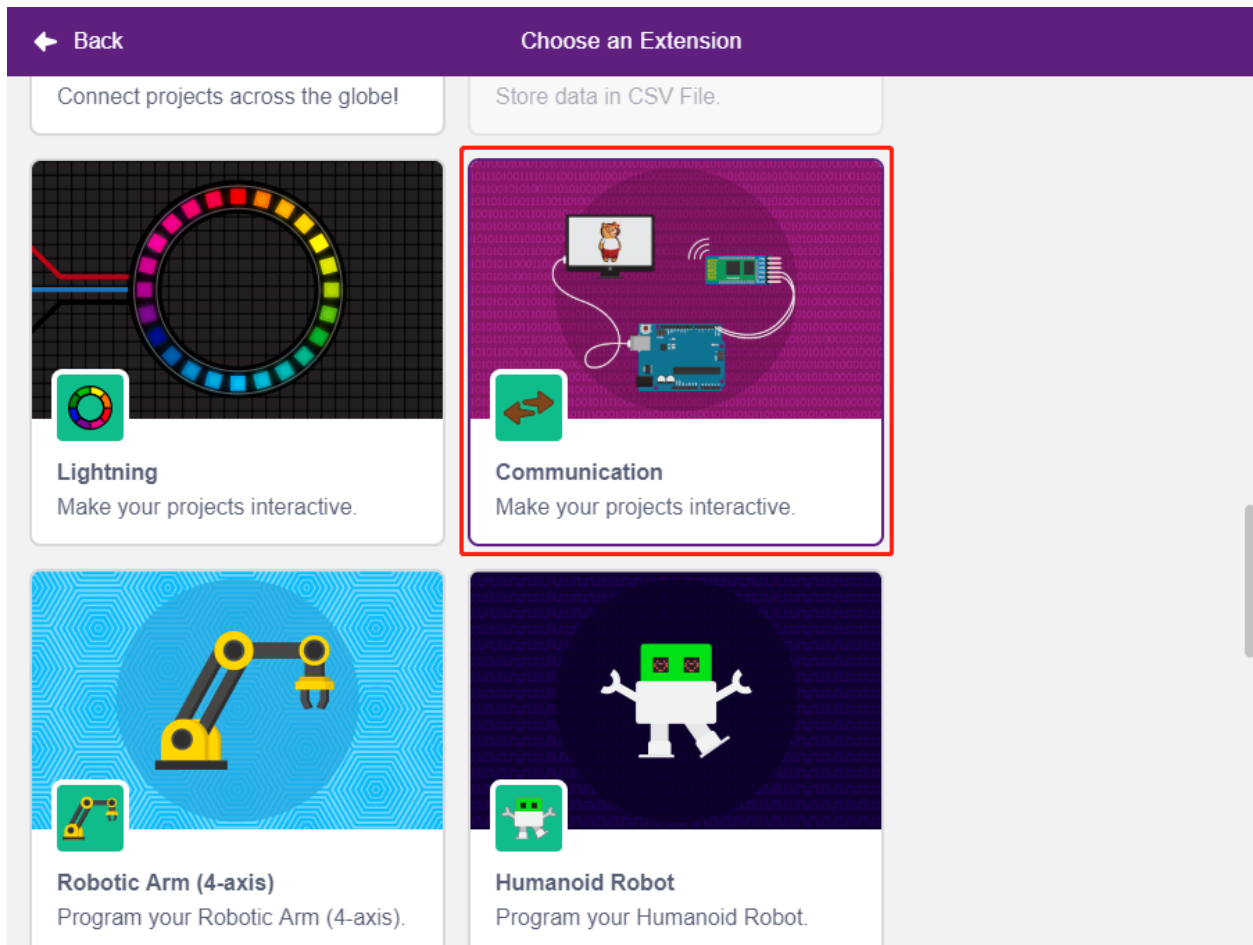
Construisez maintenant le circuit selon le schéma suivant.



## 8.12.4 Programmation

### 1. Ajout d'Extensions

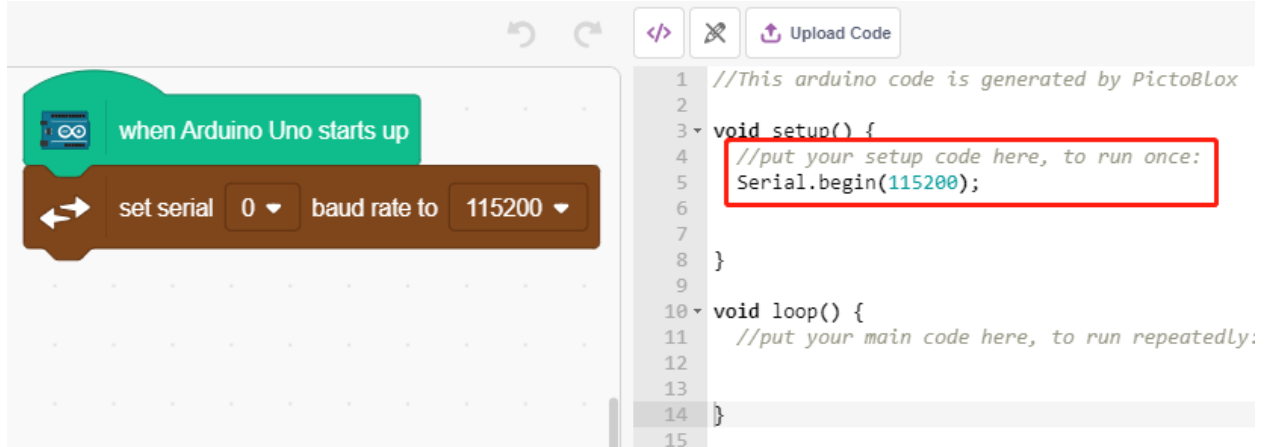
Passez en mode **Upload**, cliquez sur le bouton **Add Extension** dans le coin inférieur gauche, puis sélectionnez **Communication** pour l'ajouter, et elle apparaîtra à la fin de la zone de palette.



## 2. Initialisation de l'Arduino Uno et du Moniteur Série

En mode **Upload**, démarrez Arduino Uno puis réglez le débit en bauds du port série.

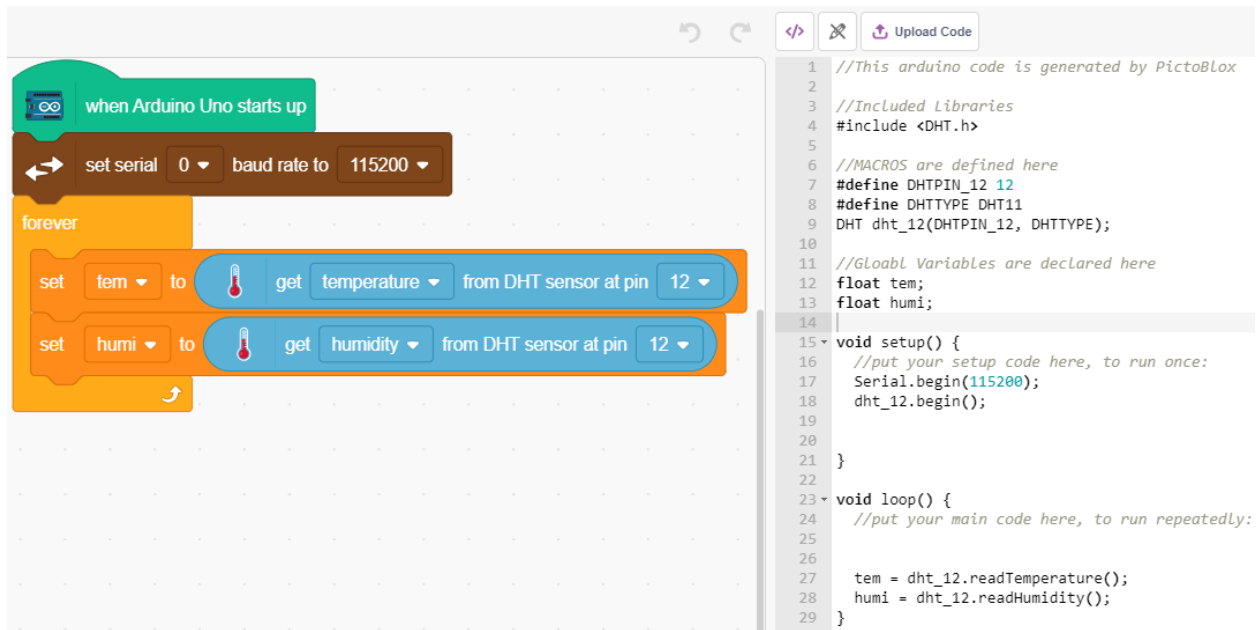
- [when Arduino Starts up] : En mode **Upload**, démarrez Arduino Uno.
- [set serial baud rate to] : De la palette **Communications**, utilisé pour régler le débit en bauds du port série 0, par défaut 115200. Si vous utilisez Mega2560, vous pouvez choisir de régler le débit en bauds des ports série 0 à 3.



## 3. Lire la température et l'humidité

Créez 2 variables **tem** et **hum** pour stocker respectivement la température et l'humidité, le code apparaîtra sur le côté

droit lorsque vous glissez et déposez le bloc.



```

1 //This arduino code is generated by PictoBlox
2
3 //Included Libraries
4 #include <DHT.h>
5
6 //MACROS are defined here
7 #define DHTPIN_12 12
8 #define DHTTYPE DHT11
9 DHT dht_12(DHTPIN_12, DHTTYPE);
10
11 //Gloabl Variables are declared here
12 float tem;
13 float humi;
14
15 void setup() {
16   //put your setup code here, to run once:
17   Serial.begin(115200);
18   dht_12.begin();
19 }
20
21
22 void loop() {
23   //put your main code here, to run repeatedly:
24
25   tem = dht_12.readTemperature();
26   humi = dht_12.readHumidity();
27 }
  
```

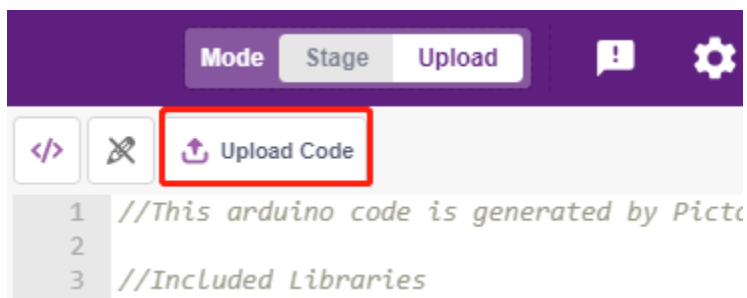
#### 4. Les imprimer sur le Moniteur Série

Écrivez la température et l'humidité lues sur le Moniteur Série. Pour éviter un transfert trop rapide causant un blocage de PictoBlox, utilisez le bloc [wait seconds] pour ajouter un intervalle de temps pour la prochaine impression.



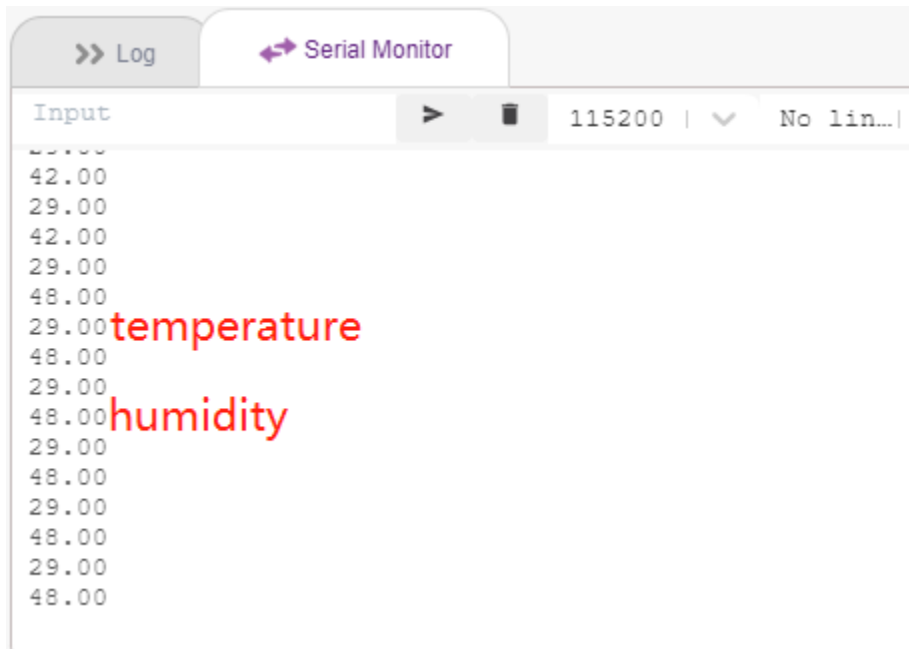
### 5. Téléversement du code

Contrairement au mode **Stage**, le code en mode **Upload** doit être téléversé sur la carte Arduino en utilisant le bouton **Upload Code** pour voir l'effet. Cela vous permet également de débrancher le câble USB tout en conservant le programme en cours d'exécution.



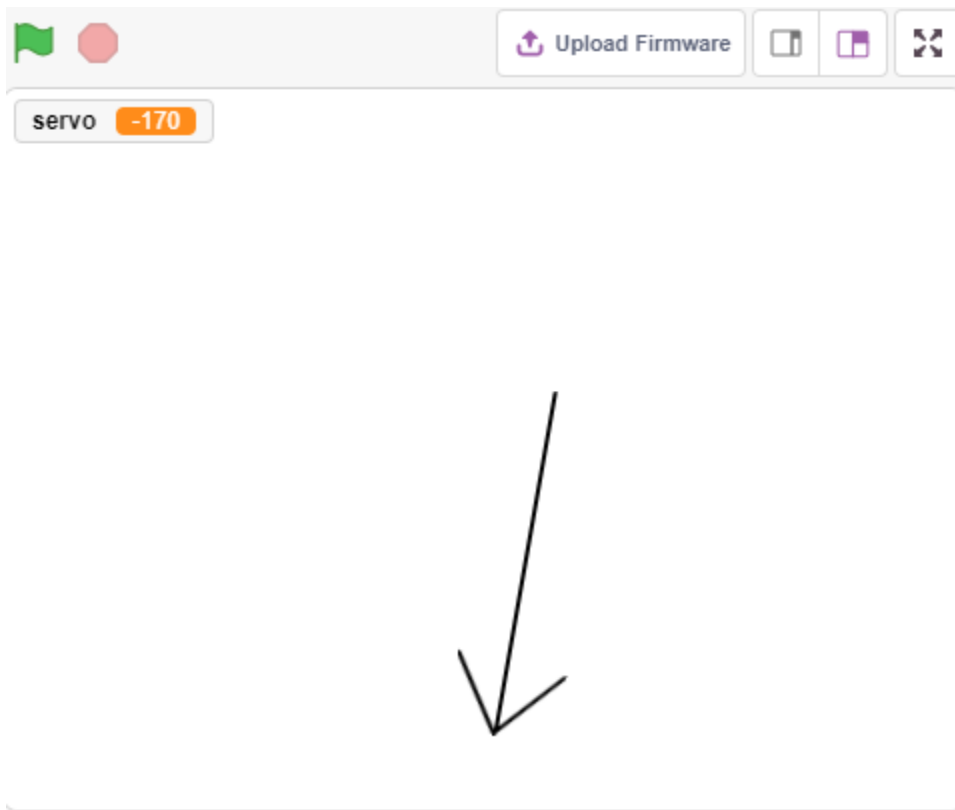
### 6. Ouvrir le moniteur série

Ouvrez maintenant le **Serial Monitor** pour voir la température et l'humidité.



## 8.13 2.10 Pendule

Dans ce projet, nous allons réaliser un pendule en forme de flèche, tandis que le servo-moteur suivra la rotation.



### 8.13.1 Vous Apprendrez

- Comment fonctionne un servo-moteur et sa plage d'angle
- Dessiner un sprite et placer le point central sur la queue.

### 8.13.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

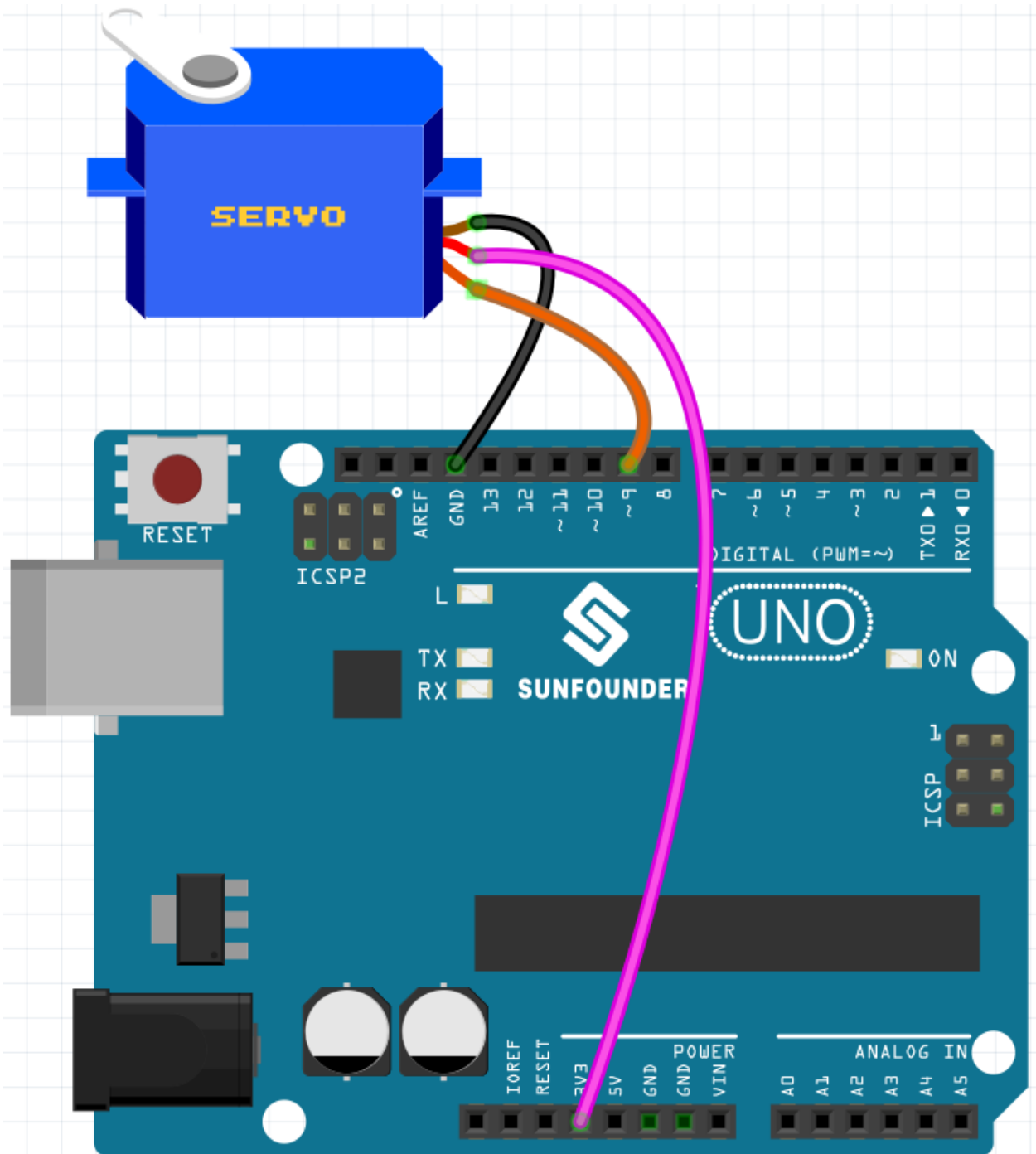
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Servomoteur</i>	

### 8.13.3 Construisez le Circuit

Un servo-moteur est un moteur engrené qui ne peut tourner que de 180 degrés. Il est contrôlé en envoyant des impulsions électriques depuis votre carte de circuit. Ces impulsions indiquent au servo-moteur la position dans laquelle il doit se déplacer.

Le servo-moteur a trois fils : le fil marron est le GND, le rouge est le VCC (à connecter à 3.3V), et l'orange est le fil de signal. La plage d'angle est de 0 à 180 degrés.

Construisez maintenant le circuit selon le schéma ci-dessous.

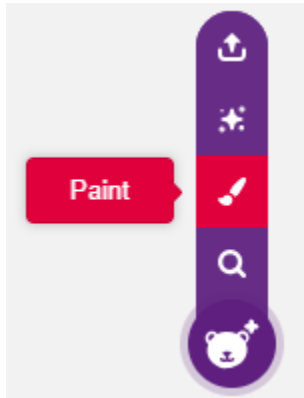




### 8.13.4 Programmation

#### 1. Peindre un sprite

Supprimez le sprite par défaut, sélectionnez le bouton Sprite et cliquez sur **Paint**, un sprite vierge **Sprite1** apparaîtra.

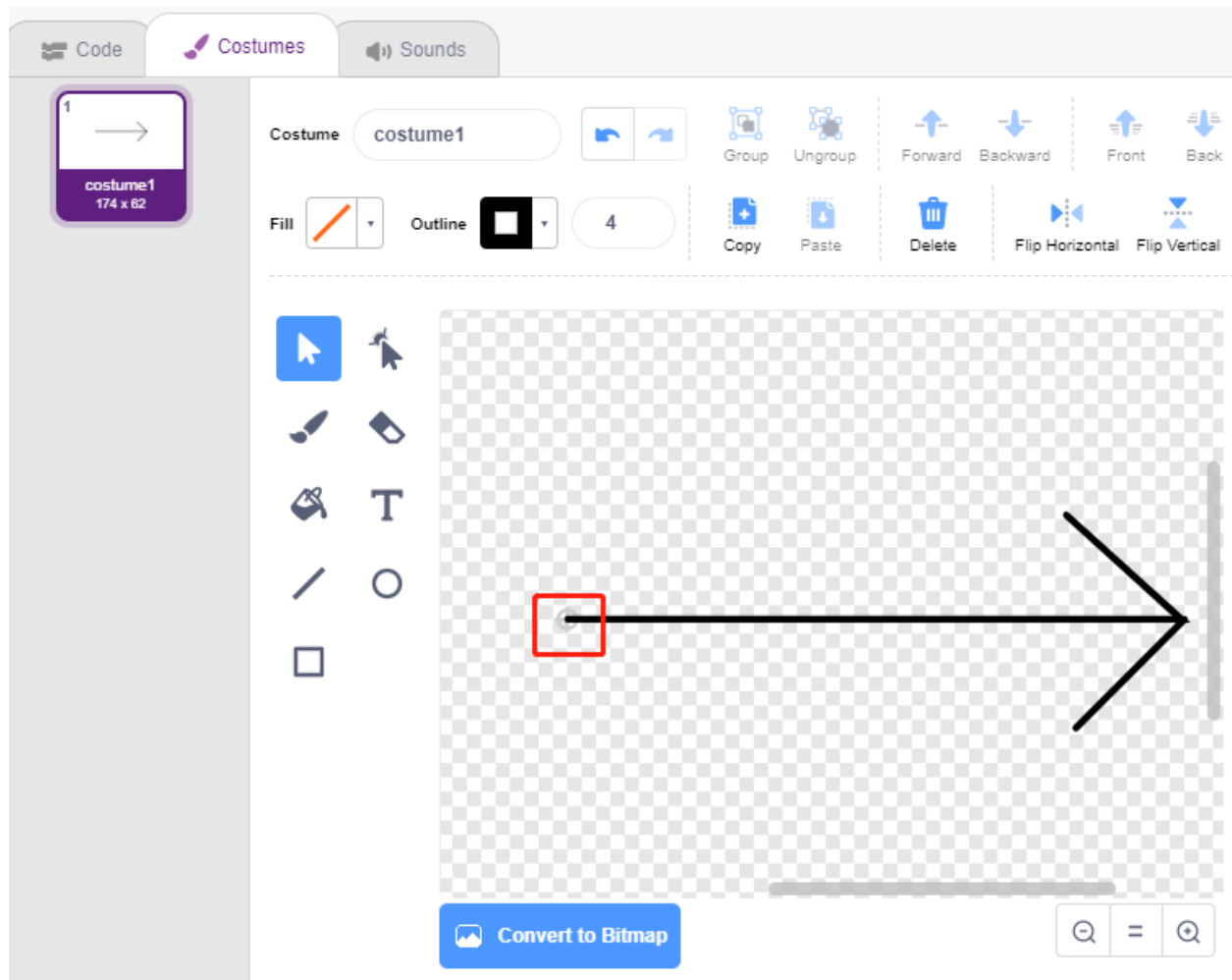


Sur la page **Costumes** ouverte, utilisez l'**Line tool** pour dessiner une flèche.

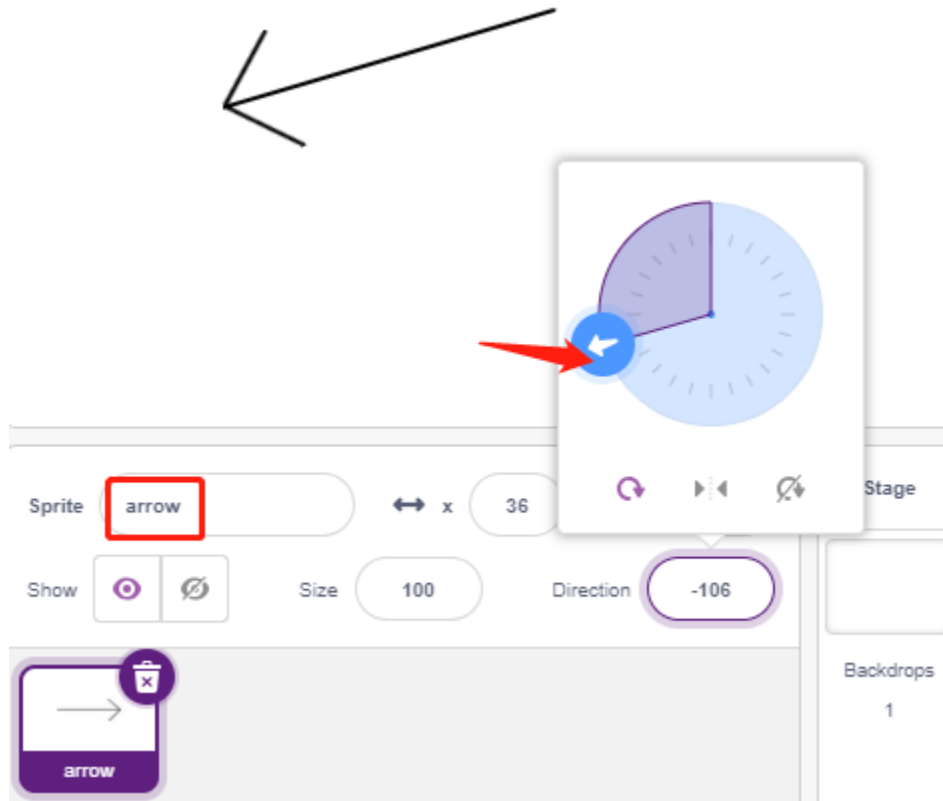
---

**Note :**

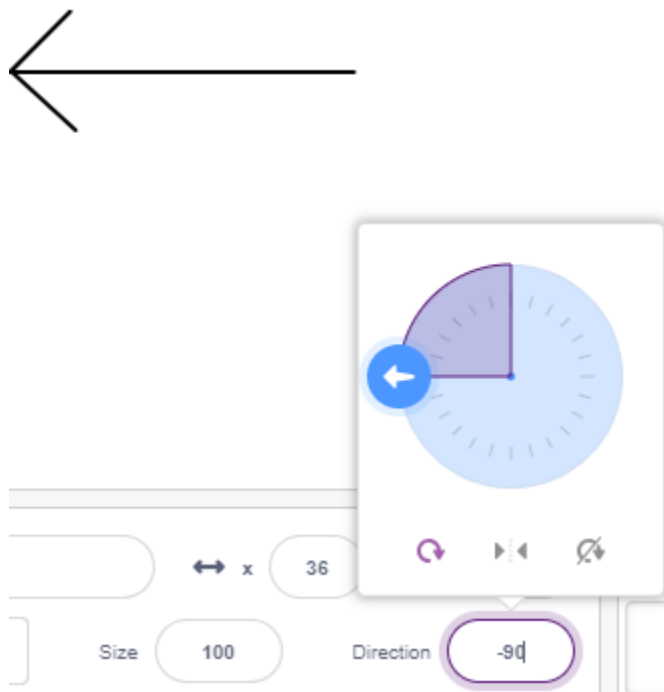
- Assurez-vous de commencer à dessiner la flèche du centre du canevas vers l'extérieur afin que la flèche tourne en cercle avec le point central comme origine.
  - Maintenez la touche Shift pour rendre l'angle de la ligne droit ou à 45 degrés.
-

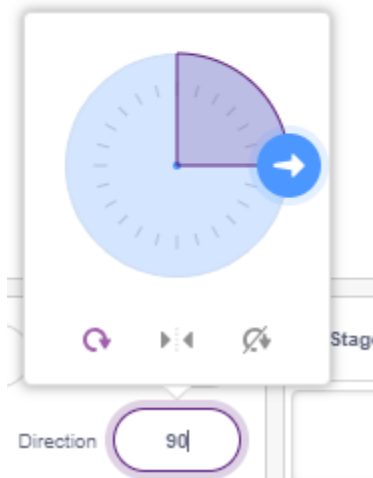
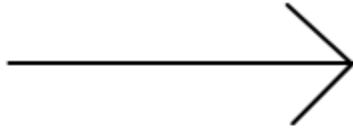


Après avoir dessiné, le sprite **arrow** s'affichera sur la scène, nommez-le **arrow**. Puis cliquez sur le nombre après **Direction**, un cadran circulaire apparaîtra, maintenant faites glisser cette flèche et voyez si le sprite **arrow** sur la scène tourne avec la queue comme origine.



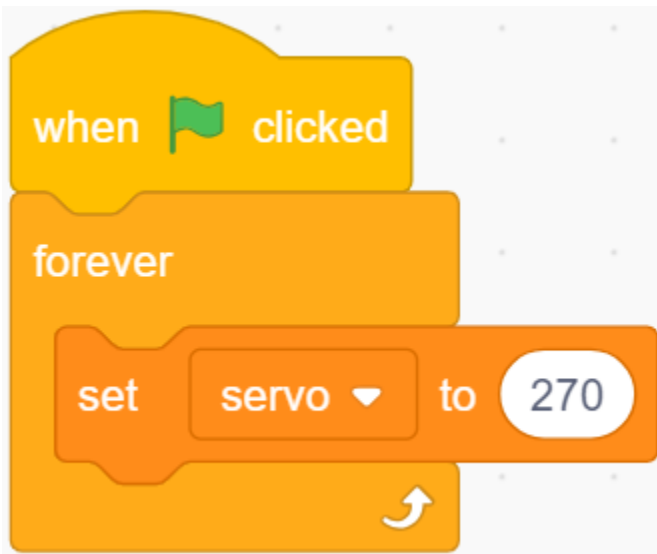
Pour faire osciller le sprite **arrow** de la gauche vers la droite, la plage d'angle est de -90 à -180, 180 à 90.





## 2. Création d'une variable.

Créez une variable appelée **servo**, qui stocke la valeur de l'angle et définit la valeur initiale à 270.



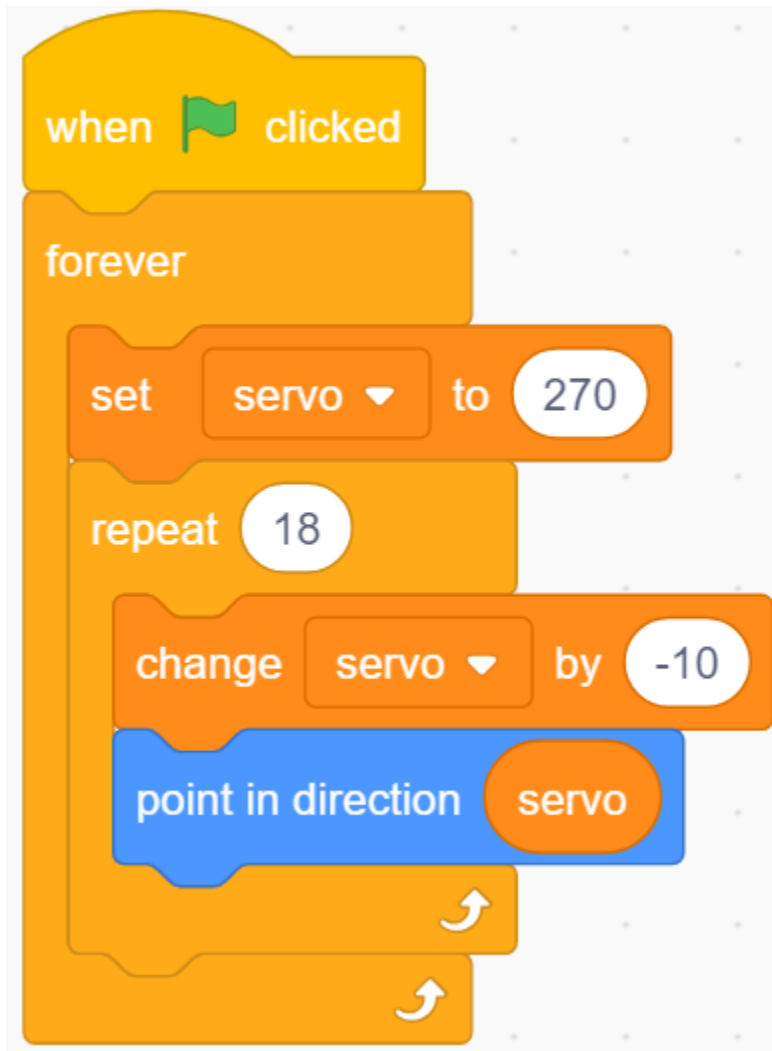
## 3. Oscillation de la gauche vers la droite

Faites maintenant osciller le sprite **arrow** de la position -90 degrés à gauche à la position 90 degrés à droite.

Avec le bloc [repeat], ajoutez -10 à la variable à chaque fois, et vous atteindrez 90 degrés en 18 passages. Ensuite, utilisez le bloc [point in block] pour faire tourner le sprite flèche vers ces angles.

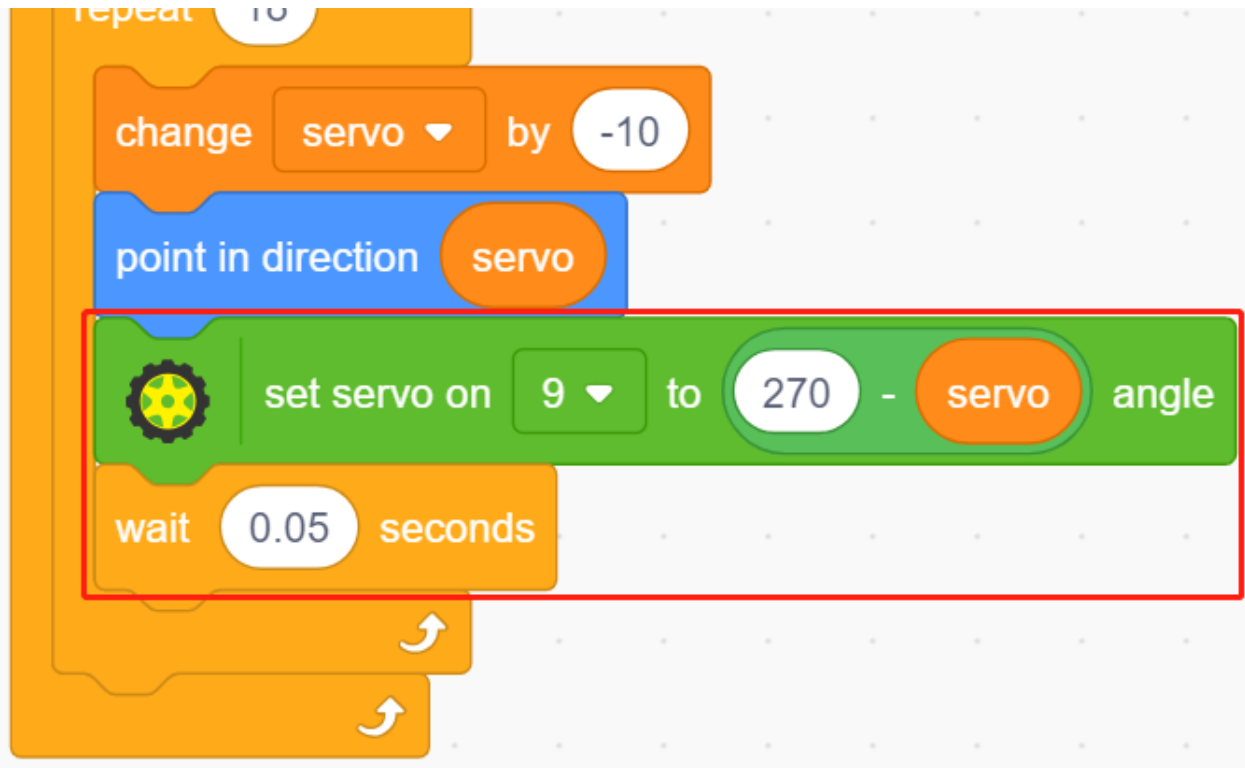
Comme l'angle de rotation du sprite est de -180 ~ 180, les angles en dehors de cette plage sont convertis par les conditions suivantes.

- Si angle > 180, alors angle -360.



#### 4. Faire tourner le Servo

Lorsque vous cliquez sur le drapeau vert, vous verrez la flèche tourner rapidement vers la droite puis revenir vers la gauche, utilisez donc un bloc [wait seconds] ici pour ralentir la rotation. Utilisez également le bloc [set servo on to angle] pour faire tourner le servo connecté à la carte Arduino vers un angle spécifique.



### 5. Oscillation de droite à gauche

De la même manière, faites tourner lentement le servo et le sprite **arrow** de la droite vers la gauche.

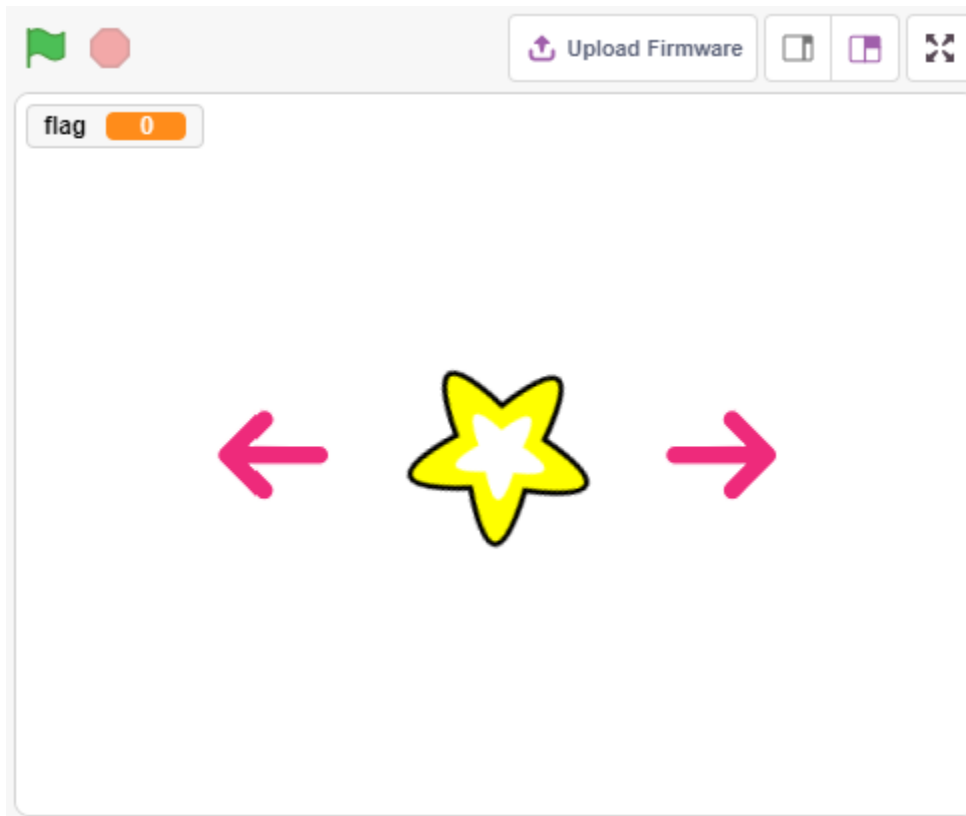
- Si  $\text{angle} > 180$ , alors  $\text{angle} - 360$ .



## 8.14 2.11 Ventilateur Rotatif

Dans ce projet, nous allons créer un sprite d'étoile tournante et un ventilateur.

Cliquer sur les sprites de flèche gauche et droite sur la scène contrôlera la rotation dans le sens horaire et antihoraire du moteur et du sprite étoile, cliquez sur le sprite étoile pour arrêter la rotation.



### 8.14.1 Vous Apprendrez

- Principe de fonctionnement d'un moteur
- Fonction de diffusion
- Arrêter d'autres scripts dans le bloc de sprite

### 8.14.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

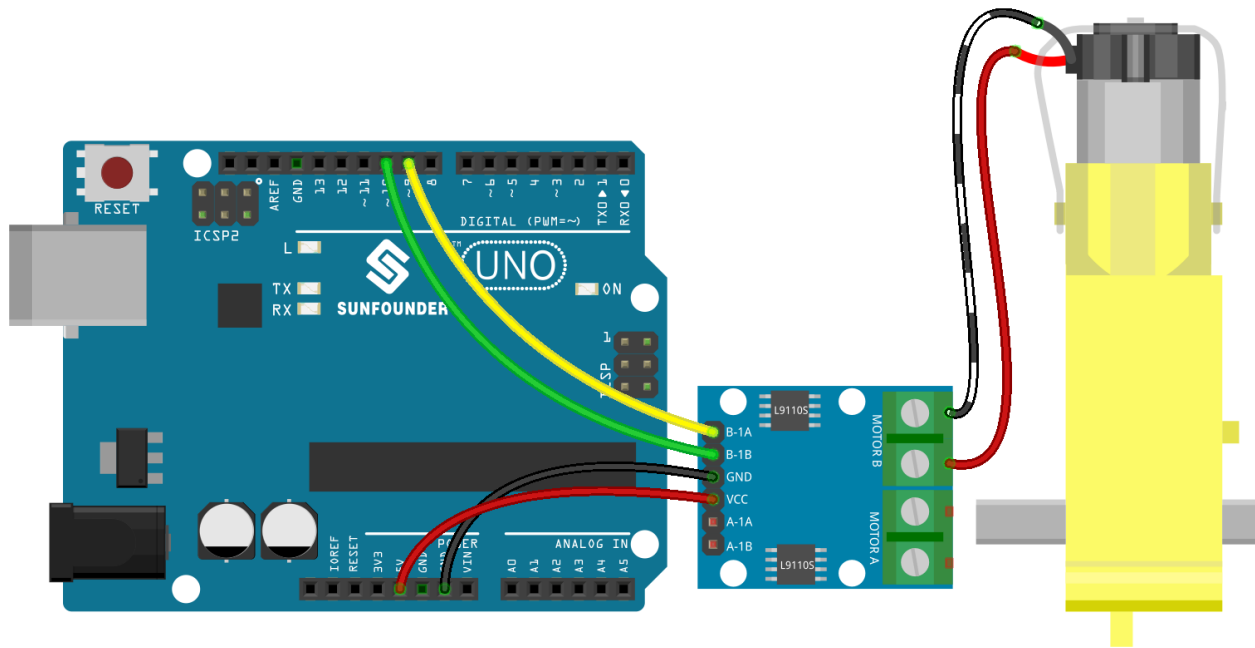
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Moteur TT</i>	-
<i>Module de Contrôle Moteur L9110</i>	-



### 8.14.3 Construisez le Circuit

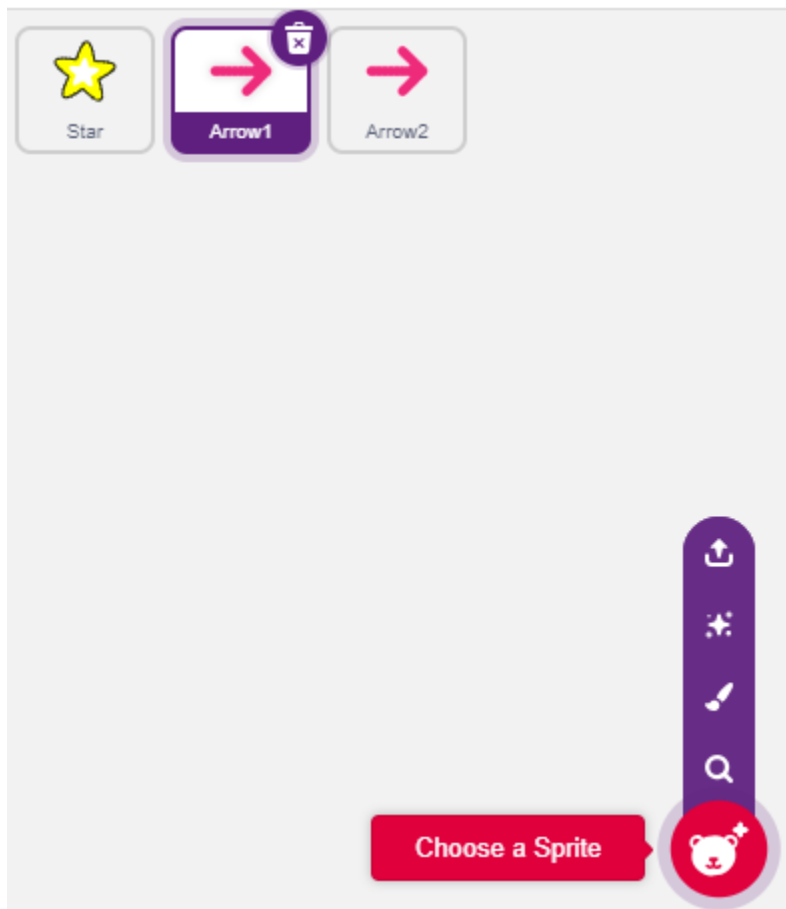


### 8.14.4 Programmation

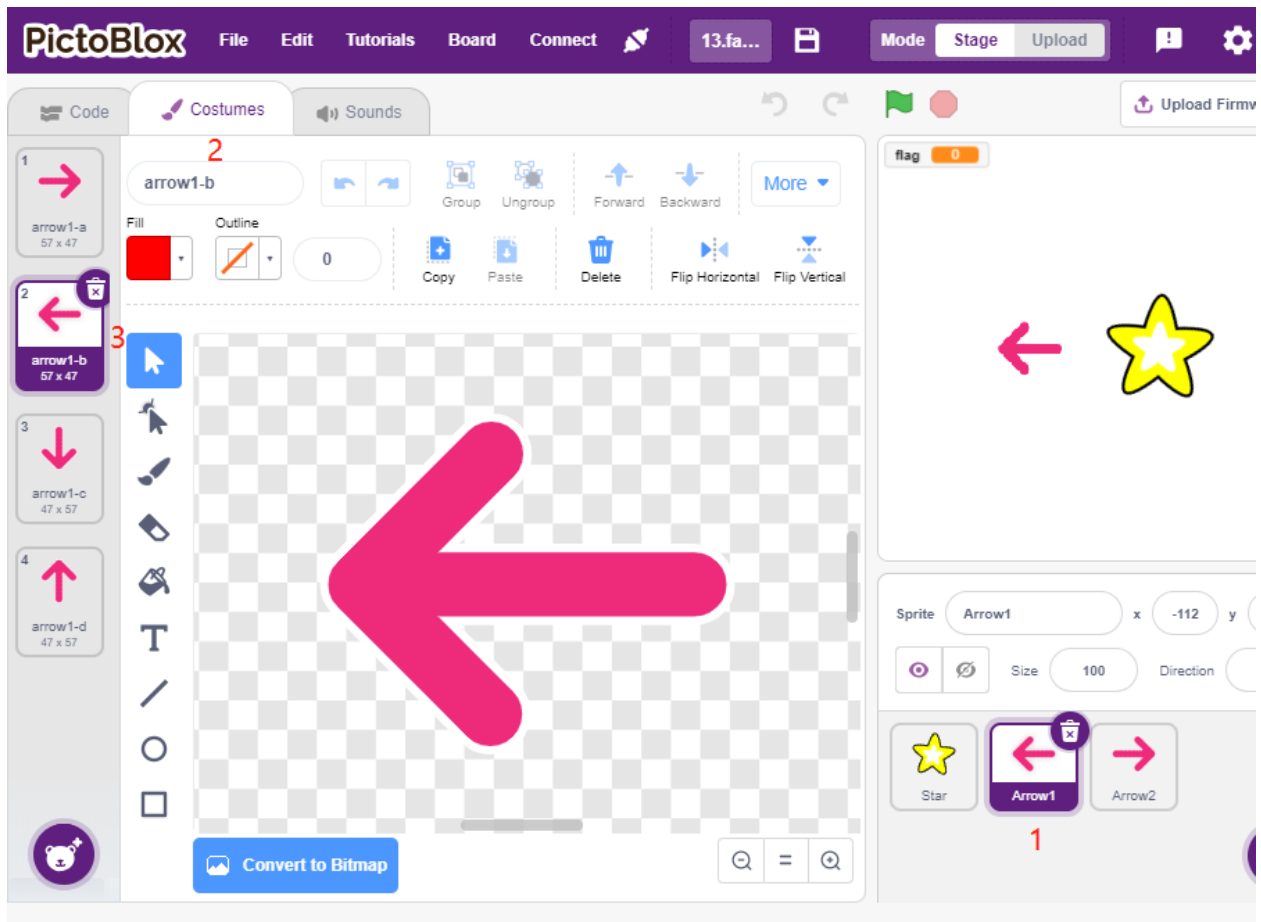
L'effet que nous voulons atteindre est d'utiliser 2 sprites flèche pour contrôler respectivement la rotation dans le sens horaire et antihoraire du moteur et du sprite étoile, cliquer sur le sprite étoile arrêtera la rotation du moteur.

#### 1. Ajouter des sprites

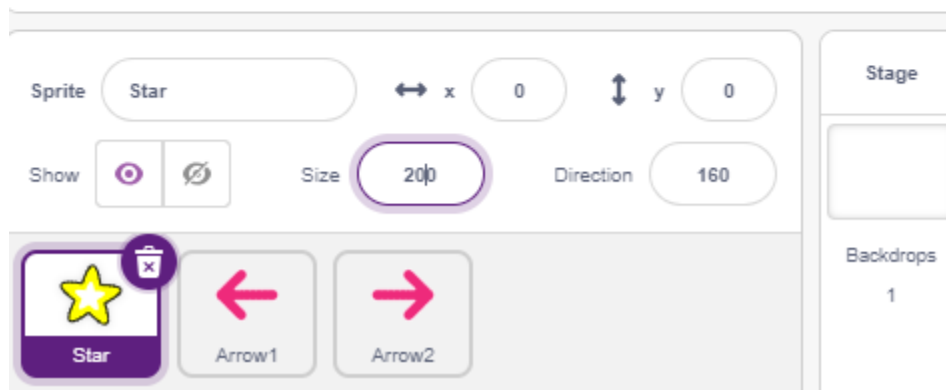
Supprimez le sprite par défaut, puis sélectionnez le sprite **Star** et le sprite **Arrow1**, et copiez **Arrow1** une fois.



Dans l'option **Costumes**, changez le sprite **Arrow1** pour un costume de direction différente.



Ajustez la taille et la position du sprite de manière appropriée.

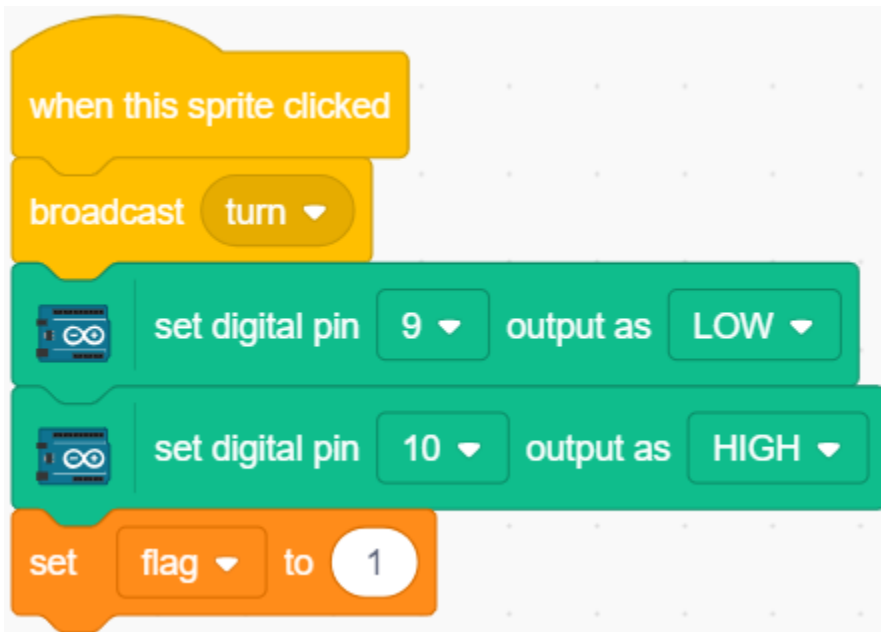


## 2. Sprite de flèche gauche

Quand ce sprite est cliqué, il diffuse un message - tourner, puis définit la broche numérique 9 à bas et la broche 10 à haut, et définit la variable **flag** à 1. Si vous cliquez sur le sprite de flèche gauche, vous constaterez que le moteur tourne dans le sens antihoraire, si votre tour est dans le sens horaire, alors vous échangez les positions des broches 9 et 10.

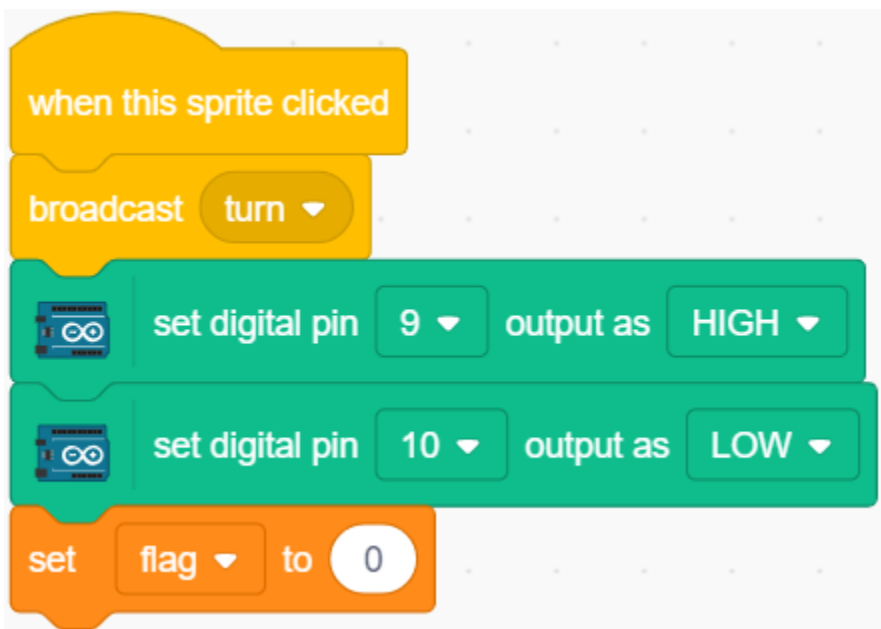
Il y a 2 points à noter ici.

- **[broadcast]** : de la palette **Events**, utilisé pour diffuser un message aux autres sprites, quand les autres sprites reçoivent ce message, ils exécuteront un événement spécifique. Par exemple, ici c'est **turn**, quand le sprite **star** reçoit ce message, il exécute le script de rotation.
- variable **flag** : La direction de rotation du sprite étoile est déterminée par la valeur de flag. Donc, quand vous créez la variable **flag**, vous devez la rendre applicable à tous les sprites.



### 3. Sprite de flèche droite

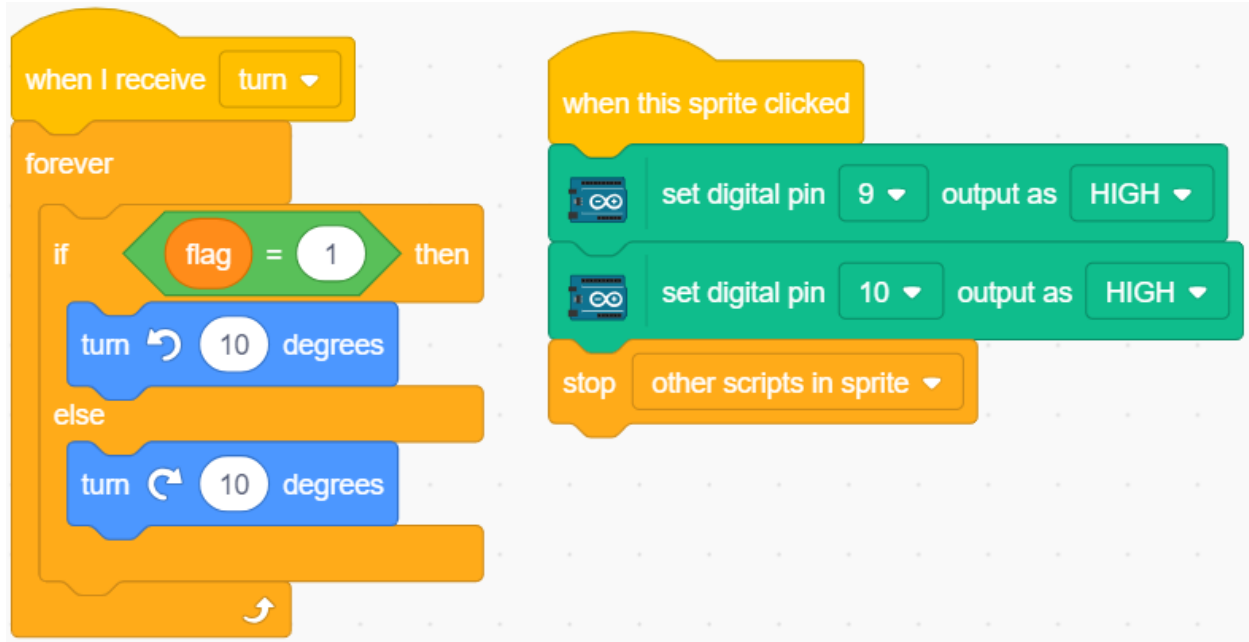
Quand ce sprite est cliqué, diffusez un message tourner, puis réglez la broche numérique 9 à haut et la broche 10 à bas pour faire tourner le moteur dans le sens horaire et définissez la variable **flag** à 0.



### 4. Sprite étoile

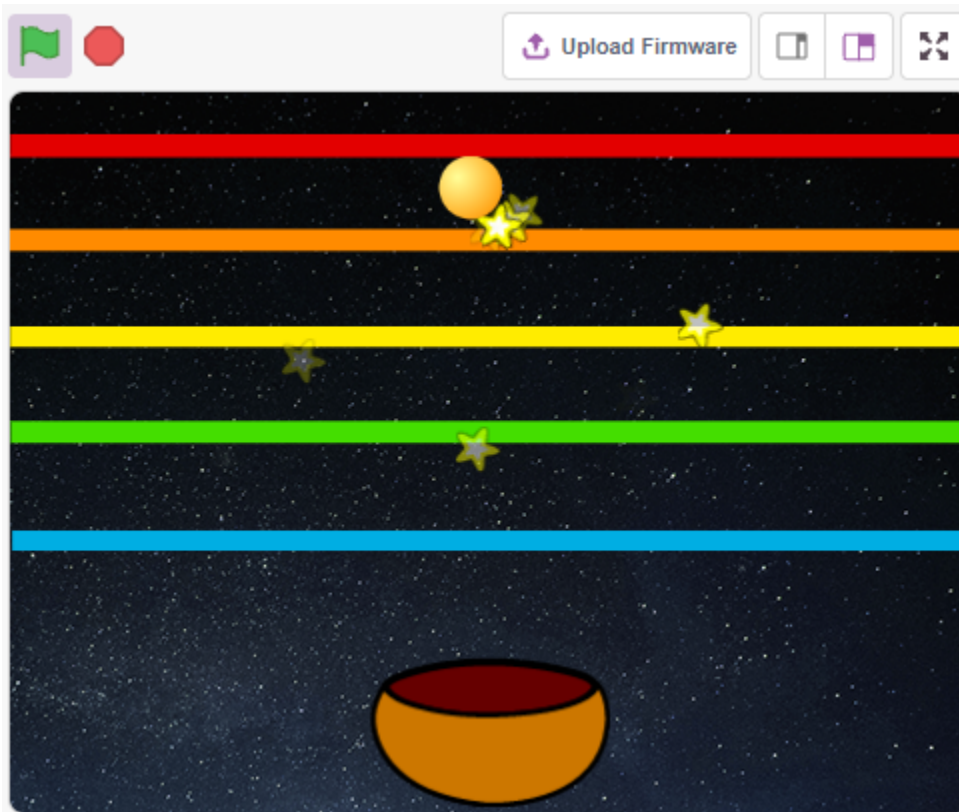
Il y a 2 événements inclus ici.

- Quand le sprite **star** reçoit le message diffusé tourner, il détermine la valeur de flag ; si flag est 1, il tourne de 10 degrés à gauche, sinon il inverse. Comme il est dans [FOREVER], il continuera à tourner.
- Quand ce sprite est cliqué, réglez les deux broches du moteur à haut pour le faire arrêter de tourner et arrêtez les autres scripts dans ce sprite.



## 8.15 2.12 Balle Sensible à la Lumière

Dans ce projet, nous utilisons une photorésistance pour faire voler la balle sur la scène vers le haut. Placez votre main au-dessus de la photorésistance pour contrôler l'intensité lumineuse qu'elle reçoit. Plus votre main est proche de la photorésistance, plus sa valeur est petite et plus la balle vole haut sur la scène, sinon elle tombera. Lorsque la balle touche la corde, elle produit un joli son ainsi qu'une lumière d'étoiles scintillantes.



### 8.15.1 Vous Apprendrez

- Remplir le sprite avec des couleurs
- Toucher entre les sprites

### 8.15.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Photorésistance</i>	

### 8.15.3 Construisez le Circuit

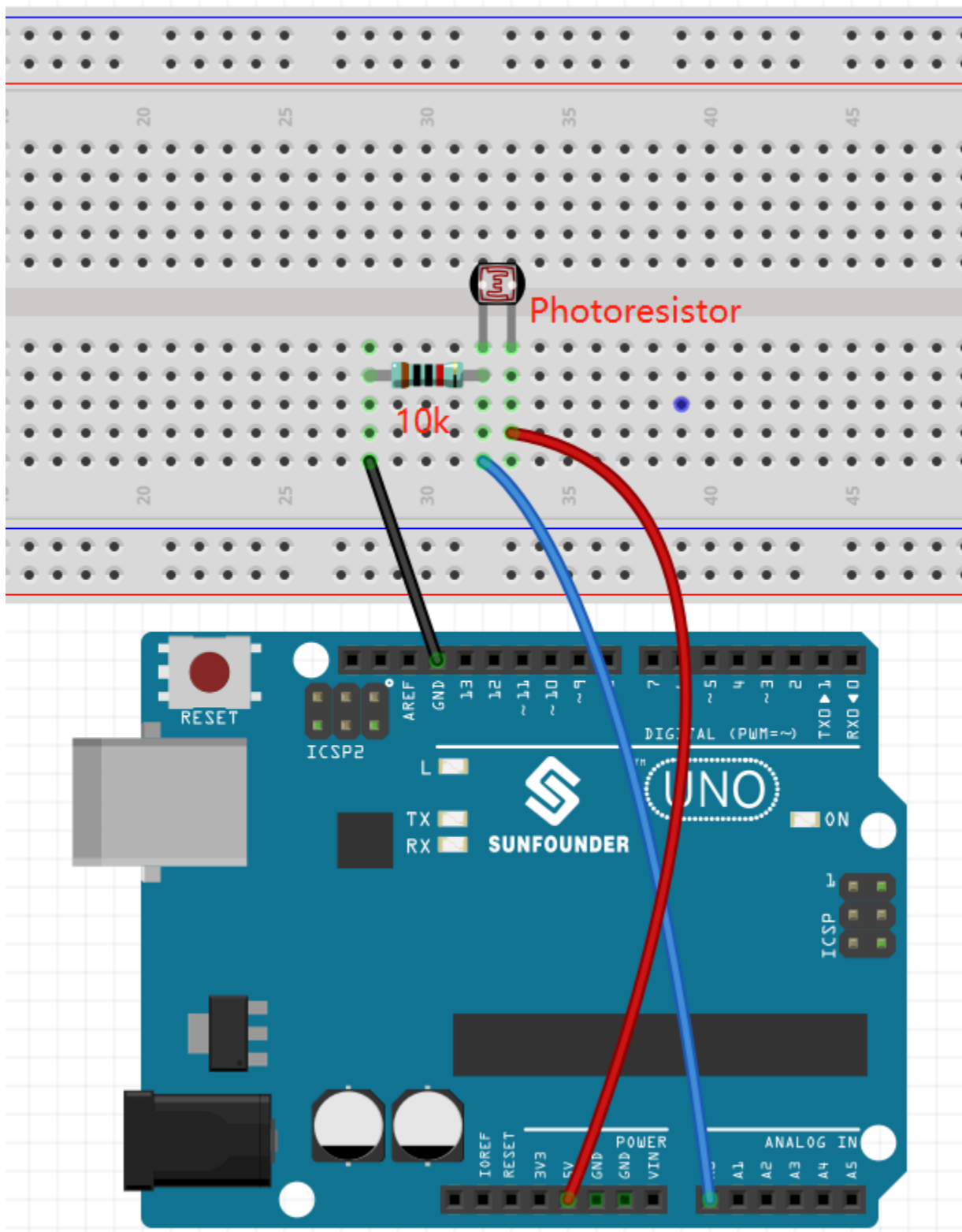
Une photorésistance ou cellule photoélectrique est une résistance variable contrôlée par la lumière. La résistance d'une photorésistance diminue avec l'augmentation de l'intensité lumineuse incidente.

Construisez le circuit selon le schéma suivant.

Connectez une extrémité de la photorésistance à 5V, l'autre à A0, et connectez une résistance de 10K en série avec GND à cette extrémité.

Ainsi, lorsque l'intensité lumineuse augmente, la résistance de la photorésistance diminue, la division de tension de la résistance de 10K augmente, et la valeur obtenue par A0 devient plus grande.



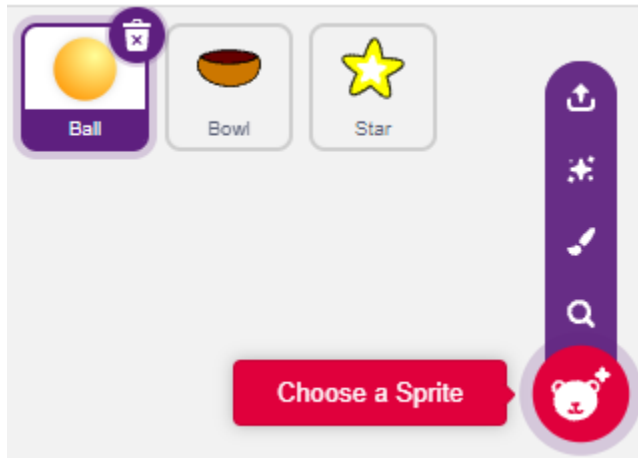


### 8.15.4 Programmation

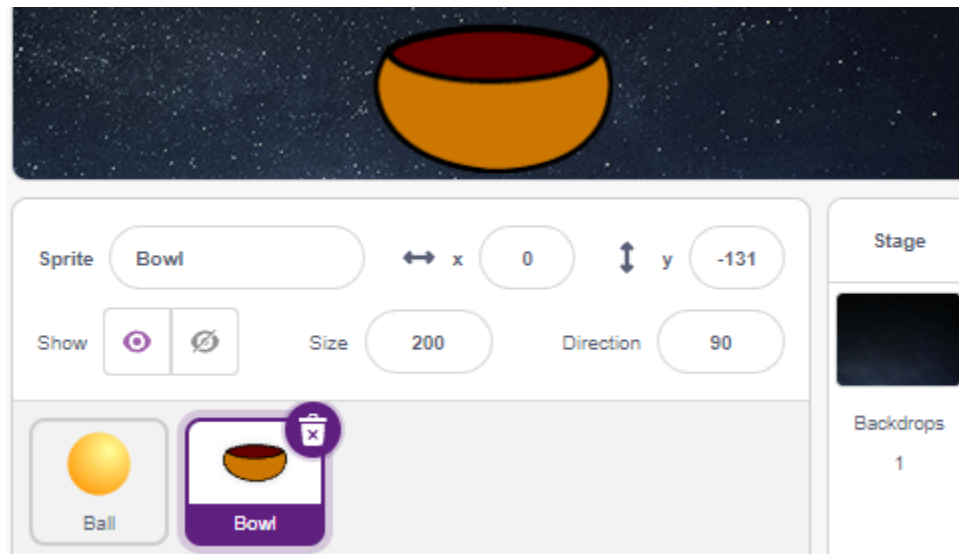
L'effet que nous voulons obtenir est que plus votre main est proche de la photorésistance, plus le sprite de la balle sur la scène continue de monter, sinon il tombera sur le sprite bol. S'il touche le sprite Ligne en montant ou en tombant, il produira un son musical et émettra des sprites d'étoiles dans toutes les directions.

#### 1. Sélectionner le sprite et le décor

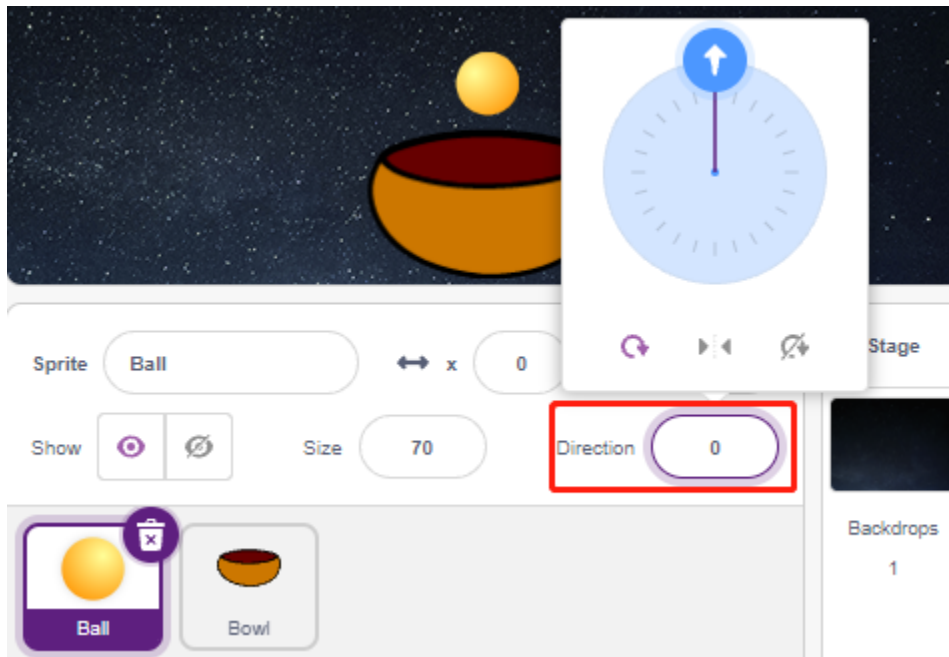
Supprimez le sprite par défaut, sélectionnez les sprites **Ball**, **Bowl** et **Star**.



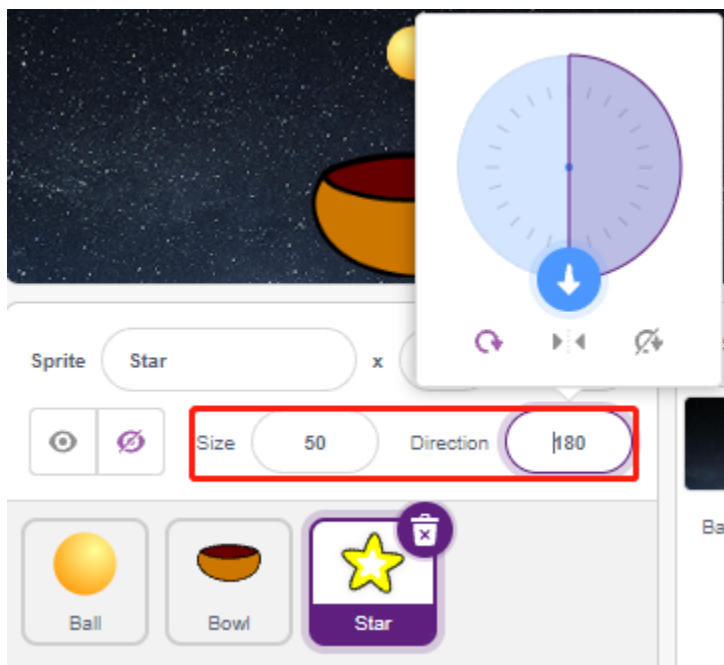
Déplacez le sprite **Bowl** au centre bas de la scène et agrandissez sa taille.



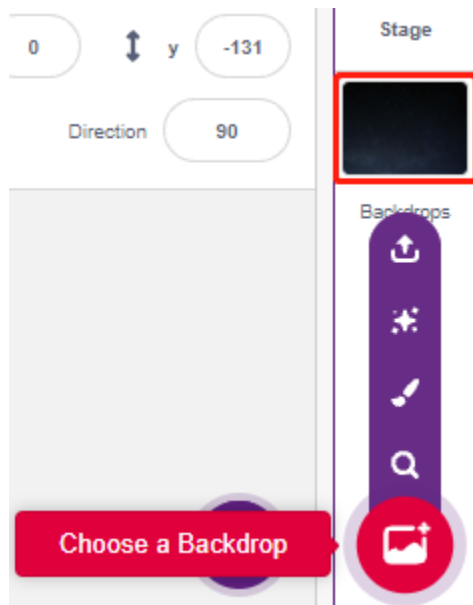
Comme nous devons le déplacer vers le haut, réglez la direction du sprite **Ball** sur 0.



Réglez la taille et la direction du sprite **Star** sur 180 car nous avons besoin qu'il tombe, ou vous pouvez le changer à un autre angle.

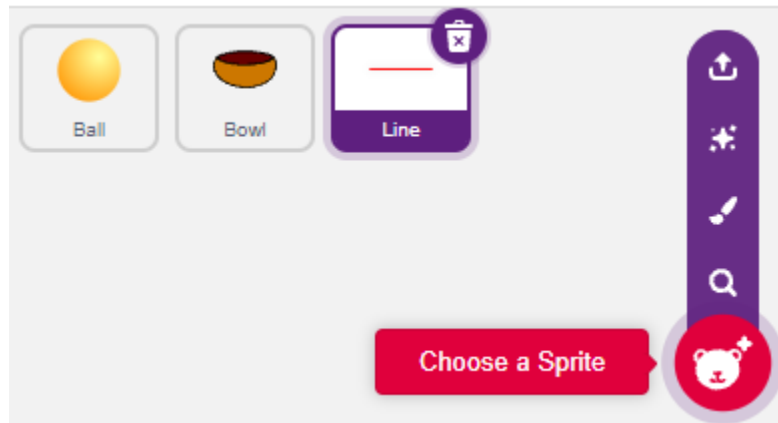


Ajoutez maintenant le décor **Stars**.

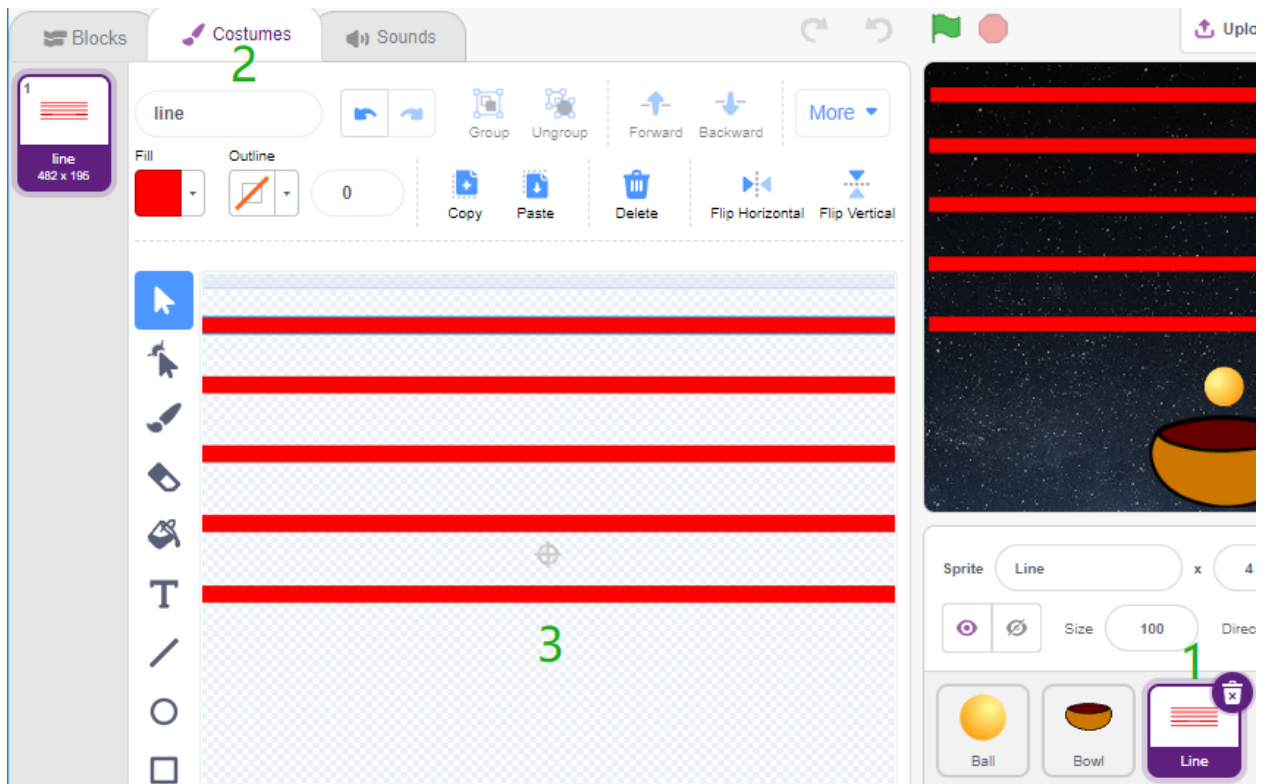


## 2. Dessiner un sprite Ligne

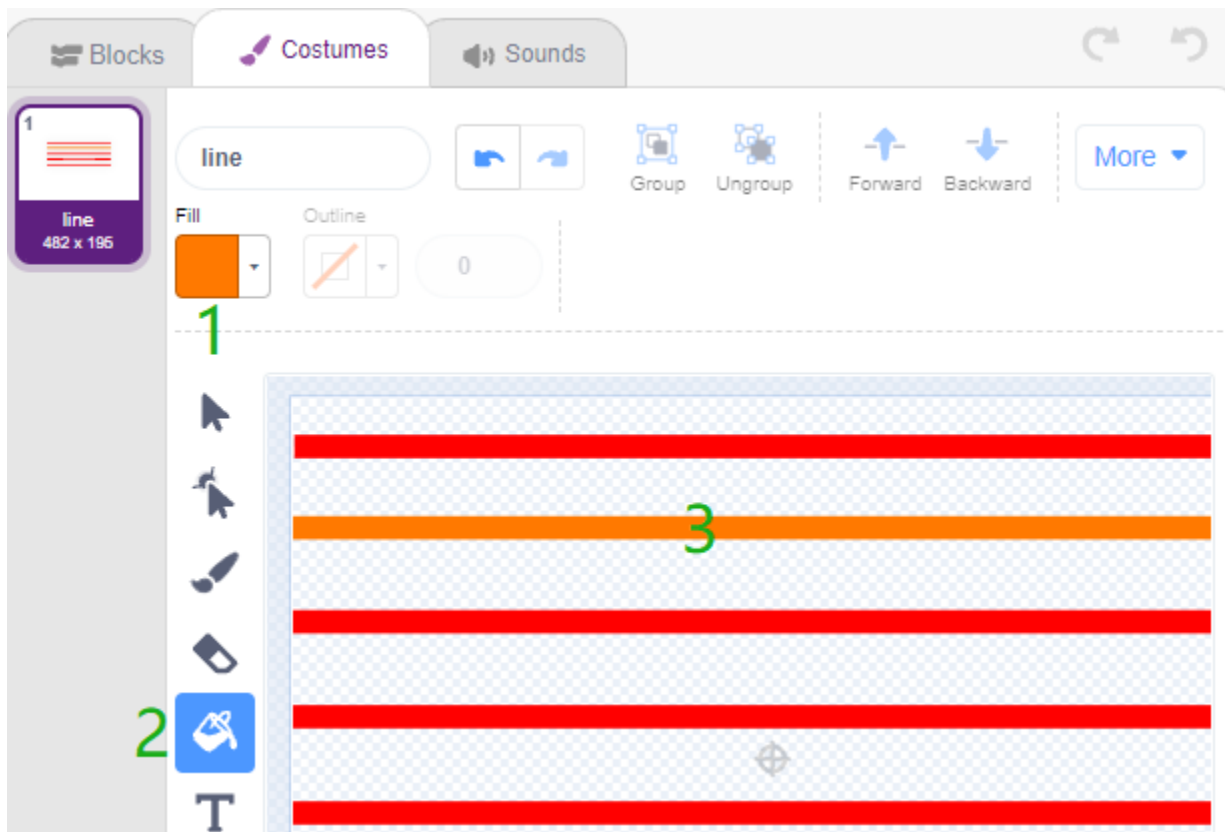
Ajoutez un sprite Ligne.



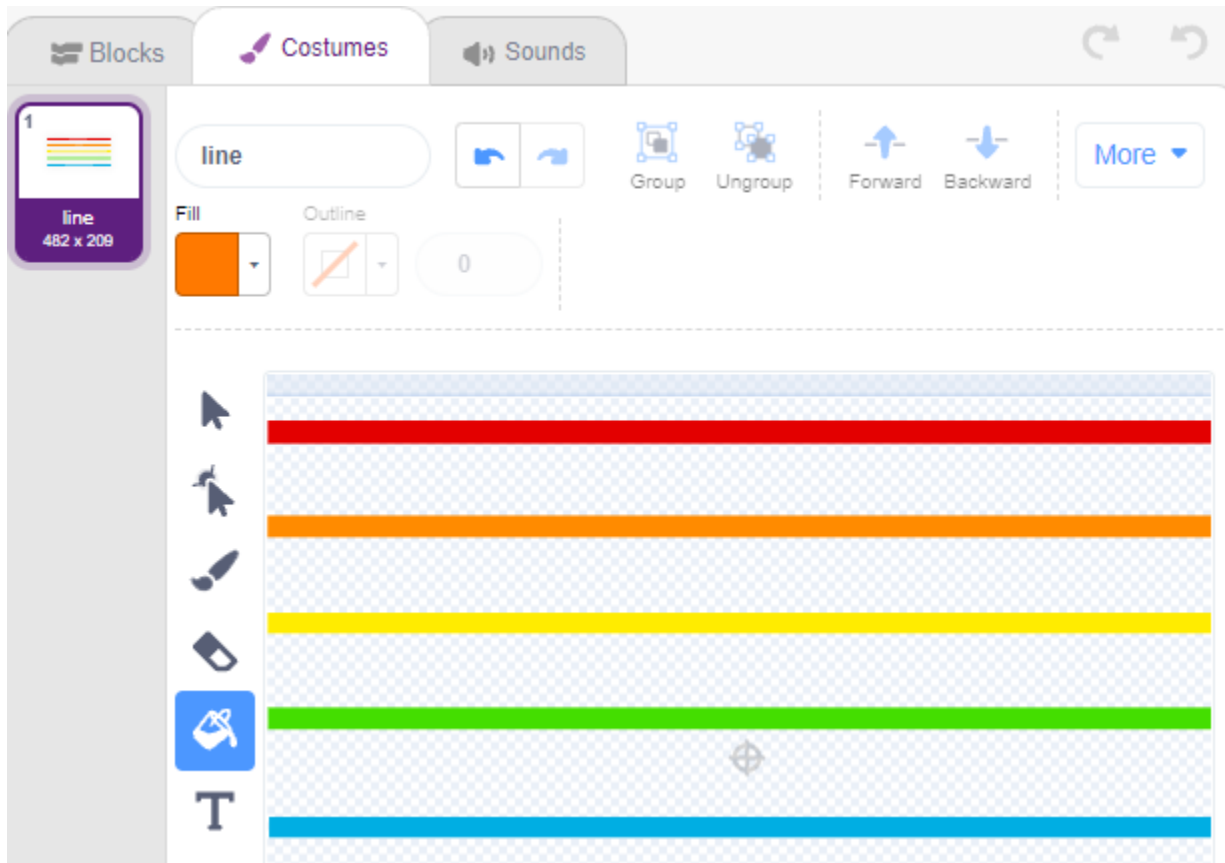
Allez sur la page **Costumes** du sprite **Line**, réduisez légèrement la largeur de la ligne rouge sur le canevas, puis copiez-la 5 fois et alignez les lignes.



Remplissez maintenant les lignes avec différentes couleurs. Choisissez d'abord une couleur que vous aimez, puis cliquez sur l'outil **Fill** et déplacez la souris sur la ligne pour la remplir de couleur.



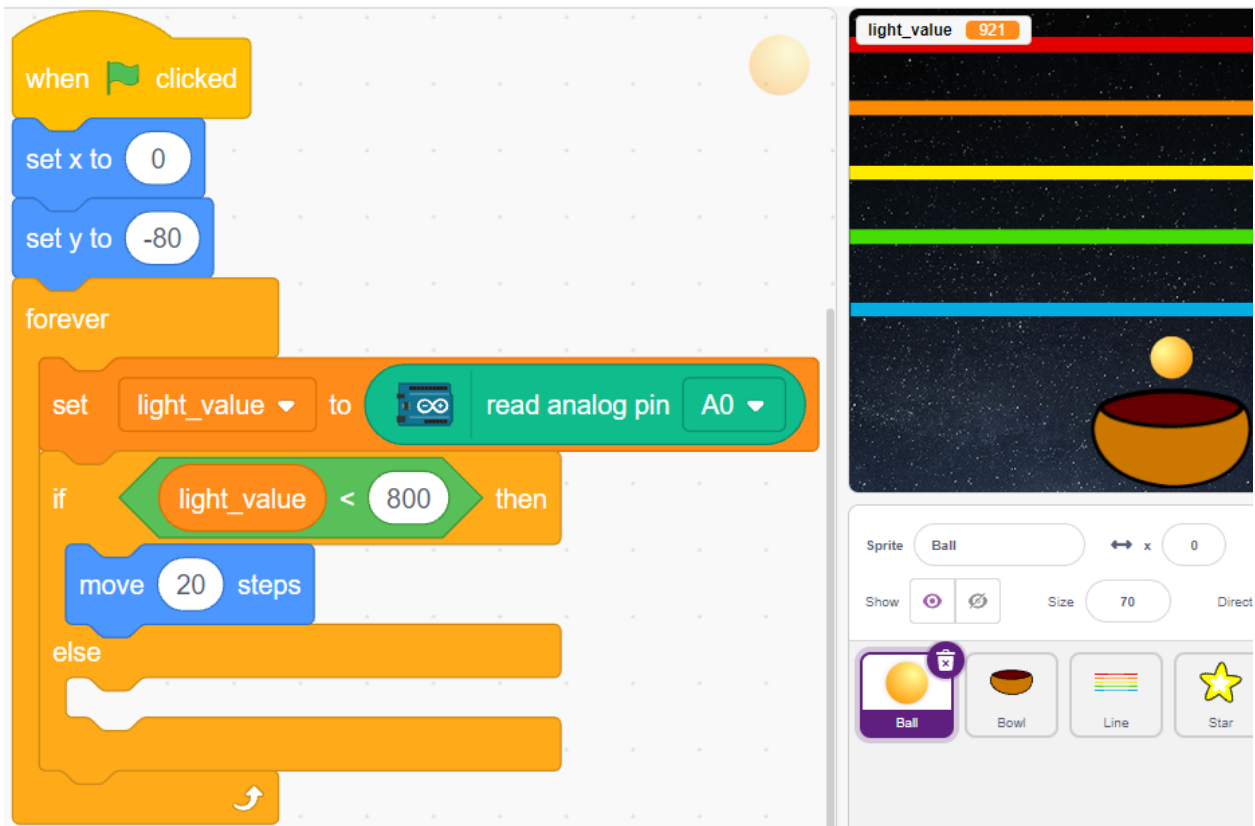
Suivez la même méthode pour changer la couleur des autres lignes.



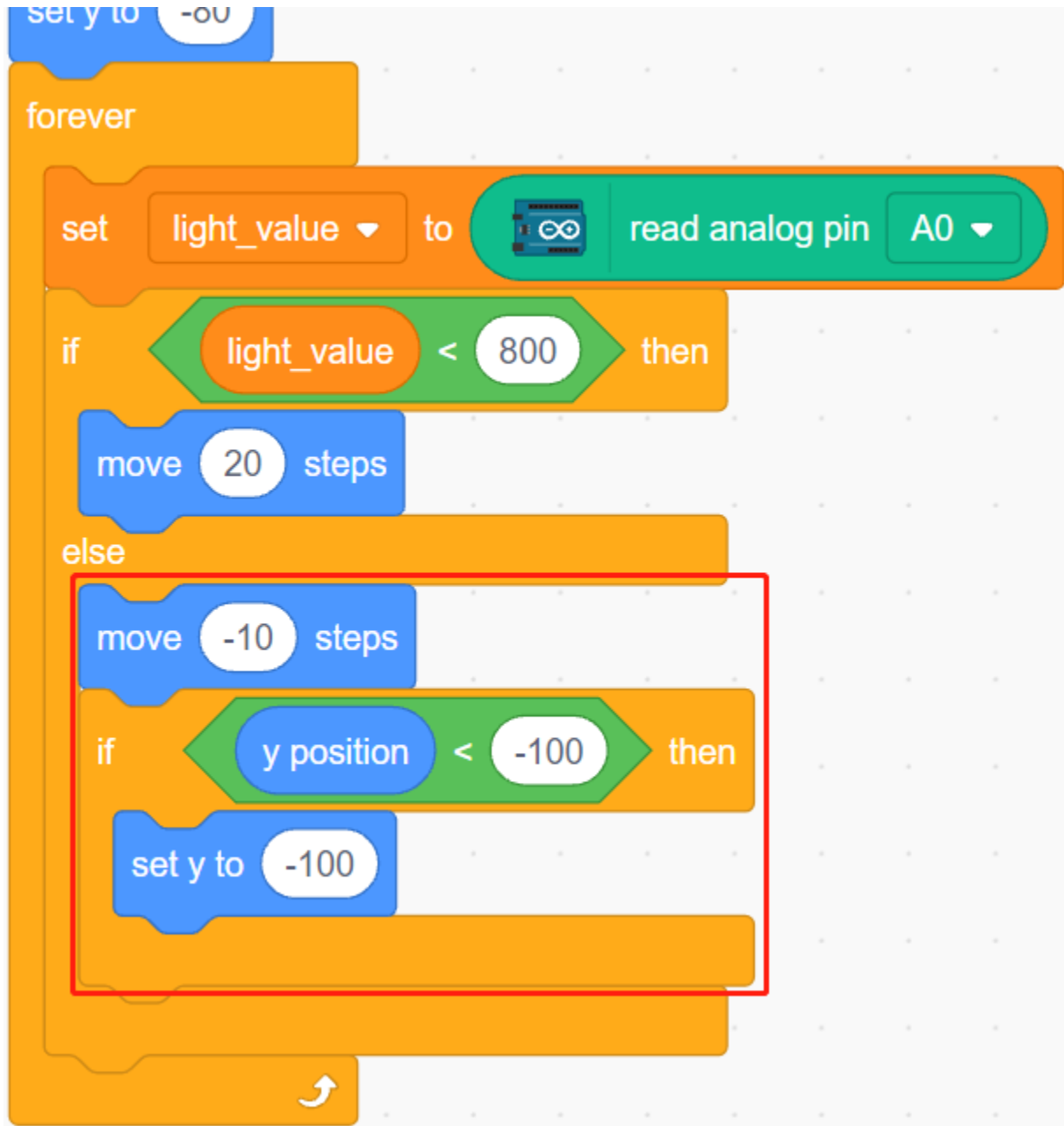
### 3. Scripter le sprite Balle

Définissez la position initiale du sprite **Ball**, puis lorsque la valeur de lumière est inférieure à 800 (cela peut être toute autre valeur, selon votre environnement actuel), laissez la Balle monter.

Vous pouvez faire apparaître la variable `light_value` sur la scène pour observer le changement d'intensité lumineuse à tout moment.

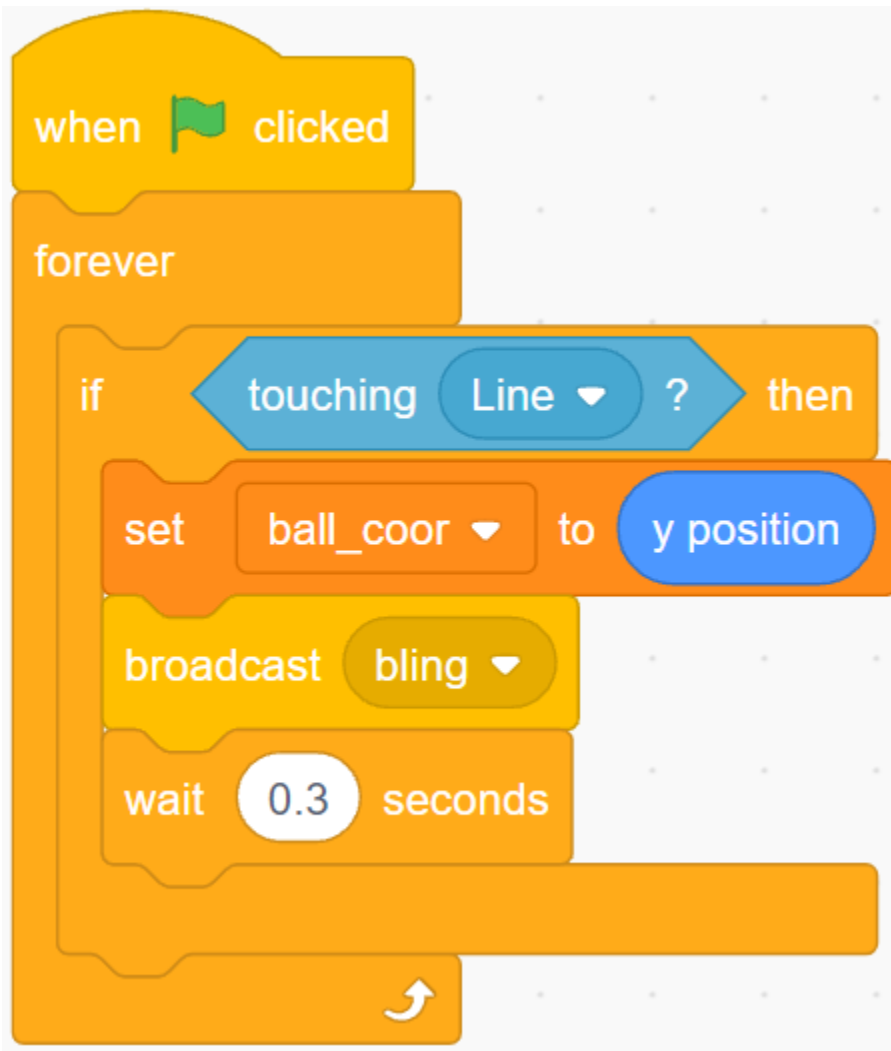


Sinon, le sprite **Ball** tombera et limitera sa coordonnée Y à un minimum de -100. Cela peut être modifié pour qu'il semble tomber sur le sprite **Bowl**.



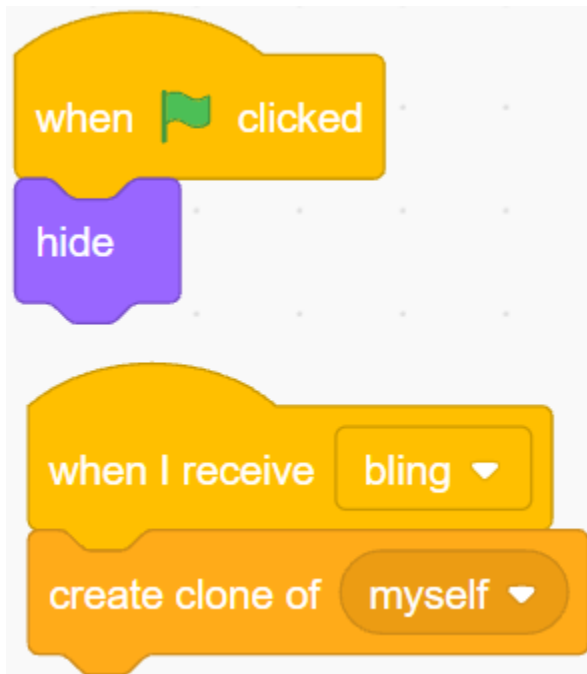
Lorsque le sprite **Line** est touché, la coordonnée Y actuelle est enregistrée dans la variable **ball\_coor** et un message **Bling** est diffusé.



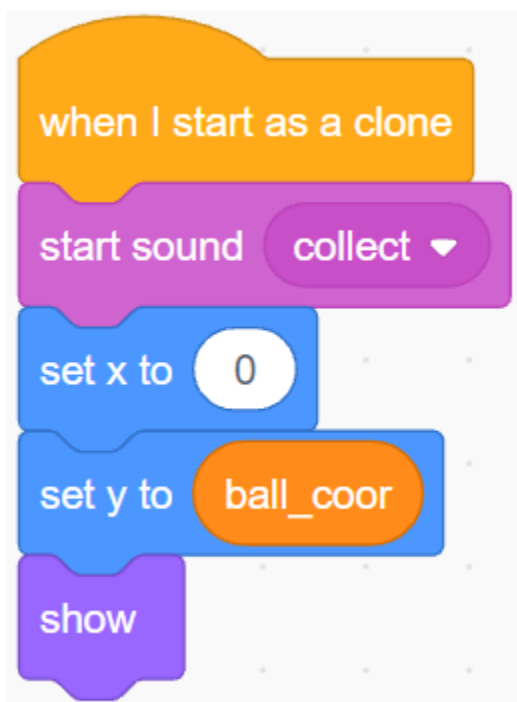


#### 4. Scripter le sprite Étoile

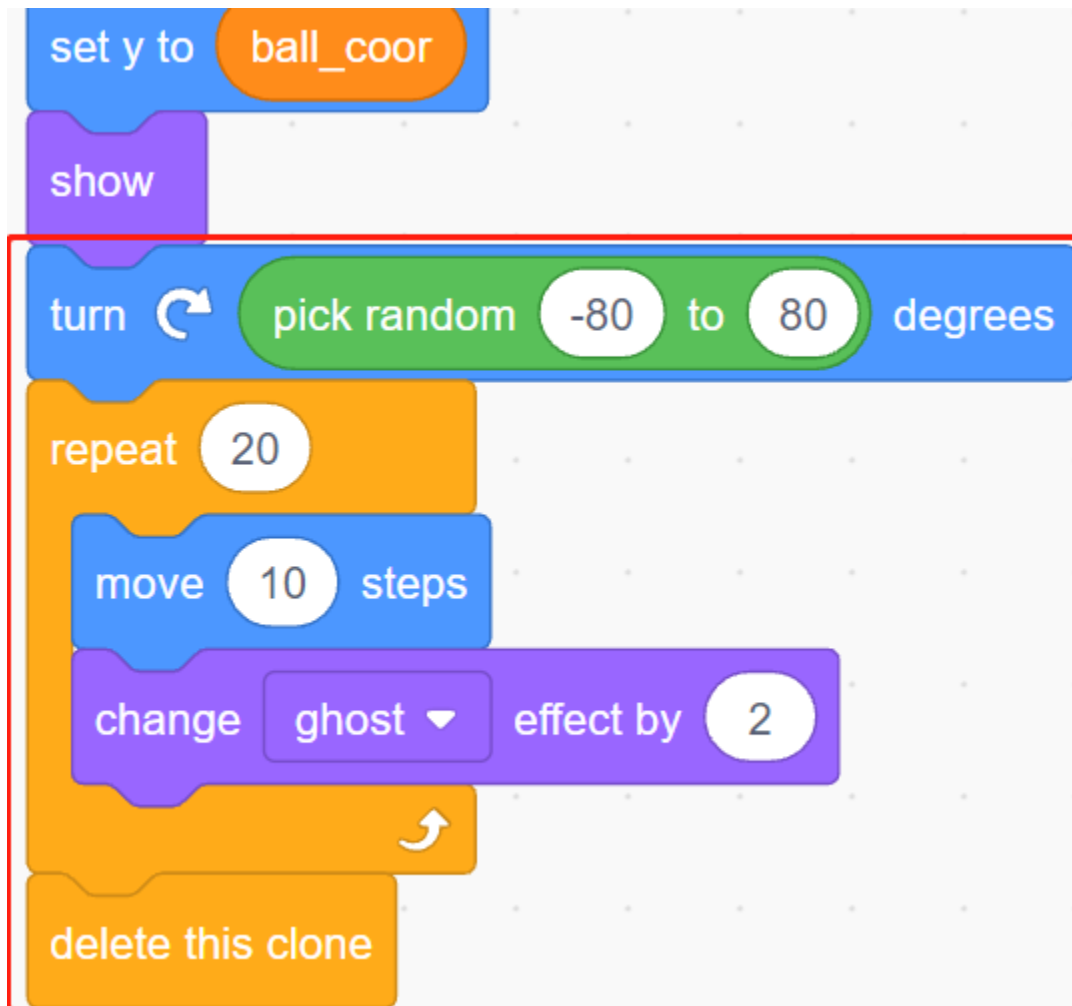
Lorsque le script démarre, cachez d'abord le sprite **Star**. Lorsque le message **Bling** est reçu, clonez le sprite **Star**.



Lorsque le sprite **Star** apparaît en tant que clone, jouez l'effet sonore et réglez ses coordonnées pour être synchronisées avec le sprite **Ball**.



Créez l'effet de l'apparition du sprite **Star** et ajustez-le selon les besoins.

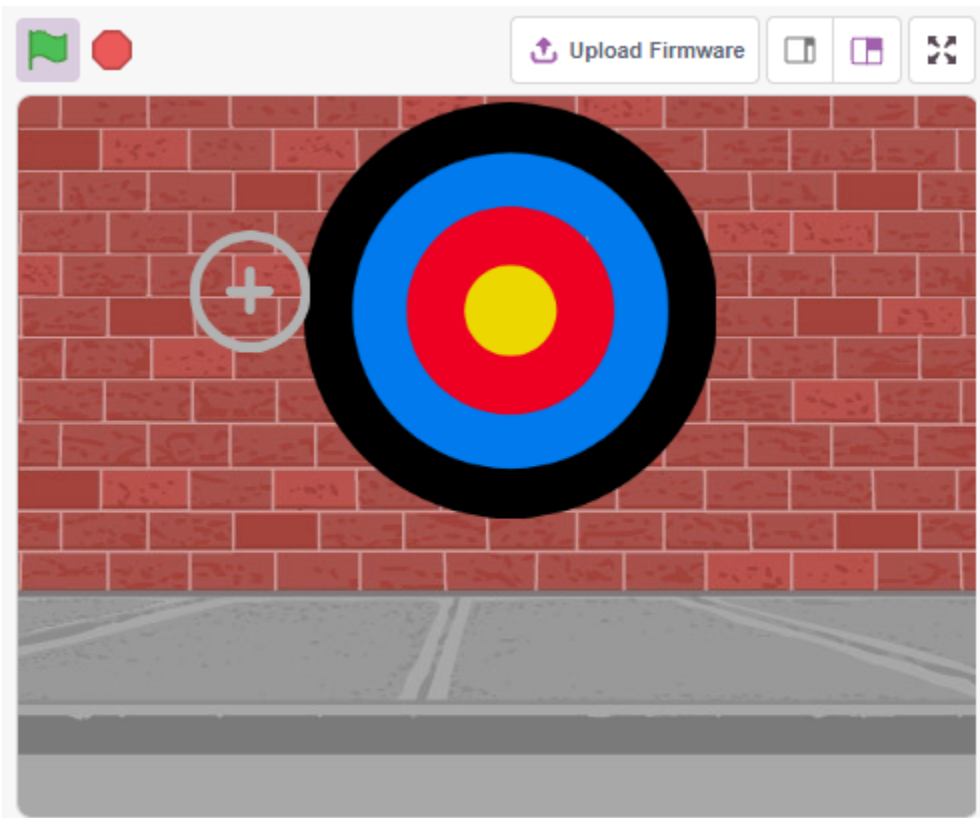


## 8.16 2.13 JEU - Tir

Avez-vous déjà vu ces jeux de tir à la télévision ? Plus un concurrent tire une balle près du centre de la cible, plus son score est élevé.

Aujourd'hui, nous réalisons également un jeu de tir sur Scratch. Dans le jeu, faites tirer le Réticule le plus loin possible vers le centre de la cible pour obtenir un score plus élevé.

Cliquez sur le drapeau vert pour commencer. Utilisez le module d'Évitement d'Obstacles pour tirer une balle.



8.16.1 Vous Apprendrez

- Comment fonctionne le module d’Évitement d’Obstacles et sa plage d’angle
- Peindre différents sprites
- Toucher des couleurs

8.16.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.  
Il est certainement pratique d’acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

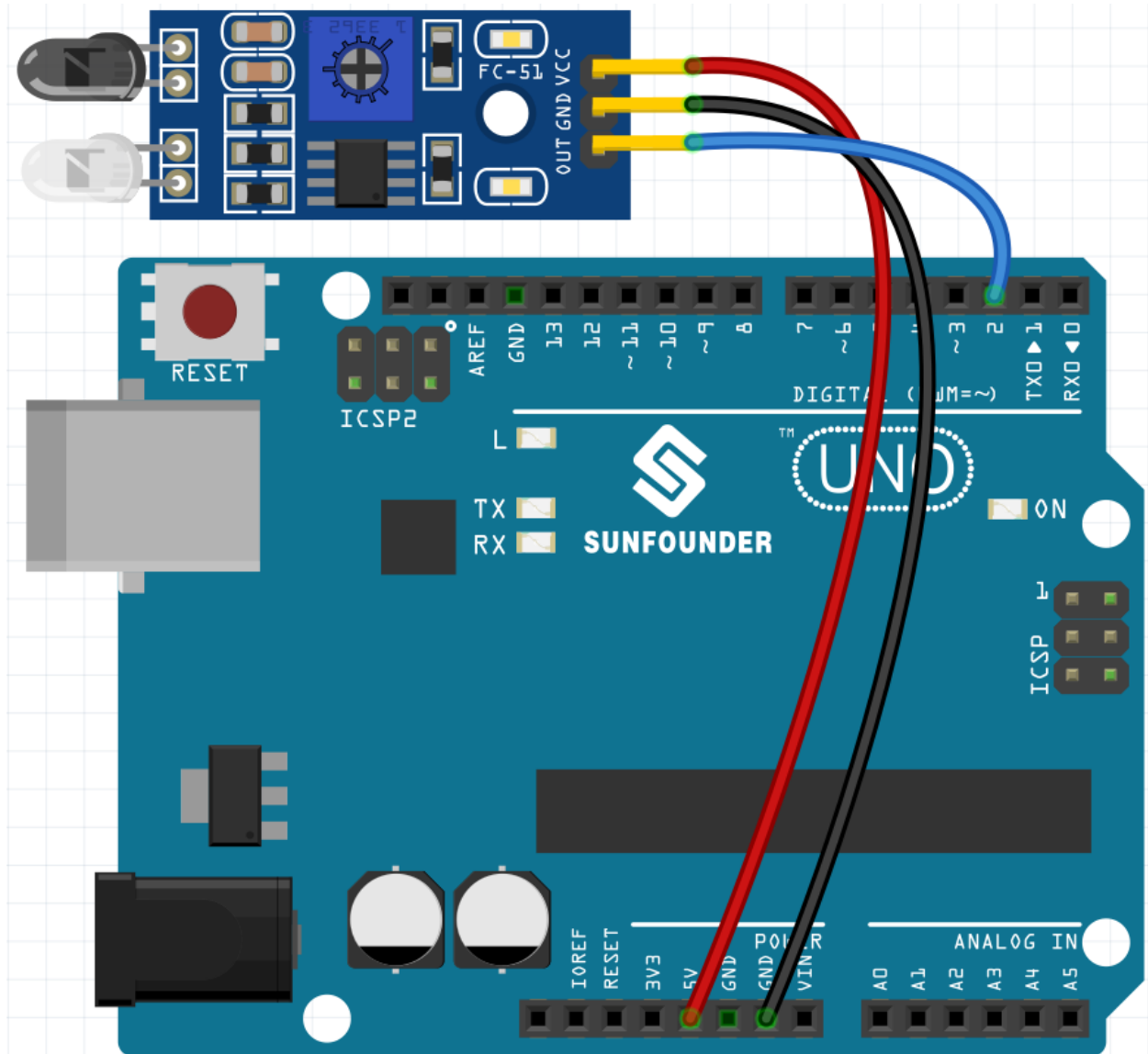
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D’ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module d’Évitement d’Obstacle</i>	

### 8.16.3 Construisez le Circuit

Le module d'évitement d'obstacles est un capteur infrarouge de proximité à distance réglable dont la sortie est normalement haute et devient basse lorsqu'un obstacle est détecté.

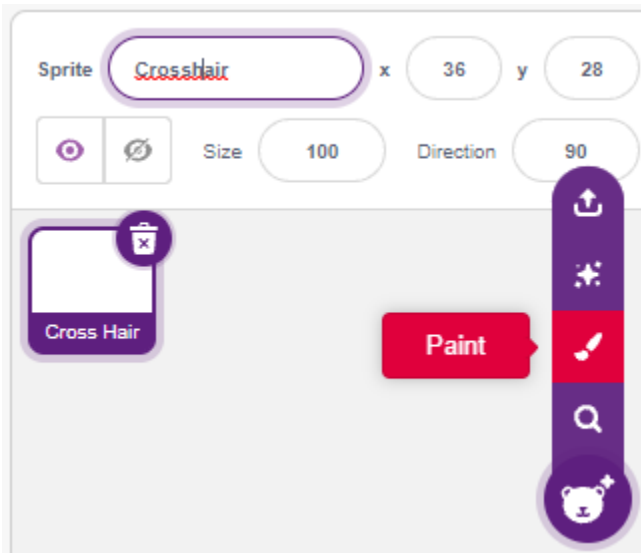
Construisez maintenant le circuit selon le schéma ci-dessous.



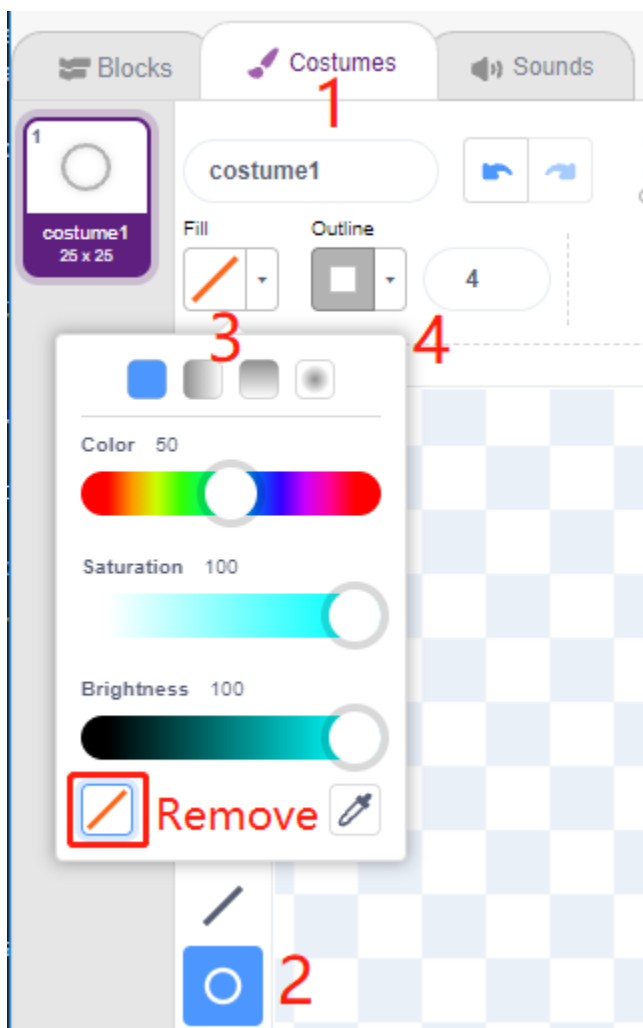
### 8.16.4 Programmation

#### 1. Peindre le sprite Réticule

Supprimez le sprite par défaut, sélectionnez le bouton **Sprite** et cliquez sur **Paint**, un sprite vierge **Sprite1** apparaîtra et nommez-le **Crosshair**.

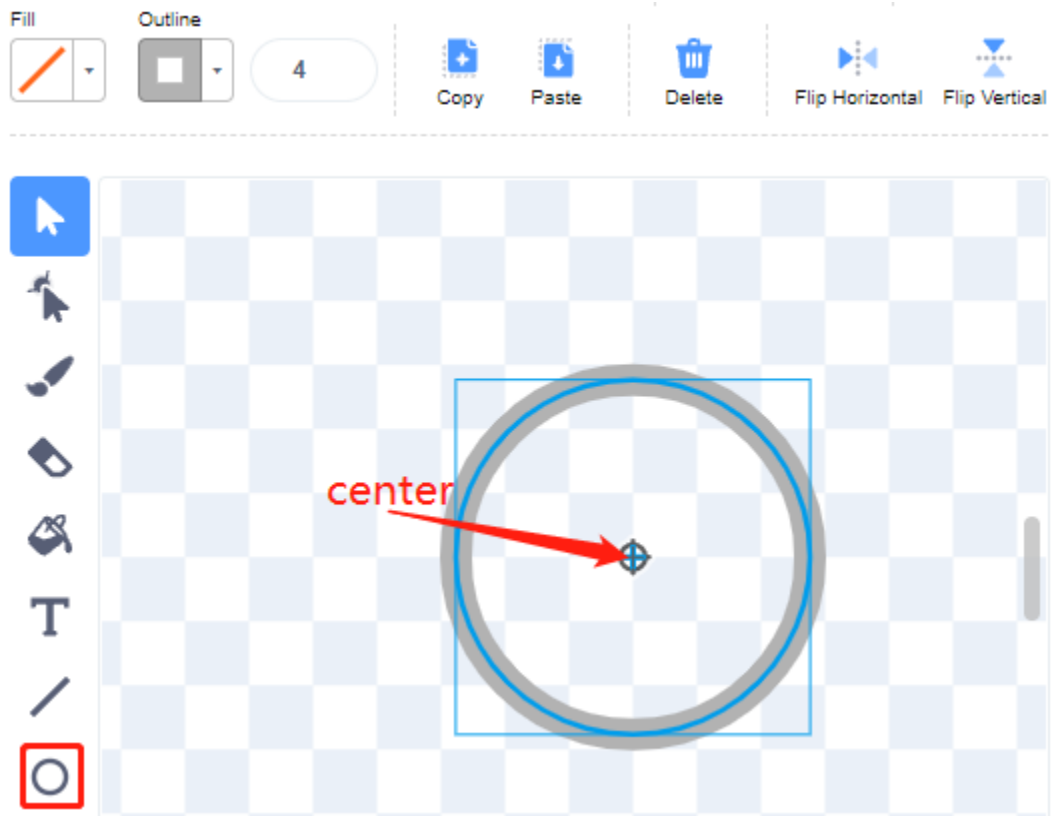


Allez à la page **Costumes** du sprite **Crosshair**. Cliquez sur l'outil **Circle**, retirez la couleur de remplissage et définissez la couleur et la largeur du contour.

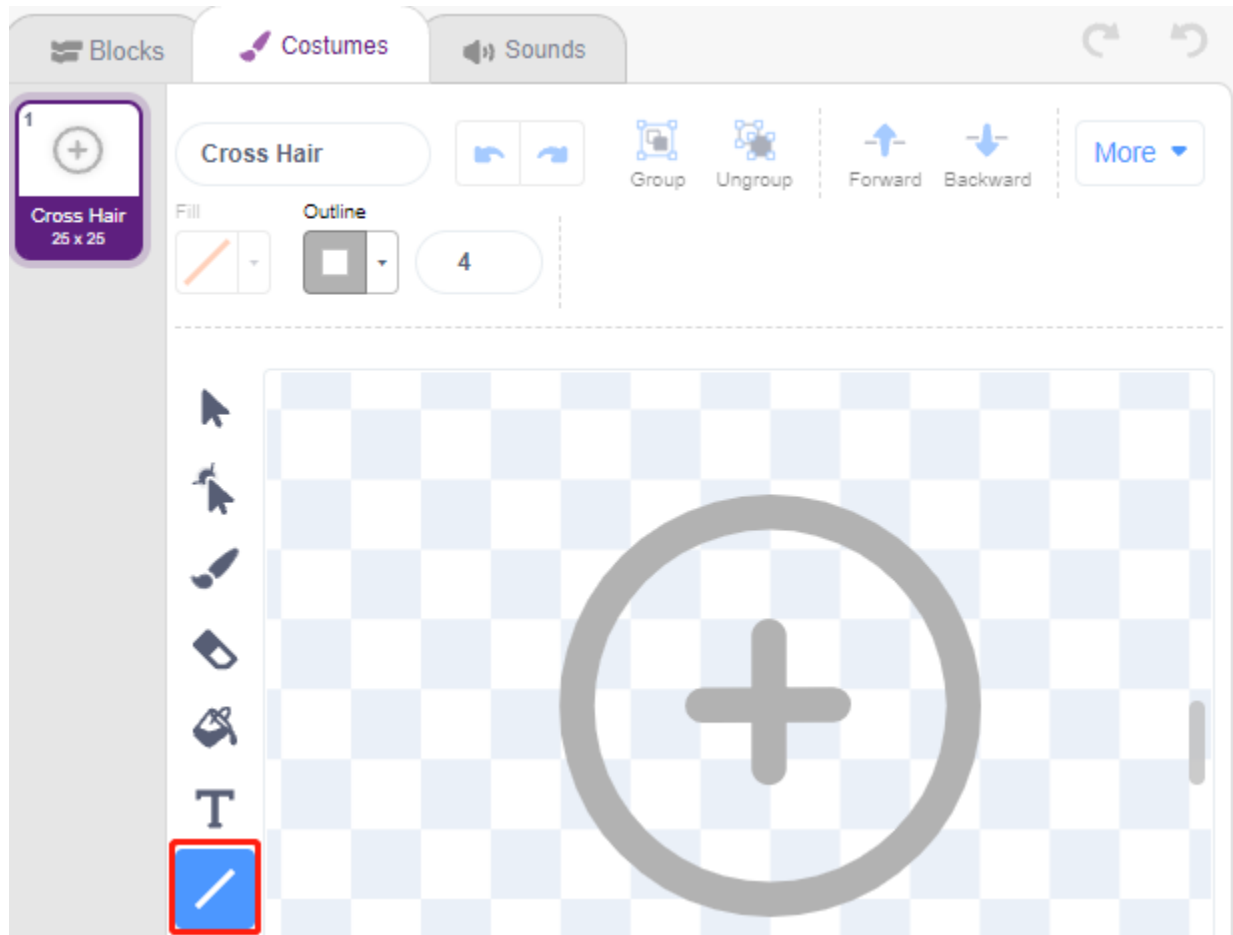


Dessinez maintenant un cercle avec l'outil **Circle**. Après le dessin, vous pouvez cliquer sur l'outil **Select** et déplacer le

cercle de sorte que le point d'origine soit aligné avec le centre du canevas.

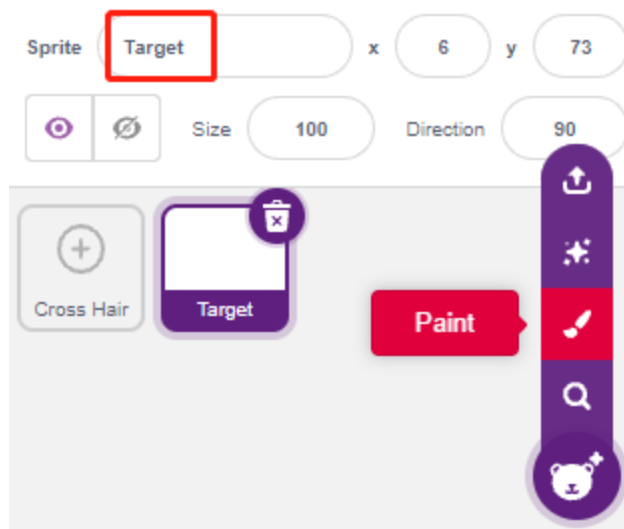


Utilisez l'outil **Line** pour dessiner une croix à l'intérieur du cercle.



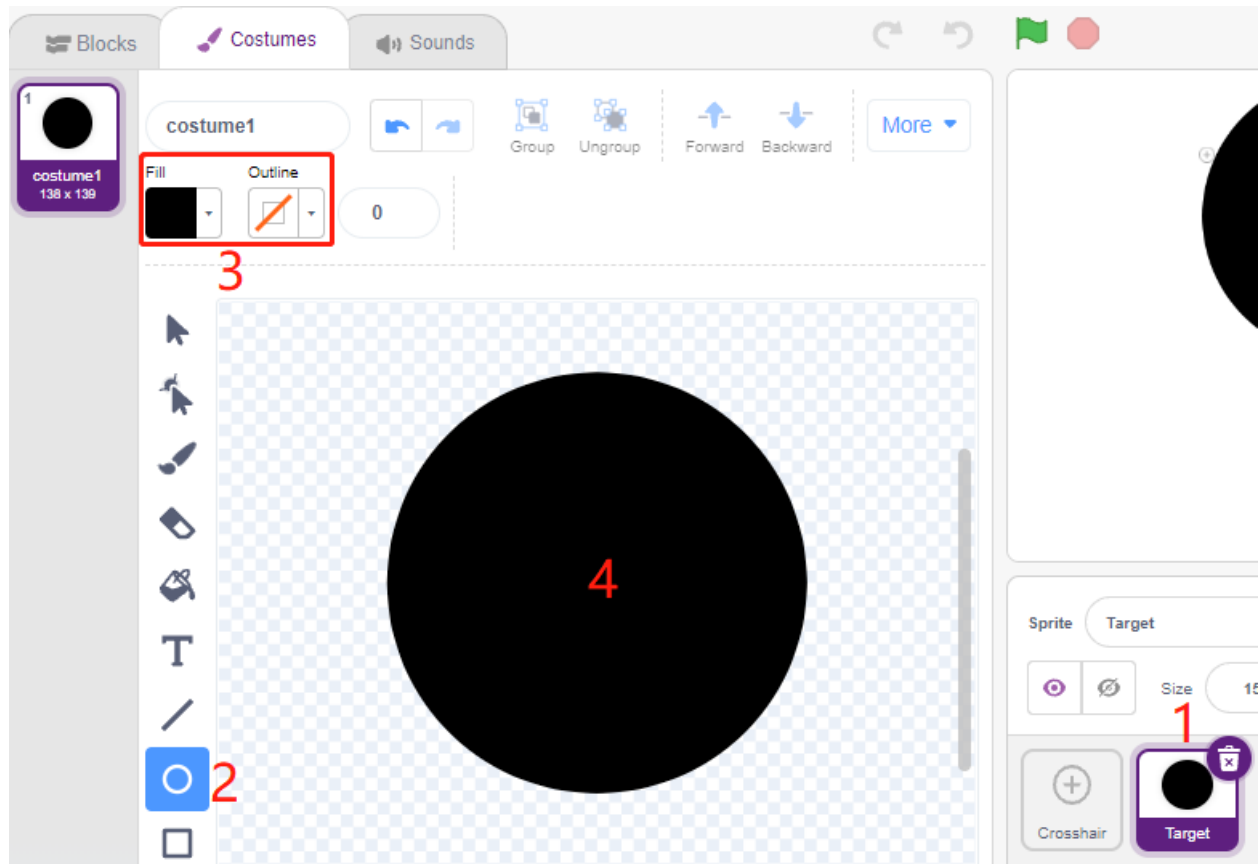
### Peindre le sprite Cible

Créez un nouveau sprite appelé sprite **Target**.

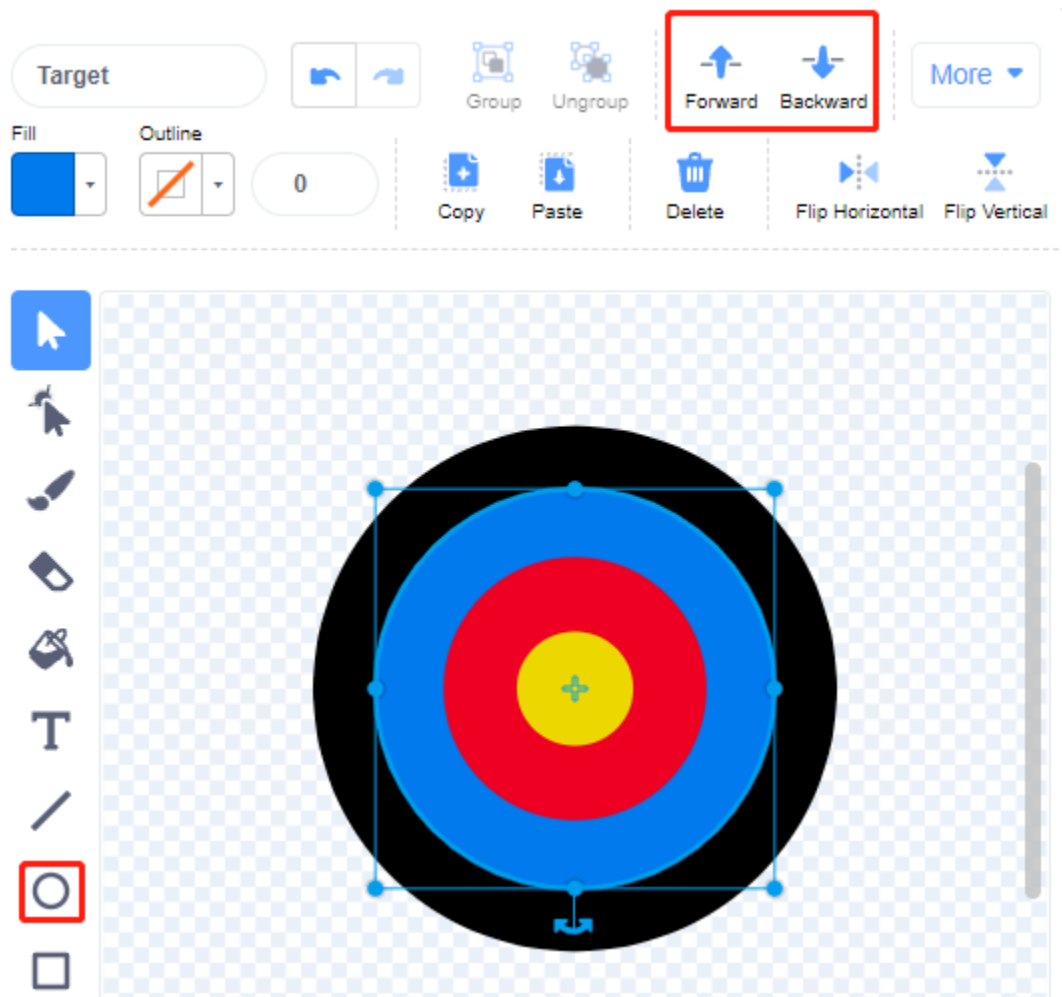


Allez à la page Costumes du sprite **Target**, cliquez sur l'outil **Circle**, sélectionnez une couleur de remplissage et supprimez le contour et peignez un grand cercle.



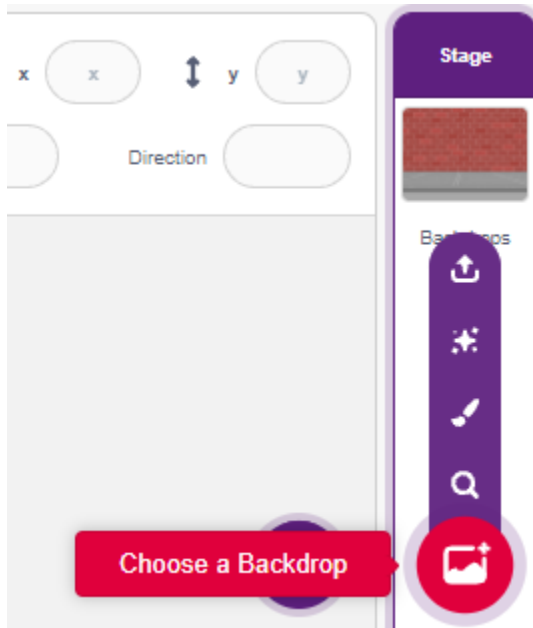


Utilisez la même méthode pour dessiner des cercles supplémentaires, chacun de couleur différente, et vous pouvez utiliser les outils **Forward** ou **Backward** pour changer la position des cercles qui se chevauchent. Notez que vous devez également sélectionner l'outil pour déplacer les cercles, afin que l'origine de tous les cercles et le centre du canevas soient alignés.



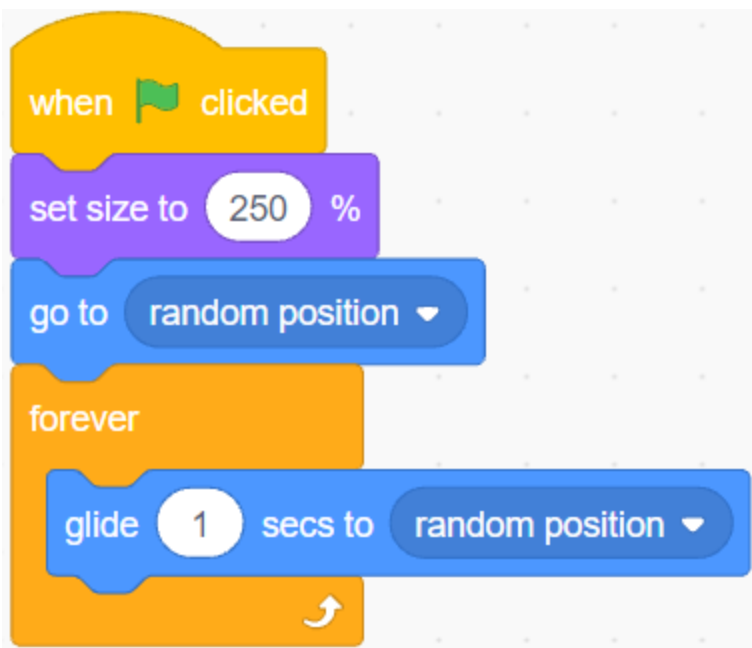
### 3. Ajouter un décor

Ajoutez un fond approprié qui de préférence n'a pas trop de couleurs et ne correspond pas aux couleurs dans le sprite **Target**. Ici, j'ai choisi le décor **Wall1**.

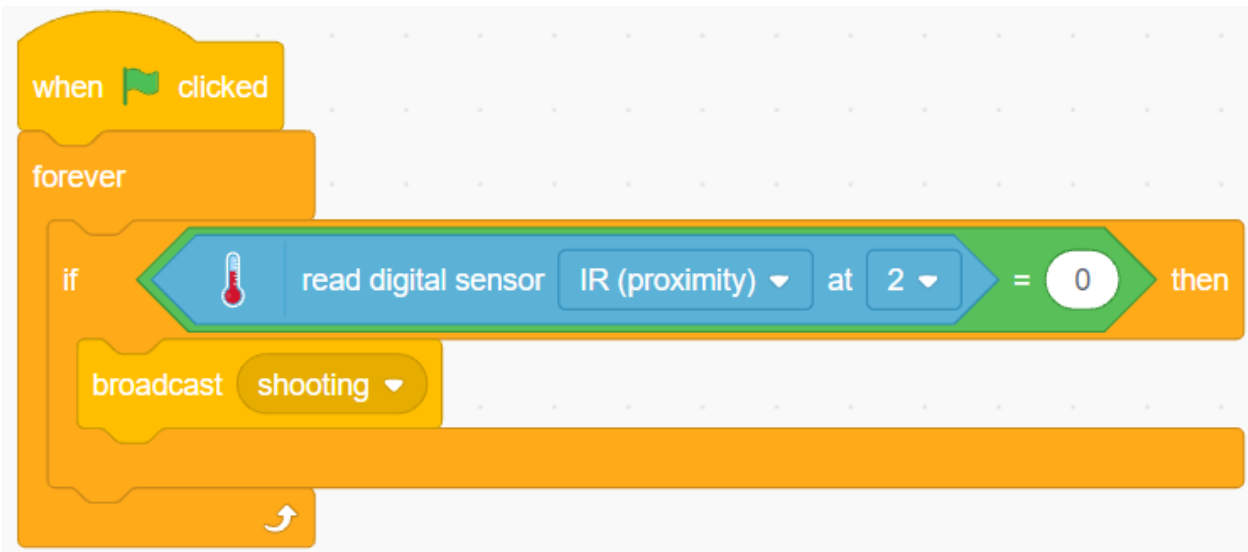


#### 4. Scripter le sprite Réticule

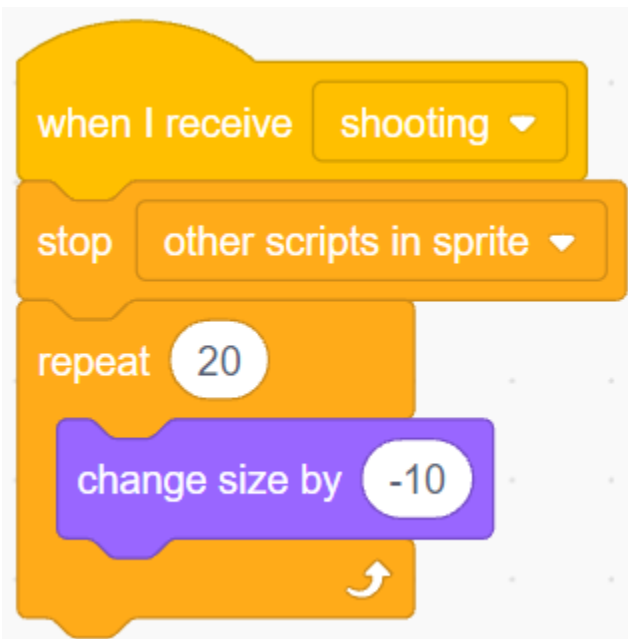
Définissez la position et la taille aléatoires du sprite **Crosshair**, et laissez-le se déplacer de manière aléatoire.



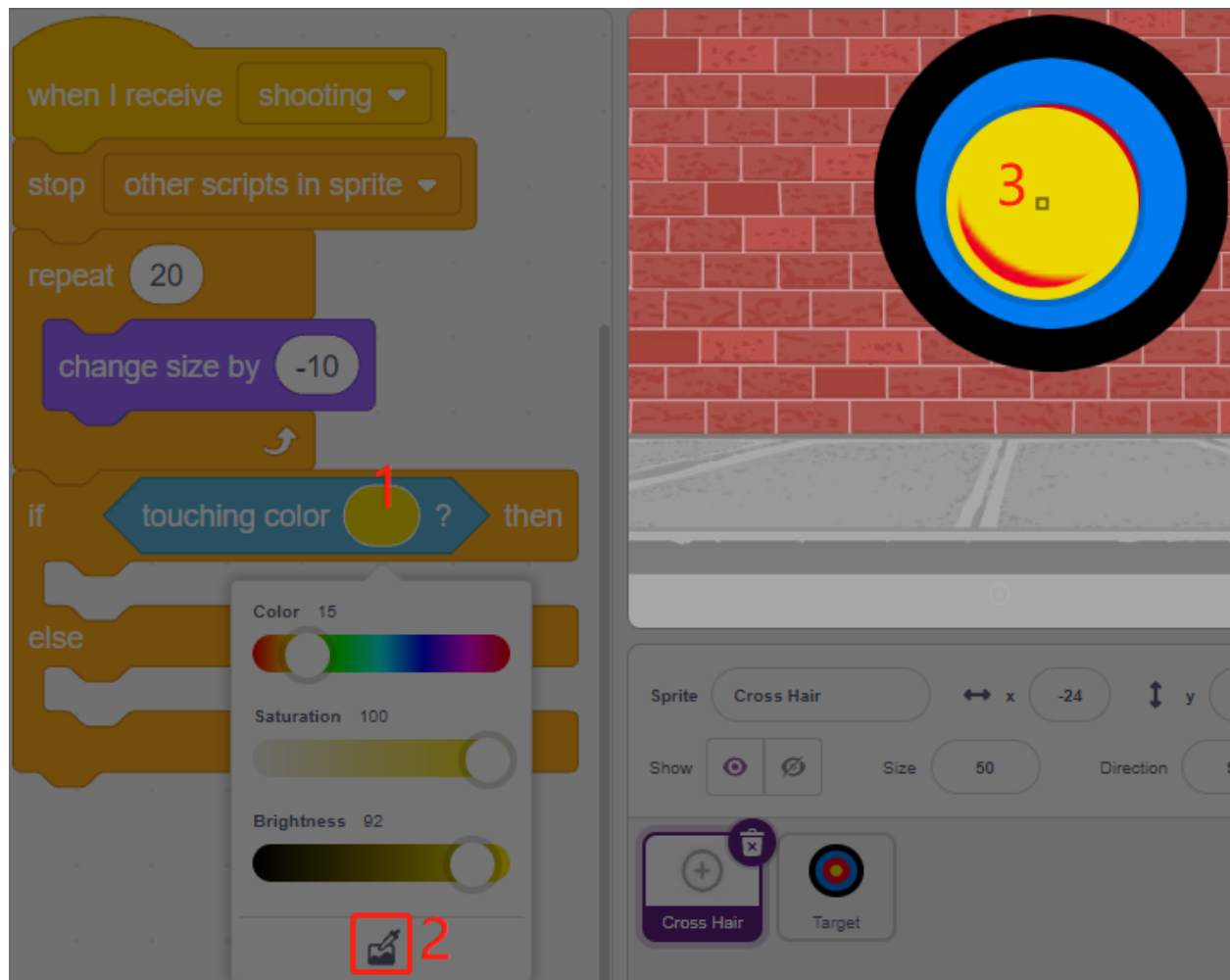
Lorsqu'une main est placée devant le module d'évitement d'obstacles, il émet un niveau bas comme signal de transmission.



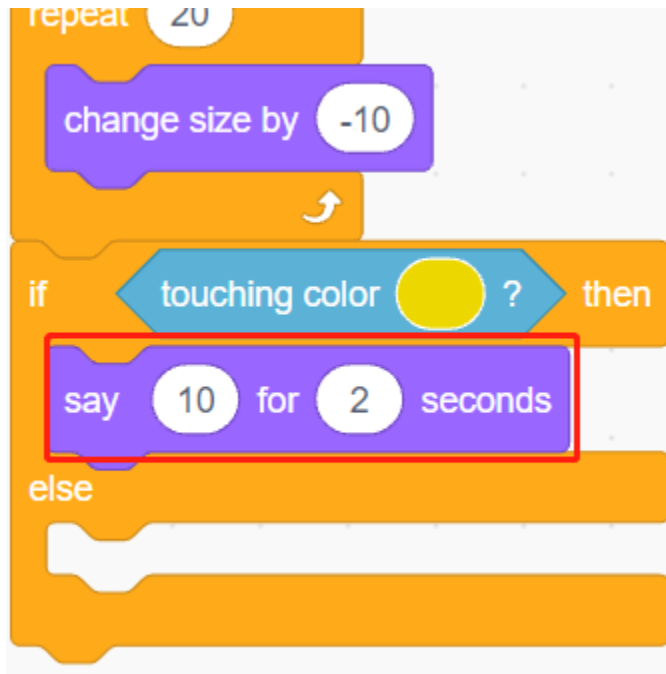
Lorsque le message **shooting** est reçu, le sprite arrête de bouger et rétrécit lentement, simulant ainsi l'effet d'un tir de balle.



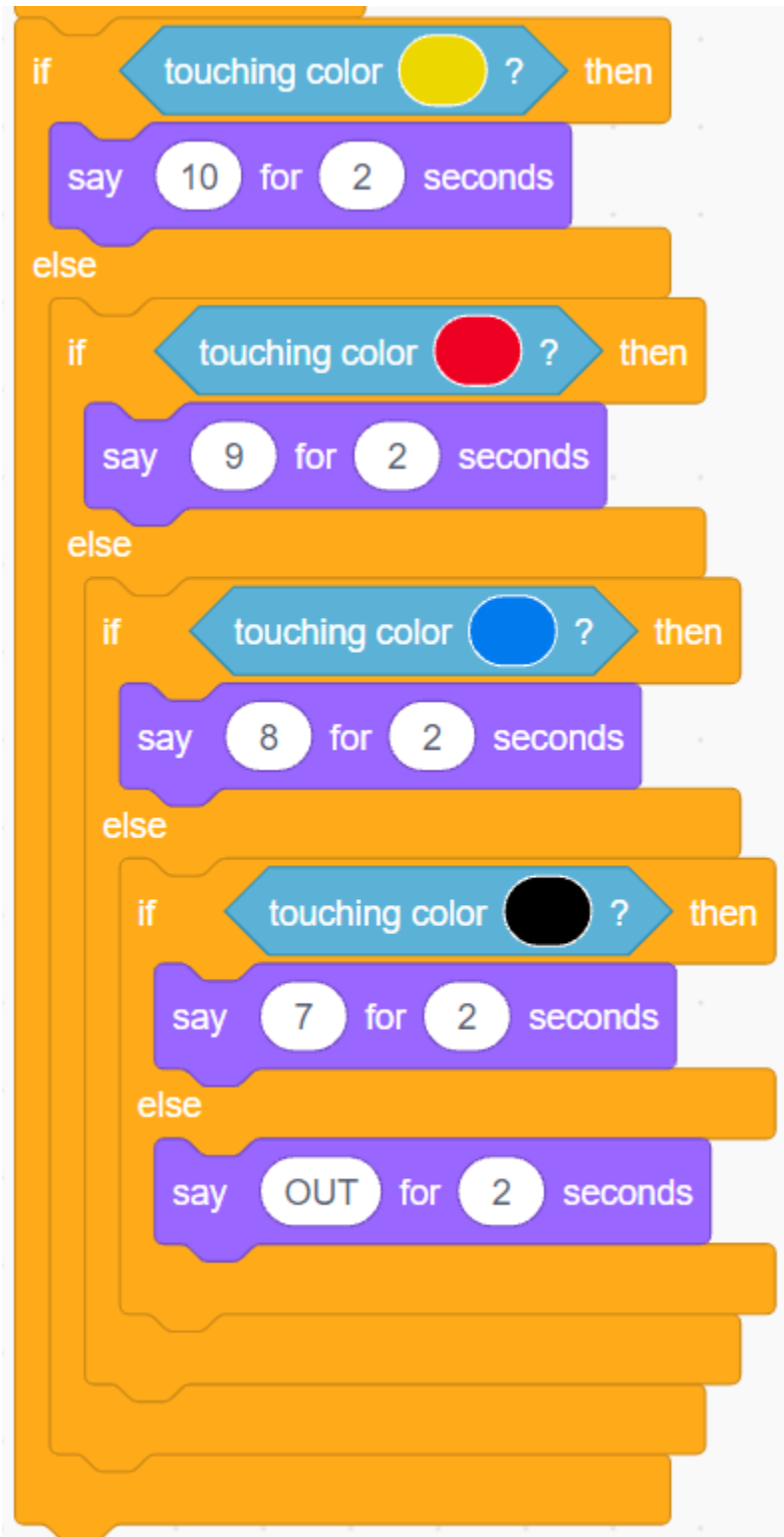
Utilisez le bloc [Touch color ()] pour déterminer la position du tir.



Lorsque le tir est à l'intérieur du cercle jaune, 10 est annoncé.



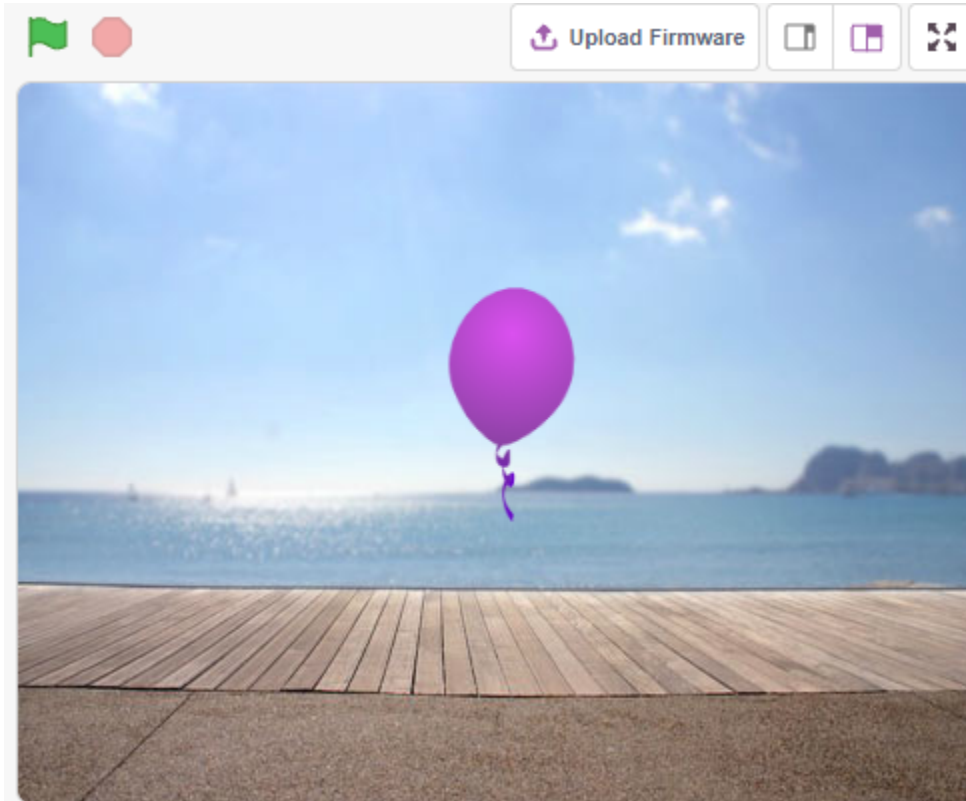
Utilisez la même méthode pour déterminer la position du tir, si ce n'est pas sur le sprite **Target**, cela signifie qu'il est en dehors du cercle.



## 8.17 2.14 JEU - Gonfler le Ballon

Ici, nous allons jouer à un jeu de gonflage de ballon.

Après avoir cliqué sur le drapeau vert, le ballon deviendra de plus en plus gros. Si le ballon est trop gros, il explosera ; si le ballon est trop petit, il tombera ; vous devez juger quand appuyer sur le bouton pour le faire monter.



### 8.17.1 Vous Apprendrez

— Peindre un costume pour le sprite

### 8.17.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

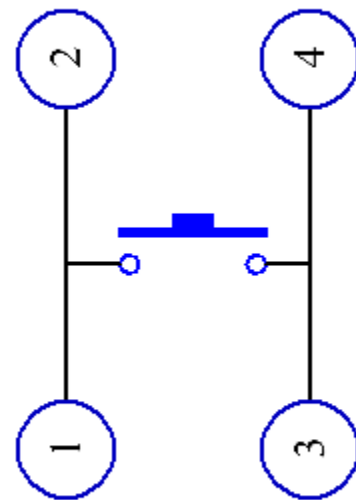
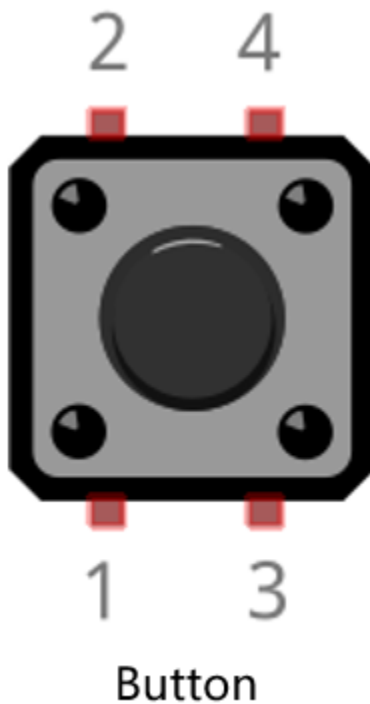
Vous pouvez également les acheter séparément via les liens ci-dessous.



INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Condensateur</i>	
<i>Bouton</i>	

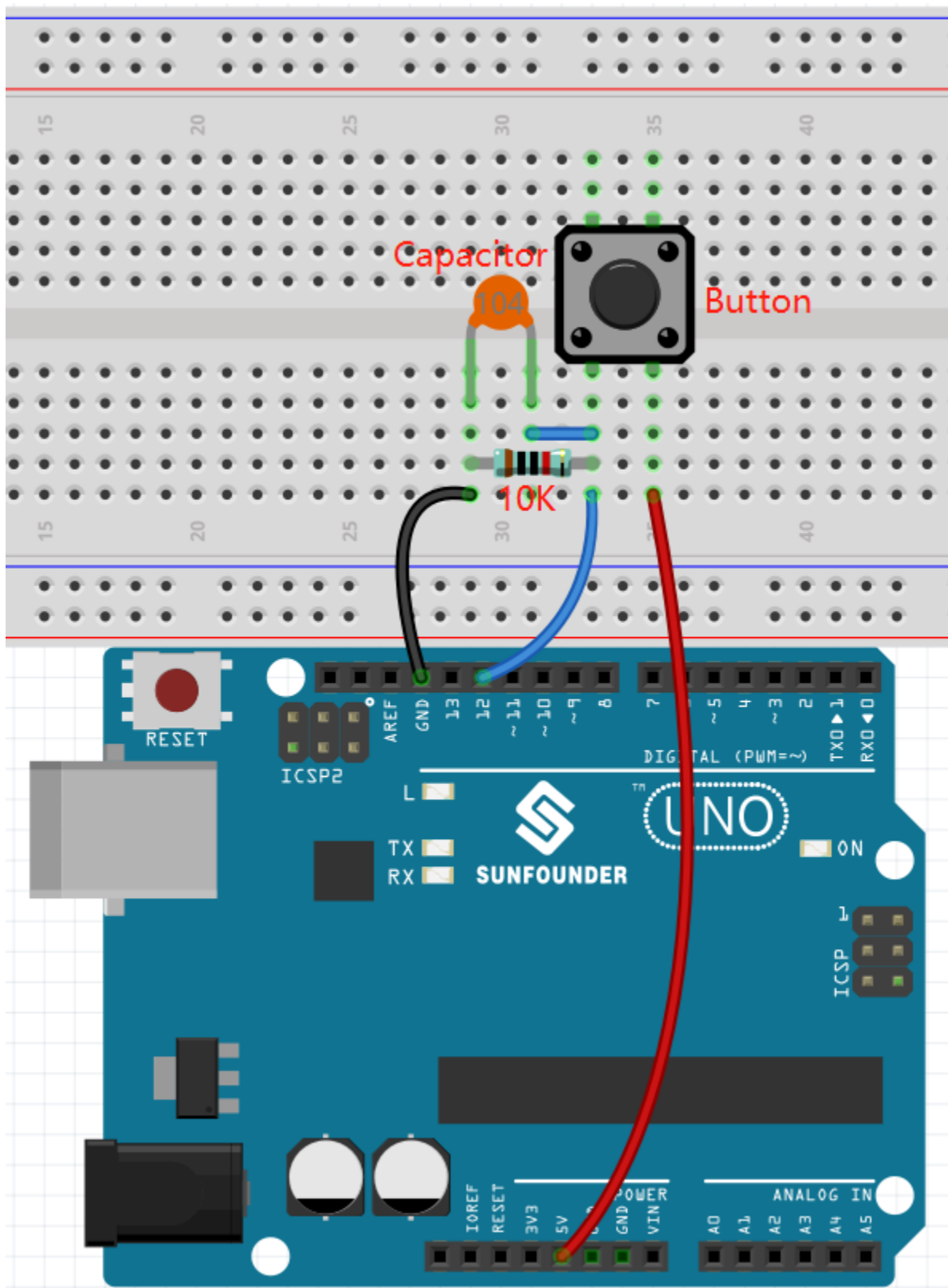
### 8.17.3 Construisez le Circuit

Le bouton est un dispositif à 4 broches, puisque la broche 1 est connectée à la broche 2, et la broche 3 à la broche 4, lorsque le bouton est pressé, les 4 broches sont connectées, fermant ainsi le circuit.



Construisez le circuit selon le schéma suivant.

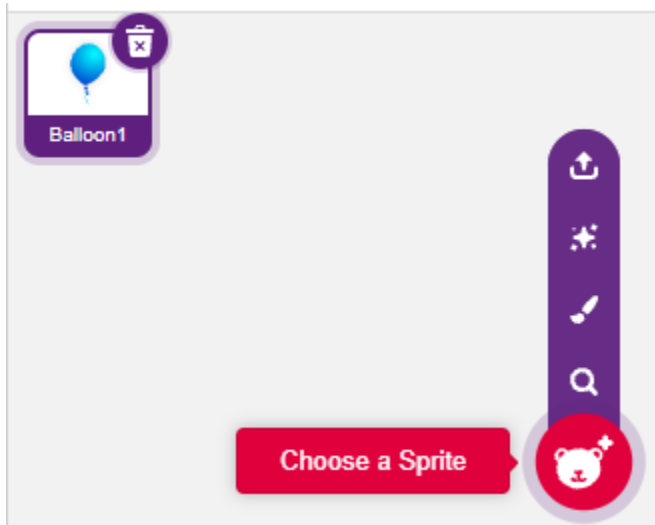
- Connectez l'une des broches du côté gauche du bouton à la broche 12, qui est connectée à une résistance de tirage et à un condensateur de 0.1uF (104) (pour éliminer le jitter et émettre un niveau stable lorsque le bouton fonctionne).
- Connectez l'autre extrémité de la résistance et du condensateur à GND, et l'une des broches du côté droit du bouton à 5V.



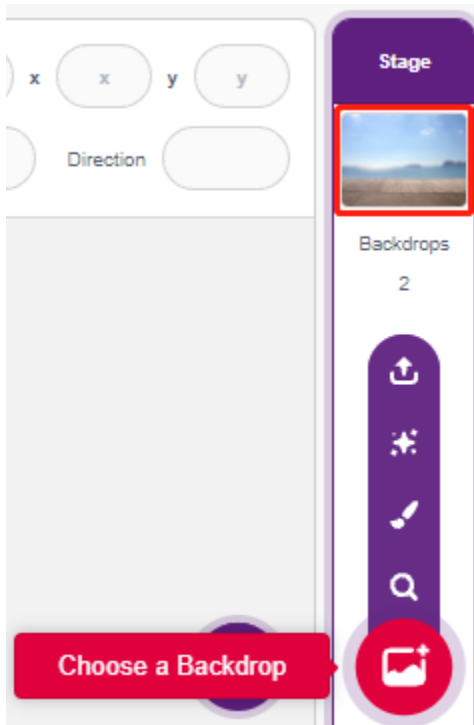
## 8.17.4 Programmation

### 1. Ajouter un sprite et un décor

Supprimez le sprite par défaut, cliquez sur le bouton **Choose a Sprite** dans le coin inférieur droit de la zone de sprite, puis sélectionnez le sprite **Balloon1**.



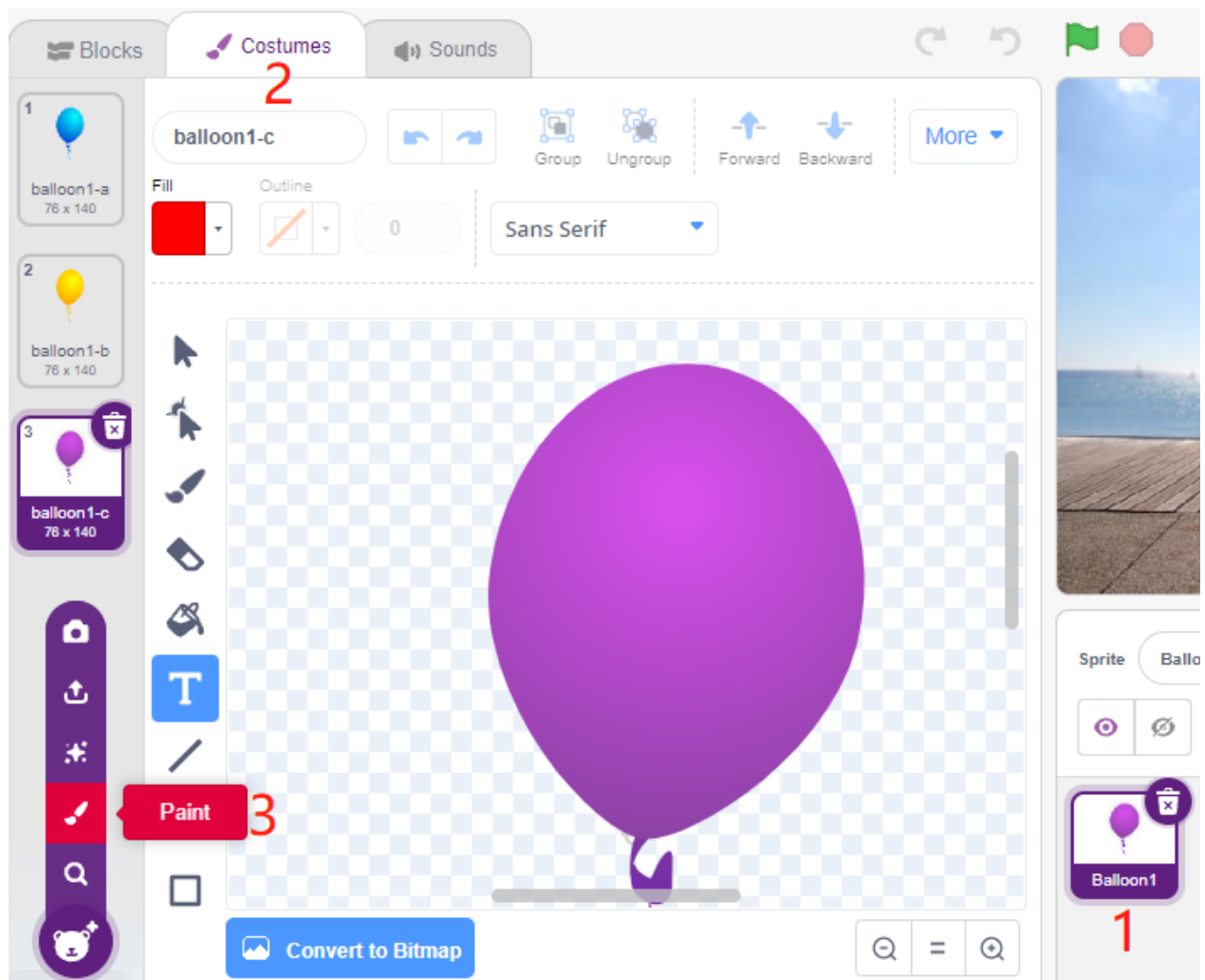
Ajoutez un décor **Boardwalk** via le bouton **Choose a backdrop**, ou d'autres décors de votre choix.



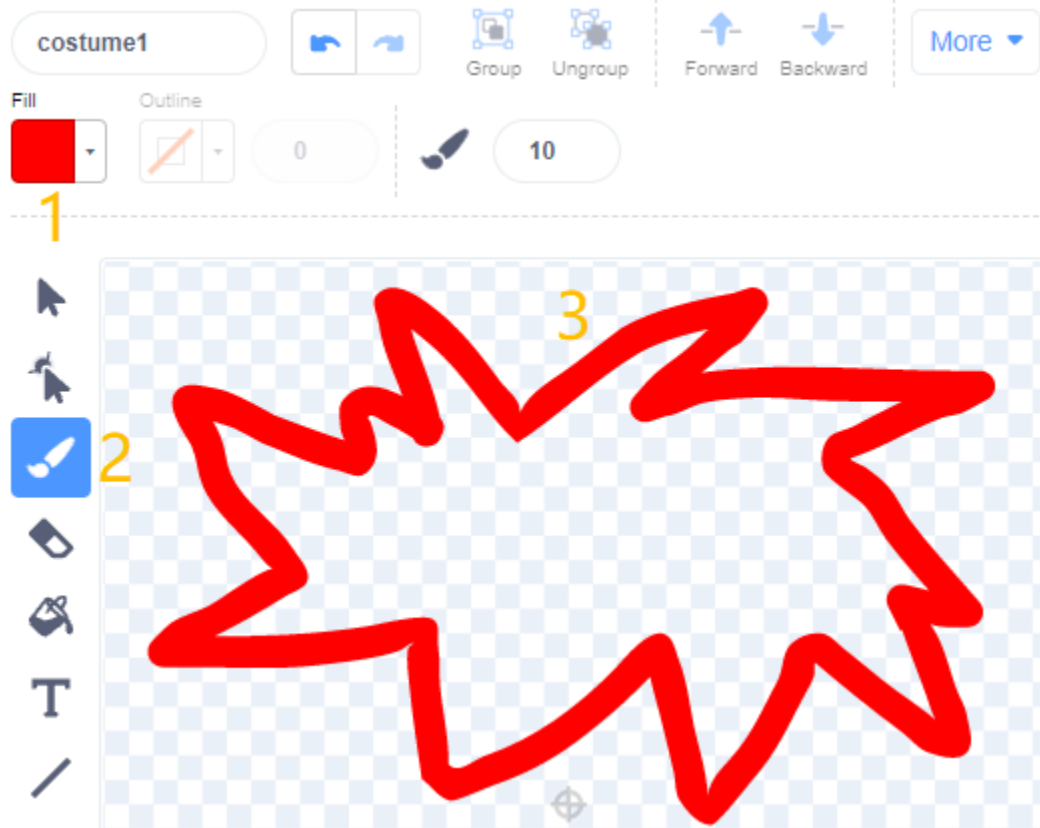
### 2. Peindre un costume pour le sprite Balloon1

Dessinez maintenant un costume d'effet d'explosion pour le sprite de ballon.

Allez à la page **Costumes** pour le sprite **Balloon1**, cliquez sur le bouton **Choose a Costume** dans le coin inférieur gauche, et sélectionnez **Paint** pour afficher un **Costume** vierge.



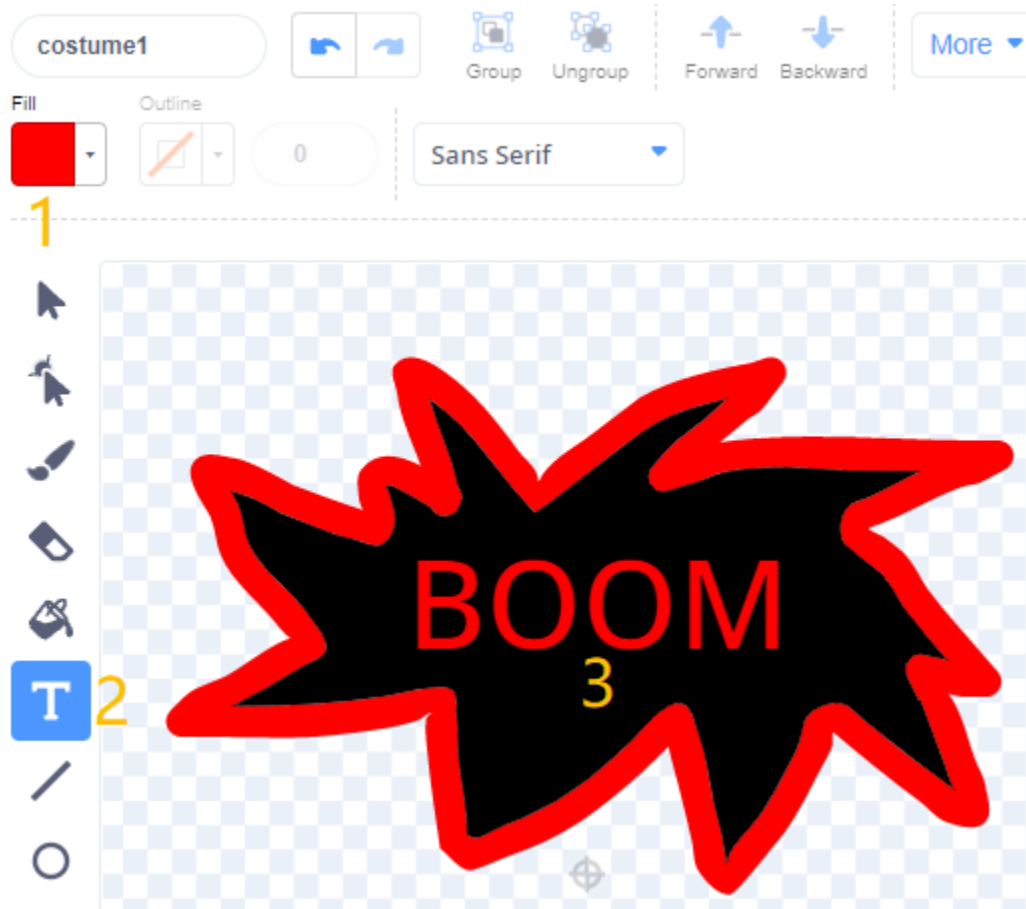
Sélectionnez une couleur, puis utilisez l'outil **Brush** pour dessiner un motif.



Sélectionnez à nouveau une couleur, cliquez sur l'outil Remplir, et déplacez la souris à l'intérieur du motif pour le remplir d'une couleur.

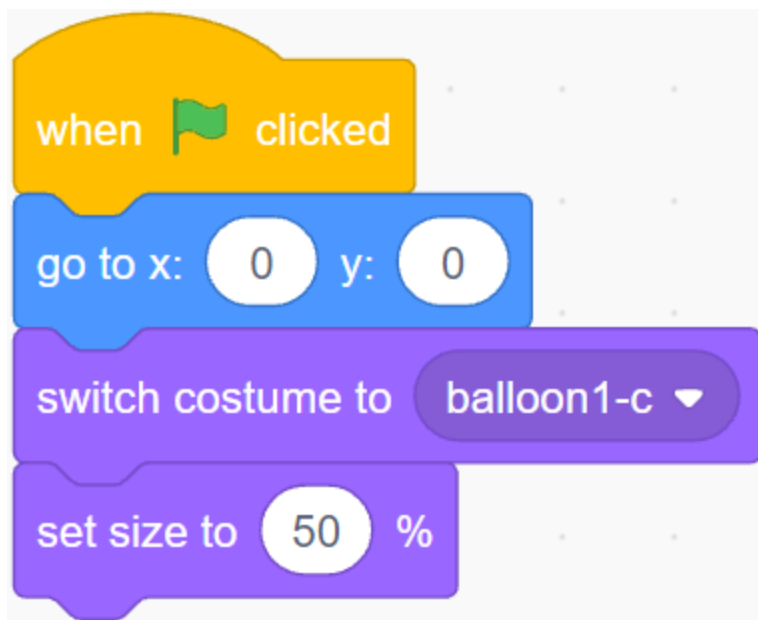


Enfin, écrivez le texte BOUM, afin qu'un costume d'effet d'explosion soit complet.

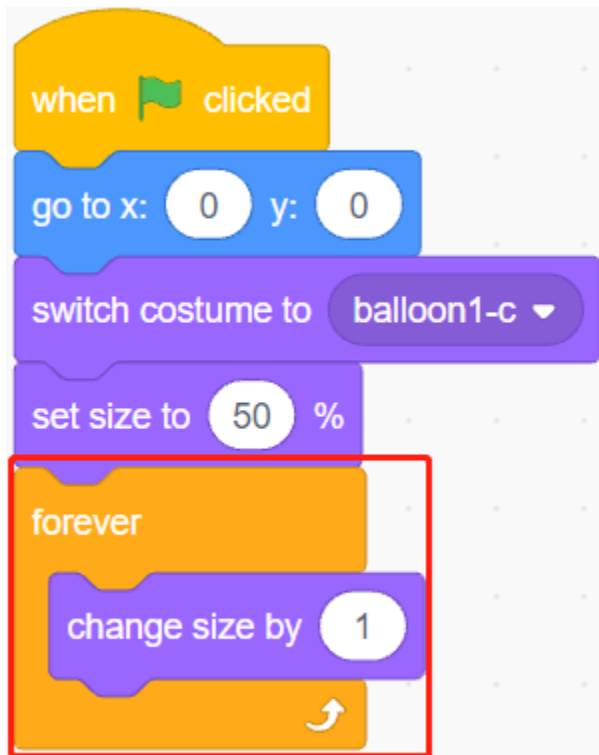


### 3. Scripter le sprite Ballon

Définissez la position et la taille initiales du sprite **Balloon1**.



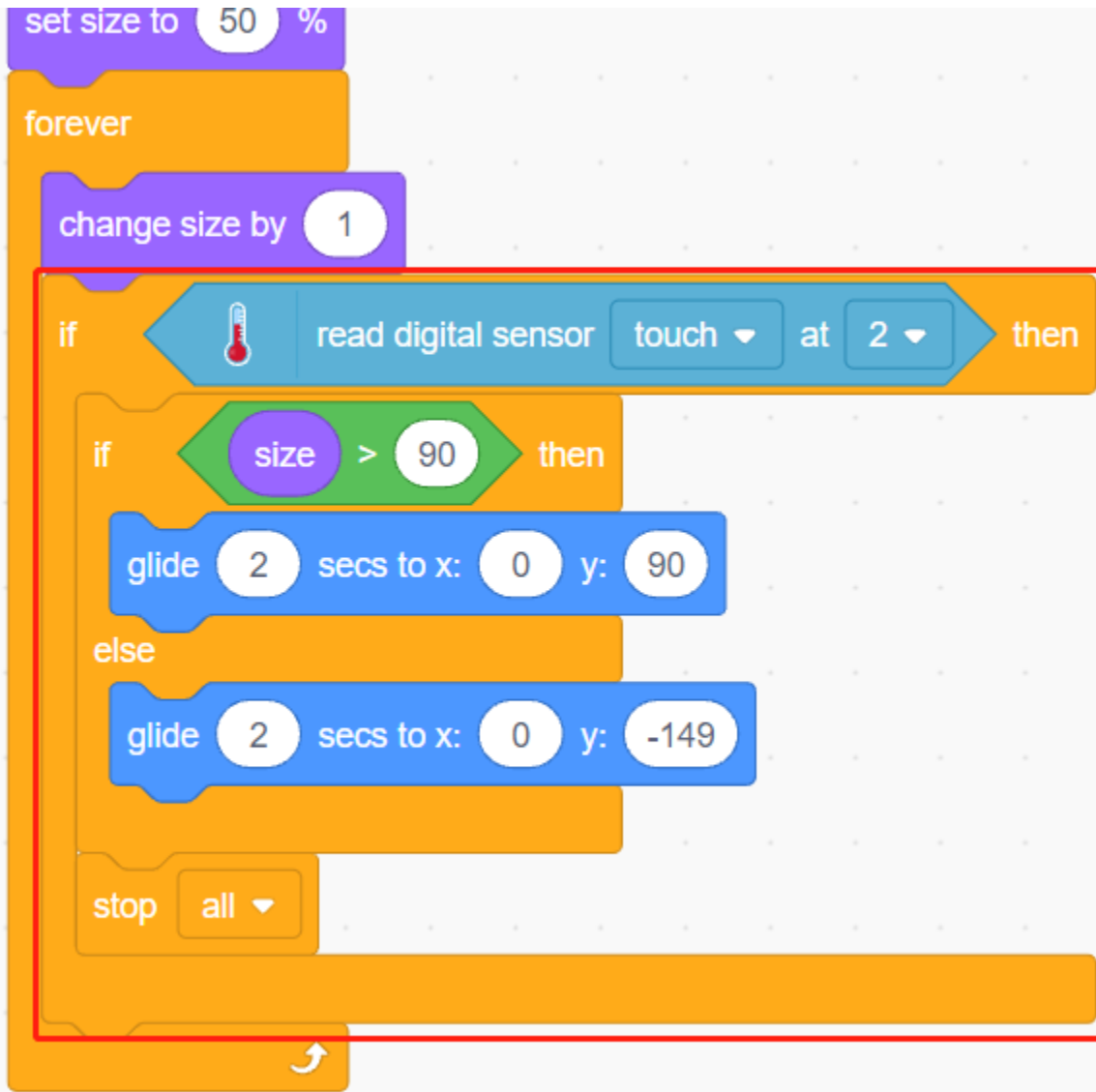
Puis laissez le sprite **Balloon** grossir lentement.



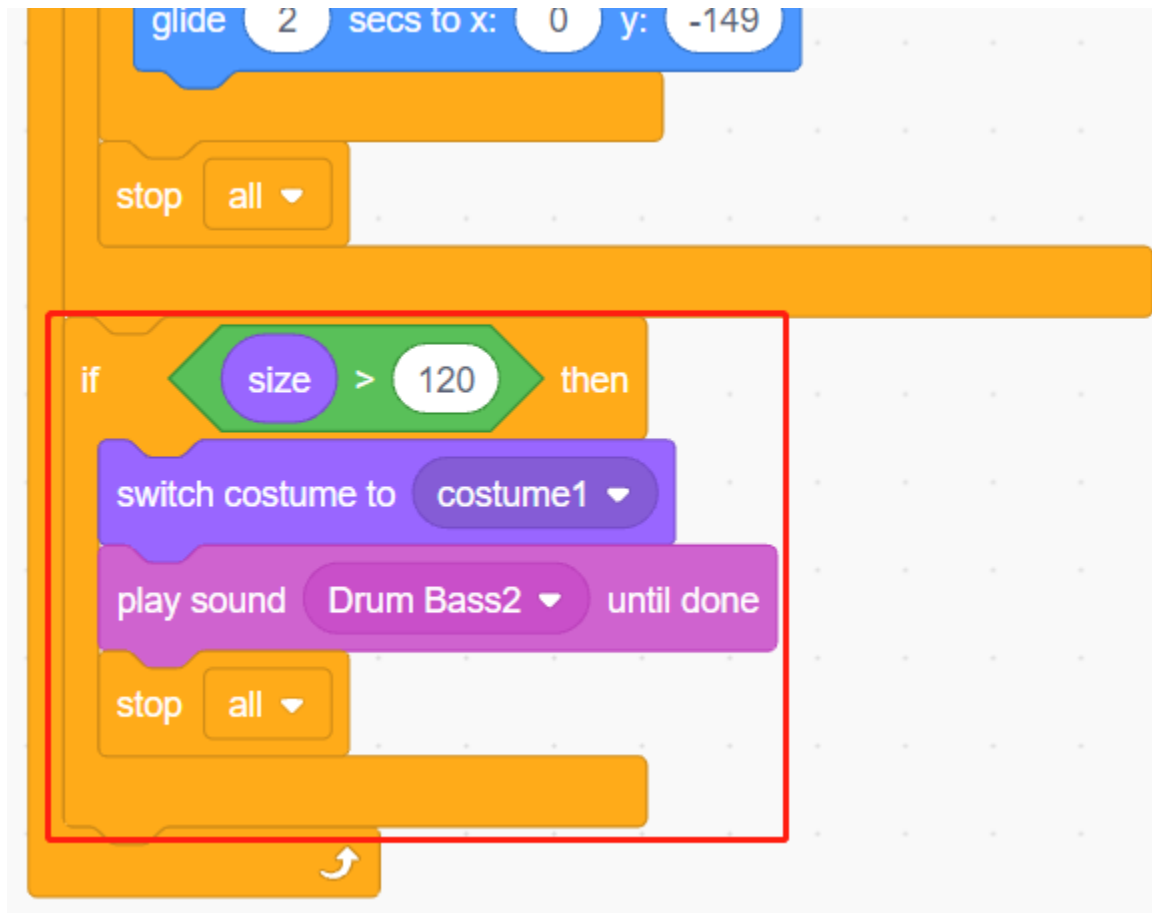
Lorsque le bouton est pressé (valeur est 1), la taille du sprite **Balloon1** cesse de grossir.

- Lorsque la taille est inférieure à 90, il tombera (la coordonnée y diminue).
- Lorsque la taille est supérieure à 90 et inférieure à 120, il volera vers le ciel (la coordonnée y augmente).





Si le bouton n'a pas été pressé, le ballon grossit lentement et quand la taille dépasse 120, il explosera (basculer vers le costume d'effet d'explosion).

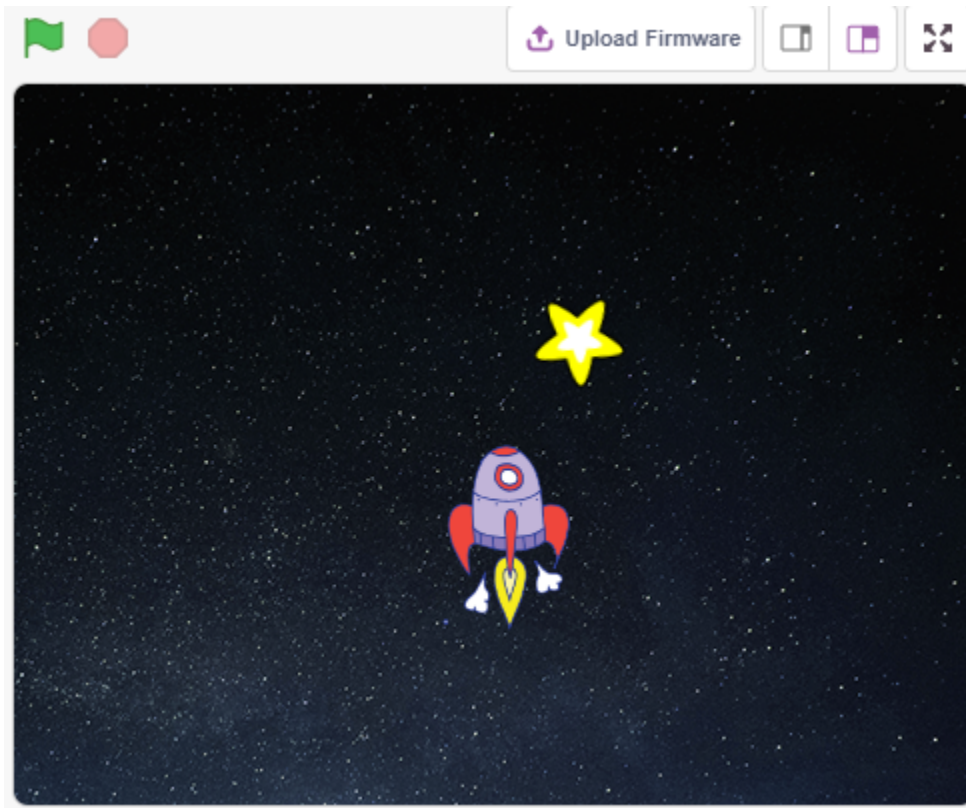


## 8.18 2.15 JEU - Étoiles Croisées

Dans les projets suivants, nous allons jouer à quelques mini-jeux amusants dans PictoBlox.

Ici, nous utilisons le module Joystick pour jouer à un jeu nommé Étoiles Croisées.

Après l'exécution du script, des étoiles apparaîtront aléatoirement sur la scène, vous devez utiliser le Joystick pour contrôler le vaisseau spatial afin d'éviter les étoiles, si vous les touchez, le jeu sera terminé.



### 8.18.1 Vous Apprendrez

- Comment fonctionne le module Joystick
- Définir les coordonnées x et y du sprite

### 8.18.2 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module de Joystick</i>	-

### 8.18.3 Construisez le Circuit

Un joystick est un dispositif d'entrée constitué d'un manche qui pivote sur une base et transmet son angle ou sa direction au dispositif qu'il contrôle. Les joysticks sont souvent utilisés pour contrôler les jeux vidéo et les robots.

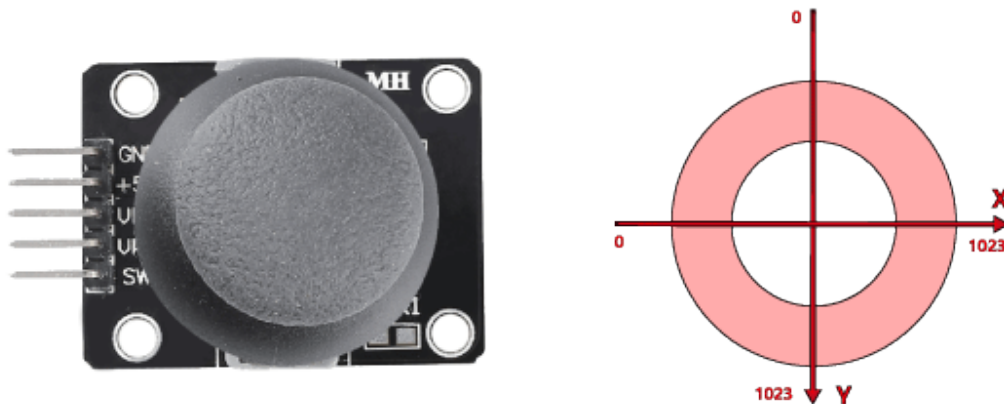
Pour communiquer une gamme complète de mouvements à l'ordinateur, un joystick doit mesurer la position du manche sur deux axes – l'axe X (de gauche à droite) et l'axe Y (de haut en bas).

Les coordonnées de mouvement du joystick sont présentées dans la figure suivante.

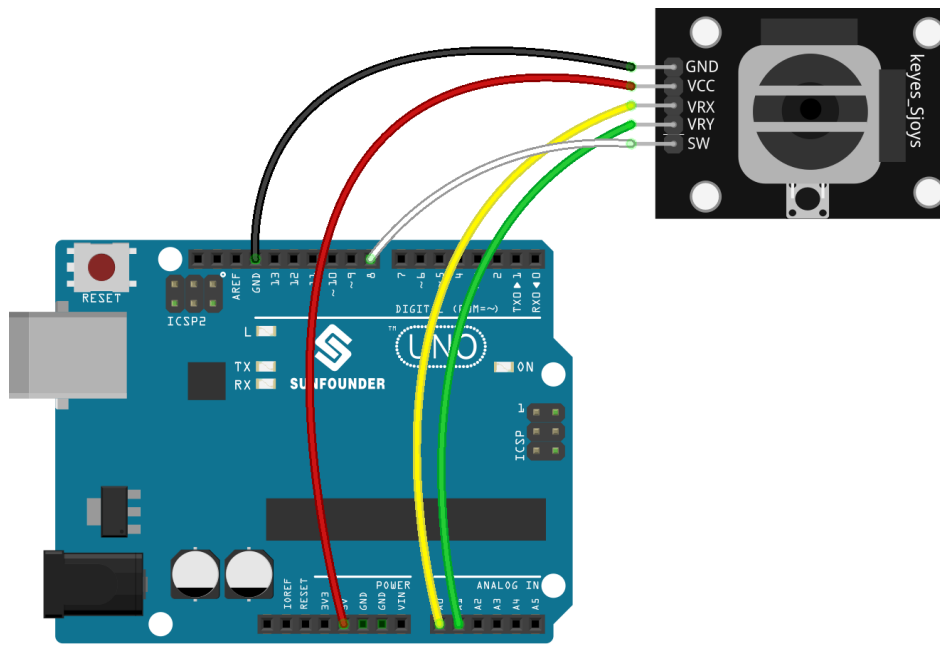
---

**Note :**

- La coordonnée x est de gauche à droite, la plage est de 0-1023.
  - La coordonnée y est de haut en bas, la plage est de 0-1023.
- 



Construisez maintenant le circuit selon le schéma suivant.

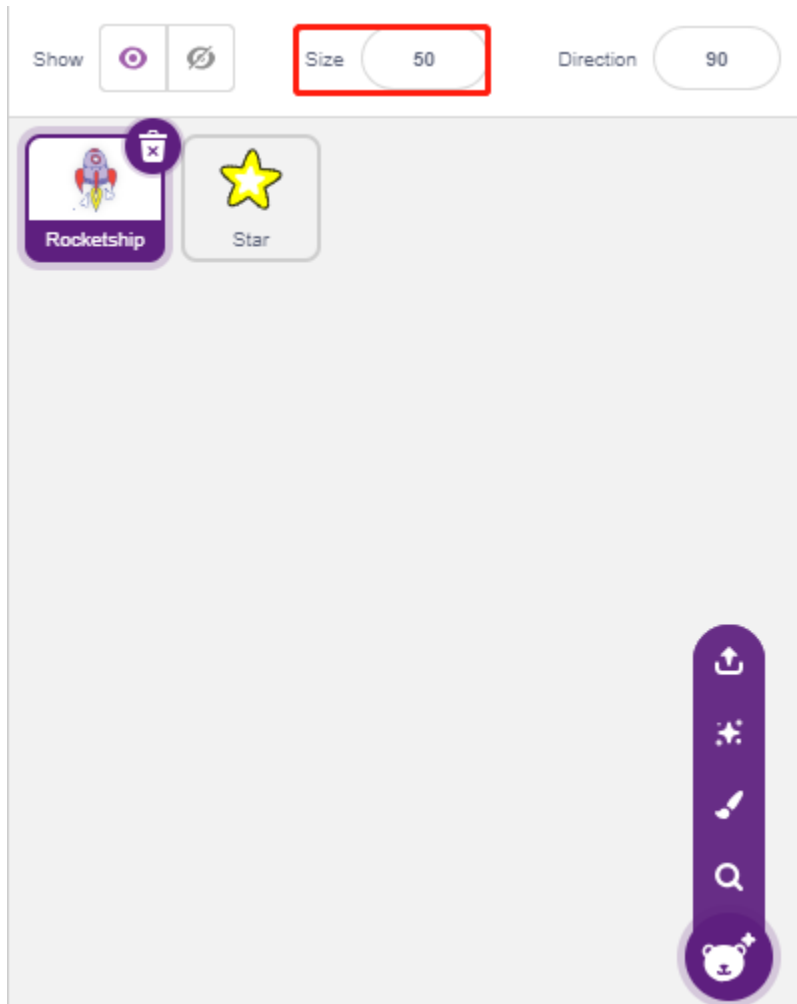


### 8.18.4 Programmation

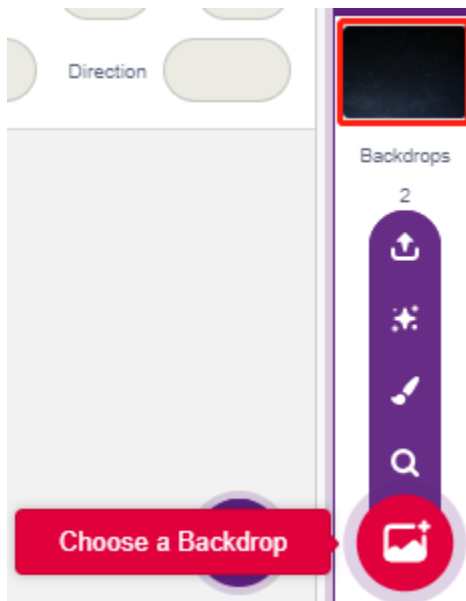
Le script complet vise à obtenir l'effet suivant : lorsque le drapeau vert est cliqué, le sprite **Stars** se déplace en courbe sur la scène et vous devez utiliser le joystick pour déplacer le **Rocketship**, afin qu'il ne soit pas touché par le sprite **Star**.

#### 1. Ajouter des sprites et des décors

Supprimez le sprite par défaut et utilisez le bouton **Choose a Sprite** pour ajouter les sprites **Rocketship** et **Star**. Notez que la taille du sprite **Rocket** est réglée à 50%.



Ajoutez maintenant le décor **Stars** via **Choose a Backdrop**.

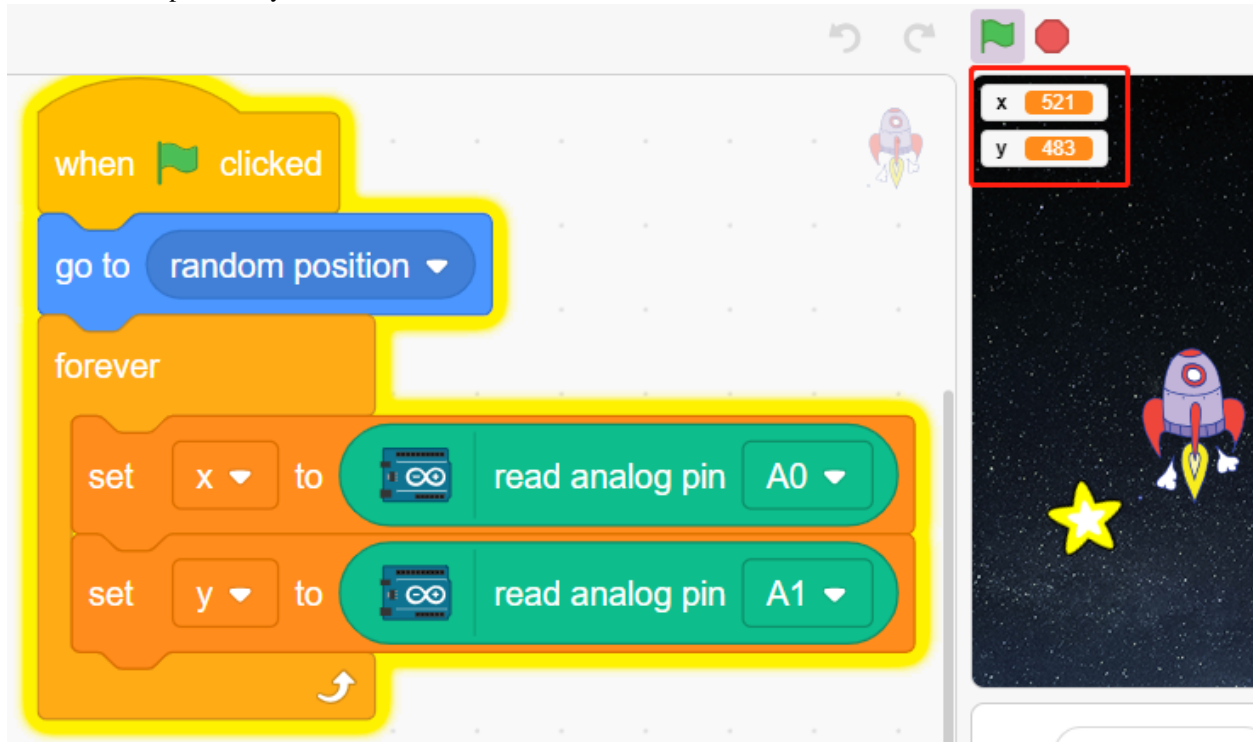


## 2. Scripter pour le Vaisseau Spatial

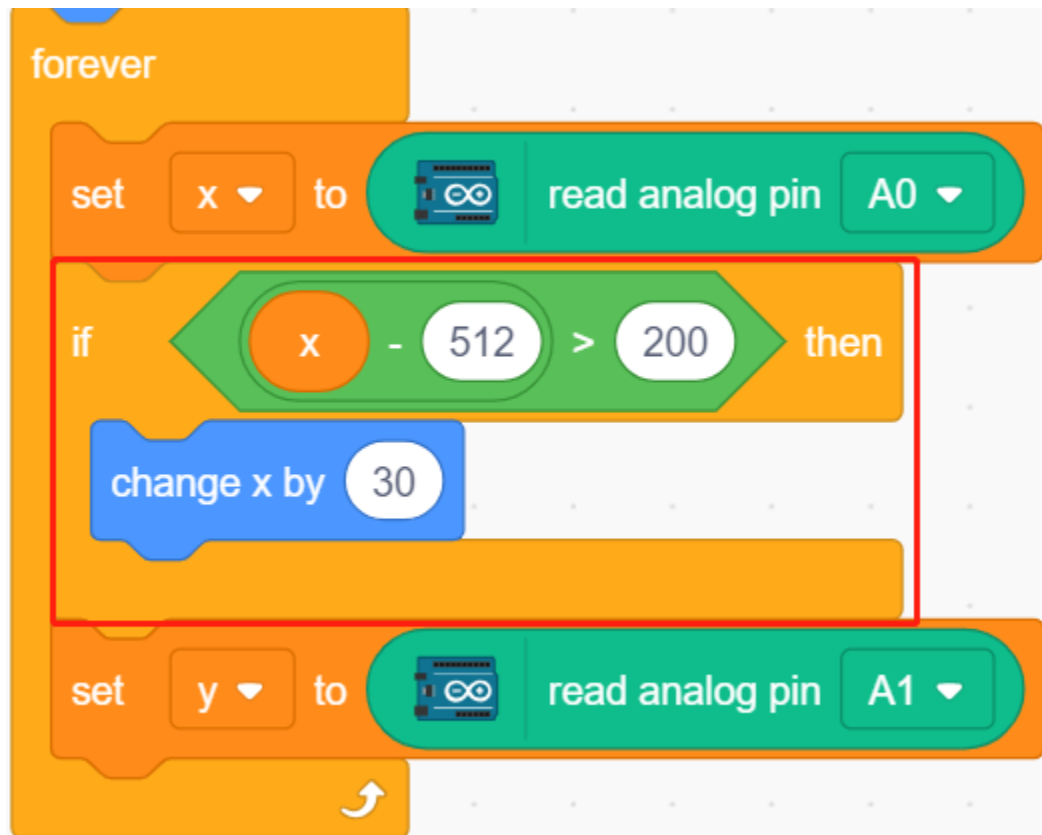
Le sprite **Rocketship** doit donner l'effet qu'il apparaîtra à une position aléatoire et sera ensuite contrôlé par le joystick pour se déplacer vers le haut, le bas, la gauche et la droite.

Le flux de travail est le suivant.

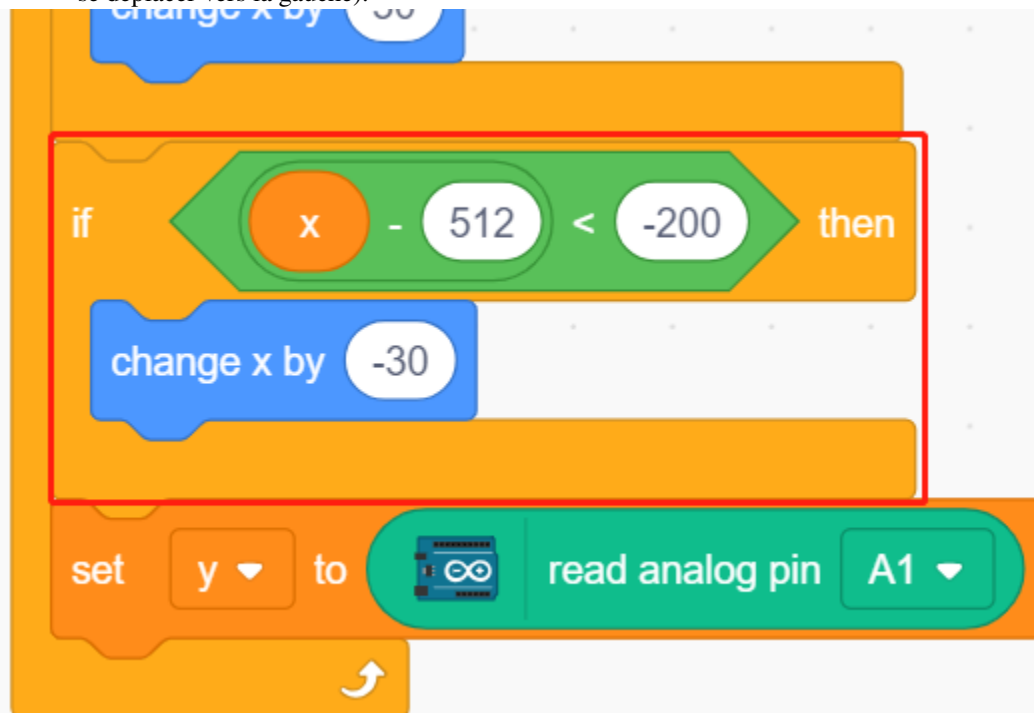
- Lorsque le drapeau vert est cliqué, faites aller le sprite à un endroit aléatoire et créez 2 variables **x** et **y**, qui stockent respectivement les valeurs lues de A0 (VRX du Joystick) et A1 (VRY du Joystick). Vous pouvez laisser le script s'exécuter, basculer le joystick vers le haut et vers le bas, à gauche et à droite, pour voir la plage de valeurs pour x et y.



- La valeur de A0 est dans la plage 0-1023 (le milieu est environ 512). Utilisez  $x - 512 > 200$  pour déterminer si le Joystick bascule vers la droite, et si c'est le cas, faites +30 à la coordonnée x du sprite (pour déplacer le sprite vers la droite).

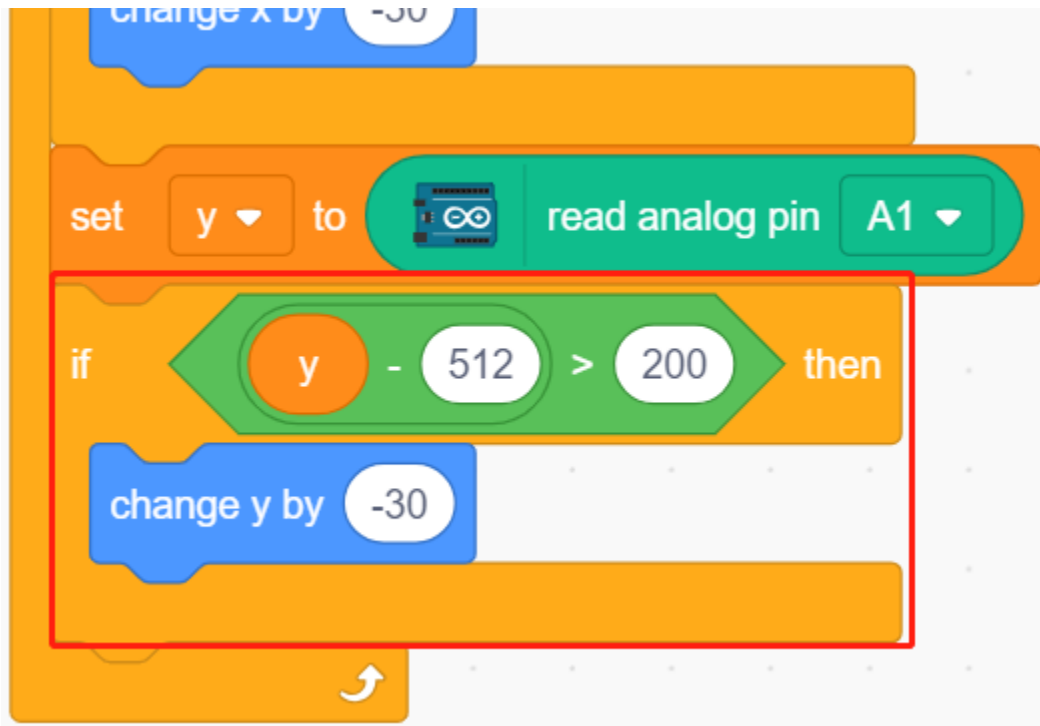


- Si le Joystick bascule vers la gauche ( $x - 512 < -200$ ), laissez la coordonnée x du sprite être -30 (laissez le sprite se déplacer vers la gauche).

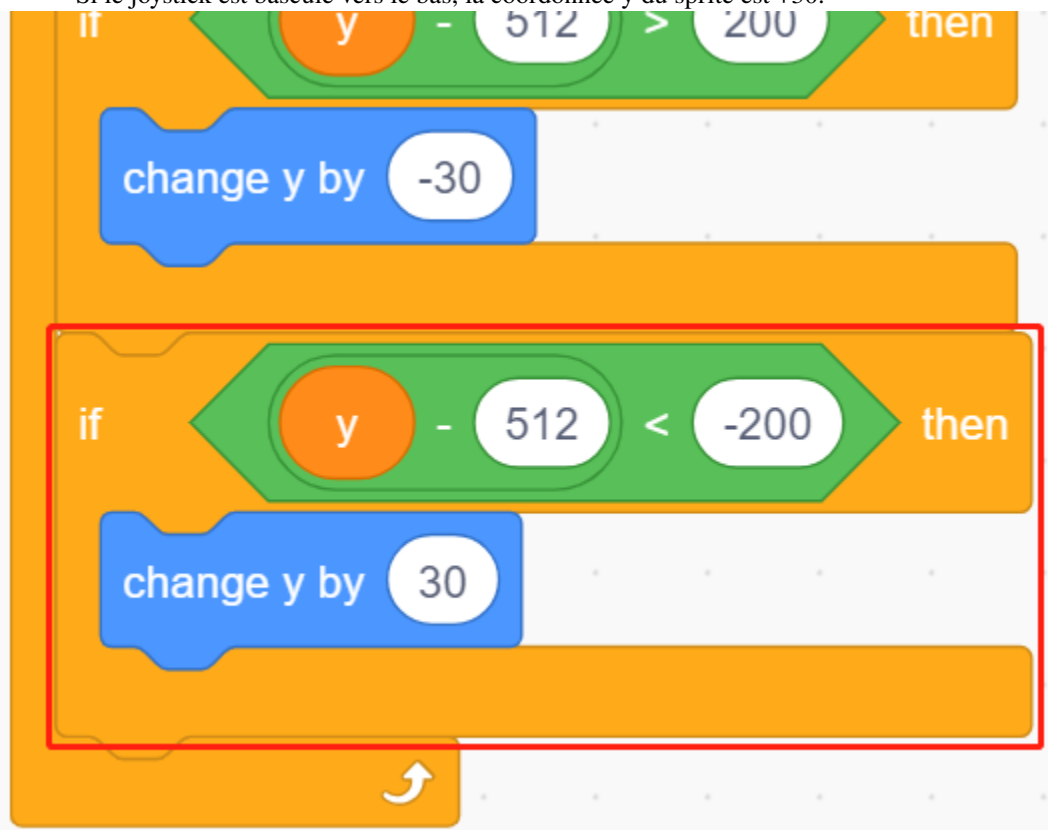


- Comme la coordonnée y du Joystick est de haut (0) à bas (1023), et la coordonnée y du sprite est de bas en haut. Donc, pour déplacer le Joystick vers le haut et le sprite vers le haut, la coordonnée y doit être -30 dans le script.





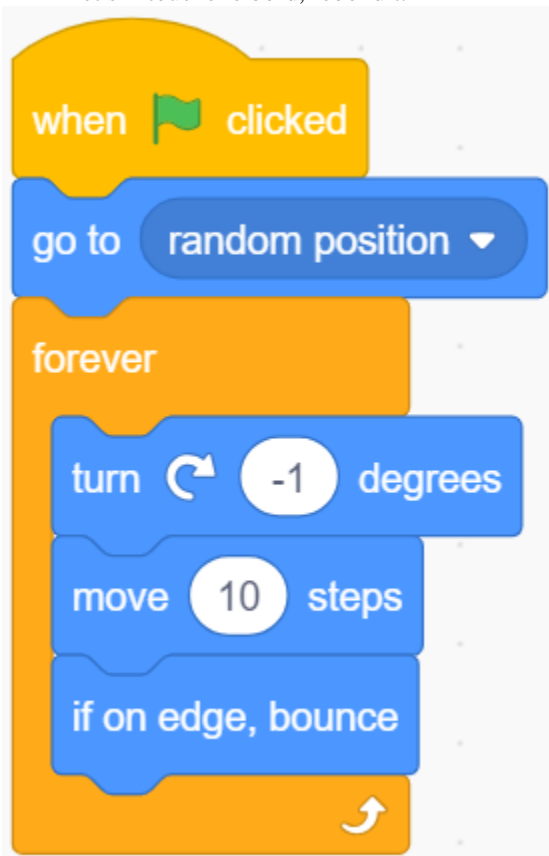
— Si le joystick est basculé vers le bas, la coordonnée y du sprite est +30.



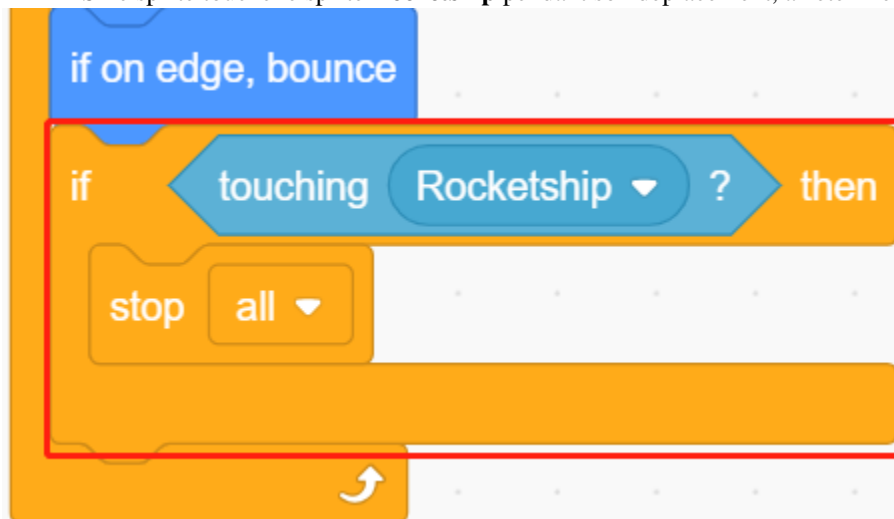
### 3. Scripter pour l'Étoile

L'effet à atteindre par le sprite **Star** est d'apparaître à un endroit aléatoire, et s'il touche **Rocketship**, le script s'arrête de fonctionner et le jeu se termine.

- Lorsque le drapeau vert est cliqué et que le sprite va à un endroit aléatoire, le bloc [turn degrees] est pour faire avancer le sprite **Star** avec un peu de changement d'angle pour que vous puissiez voir qu'il se déplace en courbe et s'il touche le bord, rebondit.



- Si le sprite touche le sprite **Rocketship** pendant son déplacement, arrêtez l'exécution du script.



## 8.19 2.16 JEU - Manger la Pomme

Dans ce projet, nous jouons à un jeu qui utilise un bouton pour contrôler un scarabée afin de manger une pomme.

Lorsque le drapeau vert est cliqué, appuyez sur le bouton et le scarabée tournera, appuyez à nouveau sur le bouton et le scarabée s'arrête de courir et avance dans cet angle. Vous devez contrôler l'angle du scarabée pour qu'il avance sans toucher la ligne noire sur la carte jusqu'à ce qu'il mange la pomme. Si il touche la ligne noire, le jeu est terminé.



### 8.19.1 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

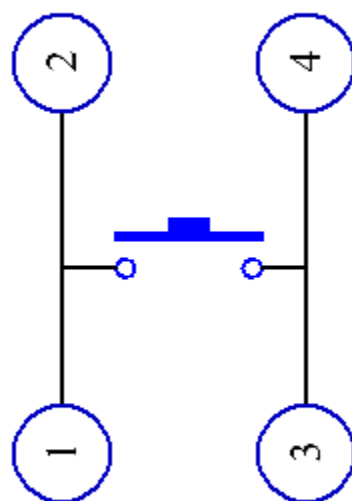
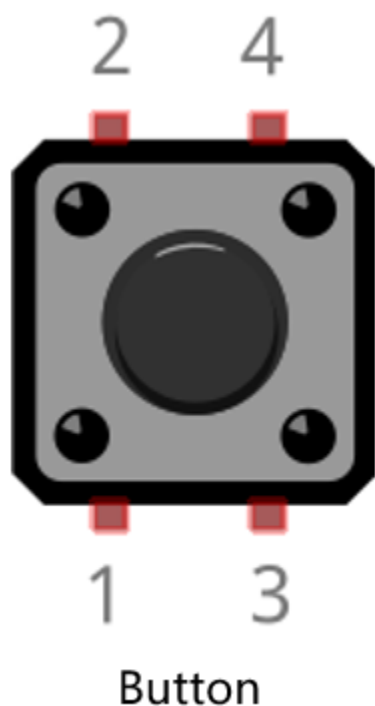
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Condensateur</i>	
<i>Bouton</i>	

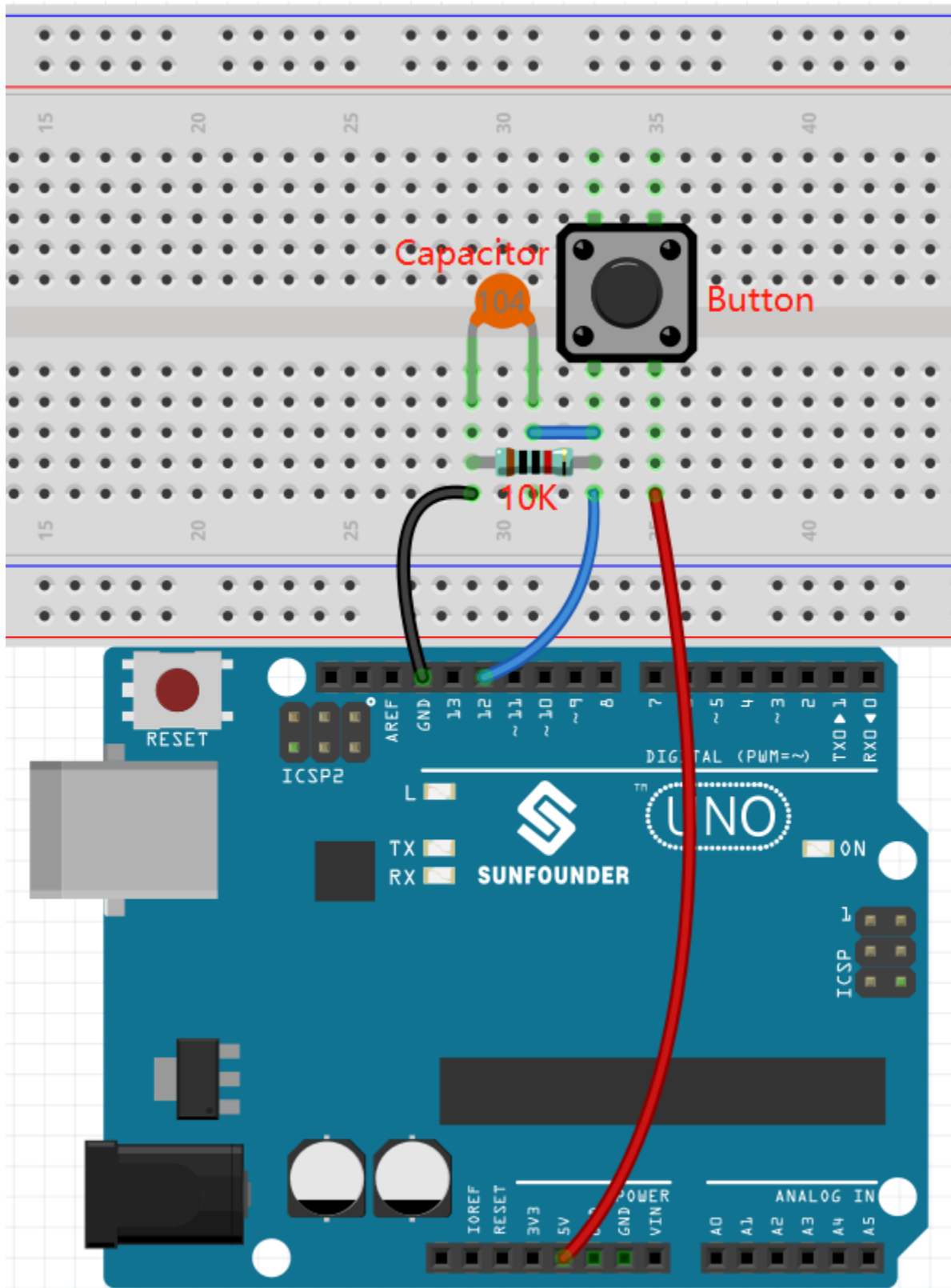
### 8.19.2 Construisez le Circuit

Le bouton est un dispositif à 4 broches, puisque la broche 1 est connectée à la broche 2, et la broche 3 à la broche 4, lorsque le bouton est pressé, les 4 broches sont connectées, fermant ainsi le circuit.



Construisez le circuit selon le schéma suivant.

- Connectez l'une des broches du côté gauche du bouton à la broche 12, qui est connectée à une résistance de tirage et à un condensateur de 0.1uF (104) (pour éliminer le jitter et émettre un niveau stable lorsque le bouton fonctionne).
- Connectez l'autre extrémité de la résistance et du condensateur à GND, et l'une des broches du côté droit du bouton à 5V.



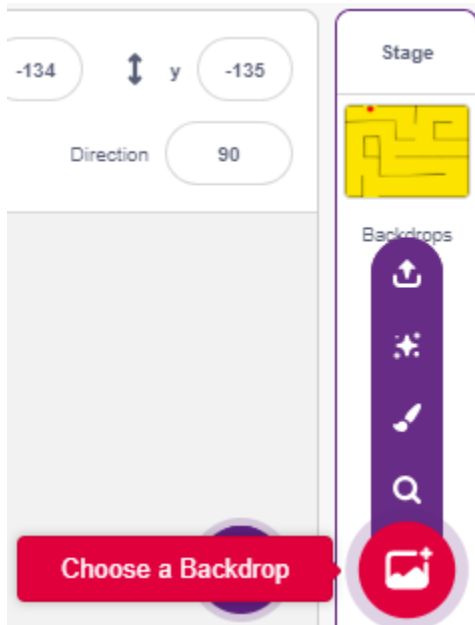
### 8.19.3 Programmation

L'effet que nous voulons atteindre est d'utiliser le bouton pour contrôler la direction du sprite **Beetle** pour avancer et manger la pomme sans toucher la ligne noire sur le décor **Maze**, ce qui changera le décor lorsqu'elle sera mangée.

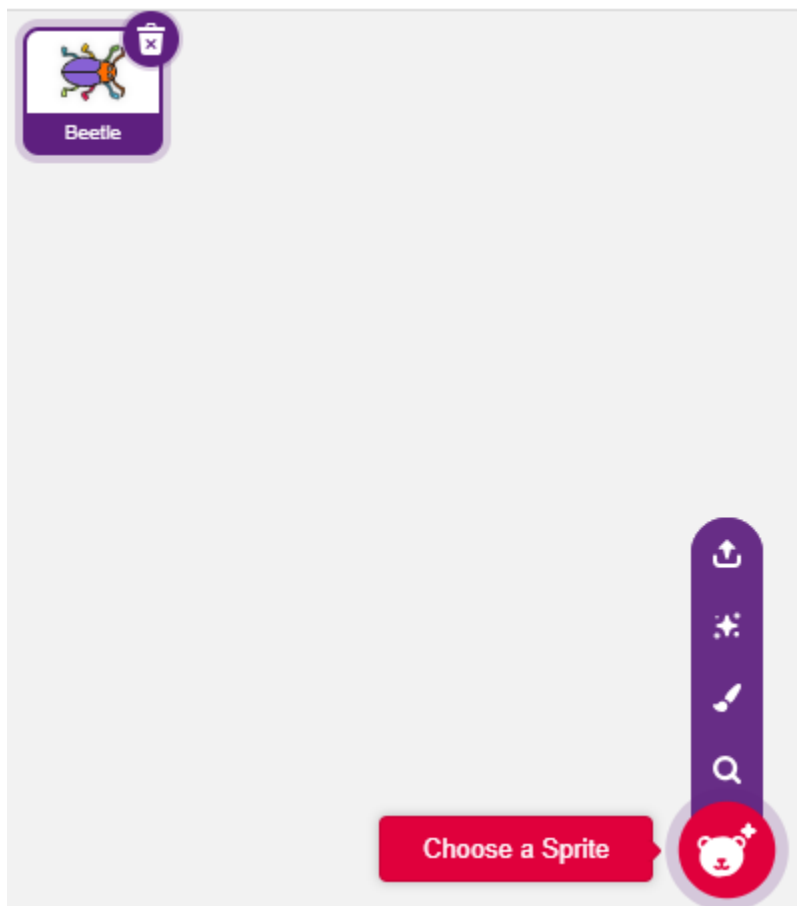
Ajoutez maintenant les décors et sprites pertinents.

#### 1. Ajouter des décors et des sprites

Ajoutez un décor **Maze** via le bouton **Choose a backdrop**.



Supprimez le sprite par défaut, puis sélectionnez le sprite **Beetle**.



Placez le sprite **Beetle** à l'entrée du décor **Maze**, en notant les valeurs des coordonnées x, y à ce point, et redimensionnez le sprite à 40%.

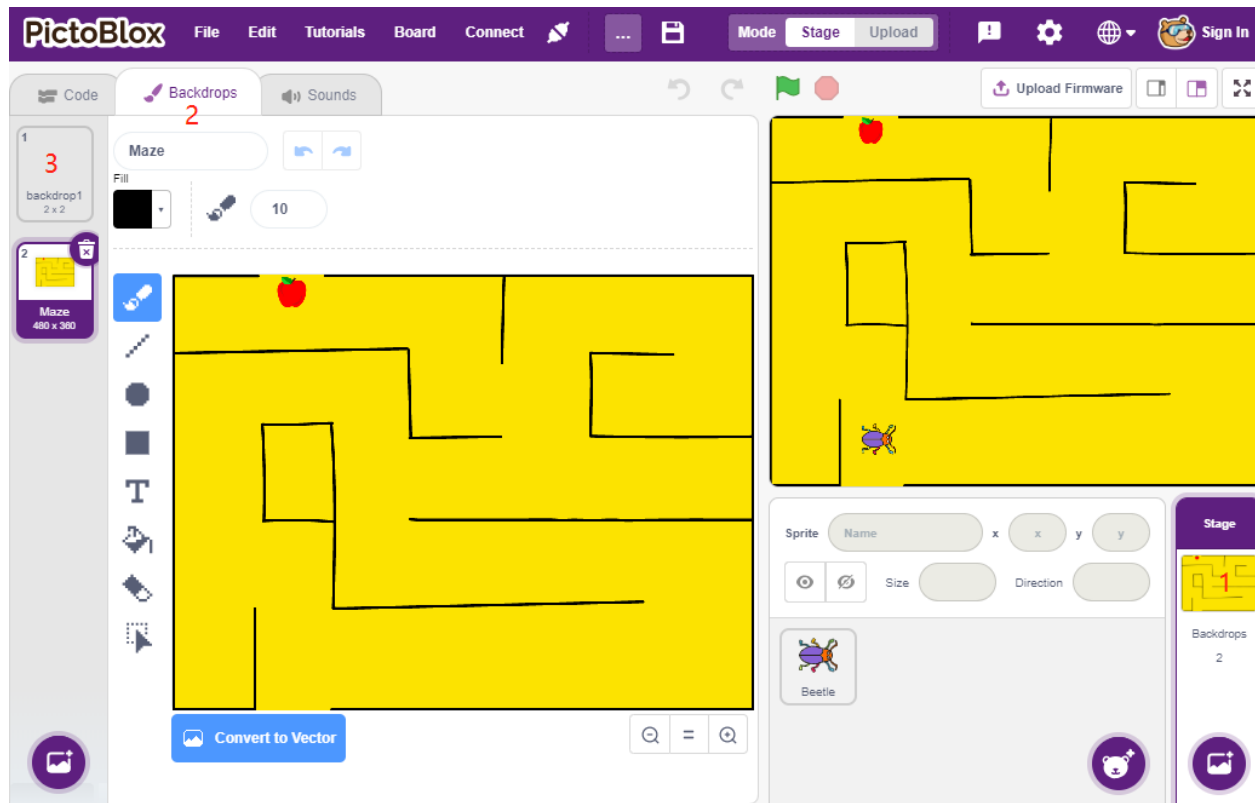


## 2. Dessiner un décor

Il est maintenant temps de dessiner simplement un décor avec le personnage GAGNÉ ! apparaissant dessus.

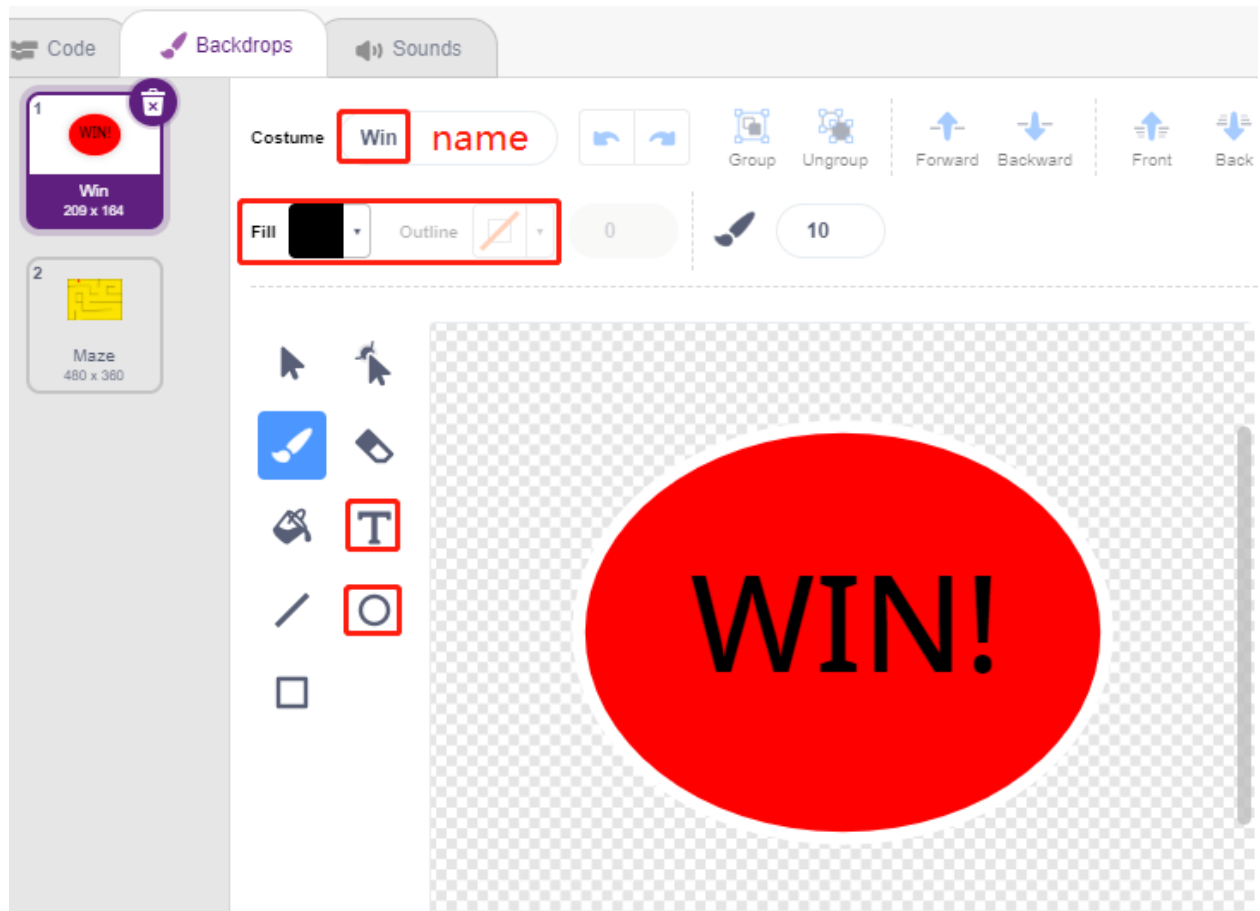
Cliquez d'abord sur la miniature du décor pour aller à la page **Backdrops** et cliquez sur le décor vierge1.





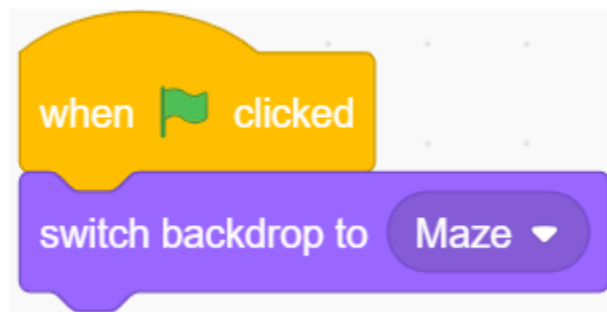
Commencez maintenant à dessiner, vous pouvez vous référer à l'image ci-dessous pour dessiner, ou vous pouvez dessiner un décor par vous-même, tant que l'expression est gagnante.

- Utilisez l'outil **Circle** pour dessiner une ellipse avec la couleur réglée sur rouge et sans contour.
- Ensuite, utilisez l'outil **Text**, écrivez le caractère "GAGNÉ!", réglez la couleur du caractère sur noir et ajustez la taille et la position du caractère.
- Nommez le décor **Win**.



### 3. Scripter pour le décor

Le décor doit être changé en **Maze** à chaque début de jeu.



### 4. Écrire des scripts pour le sprite Scarabée

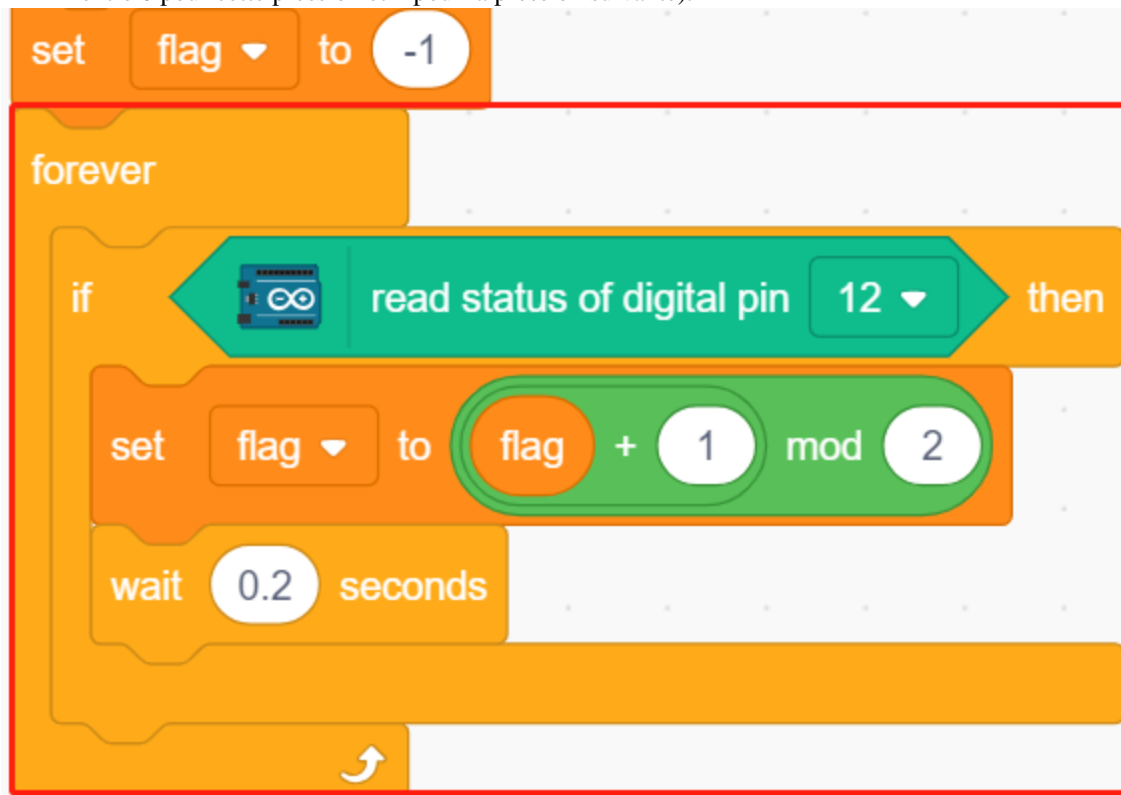
Écrivez maintenant un script pour le sprite **Beetle** pour pouvoir avancer et changer de direction sous le contrôle d'un bouton. Le flux de travail est le suivant.

- Lorsque le drapeau vert est cliqué, réglez l'angle du **Beetle** à 90, et la position à (-134, -134), ou remplacez-la par la valeur de coordonnée de votre propre position placée. Créez la variable **flag** et réglez la valeur initiale à -1.

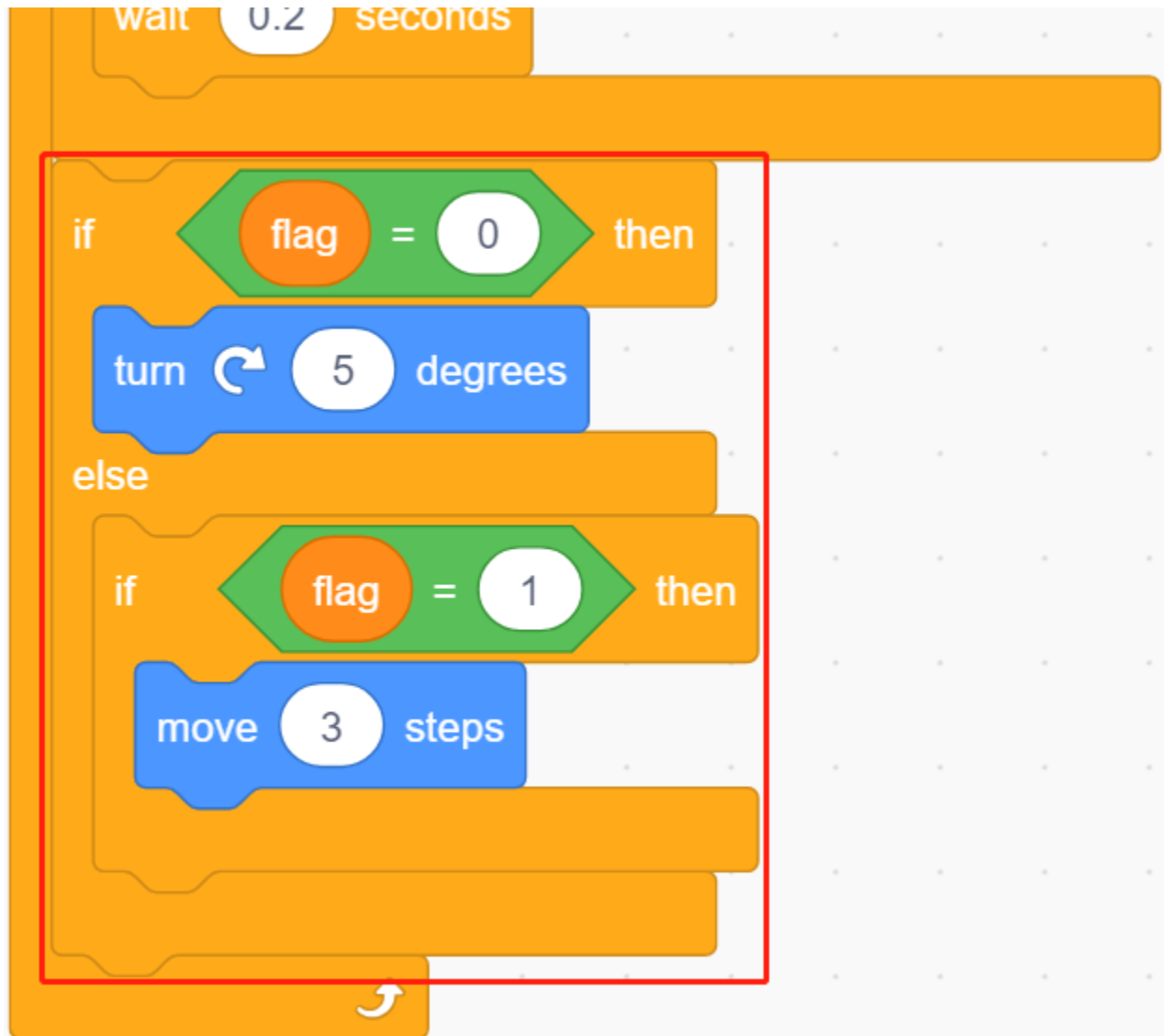


Ensuite, dans le bloc [forever], quatre blocs [if] sont utilisés pour déterminer divers scénarios possibles.

- Si la clé est 1 (pressée), utilisez le bloc [mod] pour basculer la valeur de la variable **flag** entre 0 et 1 (alternant entre 0 pour cette pression et 1 pour la pression suivante).

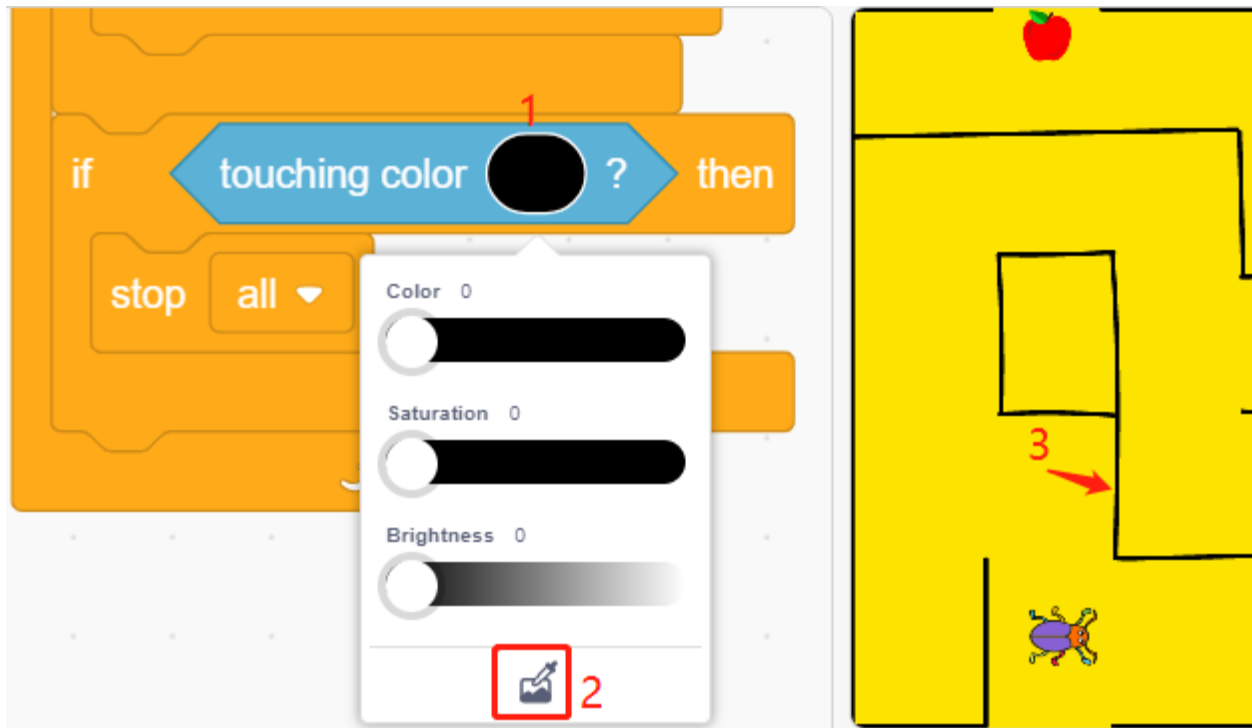


- Si  $\text{flag}=0$  (cette pression de touche), laissez le sprite **Beetle** tourner dans le sens horaire. Puis déterminez si  $\text{flag}$  est égal à 1 (touche pressée à nouveau), le sprite **Beetle** avance. Sinon, il continue à tourner dans le sens horaire.

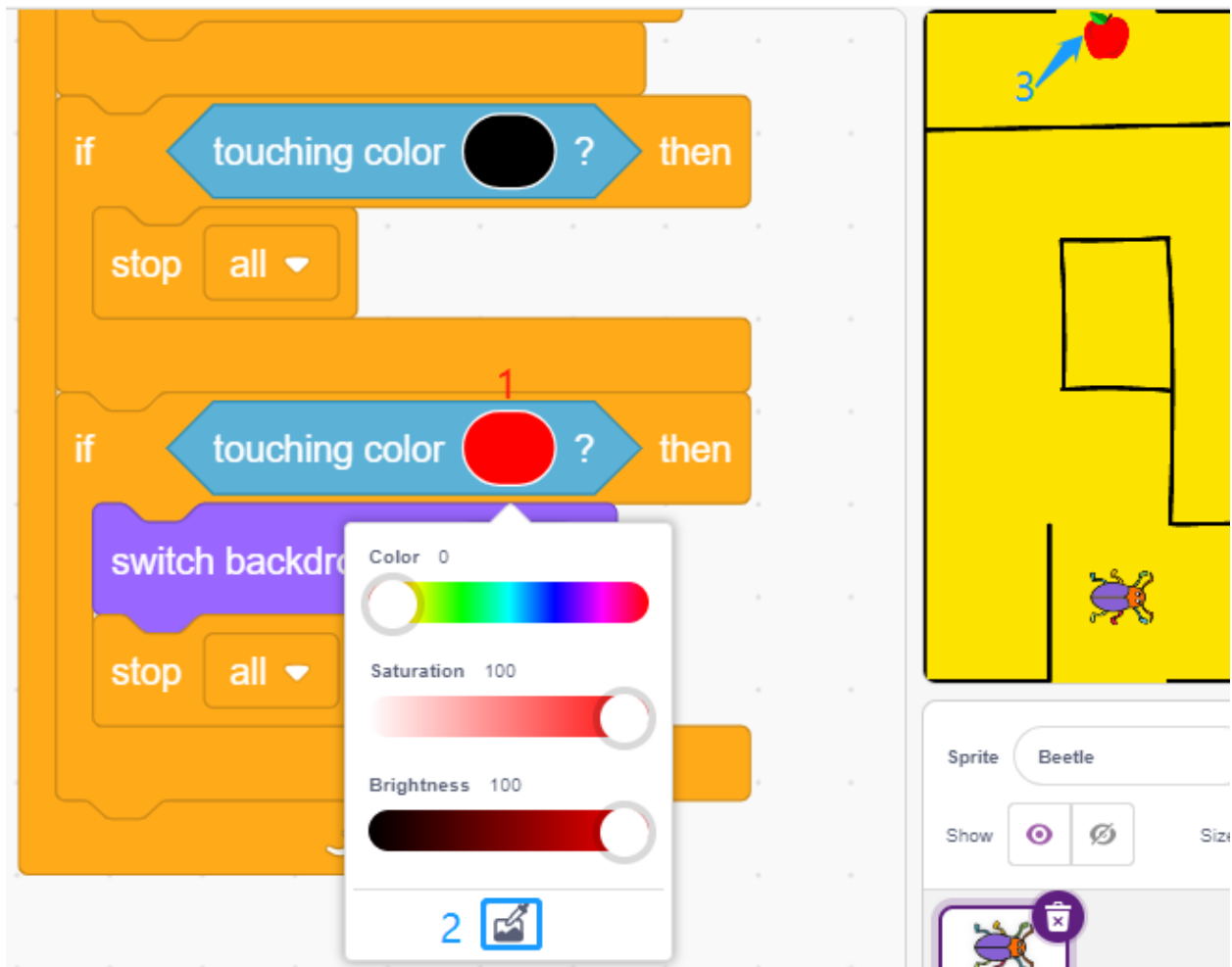


— Si le sprite Scarabée touche le noir (la ligne noire sur le décor **Maze**), le jeu se termine et le script cesse de fonctionner.

**Note :** Vous devez cliquer sur la zone de couleur dans le bloc [Touch color], puis sélectionner l'outil pipette pour prendre la couleur de la ligne noire sur la scène. Si vous choisissez un noir arbitrairement, ce bloc [Touch color] ne fonctionnera pas.



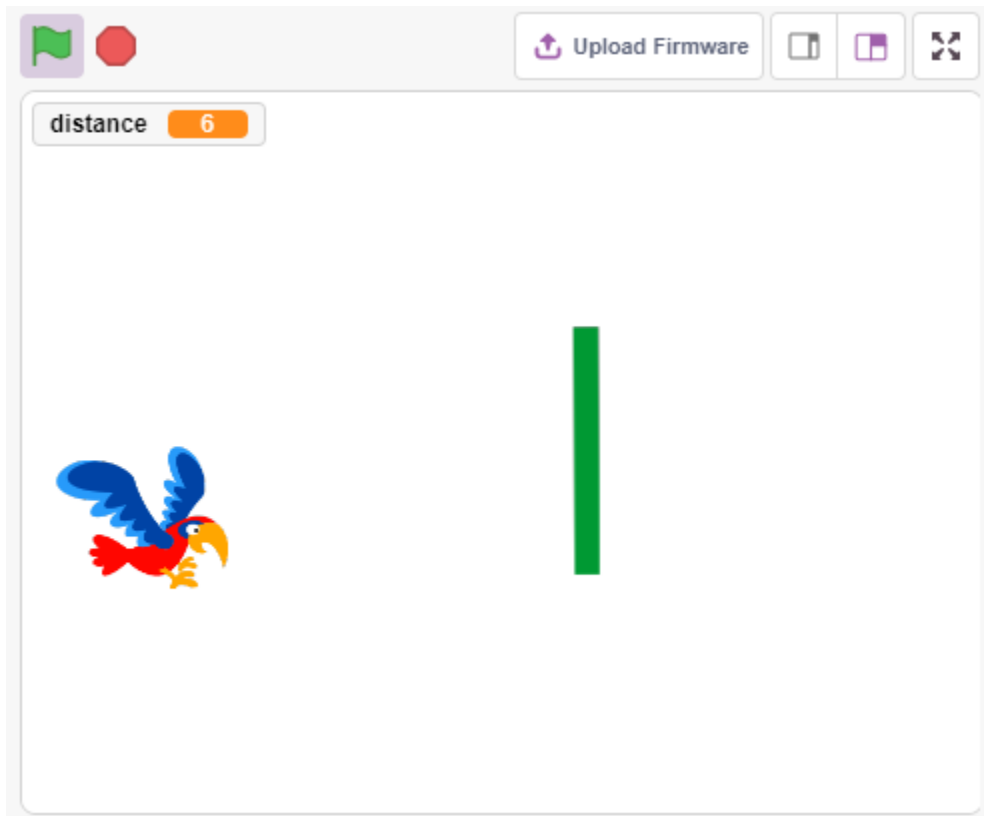
- Si le Scarabée touche le rouge (Utilisez également l'outil paille pour prendre la couleur rouge de la pomme), le décor sera changé en **Win**, ce qui signifie que le jeu réussit et arrête le script de fonctionner.



## 8.20 2.17 JEU - Perroquet Volant

Ici, nous utilisons le module ultrasonique pour jouer à un jeu de perroquet volant.

Après l'exécution du script, le bambou vert se déplacera lentement de la droite vers la gauche à une hauteur aléatoire. Maintenant, placez votre main au-dessus du module ultrasonique, si la distance entre votre main et le module ultrasonique est inférieure à 10, le perroquet s'envolera vers le haut, sinon il tombera vers le bas. Vous devez contrôler la distance entre votre main et le module ultrasonique pour que le Perroquet puisse éviter le bambou vert (Paddle), s'il le touche, le jeu est terminé.



### 8.20.1 Composants Requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

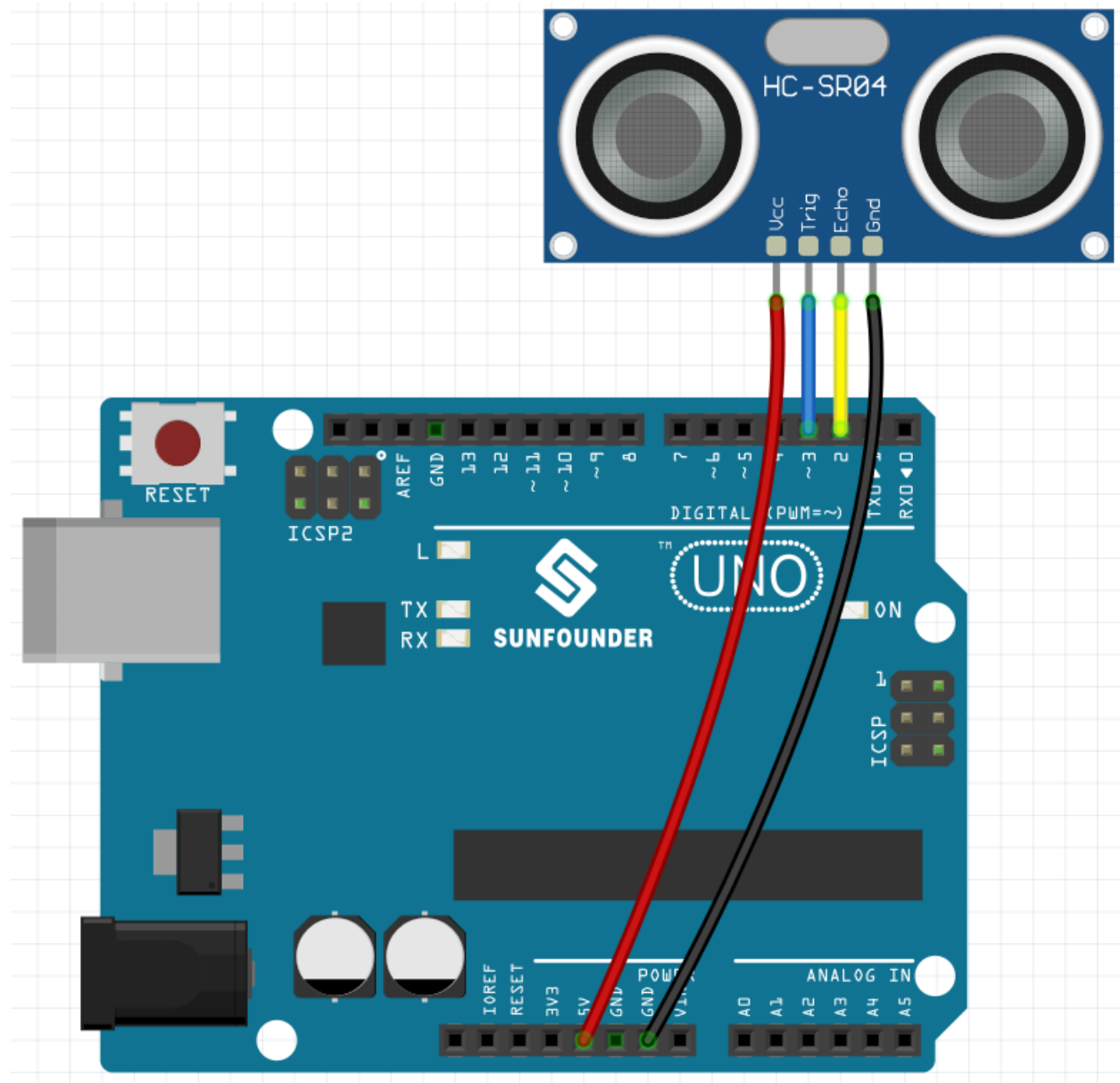
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module Ultrasonique</i>	

### 8.20.2 Construisez le Circuit

Un module capteur ultrasonique est un instrument qui mesure la distance jusqu'à un objet en utilisant des ondes sonores ultrasoniques. Il a deux sondes. L'une sert à envoyer des ondes ultrasoniques et l'autre à recevoir les ondes et à transformer le temps d'envoi et de réception en distance, détectant ainsi la distance entre l'appareil et un obstacle.

Construisez maintenant le circuit selon le schéma suivant.



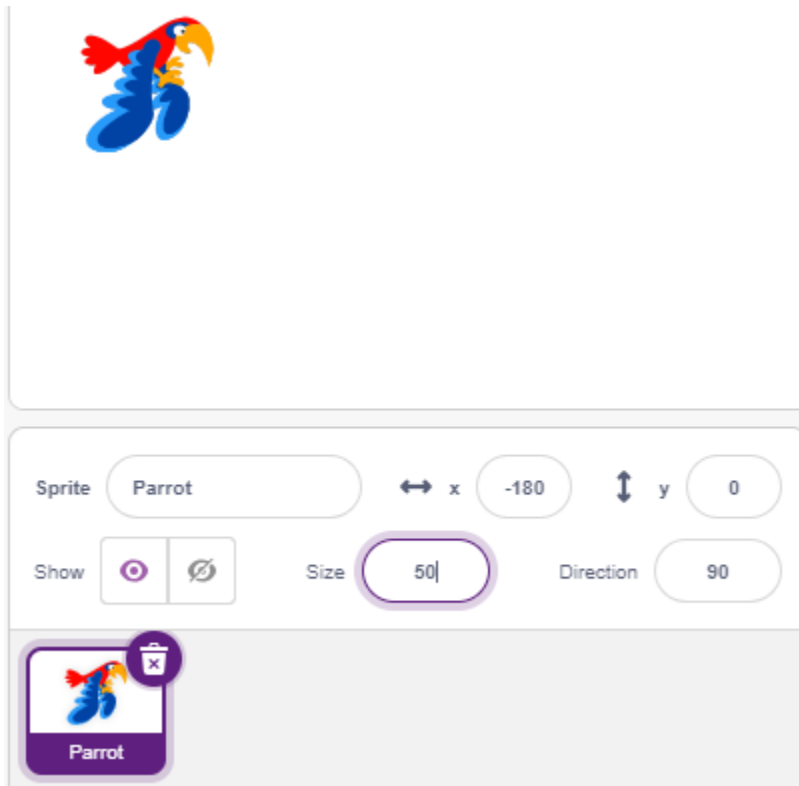


### 8.20.3 Programmation

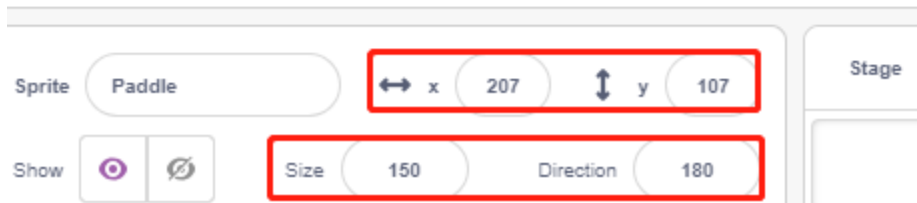
L'effet que nous voulons atteindre est d'utiliser le module ultrasonique pour contrôler la hauteur de vol du sprite **Parrot**, tout en évitant le sprite **Paddle**.

#### 1. Ajouter un sprite

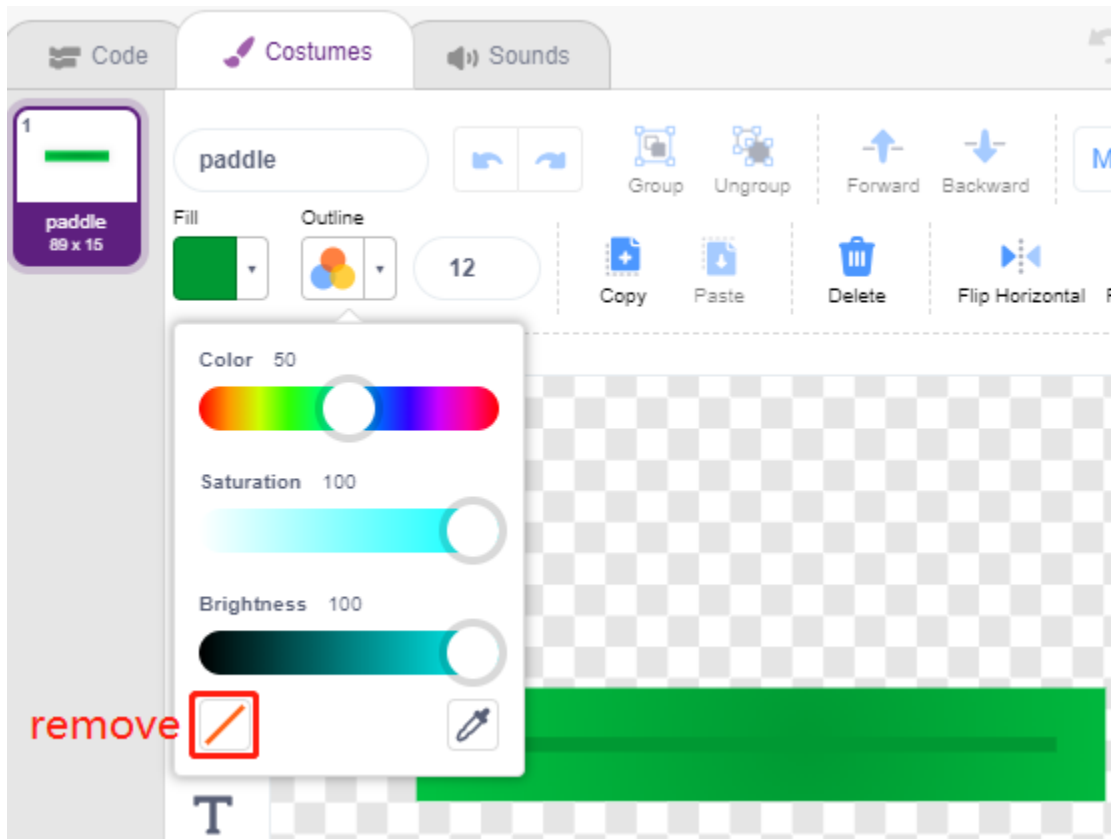
Supprimez le sprite par défaut et utilisez le bouton **Choose a Sprite** pour ajouter le sprite **Parrot**. Réglez sa taille à 50% et déplacez sa position vers le centre gauche.



Ajoutez maintenant le sprite **Paddle**, réglez sa taille à 150%, son angle à 180 et déplacez sa position initiale dans le coin supérieur droit.



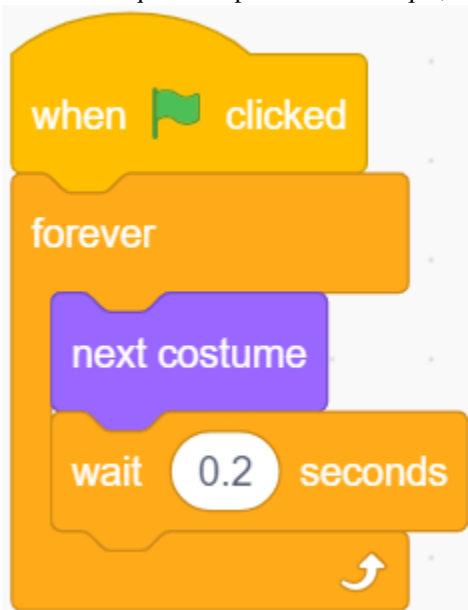
Allez à la page **Costumes** du sprite **Paddle** et supprimez le Contour.



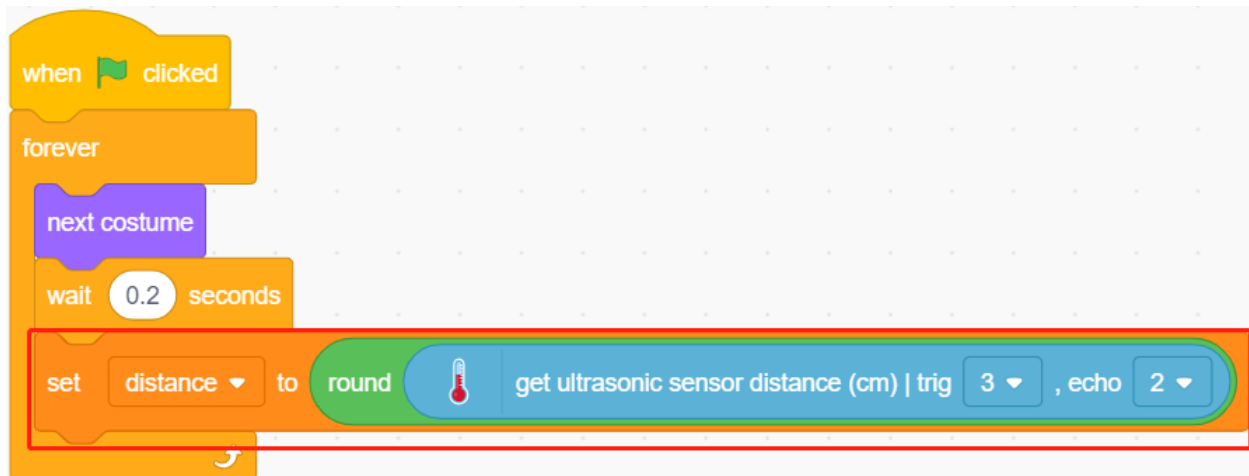
## 2. Scripter pour le sprite Perroquet

Scriptez maintenant le sprite **Parrot**, qui est en vol et dont l'altitude de vol est déterminée par la distance de détection du module ultrasonique.

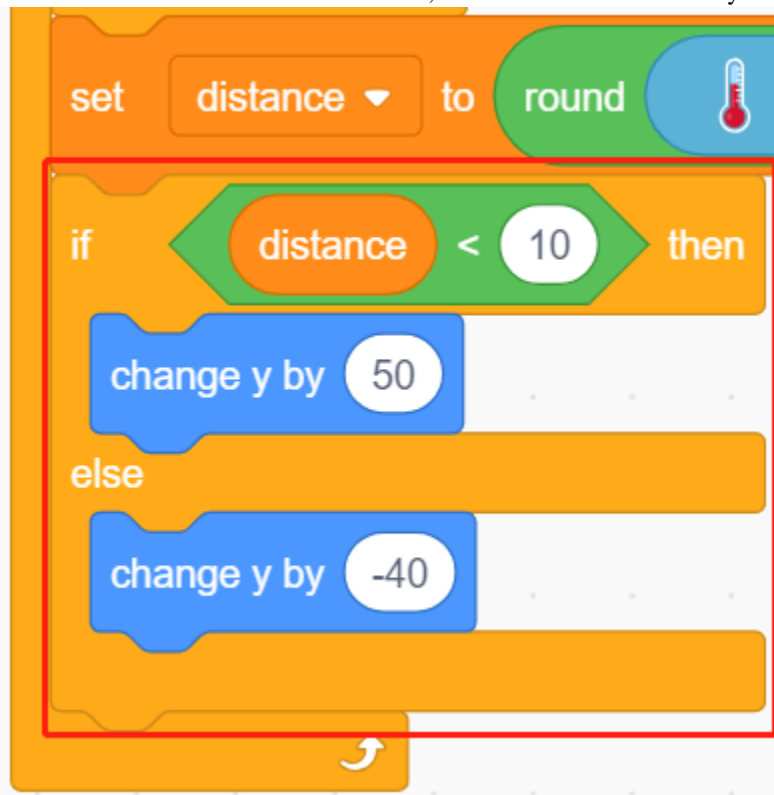
- Lorsque le drapeau vert est cliqué, changez le costume toutes les 0,2 secondes pour qu'il soit toujours en vol.



- Lisez la valeur du module ultrasonique et stockez-la dans la variable **distance** après l'avoir arrondie avec le bloc [round].



- Si la distance de détection ultrasonique est inférieure à 10 cm, laissez la coordonnée y augmenter de 50, le sprite **Parrot** volera vers le haut. Sinon, la valeur de la coordonnée y est diminuée de 40, le **Parrot** tombera.



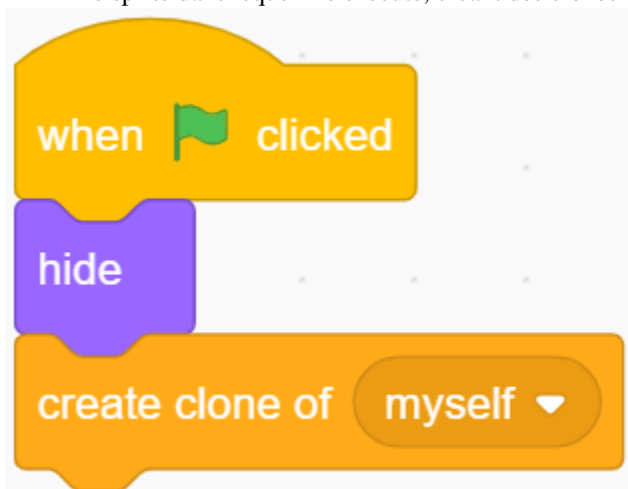
- Si le sprite **Parrot** touche le sprite **Paddle**, le jeu se termine et le script cesse de fonctionner.



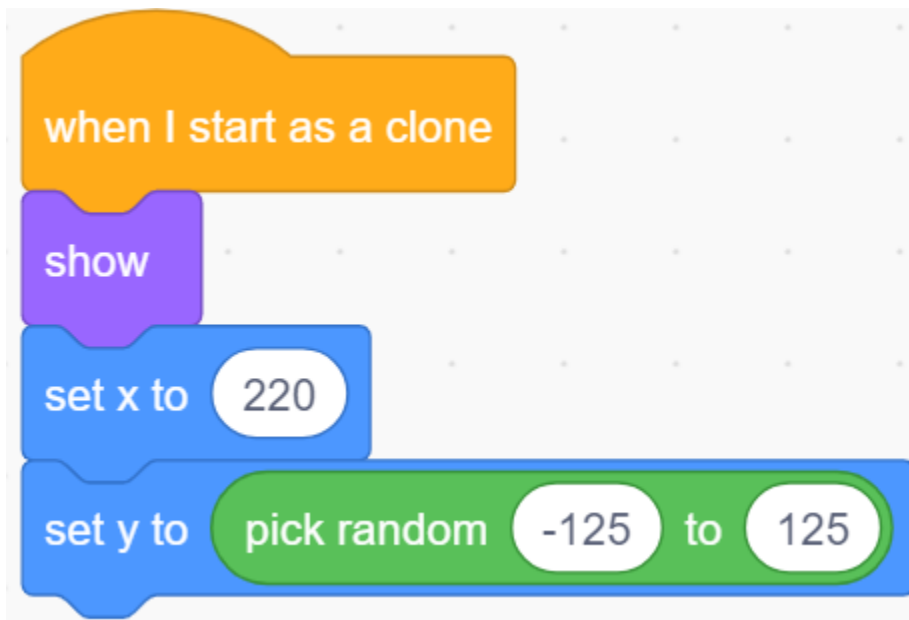
### 3. Scripter pour le sprite Pagaie

Écrivez maintenant le script pour le sprite **Paddle**, qui doit apparaître aléatoirement sur la scène.

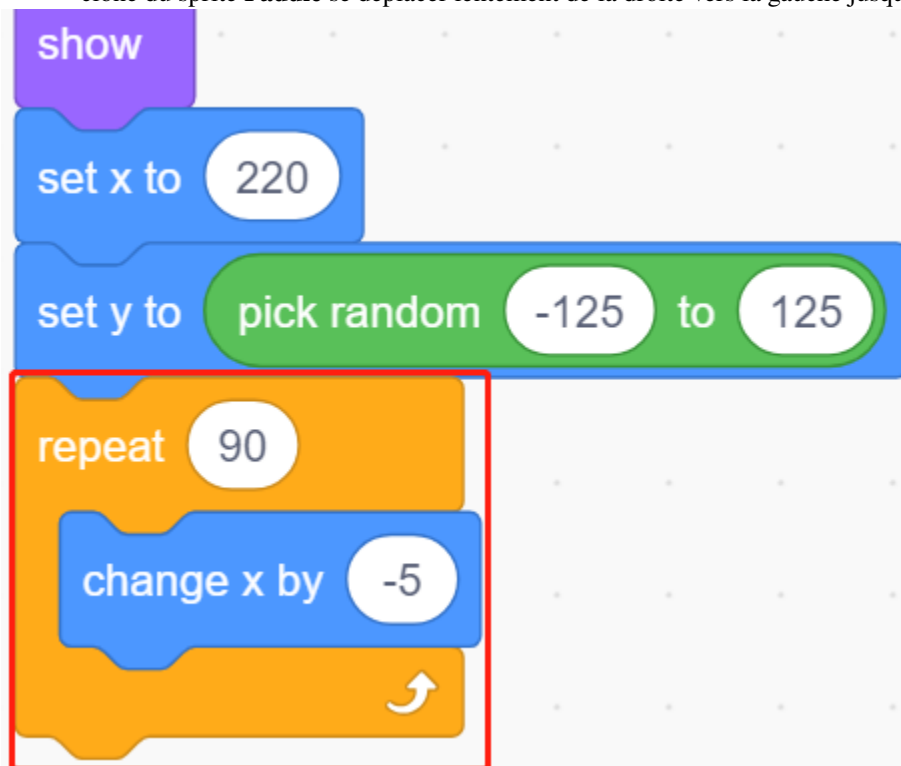
- Cachez le sprite **Paddle** lorsque le drapeau vert est cliqué, et clonez-le en même temps. Le bloc `[create clone of]` est un bloc de contrôle et un bloc empilable. Il crée un clone du sprite dans l'argument. Il peut aussi cloner le sprite dans lequel il s'exécute, créant des clones de clones, de manière récursive.



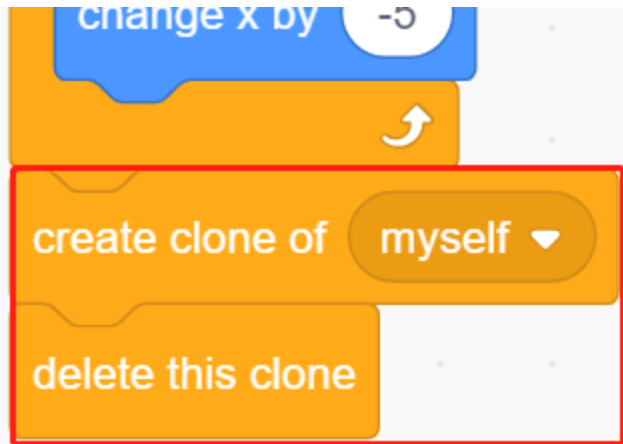
- Lorsque **Paddle** est présenté comme un clone, sa position est de 220 (le plus à droite) pour la coordonnée x et sa coordonnée y à (-125 à 125) aléatoire (hauteur aléatoire).



— Utilisez le bloc [repeat] pour faire diminuer lentement sa valeur de coordonnée x, ainsi vous pouvez voir le clone du sprite **Paddle** se déplacer lentement de la droite vers la gauche jusqu'à ce qu'il disparaisse.



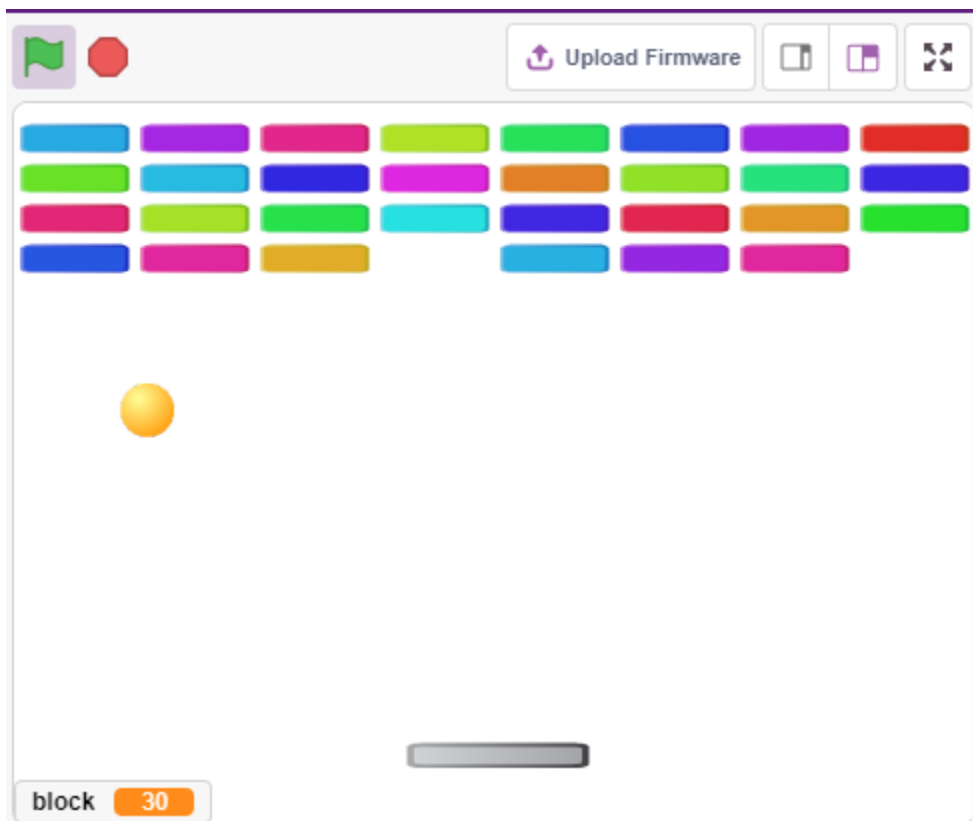
— Reclonez un nouveau sprite **Paddle** et supprimez le clone précédent.



## 8.21 2.18 JEU - Clone de Breakout

Ici, nous utilisons le potentiomètre pour jouer à un jeu clone de Breakout.

Après avoir cliqué sur le drapeau vert, vous devez utiliser le potentiomètre pour contrôler la raquette sur la scène pour attraper la balle afin qu'elle puisse monter et frapper les briques, toutes les briques disparaissent alors le jeu est gagné, si vous ne rattrapez pas la balle, le jeu est perdu.



### 8.21.1 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

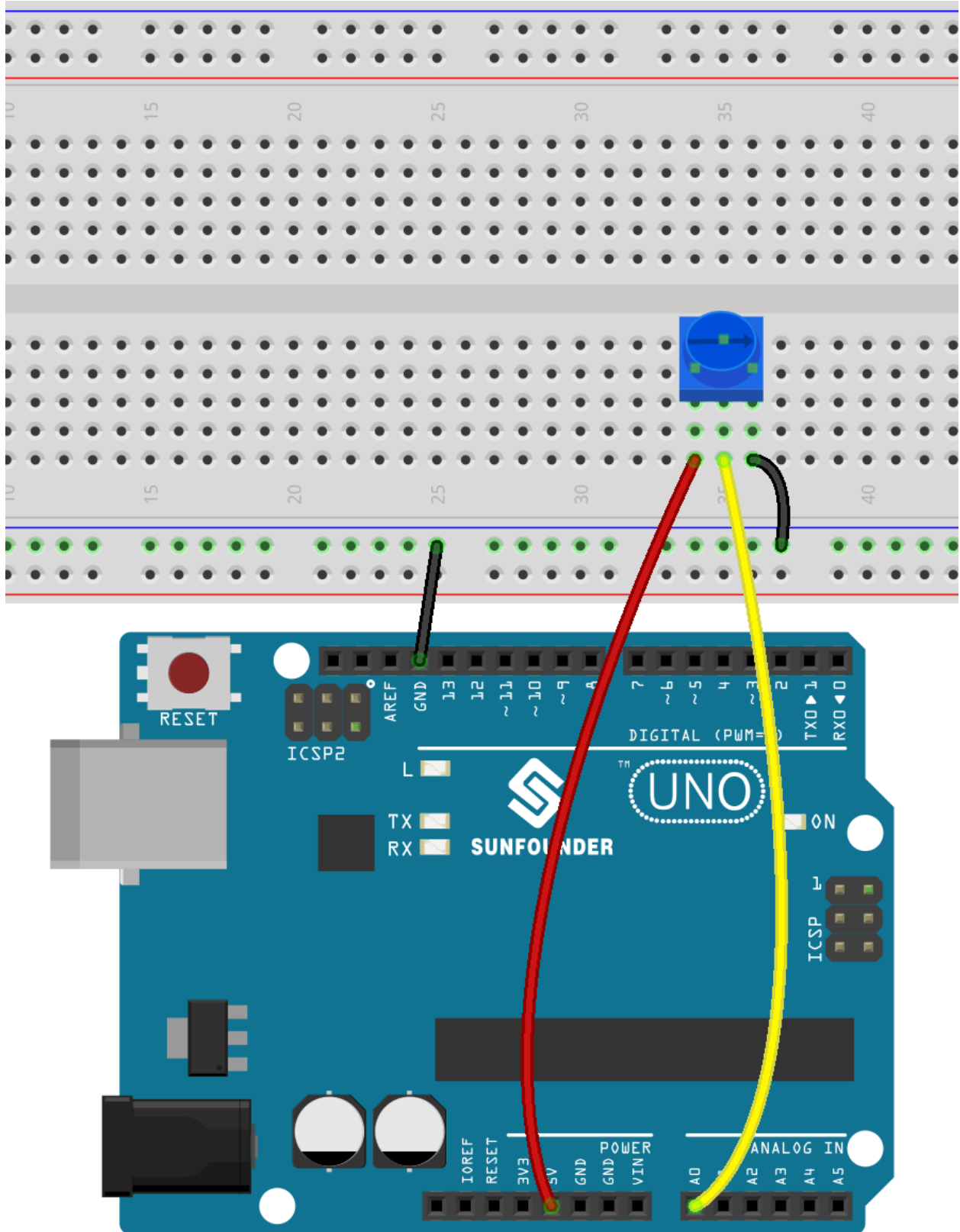
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Potentiomètre</i>	

### 8.21.2 Construisez le Circuit

Le potentiomètre est un élément résistif à 3 bornes, les 2 broches latérales sont connectées à 5V et GND, et la broche centrale est connectée à A0. Après la conversion par le convertisseur ADC de la carte Arduino, la plage de valeurs est de 0-1023.





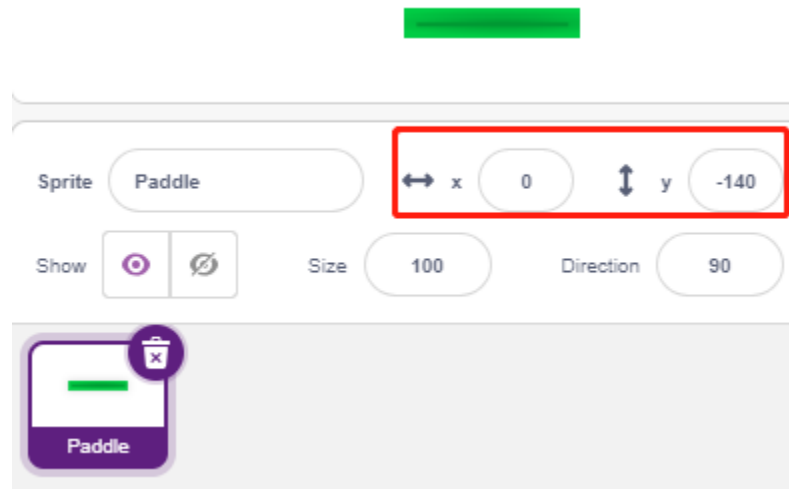
### 8.21.3 Programmation

Il y a 3 sprites sur la scène.

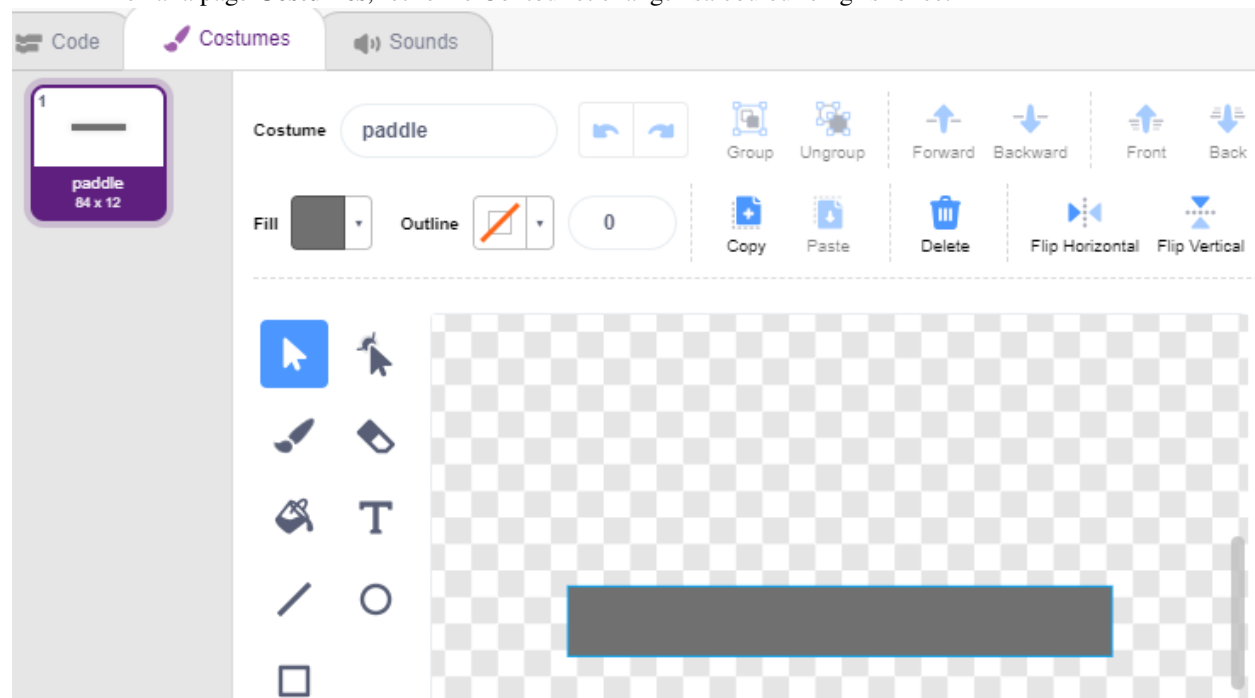
#### 1. Sprite Raquette

L'effet à atteindre par la **Paddle** est que la position initiale est au milieu du bas de la scène, et elle est contrôlée par un potentiomètre pour se déplacer vers la gauche ou vers la droite.

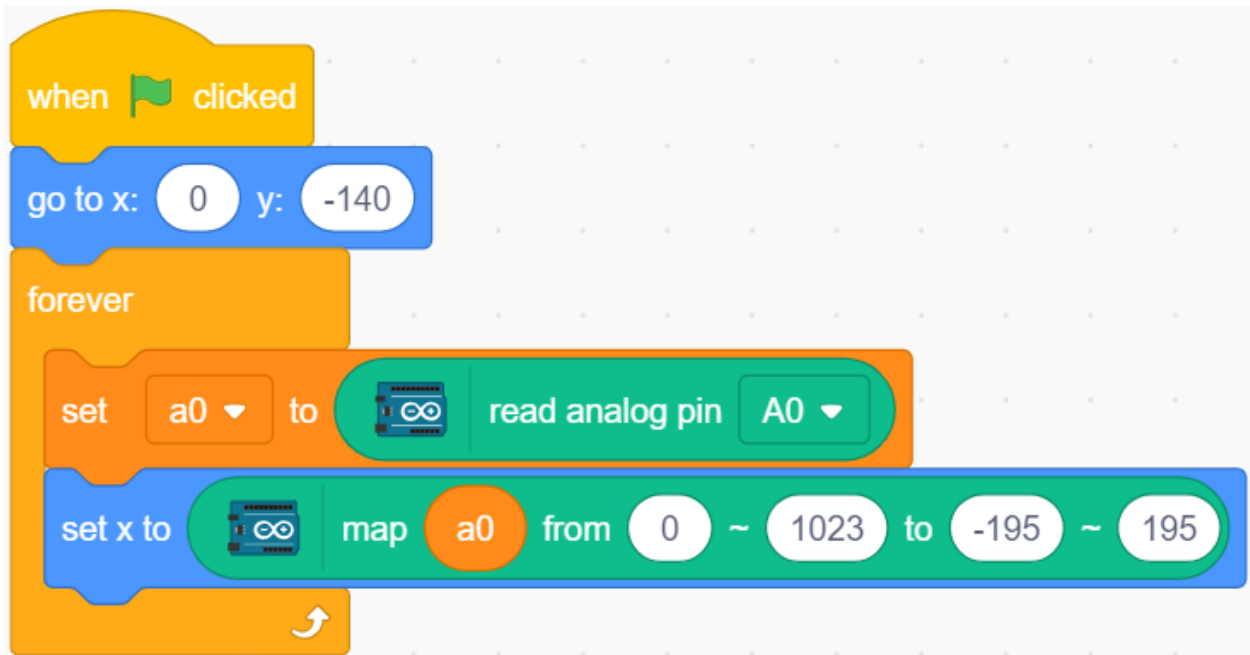
- Supprimez le sprite par défaut, utilisez le bouton **Choose a Sprite** pour ajouter le sprite **Paddle**, et réglez ses x et y à (0, -140).



- Allez à la page **Costumes**, retirez le Contour et changez sa couleur en gris foncé.



- Scriptez maintenant le sprite **Paddle** pour définir sa position initiale à (0, -140) lorsque le drapeau vert est cliqué, et lisez la valeur de A0 (potentiomètre) dans la variable **a0**. Comme le sprite **Paddle** se déplace de gauche à droite sur la scène aux coordonnées x -195~195, vous devez utiliser le bloc [map] pour mapper la plage de la variable **a0** de 0~1023 à -195~195.

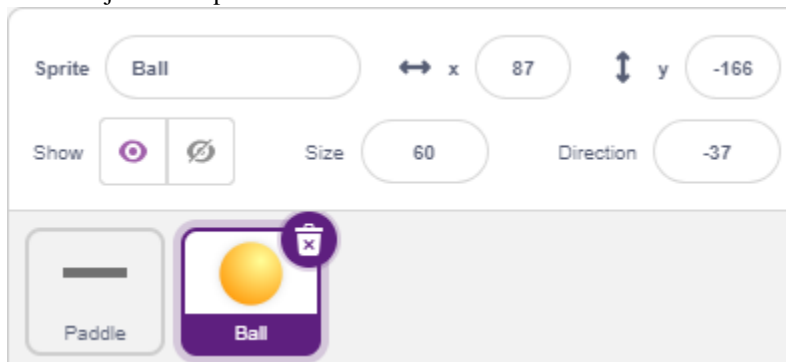


- Maintenant, vous pouvez tourner le potentiomètre pour voir si la **Paddle** peut se déplacer à gauche et à droite sur la scène.

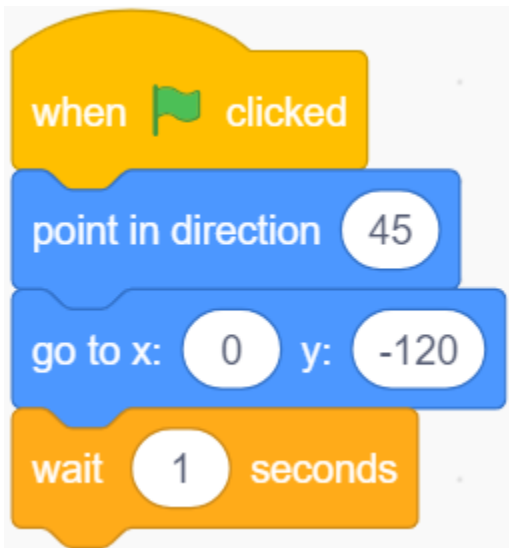
## 2. Sprite Balle

L'effet du sprite balle est qu'il se déplace autour de la scène et rebondit lorsqu'il touche le bord ; il rebondit vers le bas s'il touche le bloc au-dessus de la scène ; il rebondit vers le haut s'il touche le sprite Raquette pendant sa chute ; si ce n'est pas le cas, le script cesse de fonctionner et le jeu se termine.

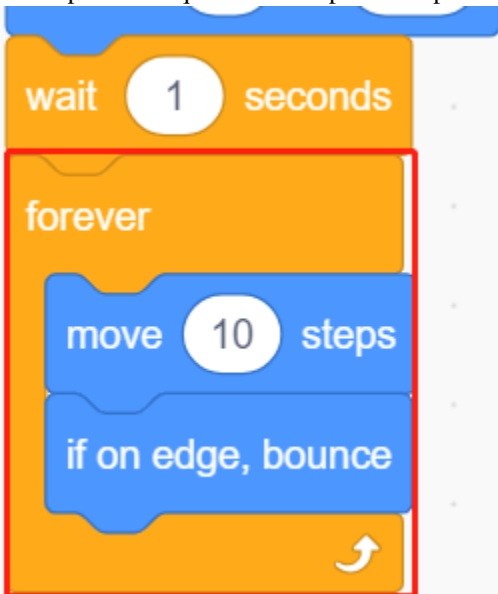
- Ajoutez le sprite **Ball**.



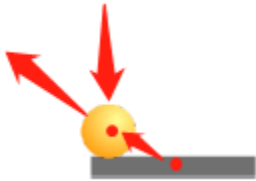
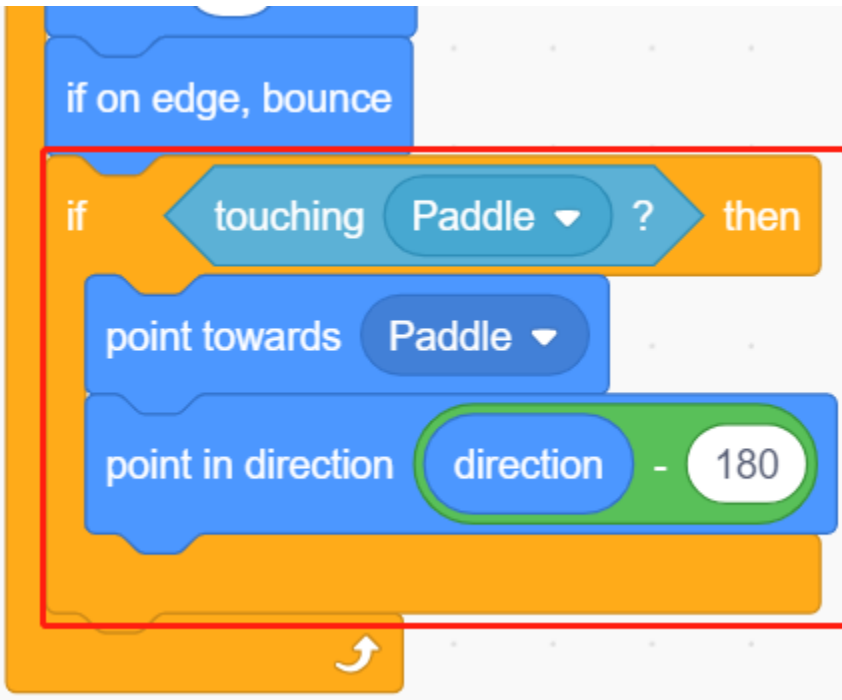
- Lorsque le drapeau vert est cliqué, réglez l'angle du sprite **Ball** à 45° et positionnez-le initialement à (0, -120).



- Laissez maintenant le sprite **Ball** se déplacer autour de la scène et rebondir lorsqu'il touche le bord, et vous pouvez cliquer sur le drapeau vert pour voir l'effet.



- Lorsque le sprite **Ball** touche le sprite **Paddle**, faites un rebond. La façon simple de faire cela est de laisser l'angle être directement inversé, mais alors vous trouverez que la trajectoire de la balle est complètement fixe, ce qui est trop ennuyeux. Par conséquent, nous utilisons le centre des deux sprites pour calculer et faire rebondir la balle dans la direction opposée au centre de la raquette.



— Lorsque le sprite **Ball** tombe au bord de la scène, le script cesse de fonctionner et le jeu se termine.



### 3. Sprite Bloc1

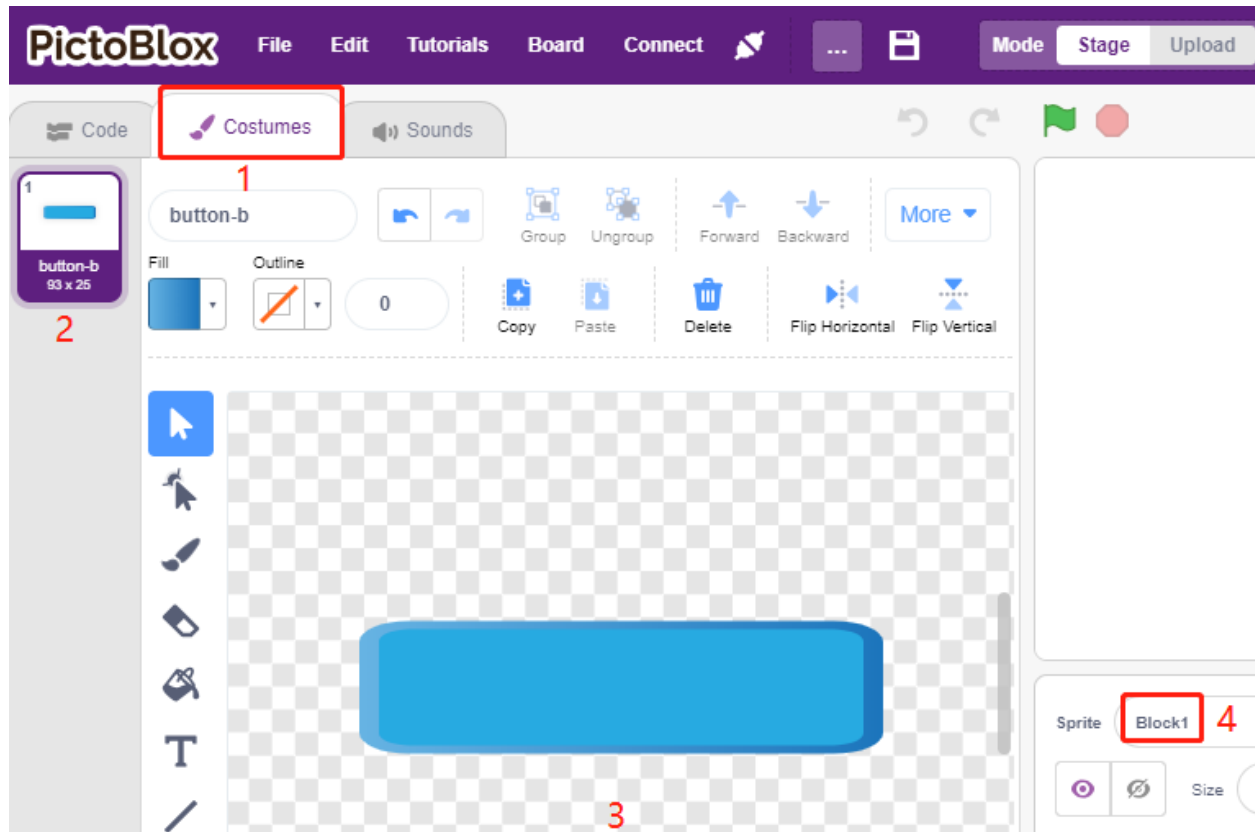
Le sprite **Block1** doit apparaître avec l'effet de clonage 4x8 de lui-même au-dessus de la scène dans une couleur aléatoire, et supprimer un clone s'il est touché par le sprite **Ball**.

Le sprite **Block1** n'est pas disponible dans la bibliothèque **PictoBlox**, vous devez le dessiner vous-même ou le modifier à partir d'un sprite existant. Ici, nous allons le modifier avec le sprite **Button3**.

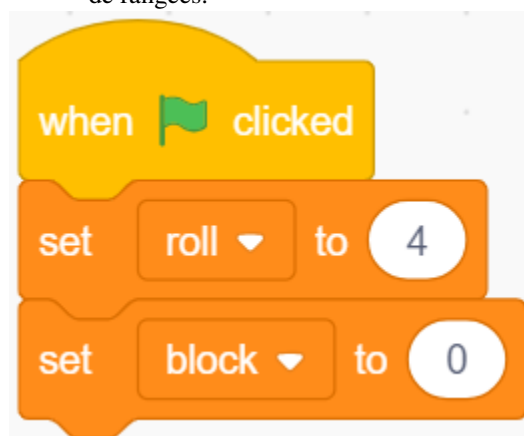
- Après avoir ajouté le sprite **Button3**, allez à la page **Costumes**. Maintenant, supprimez d'abord **button-a**, puis réduisez à la fois la largeur et la hauteur de **button-b**, et changez le nom du sprite en **Block1**, comme indiqué dans l'image suivante.

**Note :**

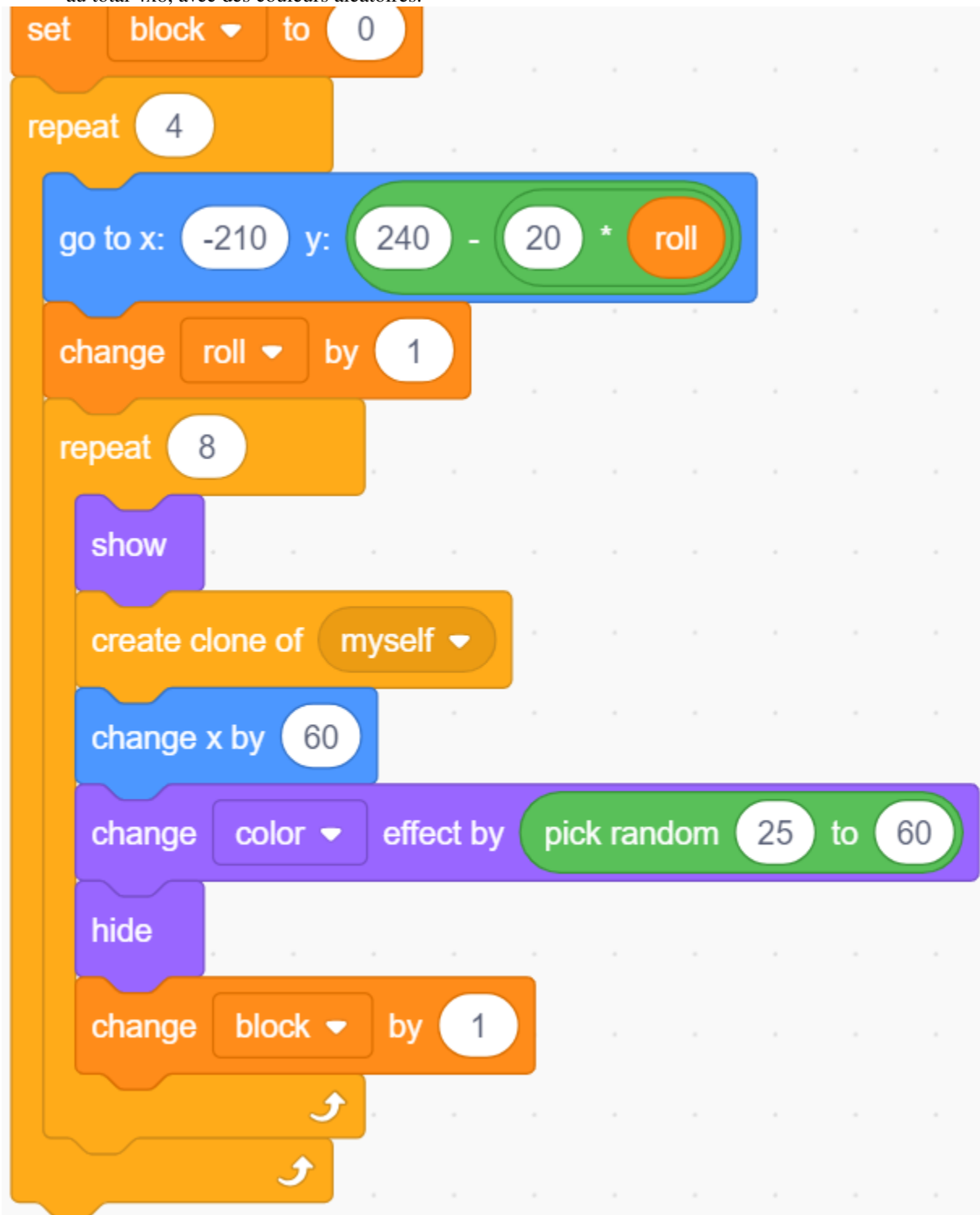
- Pour la largeur de **Block1**, vous pouvez probablement la simuler sur l'écran pour voir si vous pouvez en mettre 8 de suite, sinon, réduisez la largeur de manière appropriée.
- Lors du rétrécissement du sprite **Block1**, vous devez garder le point central au milieu du sprite.



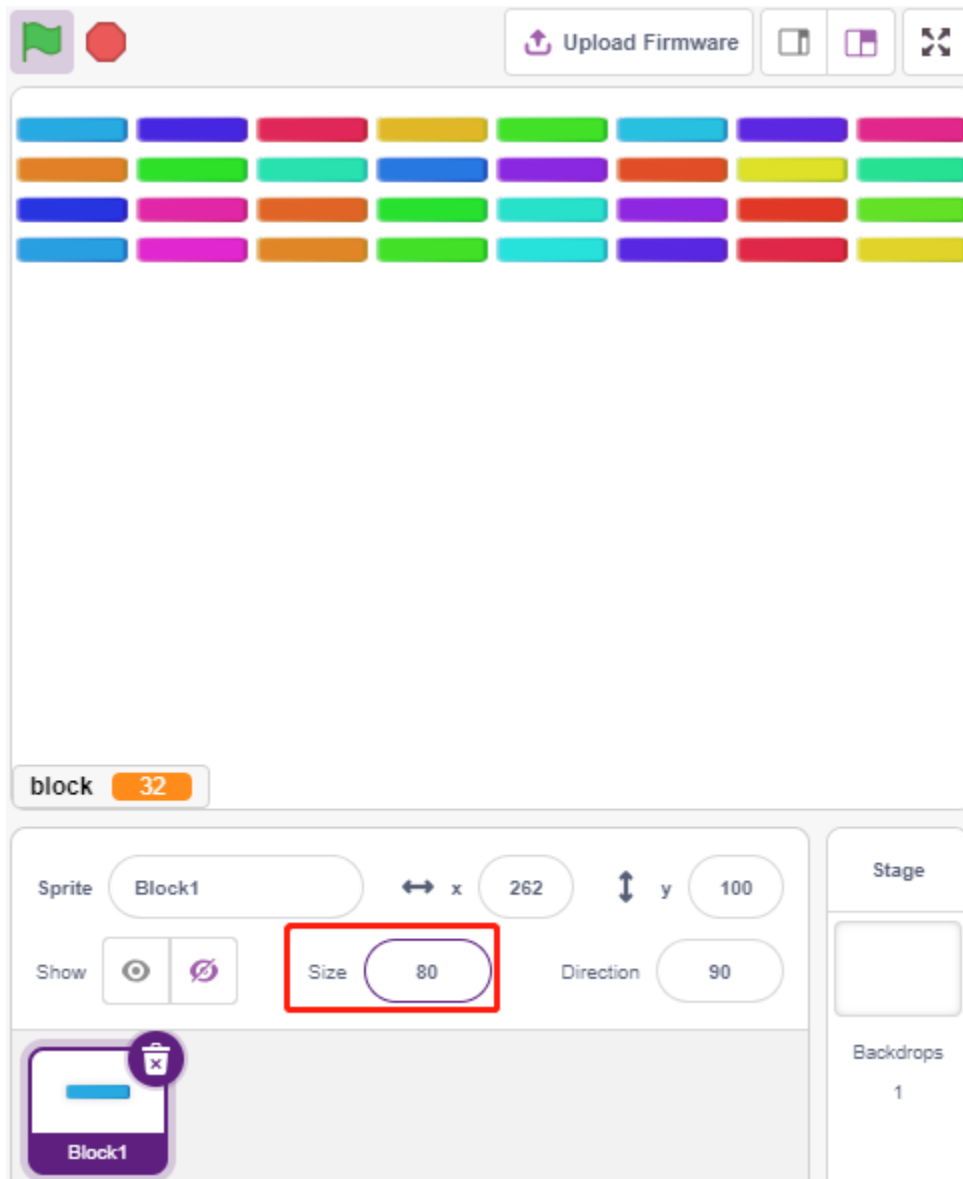
- Créez maintenant 2 variables en premier, **block** pour stocker le nombre de blocs et **roll** pour stocker le nombre de rangées.



- Nous devons faire un clone du sprite **Block1**, pour qu'il s'affiche de gauche à droite, de haut en bas, un par un, au total 4x8, avec des couleurs aléatoires.

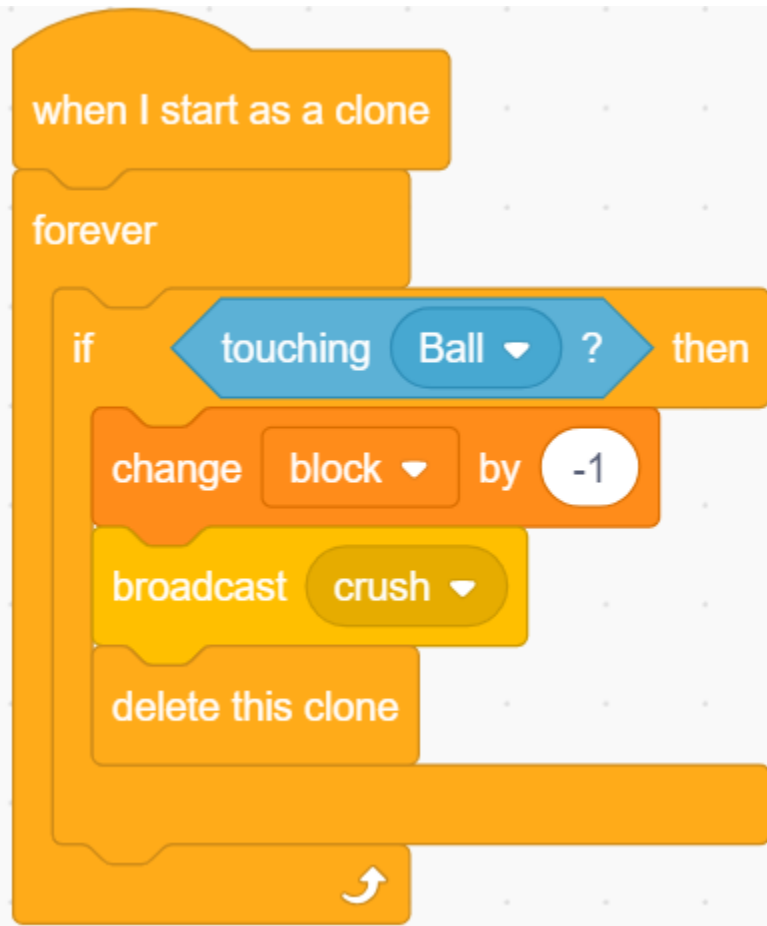


- Après avoir écrit le script, cliquez sur le drapeau vert et regardez l'affichage sur la scène, si c'est trop compact ou trop petit, vous pouvez changer la taille.

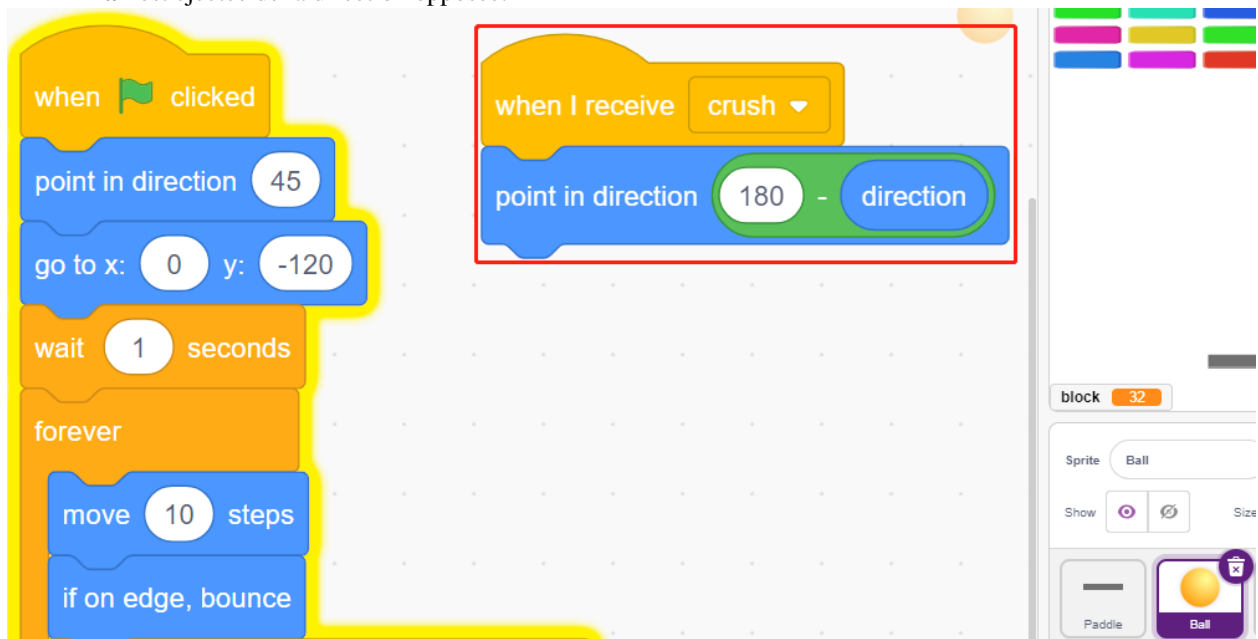


- Écrivez maintenant l'événement déclencheur. Si le sprite cloné **Block1** touche le sprite **Balle**, supprimez le clone et diffusez le message **crush**.





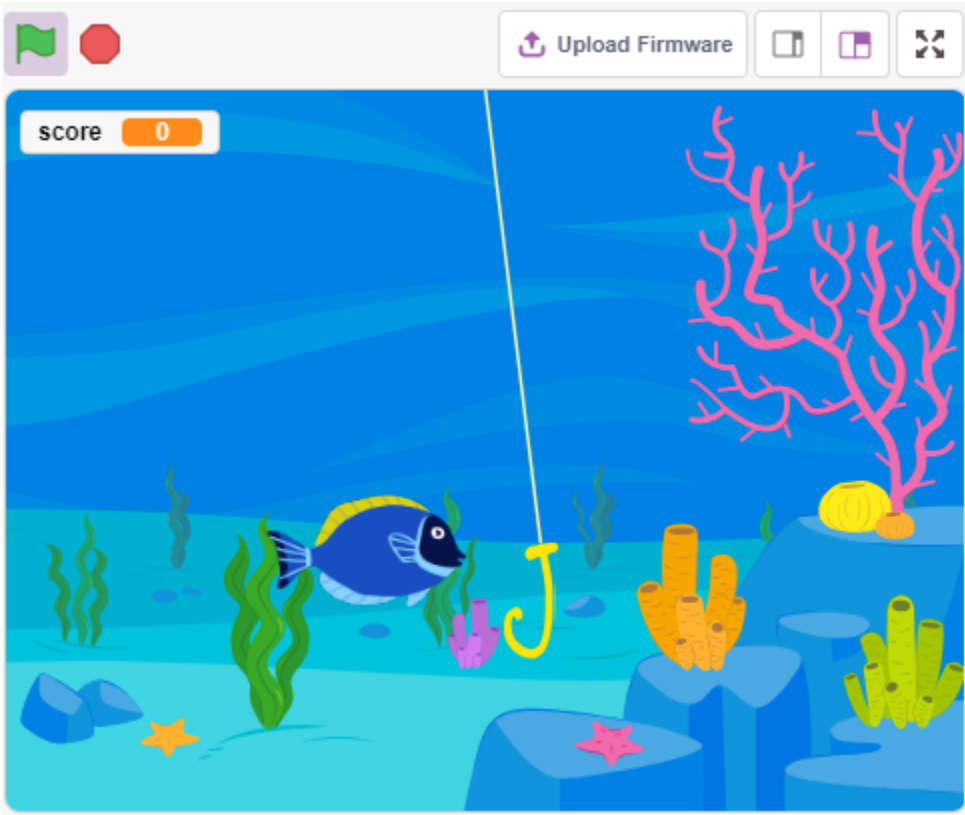
— Retour au sprite **Ball**, lorsque la diffusion **crush** est reçue (le sprite **Ball** touche le clone du sprite **Block1**), la **Ball** est éjectée de la direction opposée.



8.22 2.19 JEU - Pêche

Ici, nous jouons à un jeu de pêche avec un bouton.

Lorsque le script est en cours d'exécution, les poissons nagent de gauche à droite sur la scène, vous devez appuyer sur le bouton lorsque le poisson est presque proche de l'hameçon (il est recommandé de le presser plus longtemps) pour attraper le poisson, et le nombre de poissons attrapés sera enregistré automatiquement.



8.22.1 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

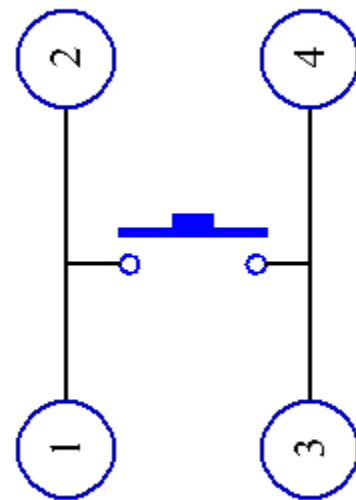
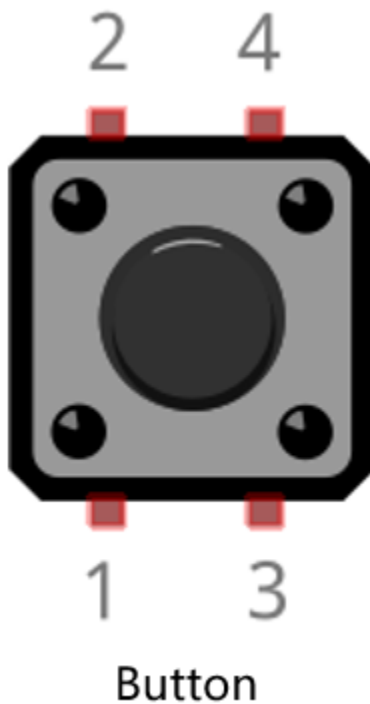
Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Résistance</i>	
<i>Condensateur</i>	
<i>Bouton</i>	

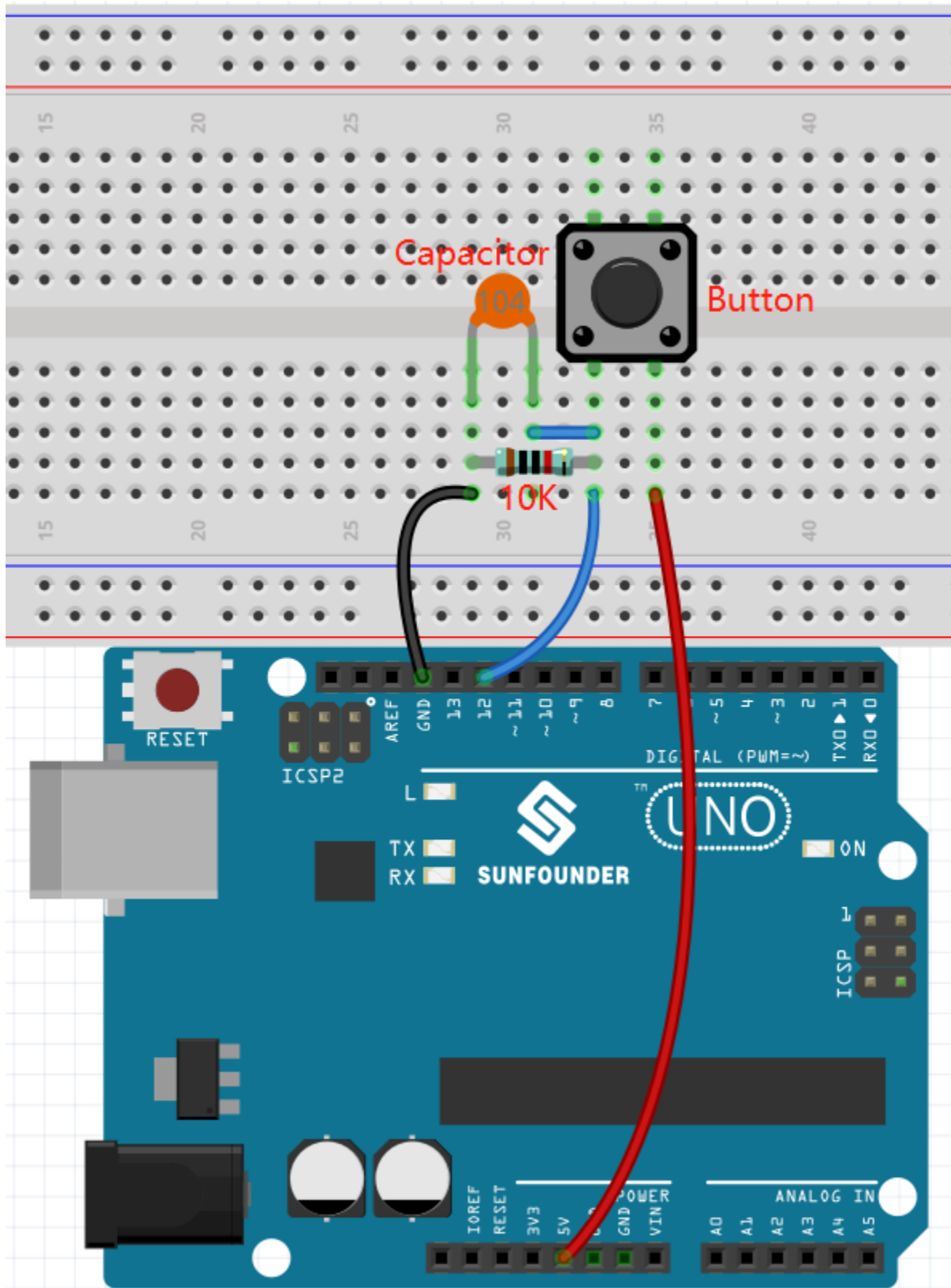
### 8.22.2 Construisez le Circuit

Le bouton est un dispositif à 4 broches, puisque la broche 1 est connectée à la broche 2, et la broche 3 à la broche 4, lorsque le bouton est pressé, les 4 broches sont connectées, fermant ainsi le circuit.



Construisez le circuit selon le schéma suivant.

- Connectez l'une des broches du côté gauche du bouton à la broche 12, qui est connectée à une résistance de tirage et à un condensateur de 0.1uF (104) (pour éliminer le jitter et émettre un niveau stable lorsque le bouton fonctionne).
- Connectez l'autre extrémité de la résistance et du condensateur à GND, et l'une des broches du côté droit du bouton à 5V.

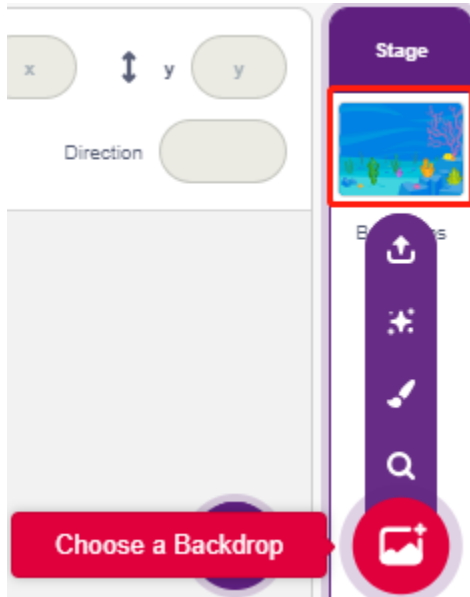


### 8.22.3 Programmation

Nous devons d'abord sélectionner un décor **Underwater**, puis ajouter un sprite **Fish** et le faire nager de gauche à droite sur la scène. Dessinez ensuite un sprite **Fishhook** et contrôlez-le par un bouton pour commencer la pêche. Lorsque le sprite **Fish** touche le sprite **Fishhook** dans l'état accroché (devient rouge), il sera attrapé.

#### 1. Ajouter un décor

Utilisez le bouton **Choose a Backdrop** pour ajouter un décor **Underwater**.

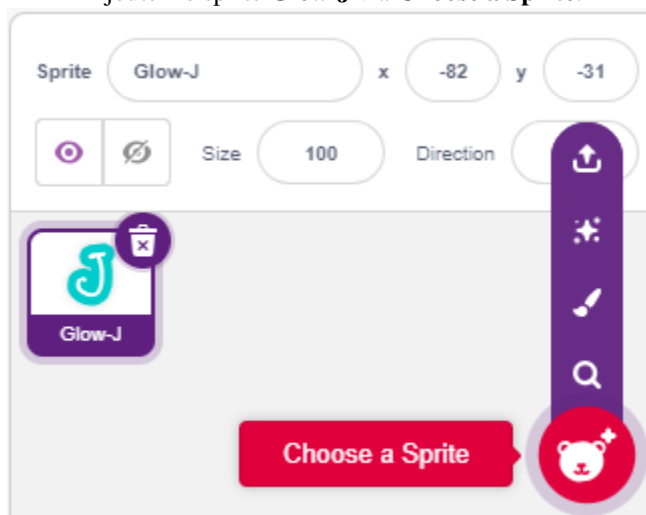


#### 2. Sprite Hameçon

Le sprite **Fishhook** reste généralement sous l'eau en état jaune ; lorsque le bouton est pressé, il est en état de pêche (rouge) et se déplace au-dessus de la scène.

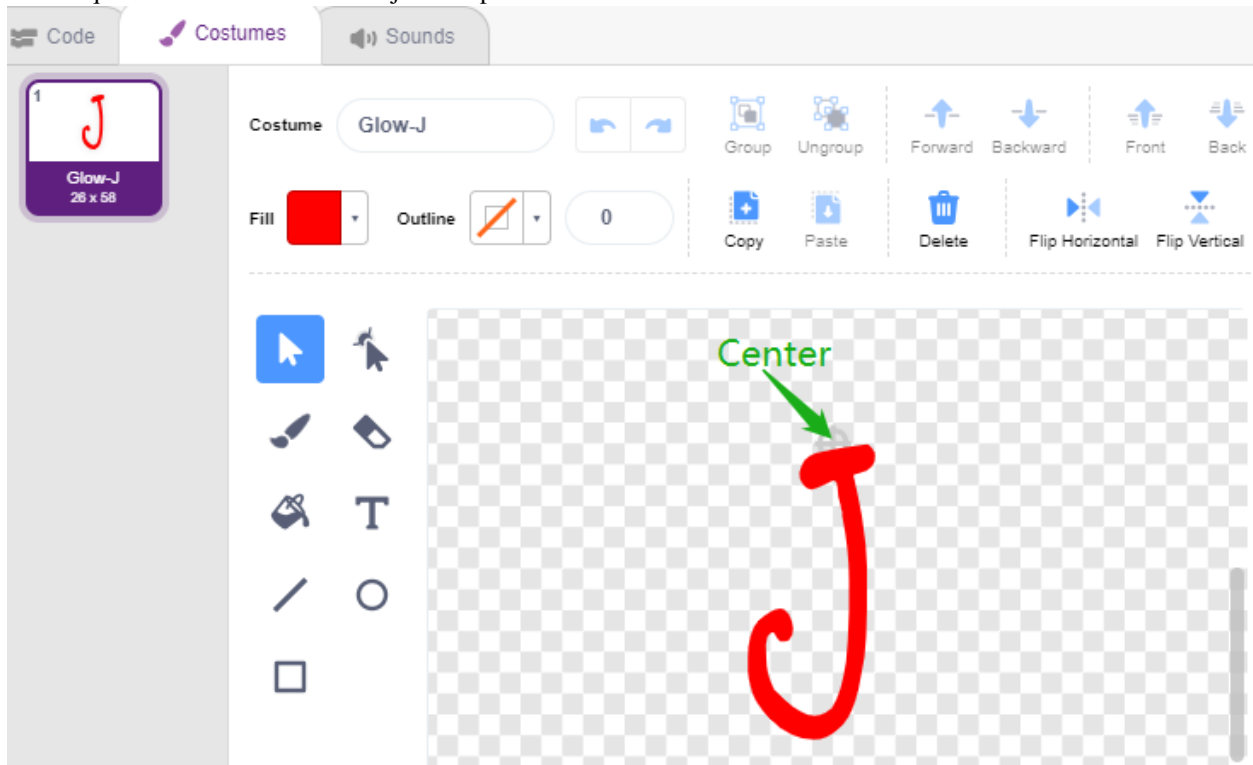
Il n'y a pas de sprite **Fishhook** dans Pictoblox, nous pouvons modifier le sprite **Glow-J** pour qu'il ressemble à un hameçon.

— Ajoutez le sprite **Glow-J** via **Choose a Sprite**.

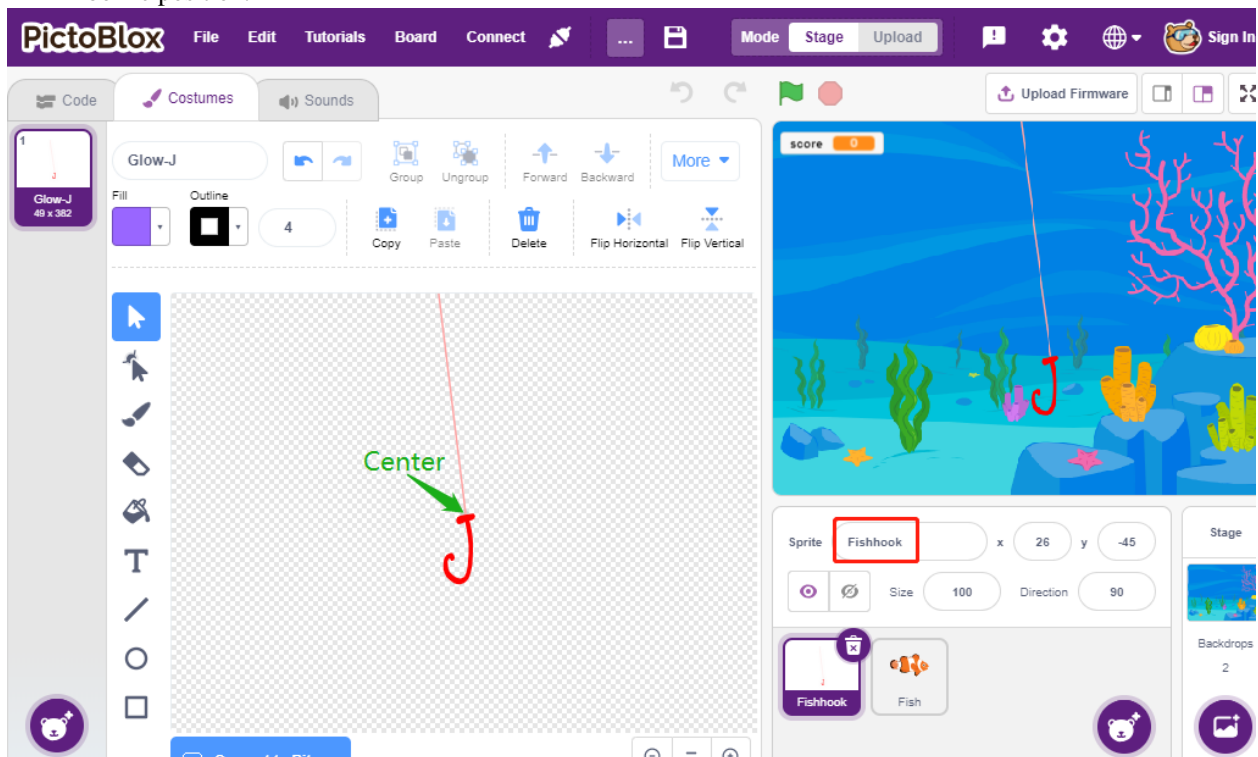


— Allez maintenant à la page **Costumes** du sprite **Glow-J**, sélectionnez le remplissage Cyan à l'écran et retirez-le. Changez ensuite la couleur J en rouge et réduisez également sa largeur. Le point le plus important à noter est

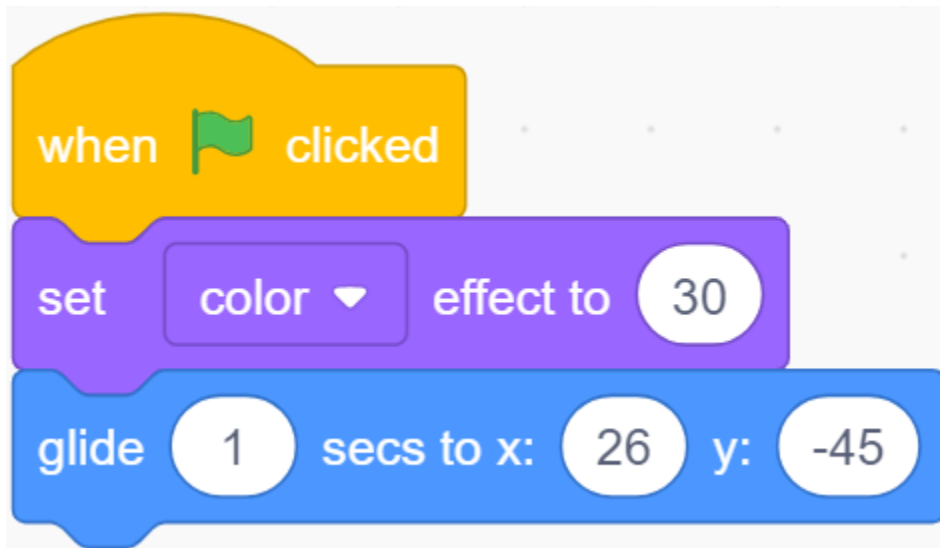
que vous devez avoir le haut juste au point central.



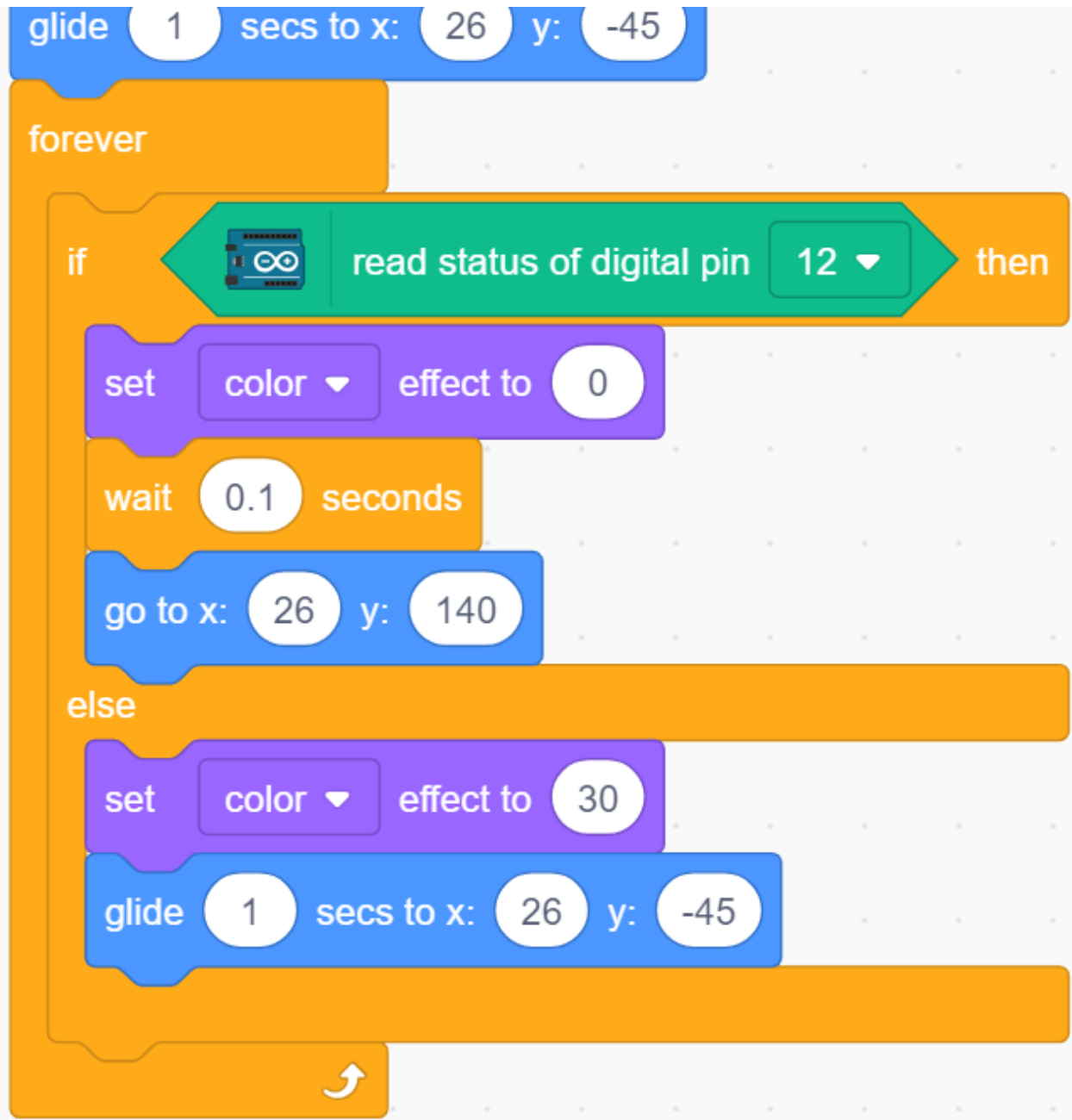
- Utilisez l'outil **Line tool** pour dessiner une ligne aussi longue que possible du point central vers le haut (ligne hors de la scène). Maintenant que le sprite est dessiné, réglez le nom du sprite sur **Fishhook** et déplacez-le à la bonne position.



- Lorsque le drapeau vert est cliqué, réglez l'effet de couleur du sprite à 30 (jaune), et définissez sa position initiale.



- Si le bouton est pressé, réglez l'effet de couleur à 0 (rouge, commencez l'état de pêche), attendez 0,1 puis déplacez le sprite **Fishhook** au-dessus de la scène. Relâchez le bouton et laissez le **Fishhook** revenir à sa position initiale.

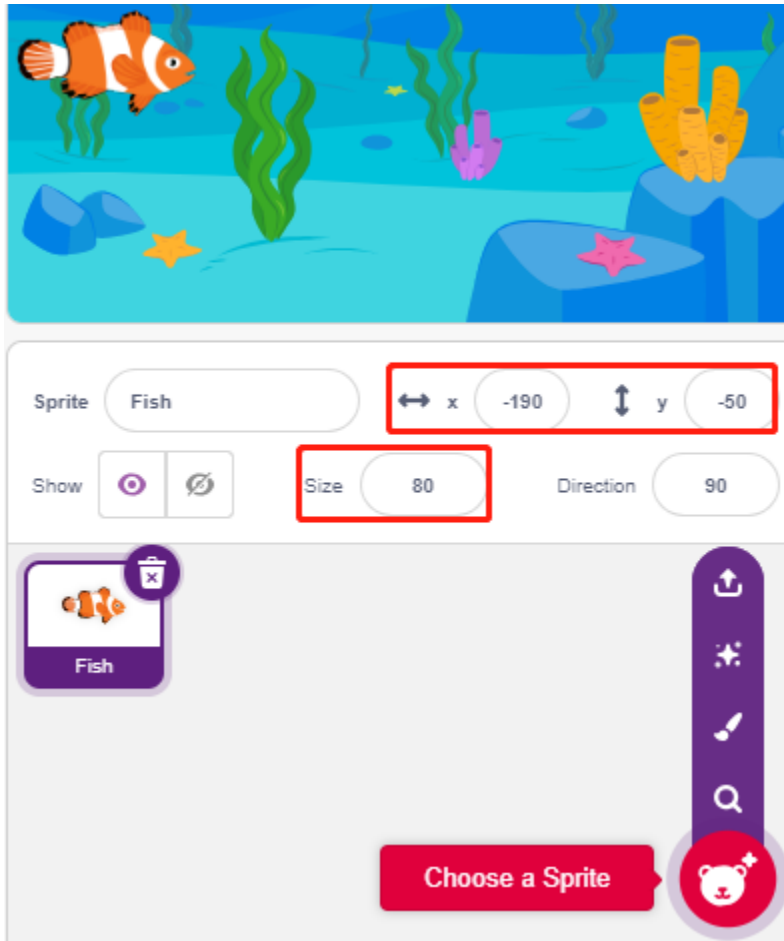


### 3. Sprite Poisson

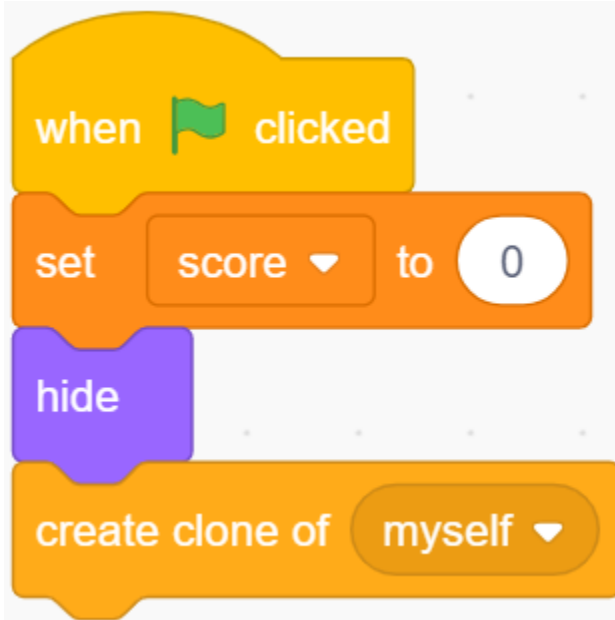
L'effet à atteindre par le sprite **Fish** est de se déplacer de gauche à droite sur la scène, et lorsqu'il rencontre un sprite **Fishhook** en état de pêche, il rétrécit et se déplace à une position spécifique puis disparaît, et clone ensuite un nouveau sprite **fish**.

- Ajoutez maintenant le sprite **fish** et ajustez sa taille et sa position.

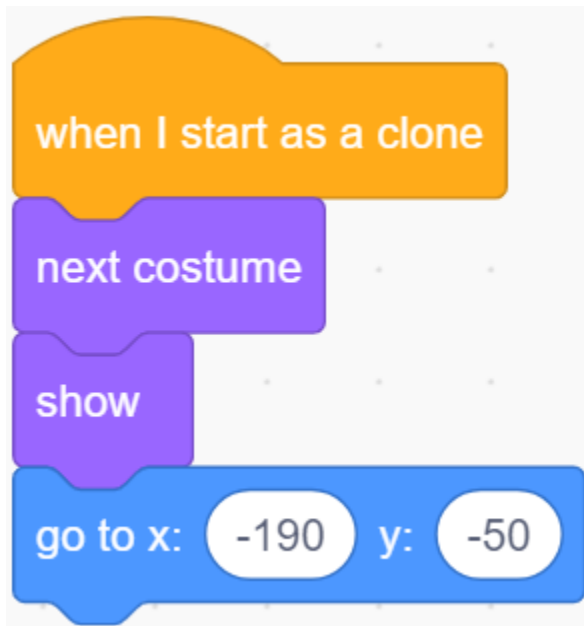




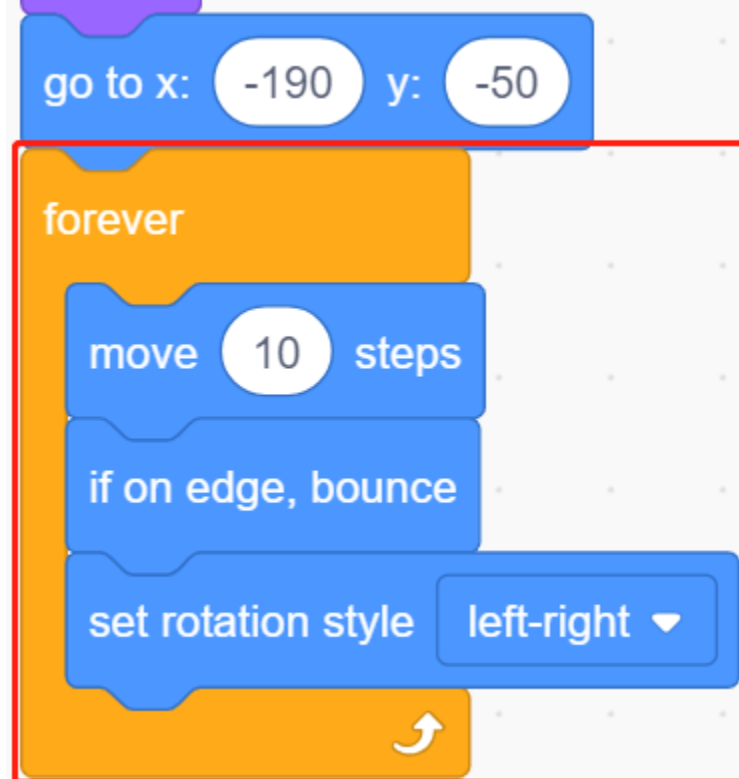
— Créez une variable **score** pour stocker le nombre de poissons attrapés, cachez ce sprite et clonez-le.



— Montrez le clone du sprite **fish**, changez son costume et enfin définissez la position initiale.



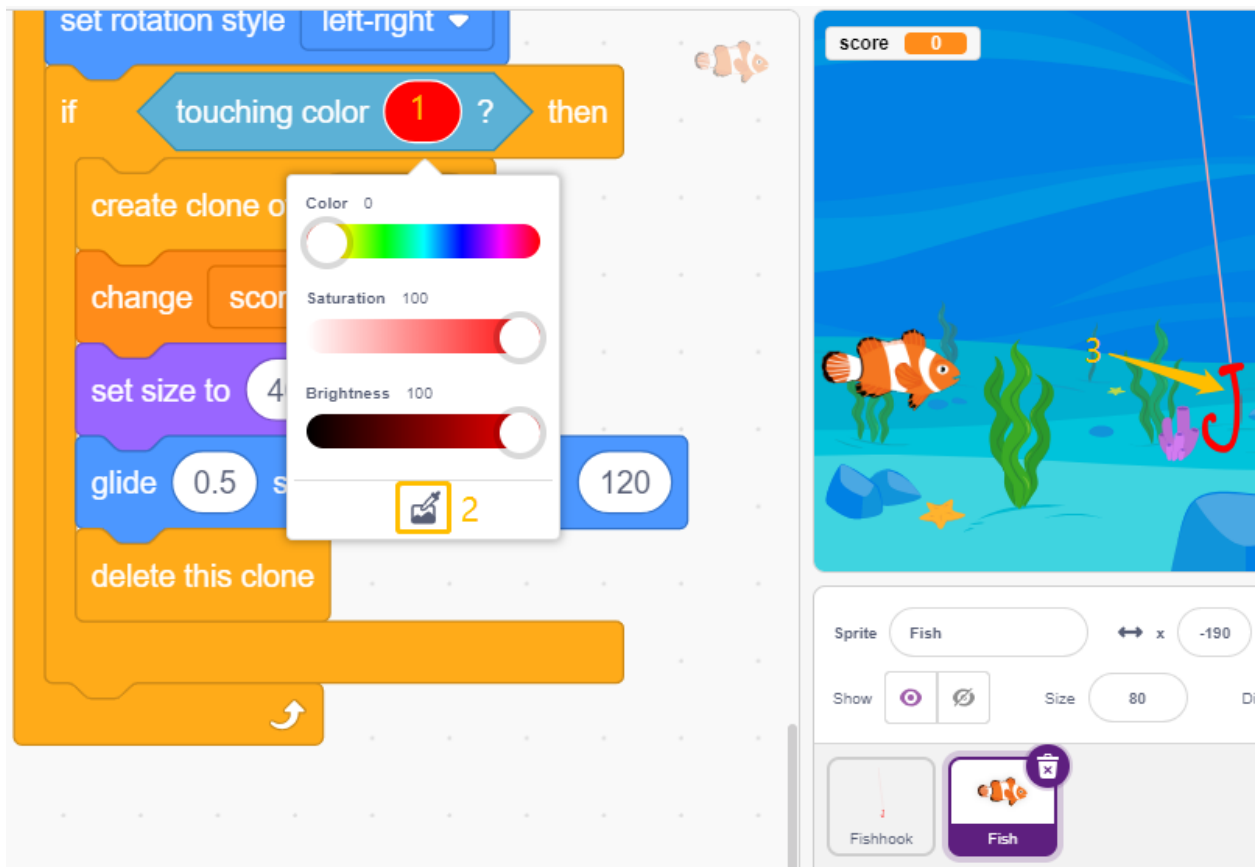
— Faites bouger le clone du sprite **fish** de gauche à droite et rebondir lorsqu'il touche le bord.



— Le sprite **fish** (du clone) ne réagit pas lorsqu'il passe l'sprite **Fishhook** ; lorsqu'il touche le sprite **Fishhook** en état de pêche (devient rouge), il sera attrapé, à ce moment le score (variable score) +1, et il montrera également une animation de score (rétrécit de 40%, se déplace rapidement vers la position du tableau de score et disparaît). En même temps, un nouveau poisson est créé (un nouveau clone de sprite poisson) et le jeu continue.

**Note :** Vous devez cliquer sur la zone de couleur dans le bloc [Touch color], puis sélectionner l'outil pipette pour prendre la couleur rouge du sprite **Fishhook** sur la scène. Si vous choisissez une couleur arbitrairement, ce bloc [Touch

color] ne fonctionnera pas.



## 8.23 2.20 JEU - Ne Touchez Pas la Tuile Blanche

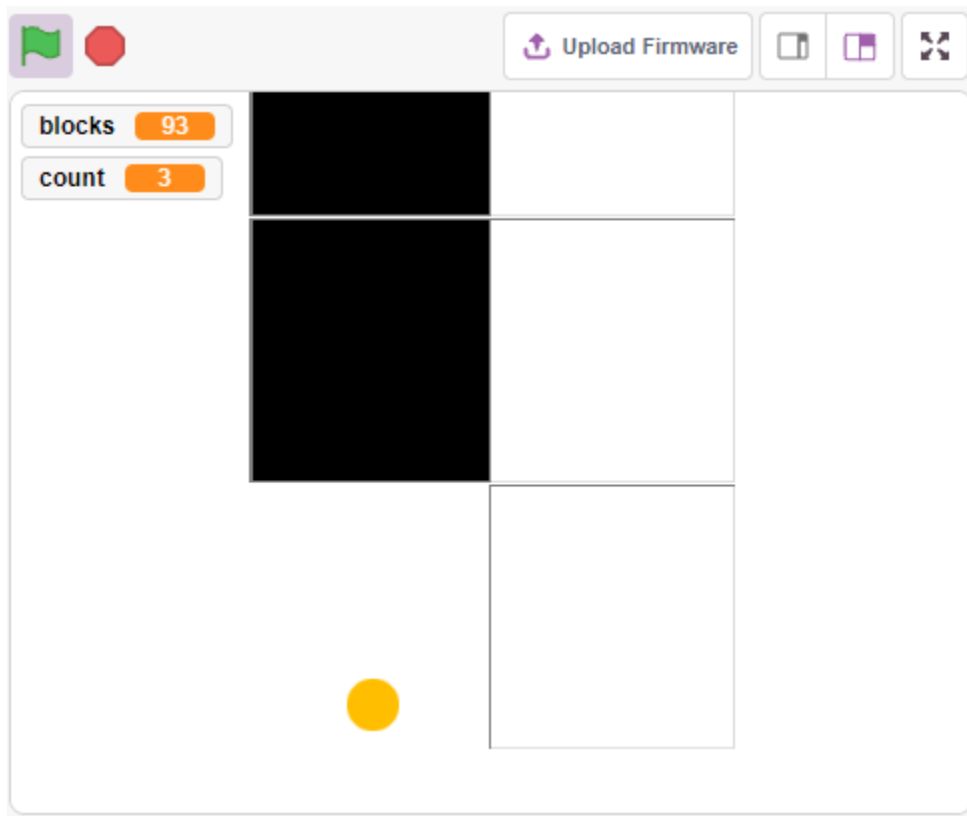
Je suis sûr que beaucoup d'entre vous ont joué à ce jeu sur vos téléphones portables. Ce jeu se joue en tapotant sur des blocs noirs apparaissant aléatoirement pour ajouter des points, la vitesse devient de plus en plus rapide, tapotez sur des blocs blancs ou manquez des blocs noirs et la partie se termine.

Maintenant, nous utilisons PictoBlox pour le reproduire.

Insérez deux modules d'évitement d'obstacles IR verticalement sur la plaque de montage, lorsque votre main est placée au-dessus de l'un des modules IR, un point clignotant apparaîtra sur la scène, représentant une touche effectuée.

Si la touche est sur le bloc noir, le score augmente de 1, touchez le bloc blanc, le score diminue de 1.

Vous devez décider de placer votre main au-dessus du module IR à gauche ou au-dessus du module IR à droite, en fonction de la position du bloc noir sur la scène.



### 8.23.1 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DE CE KIT	LIEN
3 in 1 Starter Kit	380+	

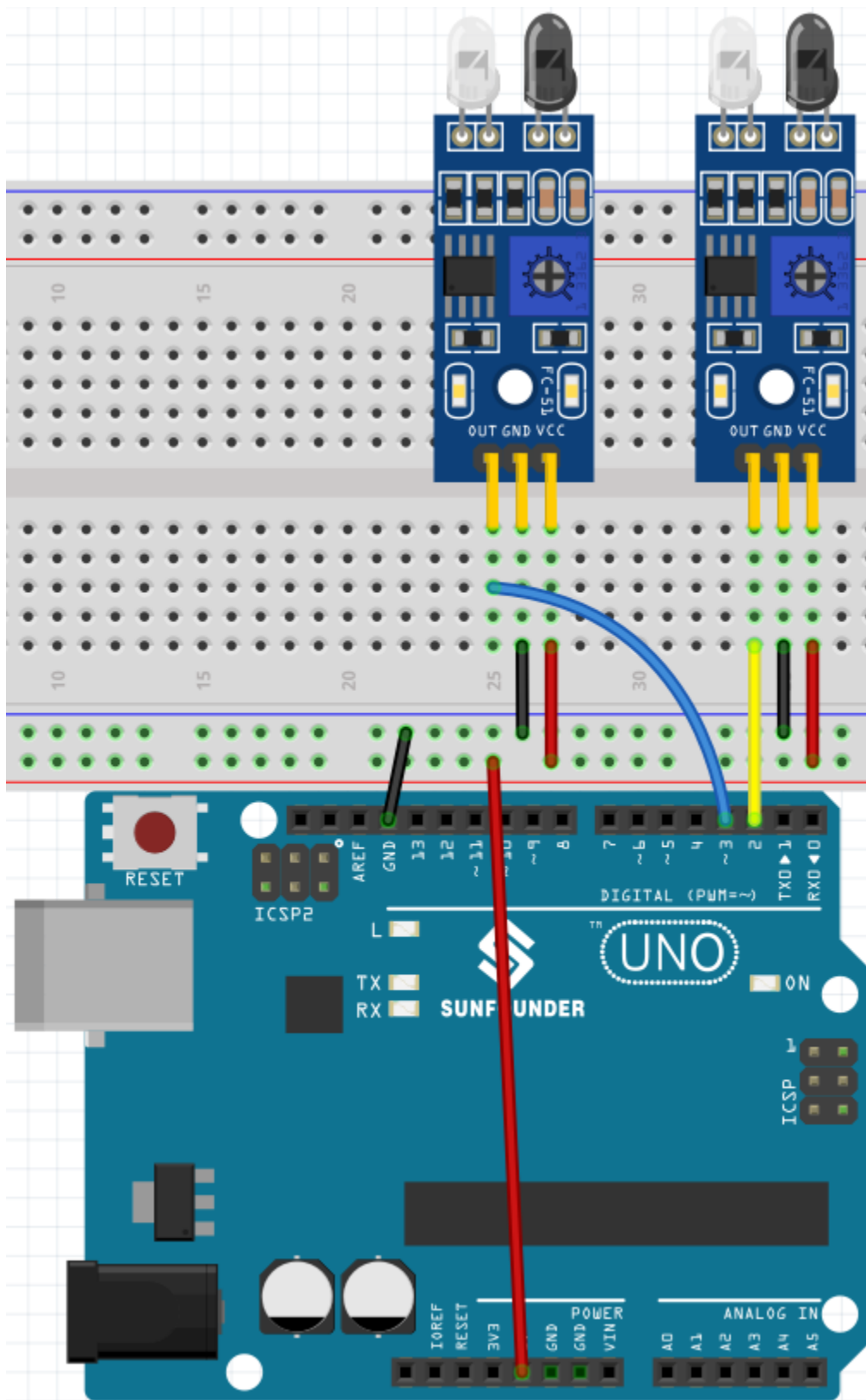
Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Plaque d'essai</i>	
<i>Fils de Cavalier</i>	
<i>Module d'Évitement d'Obstacle</i>	

### **8.23.2 Construisez le Circuit**

Le module d'évitement d'obstacles est un capteur de proximité infrarouge à distance réglable dont la sortie est normalement haute et basse lorsqu'un obstacle est détecté.

Construisez maintenant le circuit selon le schéma ci-dessous.



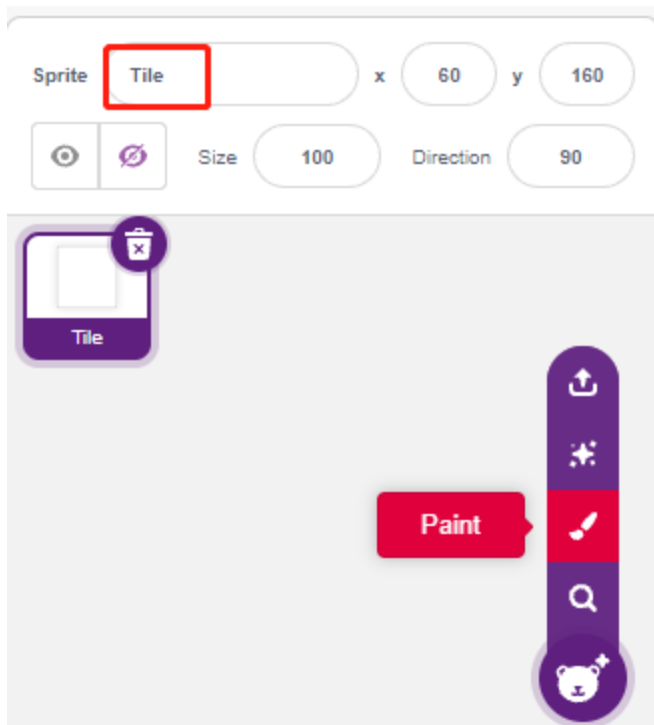
### 8.23.3 Programmation

Ici, nous avons besoin de 3 sprites, **Tile**, **Left IR** et **Right IR**.

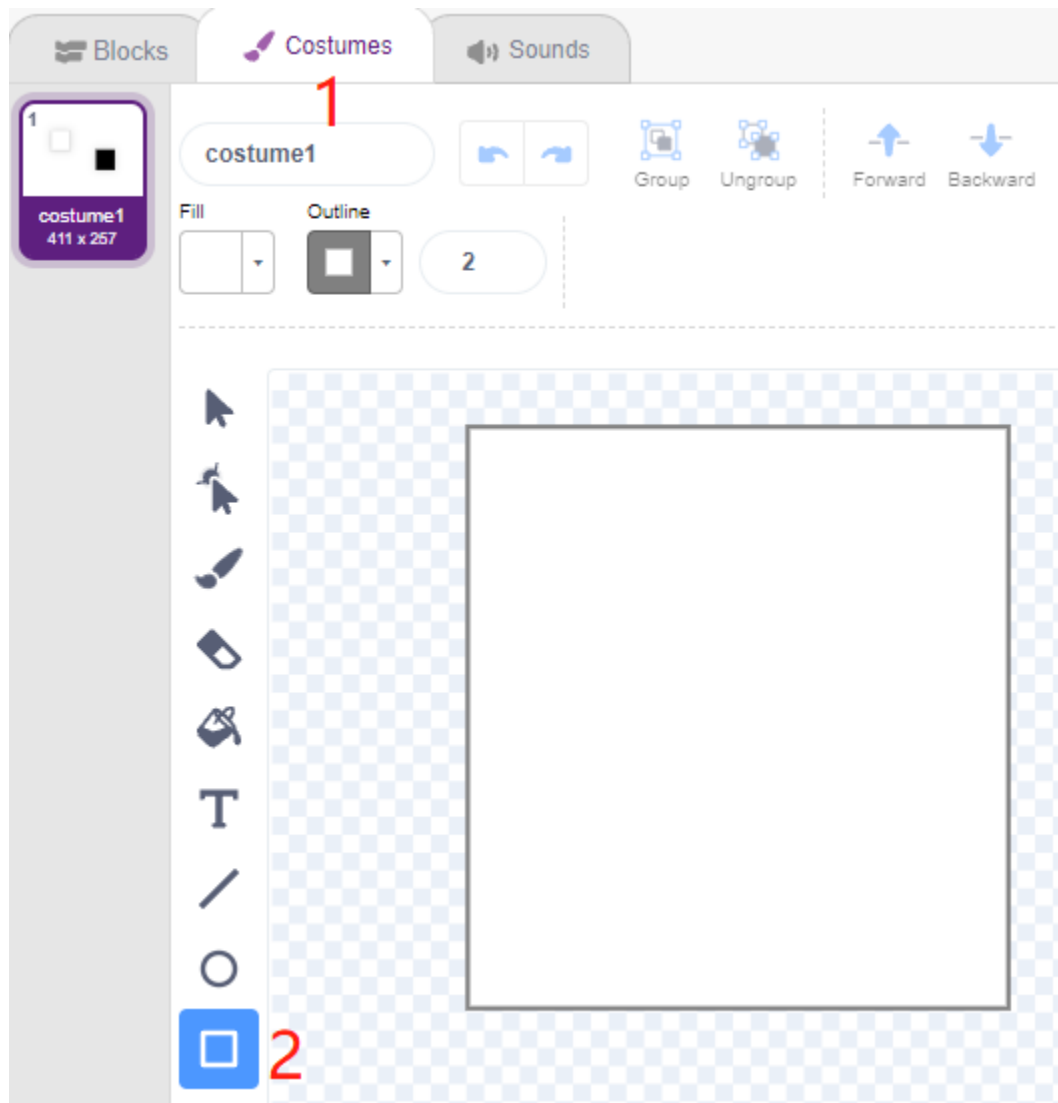
- Sprite **Tile** : utilisé pour obtenir l'effet d'alternance des tuiles noires et blanches vers le bas, dans le jeu sur téléphone portable, il y a généralement 4 colonnes, ici nous en ferons seulement deux.
- Sprite **Left IR** : utilisé pour réaliser l'effet de clic, lorsque le module IR gauche détecte votre main, il enverra un message - **left** au sprite **Left IR**, le laissant commencer à travailler. S'il touche la tuile noire sur la scène, le score augmentera de 1, sinon le score diminuera de 1.
- Sprite **Right IR** : La fonction est essentiellement la même que **Left IR**, sauf qu'il reçoit l'information **Right**.

#### 1. Peindre un sprite Tuile.

Supprimez le sprite par défaut, passez la souris sur l'icône **Add Sprite**, sélectionnez **Paint** et un sprite vierge apparaîtra et nommez-le **Tile**.

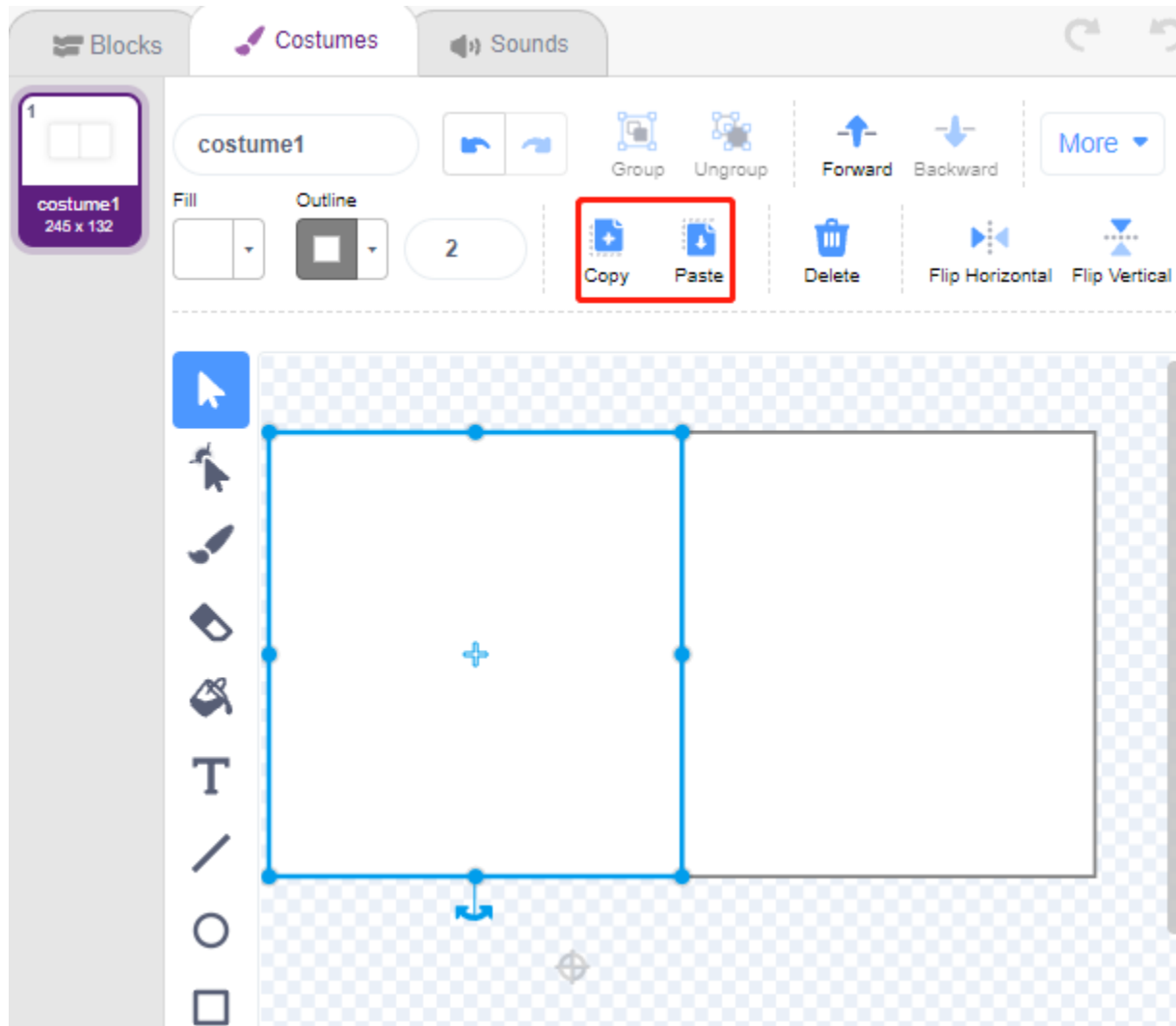


Allez à la page **Costumes** et utilisez l'outil **Rectangle** pour dessiner un rectangle.

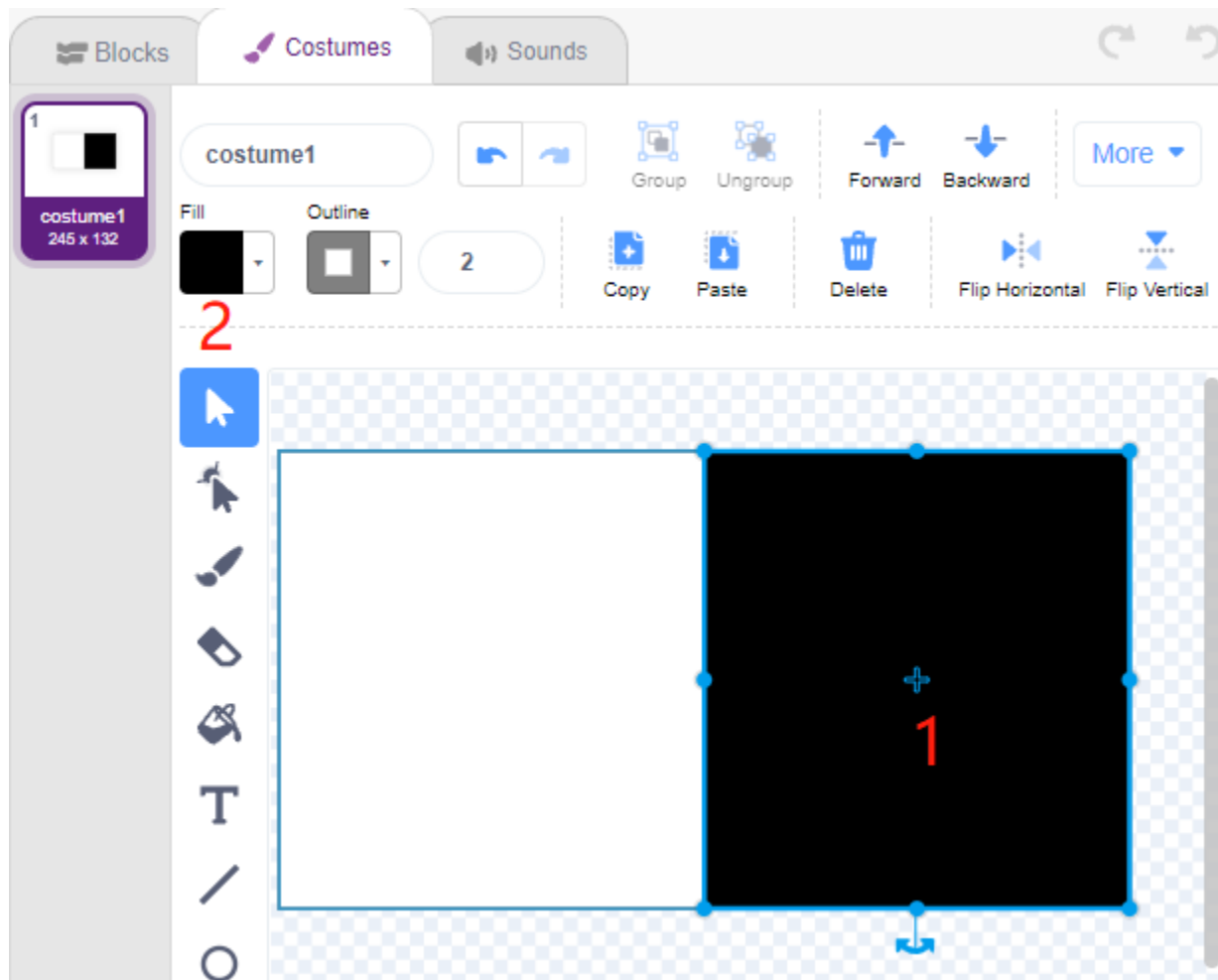


Sélectionnez le rectangle et cliquez sur **Copy** -> **Paste** pour faire un rectangle identique, puis déplacez les deux rectangles en position alignée.

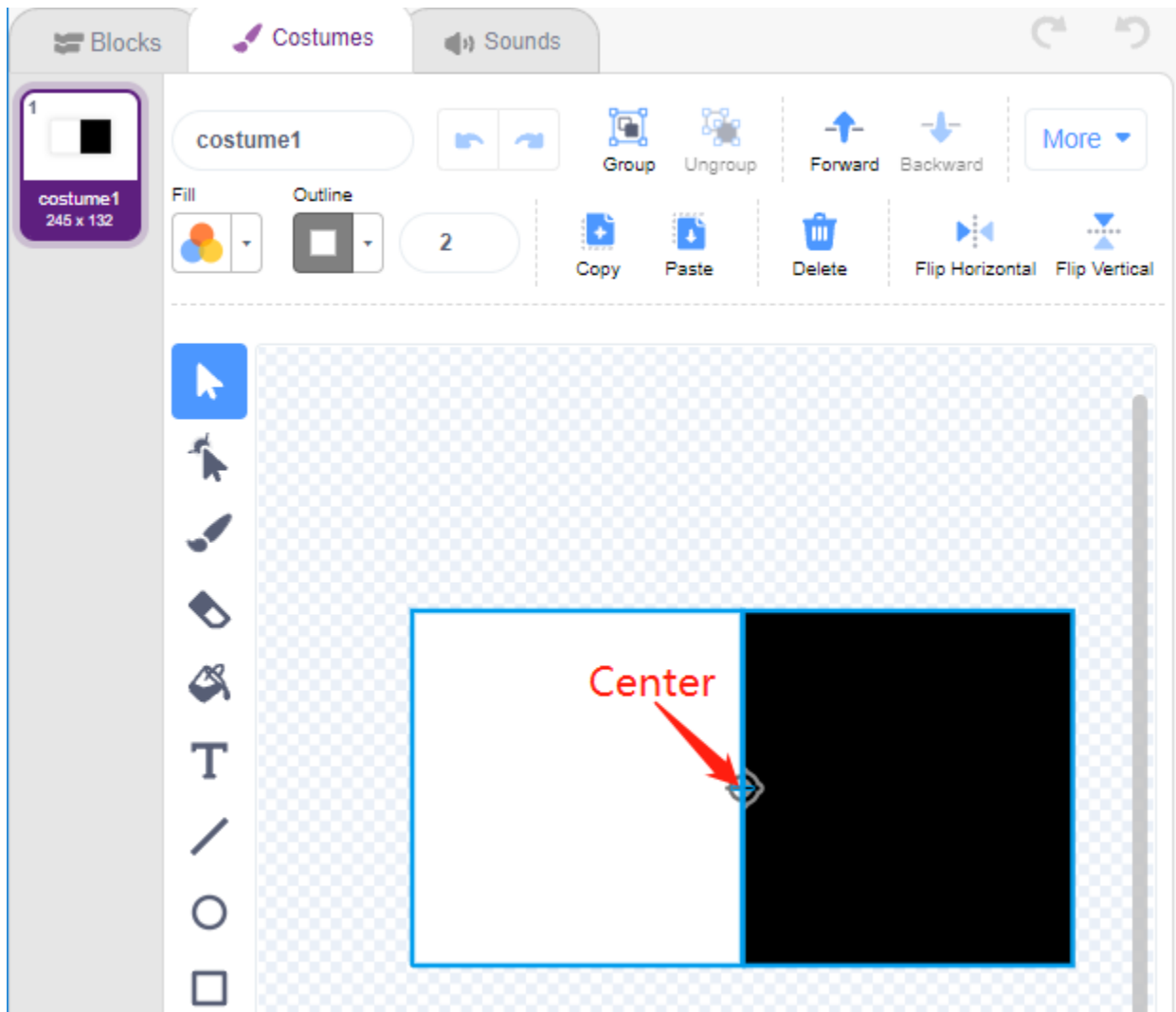




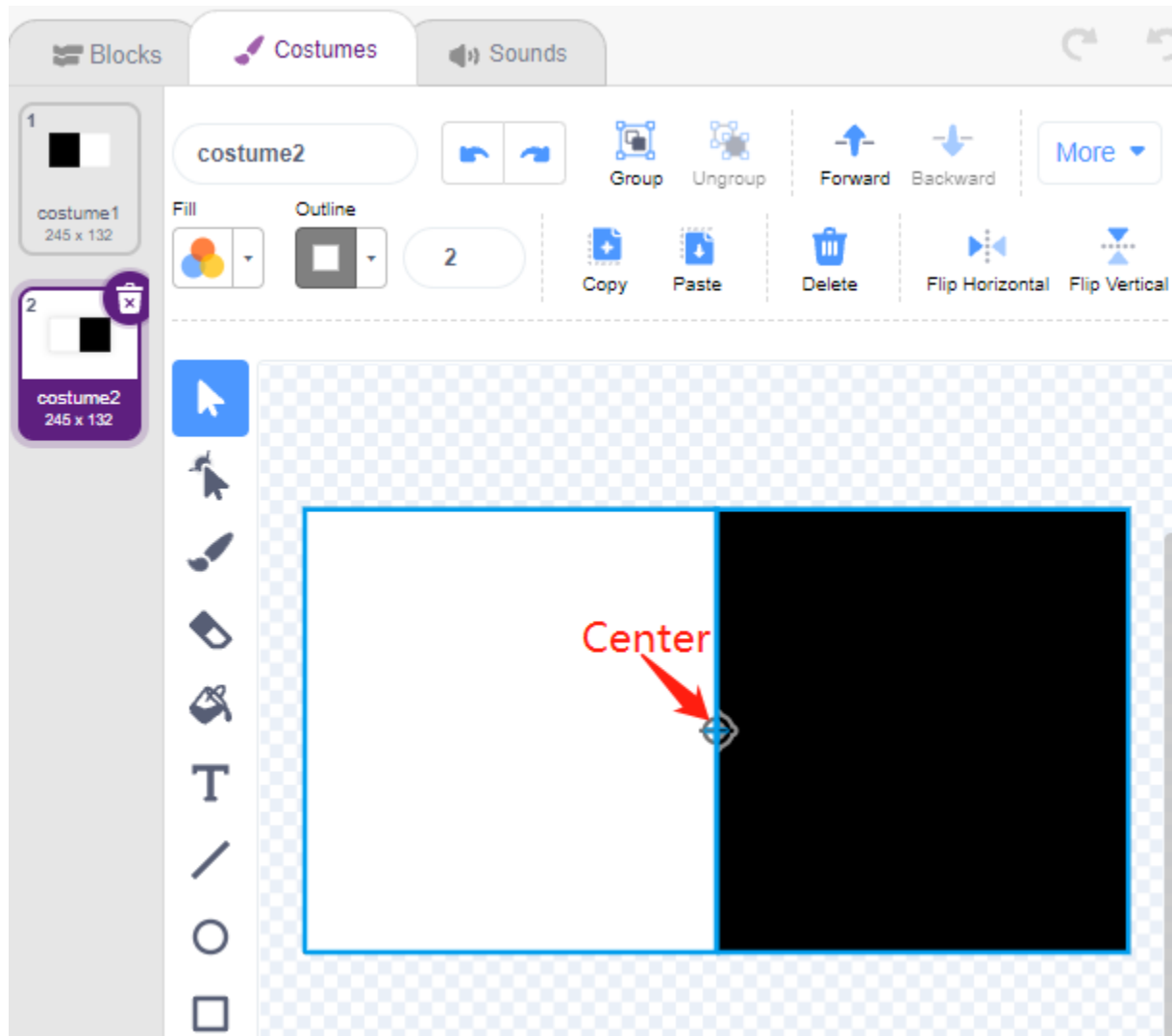
Sélectionnez l'un des rectangles et choisissez une couleur de remplissage noire.



Sélectionnez maintenant les deux rectangles et déplacez-les de sorte que leurs points centraux correspondent au centre du canevas.

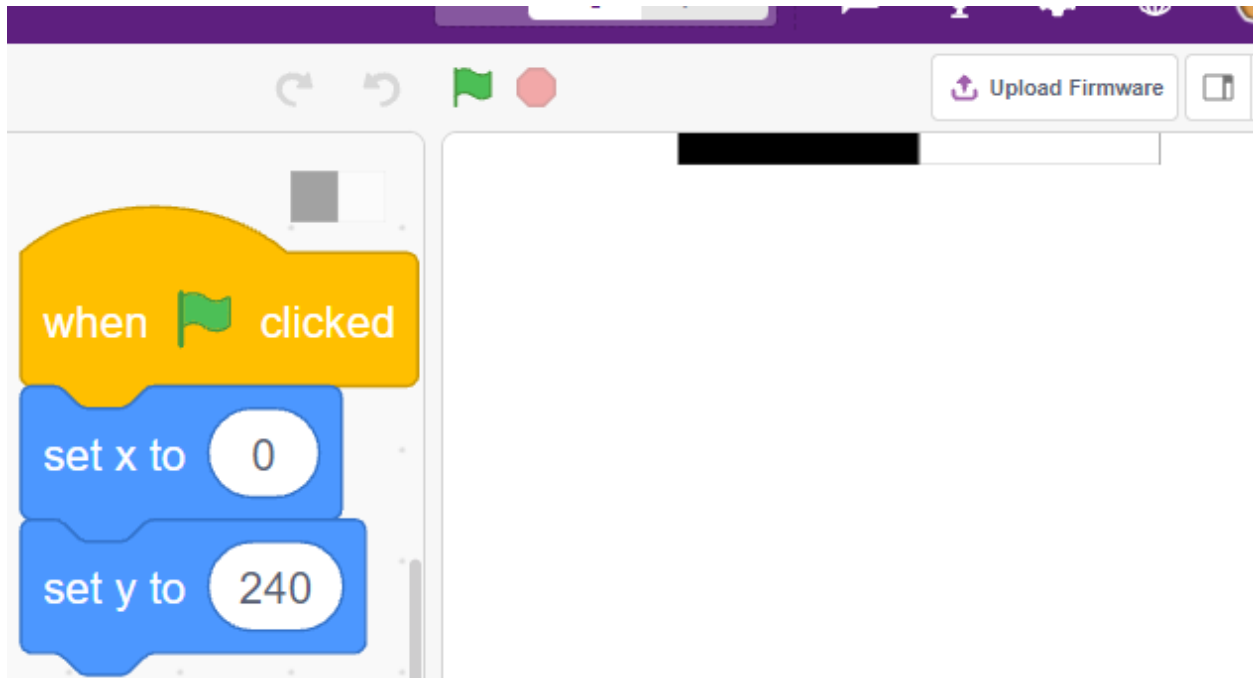


Dupliquez le costume1, en alternant les couleurs de remplissage des deux rectangles. Par exemple, la couleur de remplissage du costume1 est blanche à gauche et noire à droite, et la couleur de remplissage du costume2 est noire à gauche et blanche à droite.



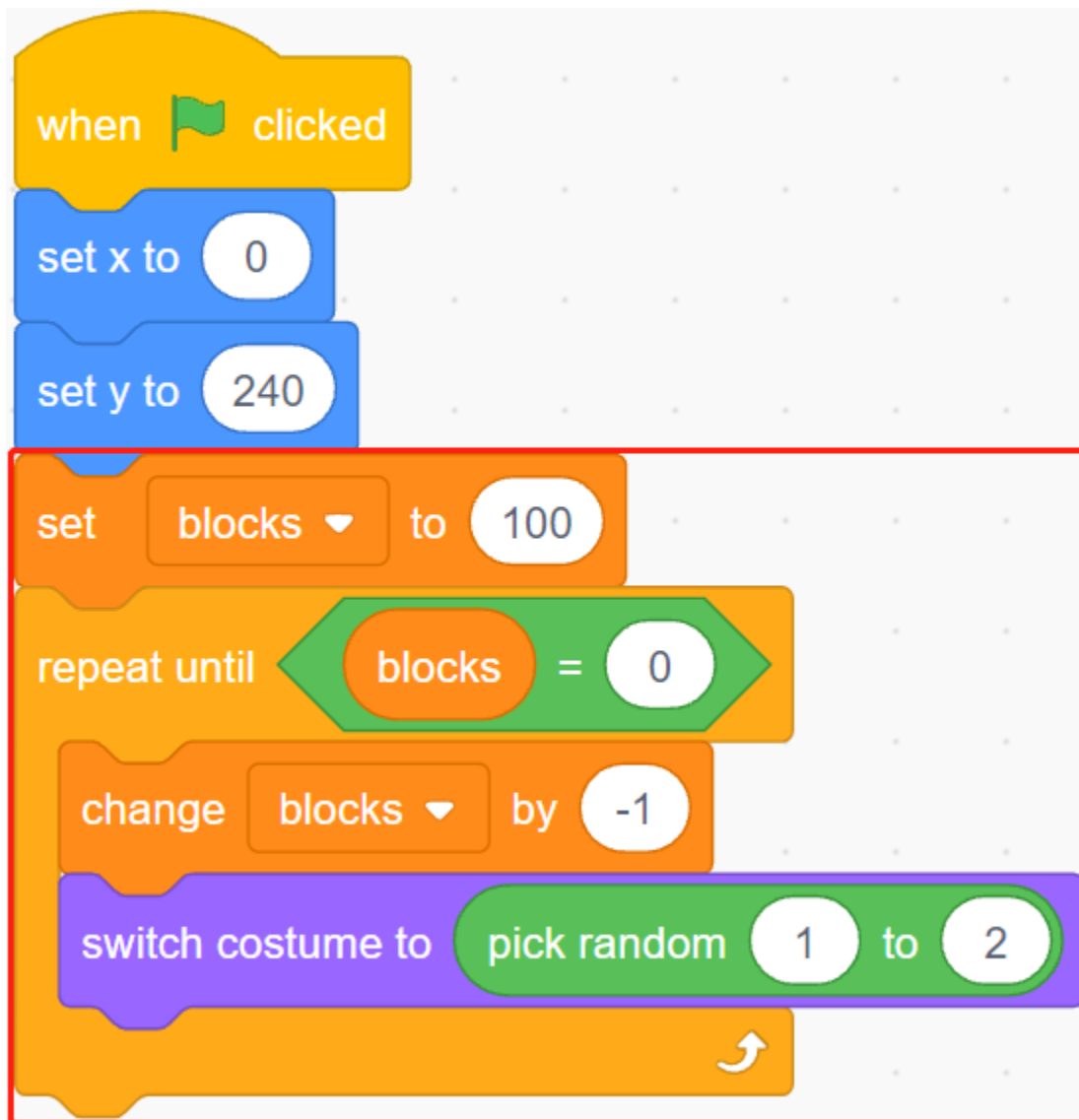
## 2. Scripter le sprite Tuile

Retournez maintenant à la page **Blocks** et réglez la position initiale du sprite **Tuile** pour qu'il soit en haut de la scène.

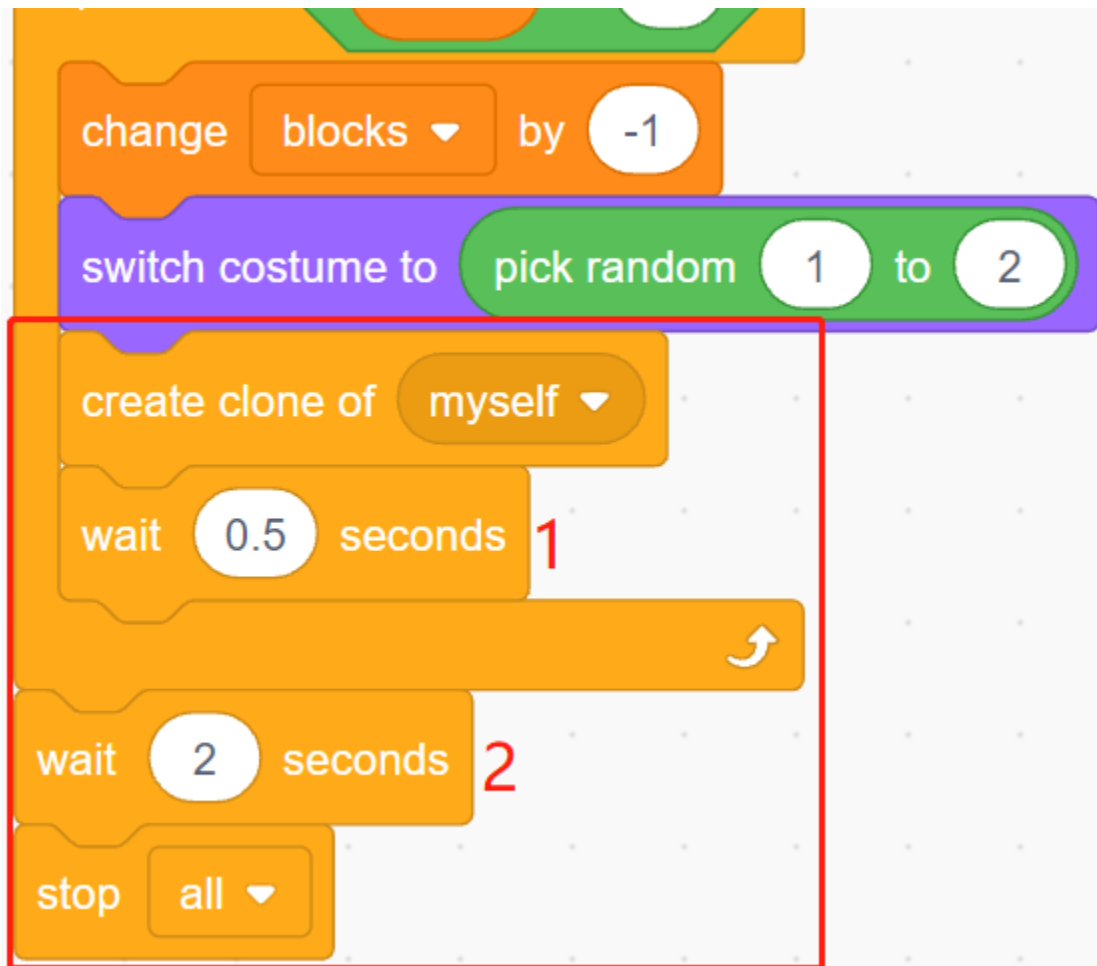


Créez une variable - **blocks** et donnez-lui une valeur initiale pour déterminer le nombre de fois où le sprite **Tile** apparaîtra. Utilisez le bloc [repeat until] pour faire diminuer progressivement la variable **blocks** jusqu'à ce que **blocks** soit 0. Pendant ce temps, faites en sorte que le sprite **Tile** change aléatoirement de costume.

Après avoir cliqué sur le drapeau vert, vous verrez le sprite **Tile** sur la scène changer rapidement de costume.



Créez des clones du sprite **Tile** pendant que la variable **blocks** diminue, et arrêtez l'exécution du script lorsque blocs est 0. Deux blocs [wait () seconds] sont utilisés ici, le premier pour limiter l'intervalle entre les clones de **Tile's** et le second pour permettre à la variable blocs de diminuer à 0 sans arrêter immédiatement le programme, donnant au dernier sprite de tuile suffisamment de temps pour se déplacer.

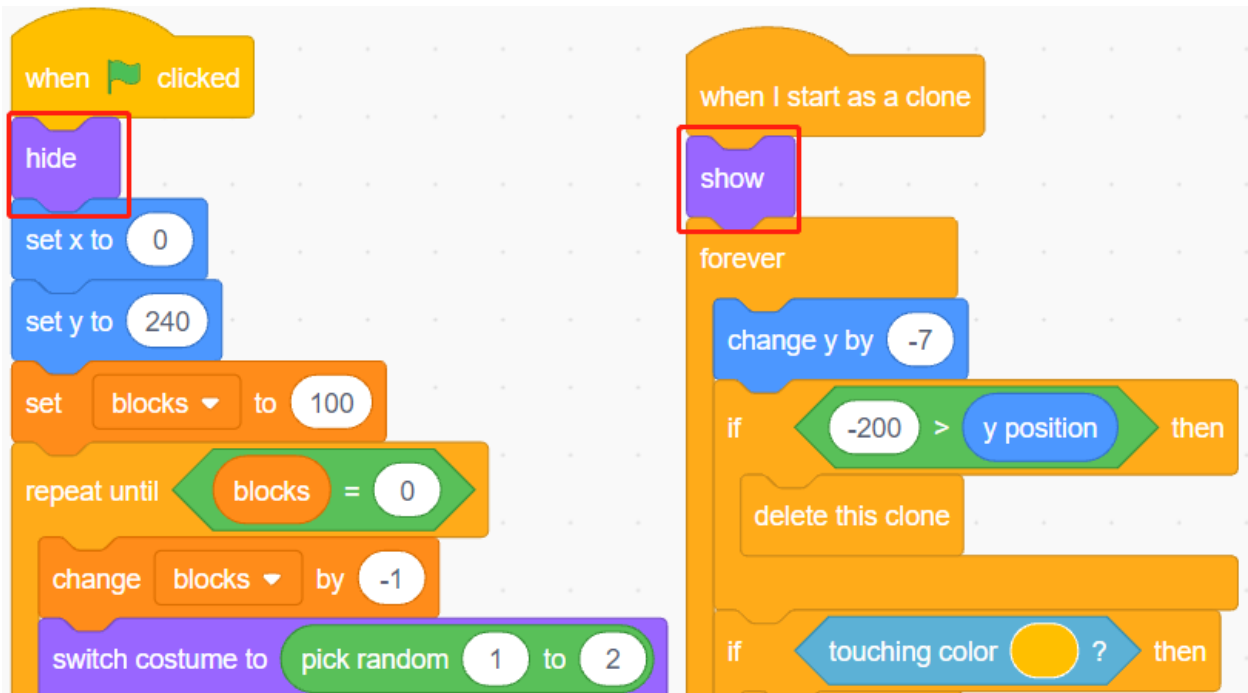


Maintenant, scriptez le clone du sprite **Tile** pour qu'il descende lentement et supprimez-le lorsqu'il atteint le bas de la scène. Le changement dans la coordonnée y affecte la vitesse de chute, plus la valeur est grande, plus la vitesse de chute est rapide.



Cachez le corps et affichez le clone.

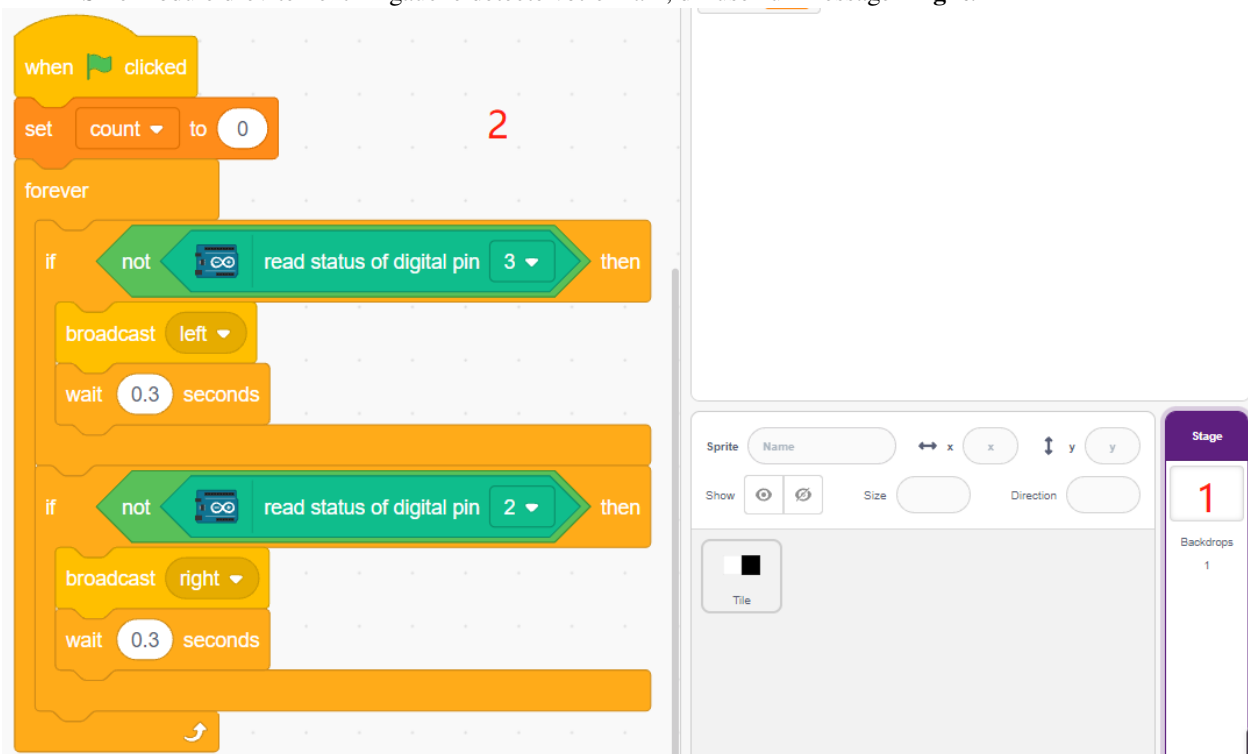




### 3. Lire les valeurs des 2 modules IR

Dans le décor, lisez les valeurs des 2 modules IR et effectuez les actions correspondantes.

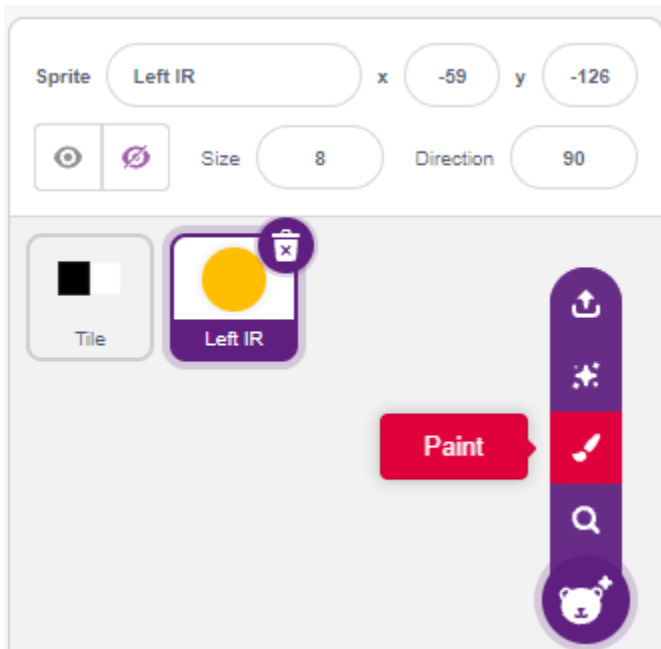
- Si le module d'évitement d'obstacles IR gauche détecte votre main, diffusez un message - **left**.
- Si le module d'évitement IR gauche détecte votre main, diffusez un message - **right**.



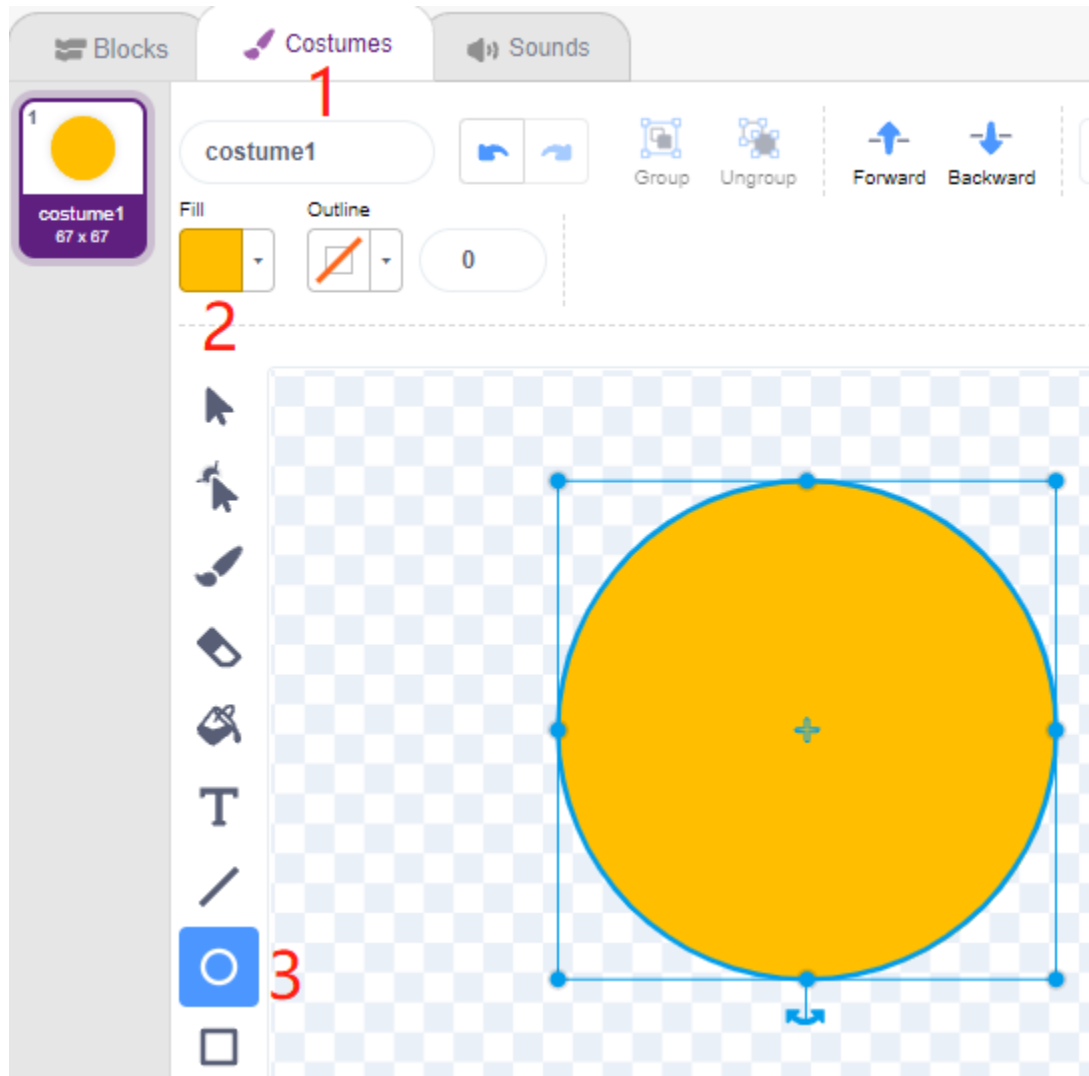
### 4. Sprite IR Gauche

Encore une fois, passez la souris sur l'icône **Add sprite** et sélectionnez **Paint** pour créer un nouveau sprite appelé **Left**

IR.



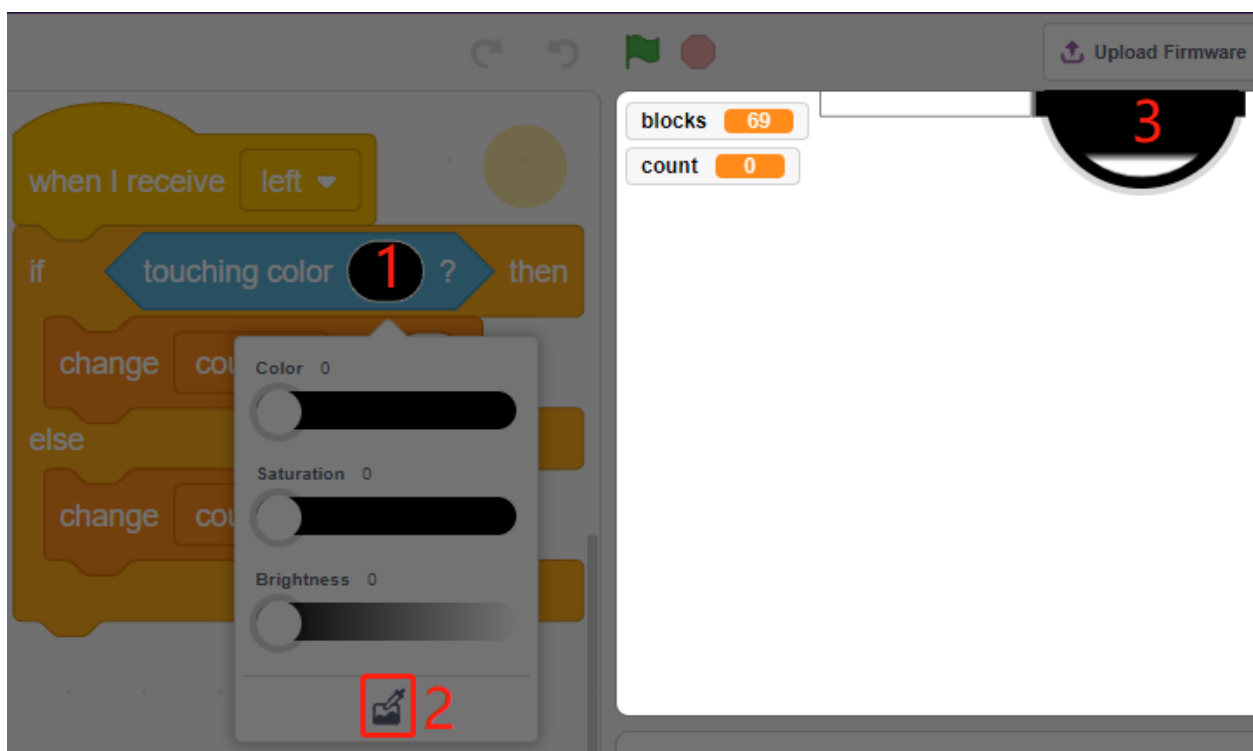
Allez à la page **Costumes** du sprite **Left IR**, sélectionnez la couleur de remplissage (n'importe quelle couleur hors noir et blanc) et dessinez un cercle.



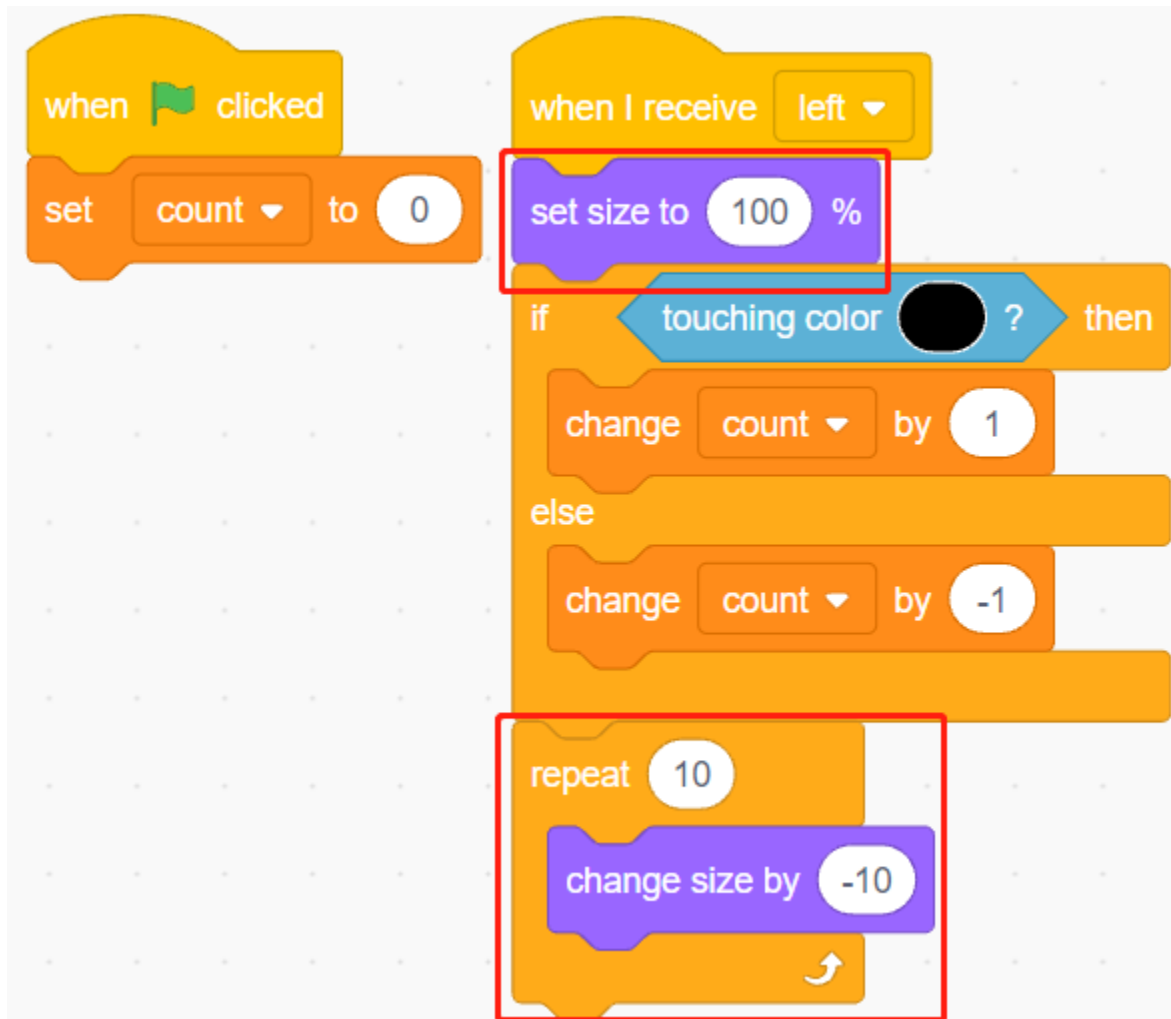
Commencez maintenant à scripter le sprite **Left IR**. Lorsque le message - **left** est reçu (le module récepteur IR à gauche détecte un obstacle), déterminez alors si le bloc noir du sprite **Tile** est touché, et si c'est le cas, laissez la variable **count** ajouter 1, sinon soustraire 1.



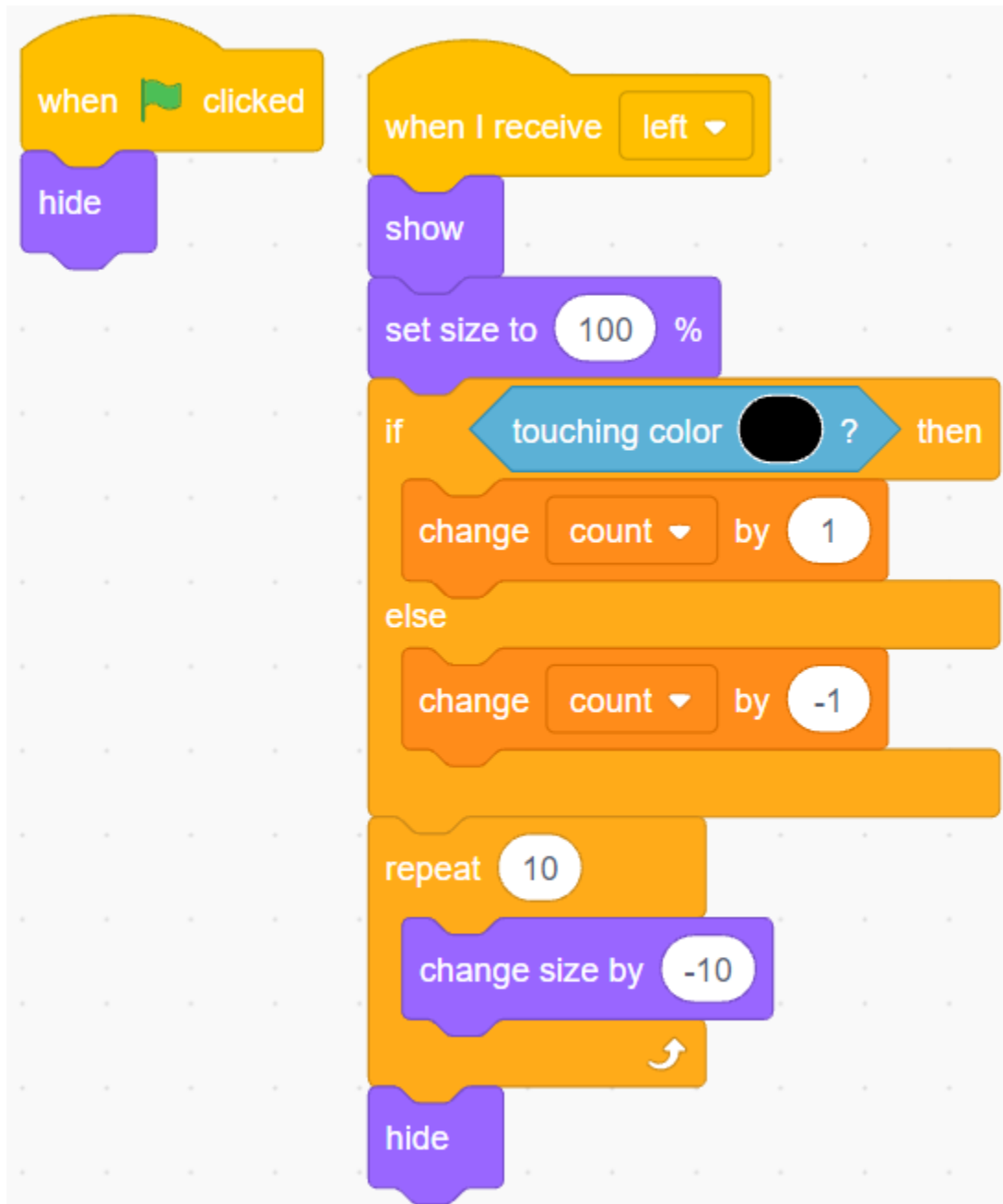
**Note :** Vous devez faire apparaître le sprite **Tile** sur la scène, puis absorber la couleur du bloc noir dans le sprite **Tile**.



Maintenant, faisons l'effet de détection (agrandir et rétrécir) pour **Left IR**.

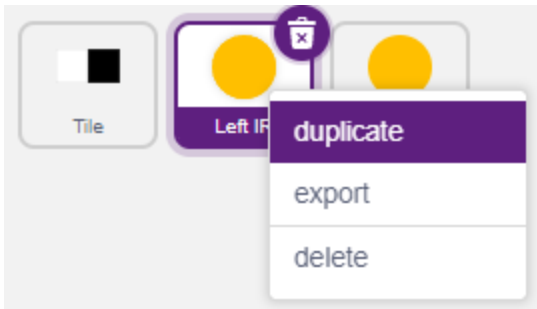


Faites en sorte que le sprite **Left IR** se cache lorsque le drapeau vert est cliqué, s'affiche lorsque le message - **left** est reçu, et se cache à nouveau enfin.

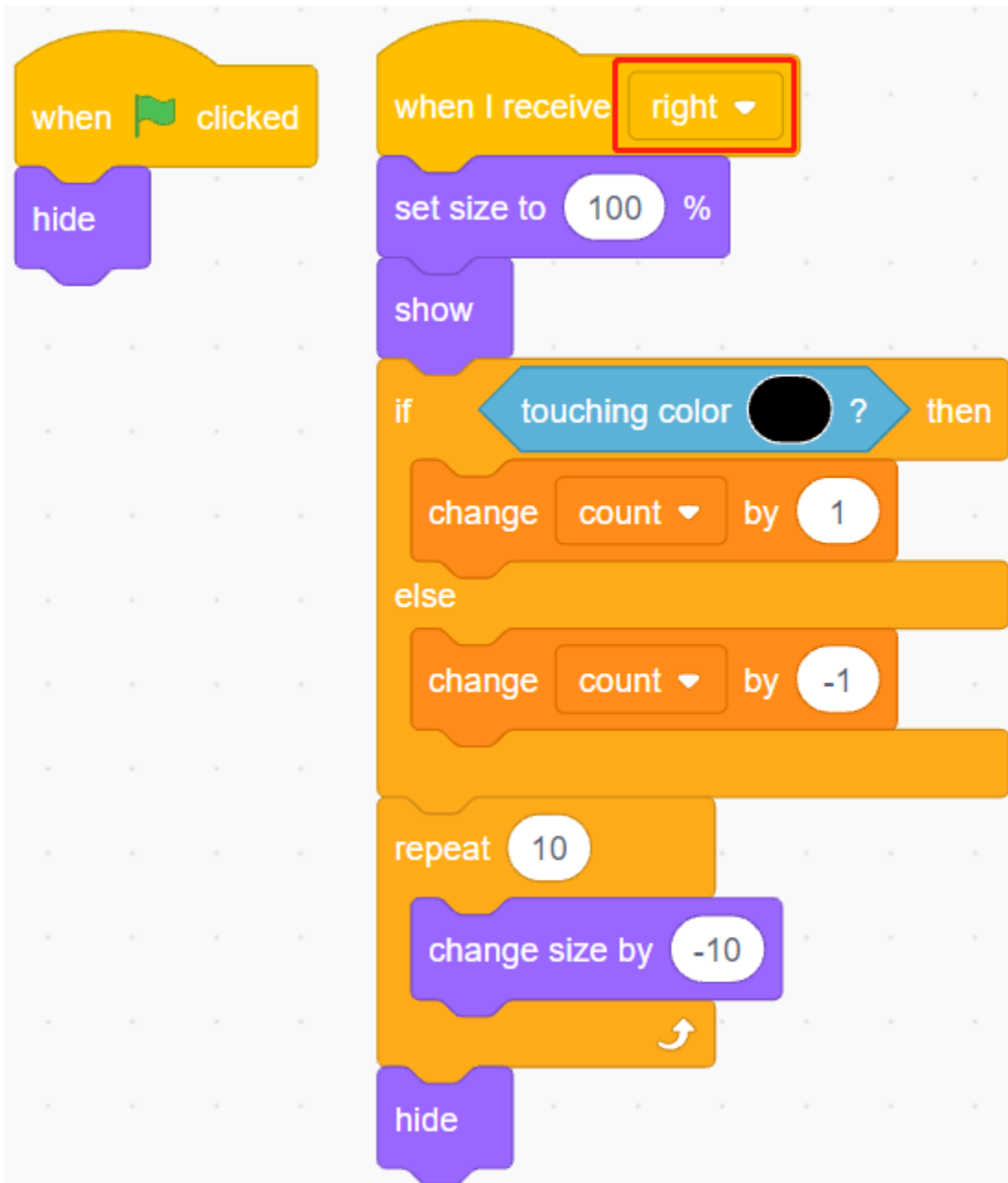


### 5. Sprite IR Droit

Copiez le sprite **Left IR** et renommez-le en **Right IR**.



Changez ensuite le message reçu en - **right**.



Maintenant que toute la programmation est terminée, vous pouvez cliquer sur le drapeau vert pour exécuter le script.

## 8.24 2.21 JEU - Protège ton Cœur

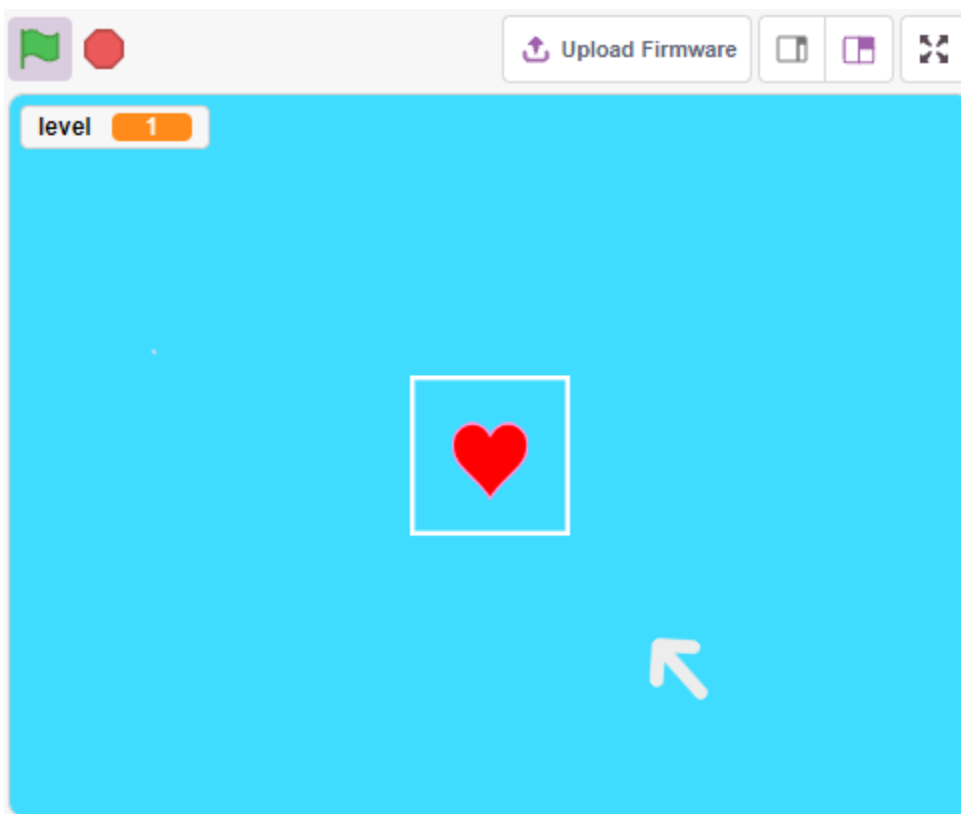
Dans ce projet, créons un jeu qui teste la vitesse de réaction.

Sur la scène, un cœur est protégé dans une boîte rectangulaire, et des flèches volent vers ce cœur depuis n'importe quelle position sur la scène. La couleur des flèches alterne aléatoirement entre noir et blanc, et leur vitesse augmente progressivement.

Si la couleur de la boîte rectangulaire et celle de la flèche sont identiques, la flèche est bloquée à l'extérieur et le niveau augmente de 1 ; si leurs couleurs diffèrent, la flèche transperce le cœur et la partie se termine.

Ici, la couleur de la boîte rectangulaire est contrôlée par le module de Suivi de Ligne. Lorsque ce module est placé sur une surface noire (une surface réfléchissante), la couleur de la boîte est noire, sinon elle est blanche.

Ainsi, vous devez décider de placer le module de Suivi de Ligne sur une surface blanche ou noire en fonction de la couleur de la flèche.



### 8.24.1 Composants requis

Pour ce projet, nous aurons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

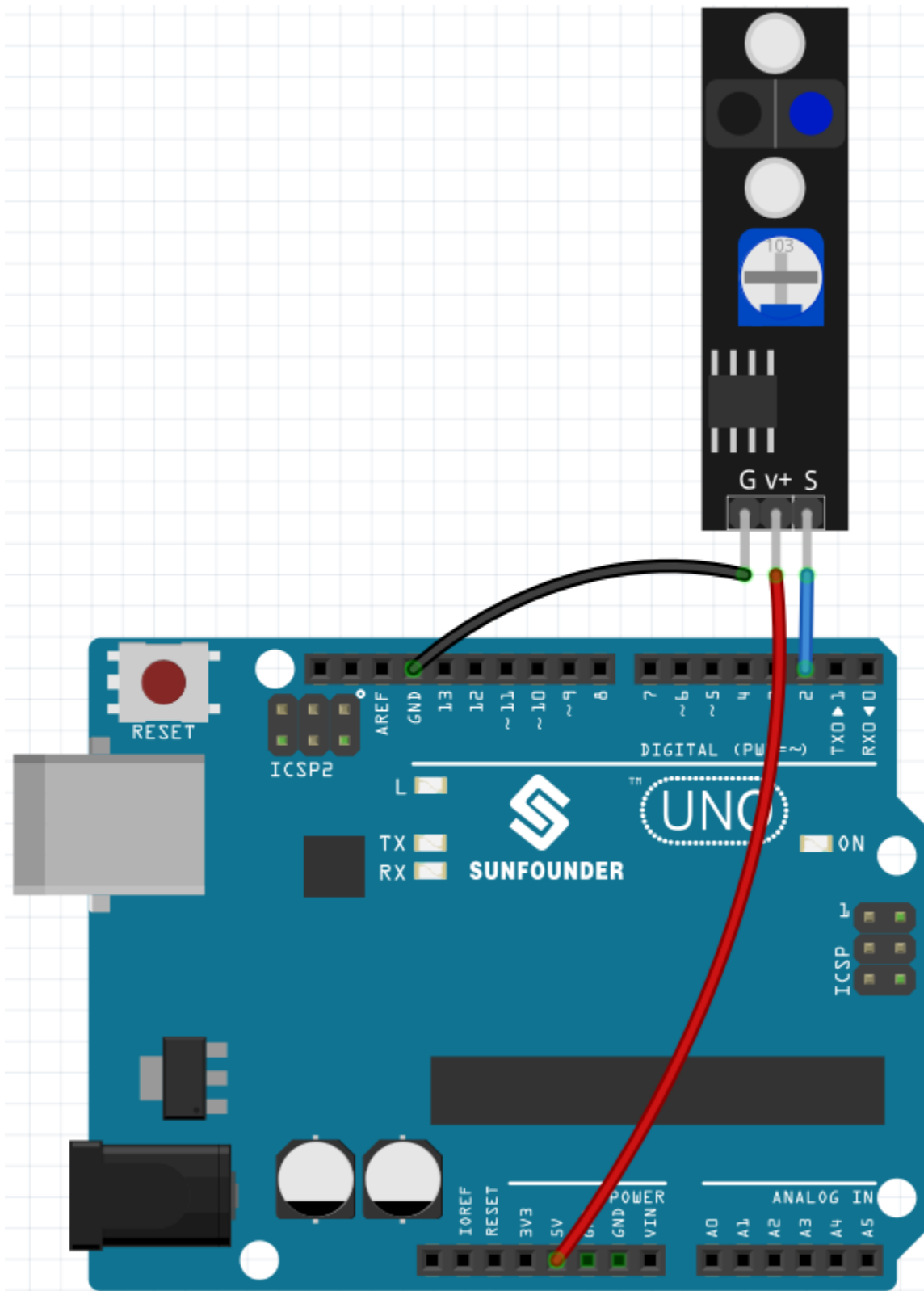


INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module de Suivi de Ligne</i>	

### 8.24.2 Construisez le Circuit

Il s'agit d'un module numérique de Suivi de Ligne, lorsqu'une ligne noire est détectée, il sort une valeur de 1 ; lorsqu'une ligne blanche est détectée, il sort une valeur de 0. De plus, vous pouvez ajuster sa distance de détection grâce au potentiomètre sur le module.

Construisez maintenant le circuit selon le schéma ci-dessous.



**Note :** Avant de commencer le projet, vous devez ajuster la sensibilité du module.

Câblez selon le schéma ci-dessus, puis alimentez la carte R3 (soit directement via le câble USB, soit via le câble de la

pile bouton 9V), sans télécharger le code.

Collez maintenant un ruban électrique noir sur le bureau, placez le module de Suivi de Ligne à une hauteur de 2 cm du bureau.

Avec le capteur orienté vers le bas, observez la LED de signal sur le module pour vous assurer qu'elle s'allume sur la table blanche et s'éteint sur le ruban noir.

Si ce n'est pas le cas, vous devez ajuster le potentiomètre sur le module, afin qu'il puisse produire l'effet ci-dessus.

### 8.24.3 Programmation

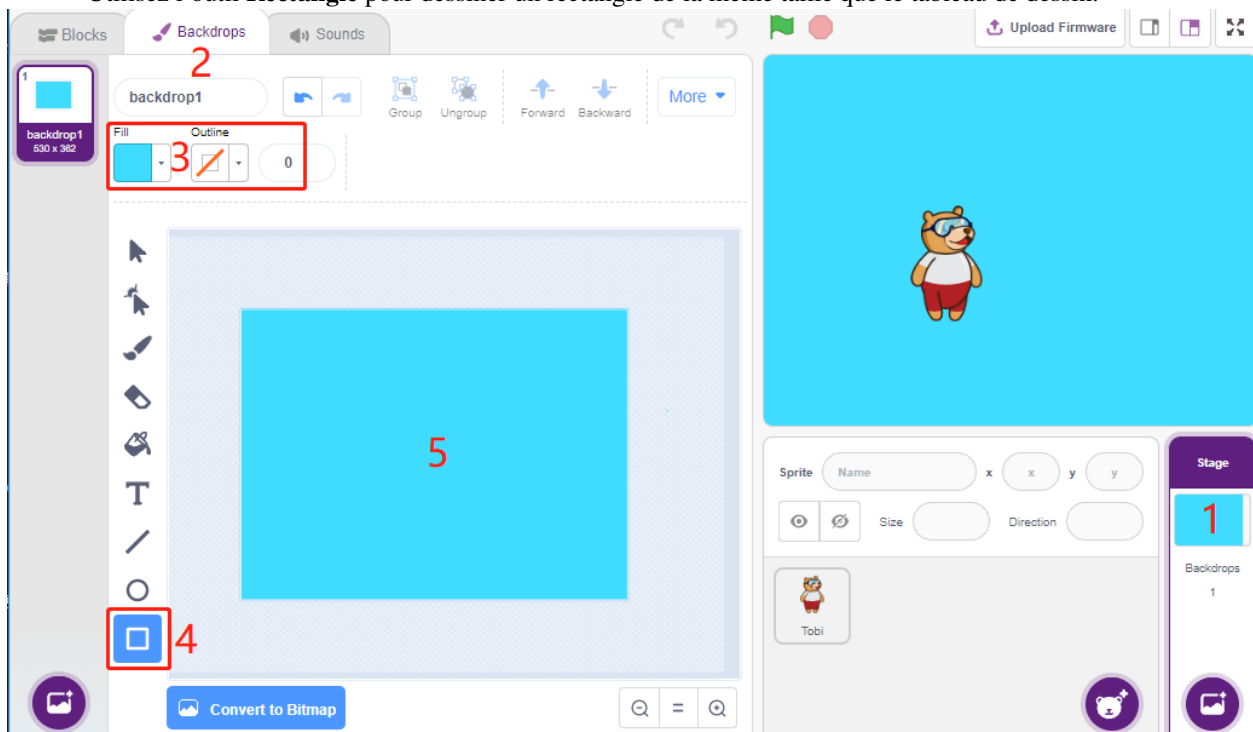
Ici, nous devons créer 3 sprites : **Heart**, **Square Box** et **Arrow1**.

- **Heart** : reste au milieu de la scène, si touché par le sprite **Arrow1**, la partie est terminée.
- **Square Box** : Deux types de costumes, noir et blanc, changeront selon la valeur du module de Suivi de Ligne.
- **Arrow** : vole vers le milieu de la scène depuis n'importe quelle position en noir/blanc ; si sa couleur correspond à celle du sprite **Square Box**, elle est bloquée et revole vers le milieu de la scène depuis une position aléatoire ; si sa couleur ne correspond pas à celle du sprite **Square Box**, elle traverse le sprite **Heart** et la partie se termine.

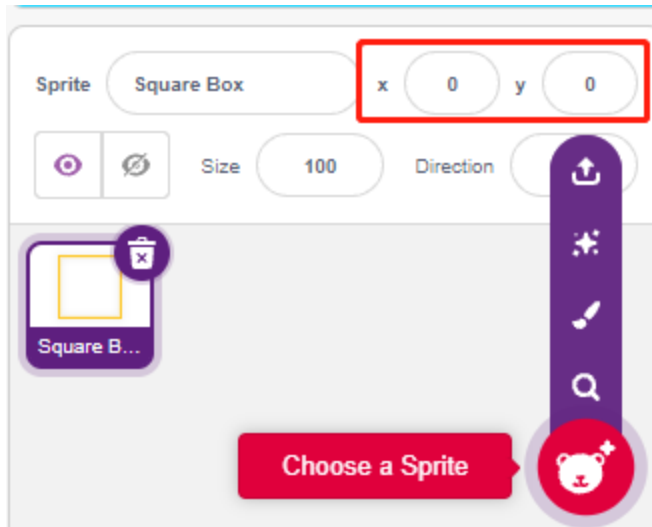
#### 1. Ajouter le sprite Boîte Carrée

Puisque les sprites Flèche1 et Boîte Carrée ont tous deux des costumes blancs, afin qu'ils puissent être affichés sur la scène, remplissez maintenant l'arrière-plan avec une couleur qui peut être n'importe quelle couleur sauf noir, blanc et rouge.

- Cliquez sur **Backdrop1** pour accéder à sa page **Backdrops**.
- Sélectionnez la couleur que vous souhaitez remplir.
- Utilisez l'outil **Rectangle** pour dessiner un rectangle de la même taille que le tableau de dessin.

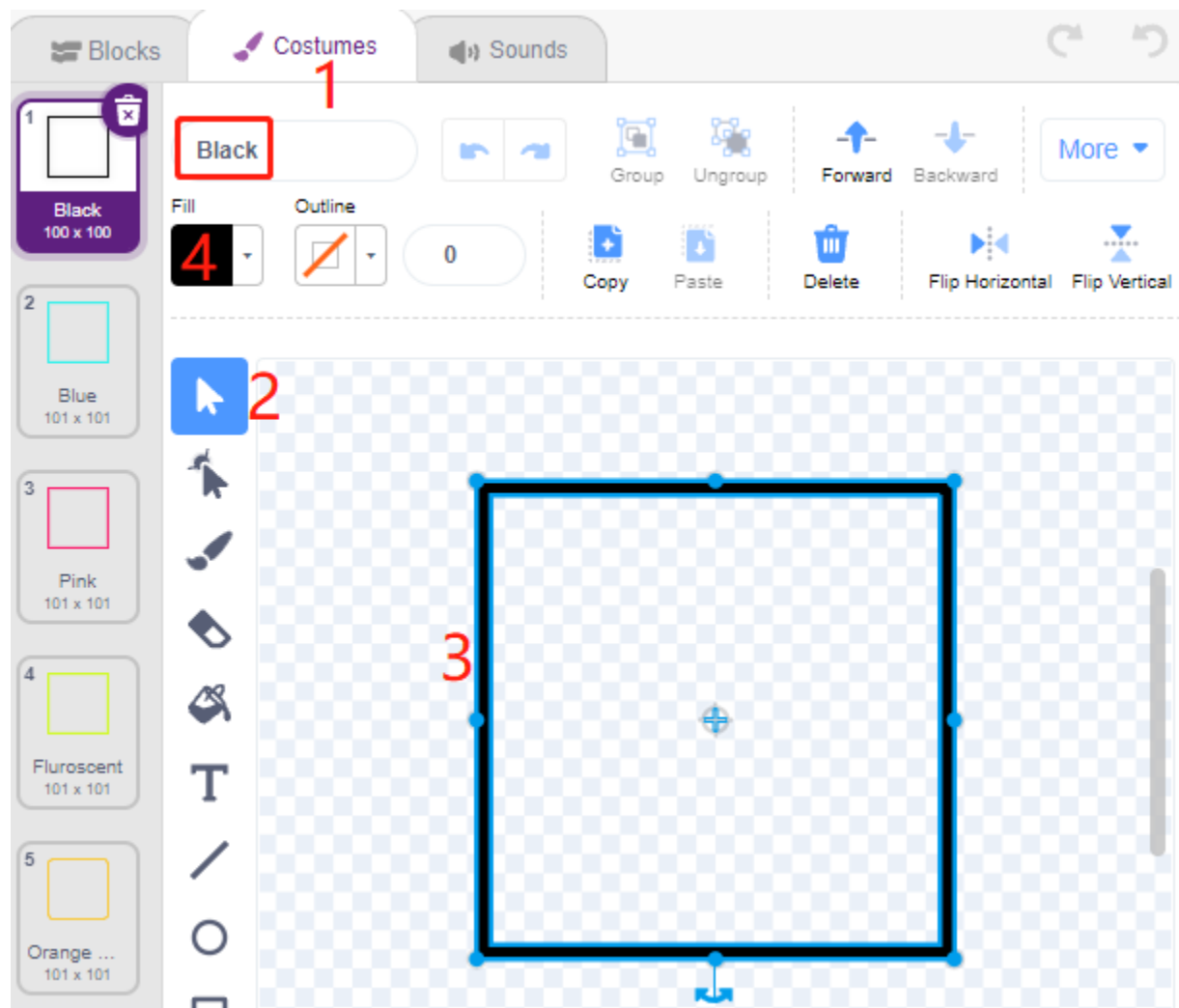


Supprimez le sprite par défaut, utilisez le bouton **Choose a Sprite** pour ajouter le sprite **Square Box**, et réglez ses coordonnées x et y à (0, 0).

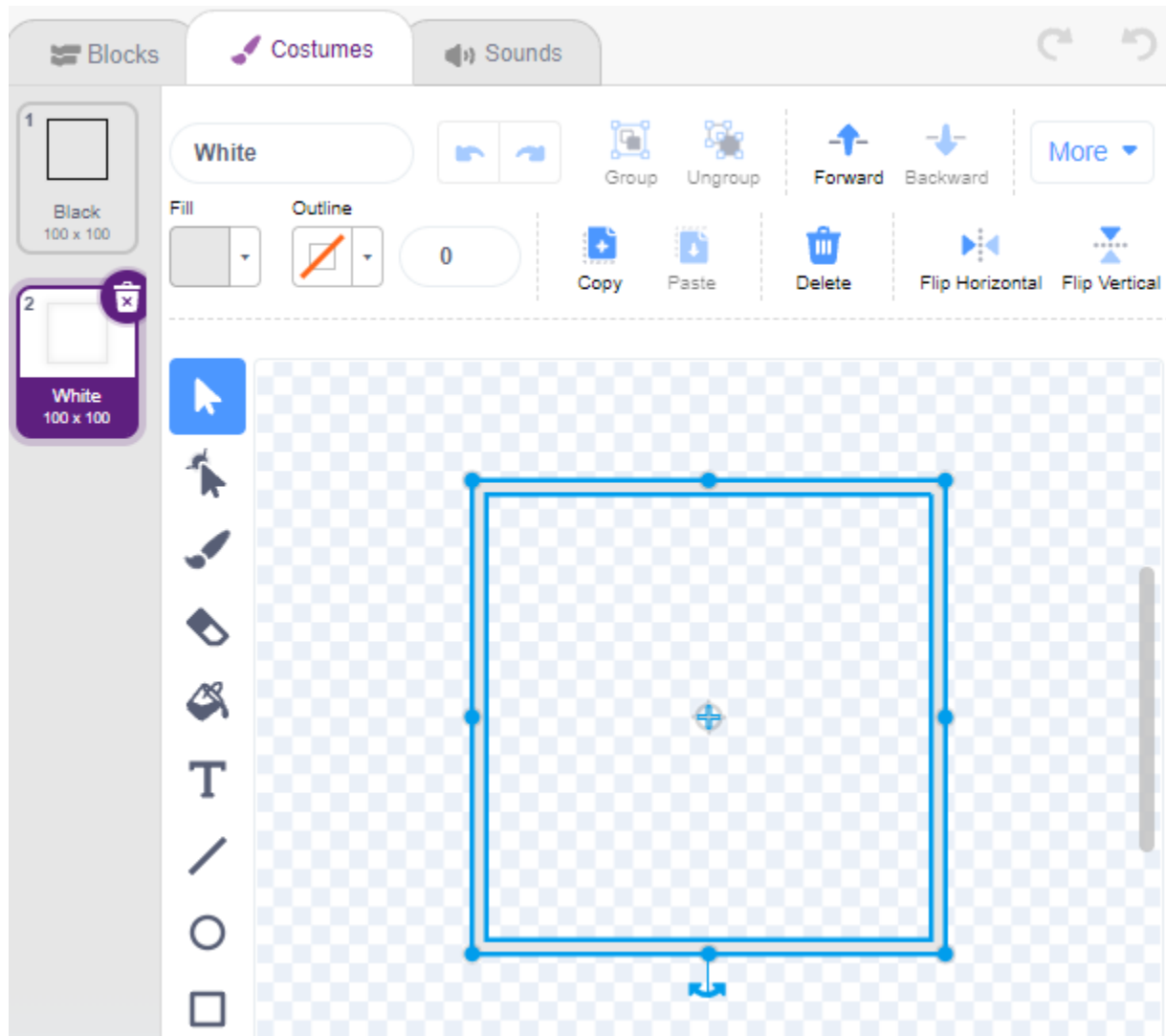


Allez à la page **Costumes** du sprite **Square Box** et réglez les costumes noir et blanc.

- Cliquez sur l'outil de sélection
- Sélectionnez le rectangle sur le canevas
- Sélectionnez la couleur de remplissage en noir
- et nommez le costume **Black**

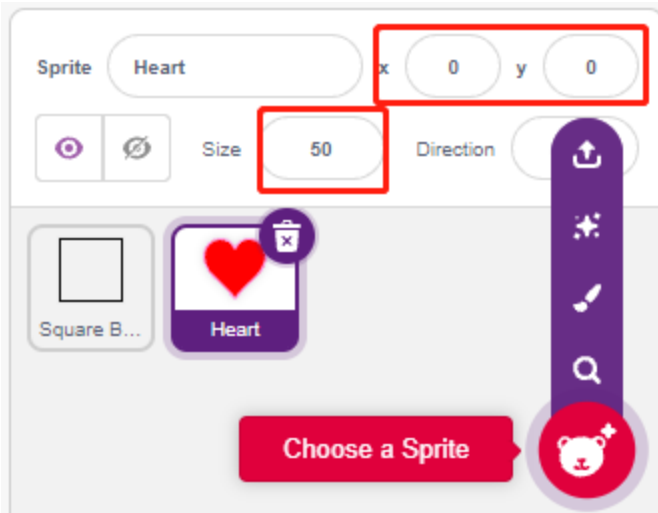


Sélectionnez le deuxième costume, réglez la couleur de remplissage en blanc, nommez-le Blanc et supprimez les autres costumes.

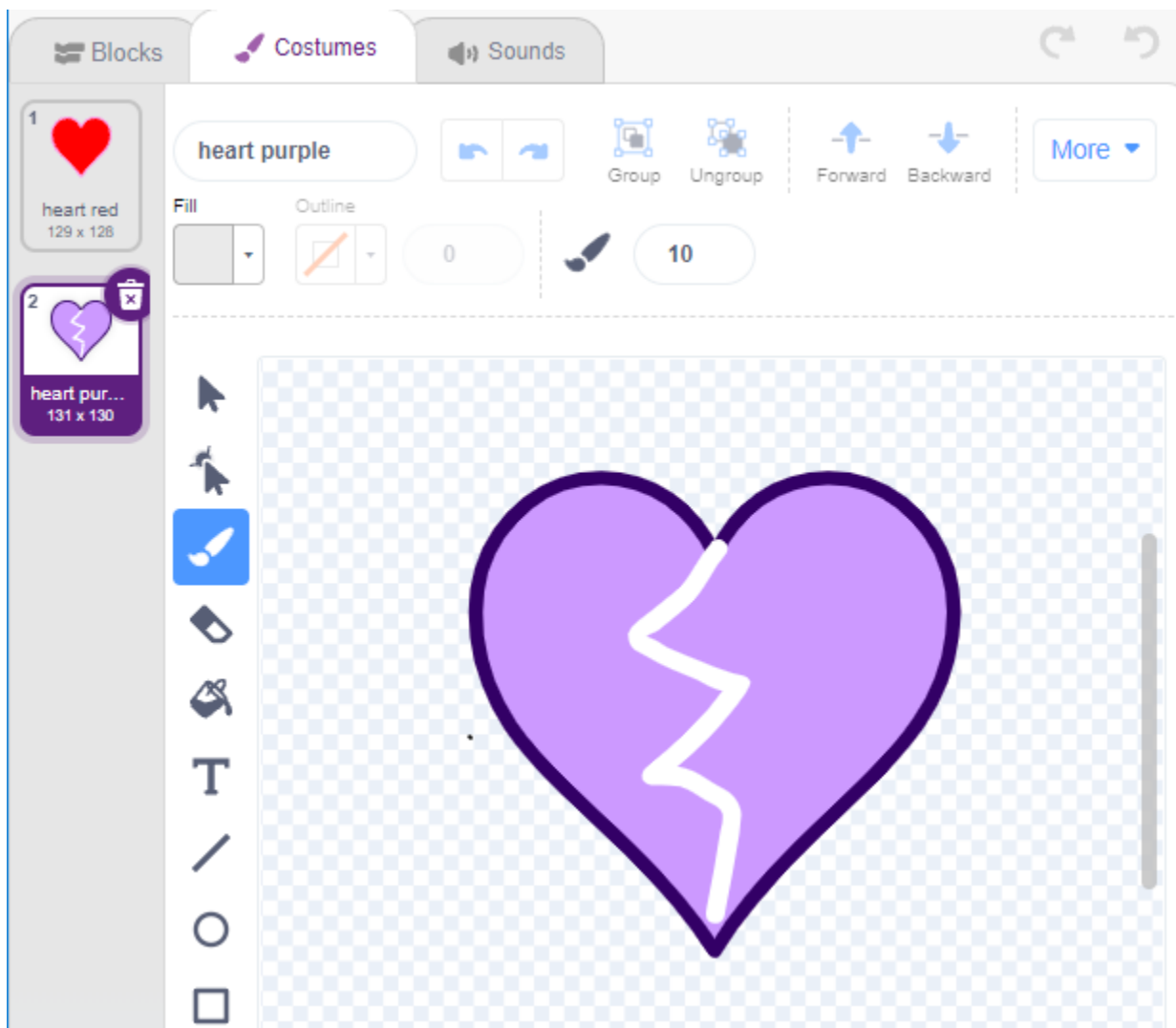


## 2. Ajouter le sprite Cœur

Ajoutez également un sprite **Heart**, réglez sa position à (0, 0), et réduisez sa taille pour qu'il semble être situé à l'intérieur de la Boîte Carrée.

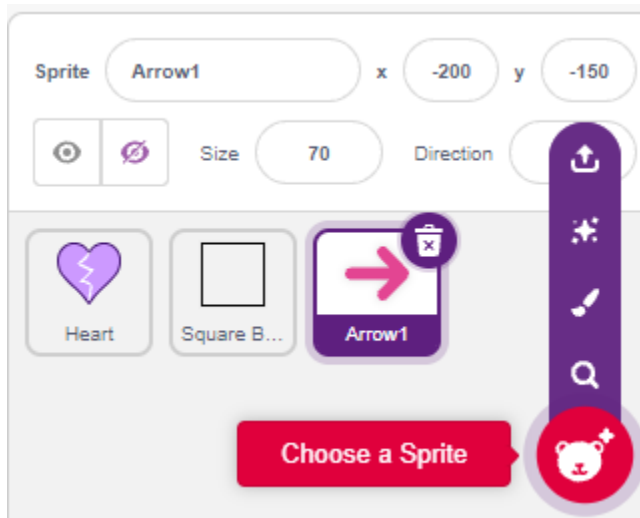


Sur la page **Costumes**, ajustez le costume violet du cœur pour qu'il semble être brisé.

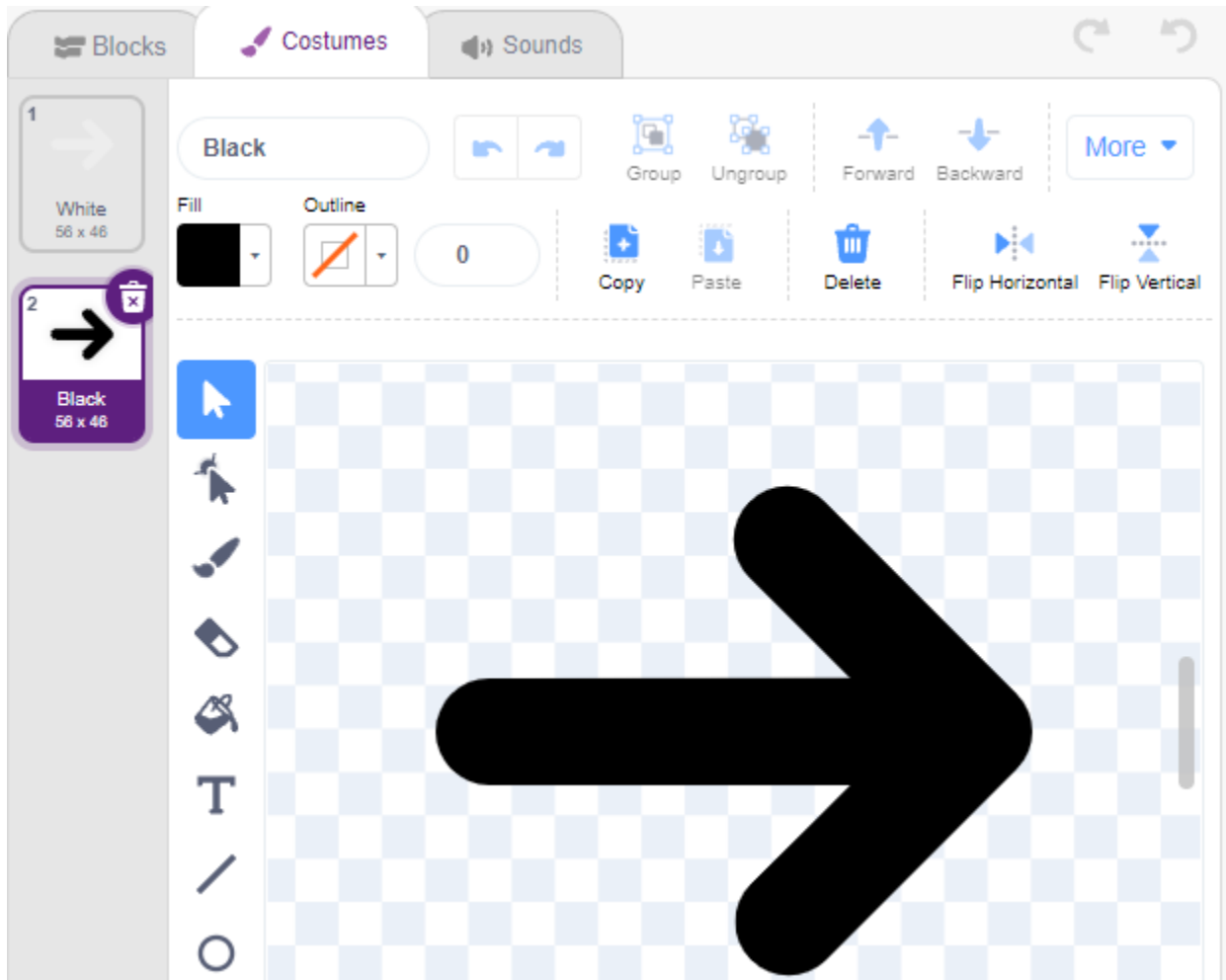


### 3. Ajouter le sprite Flèche1

Ajoutez un sprite **Arrow1**.



Sur la page **Costumes**, gardez et copiez le costume orienté vers la droite et réglez sa couleur en noir et blanc.

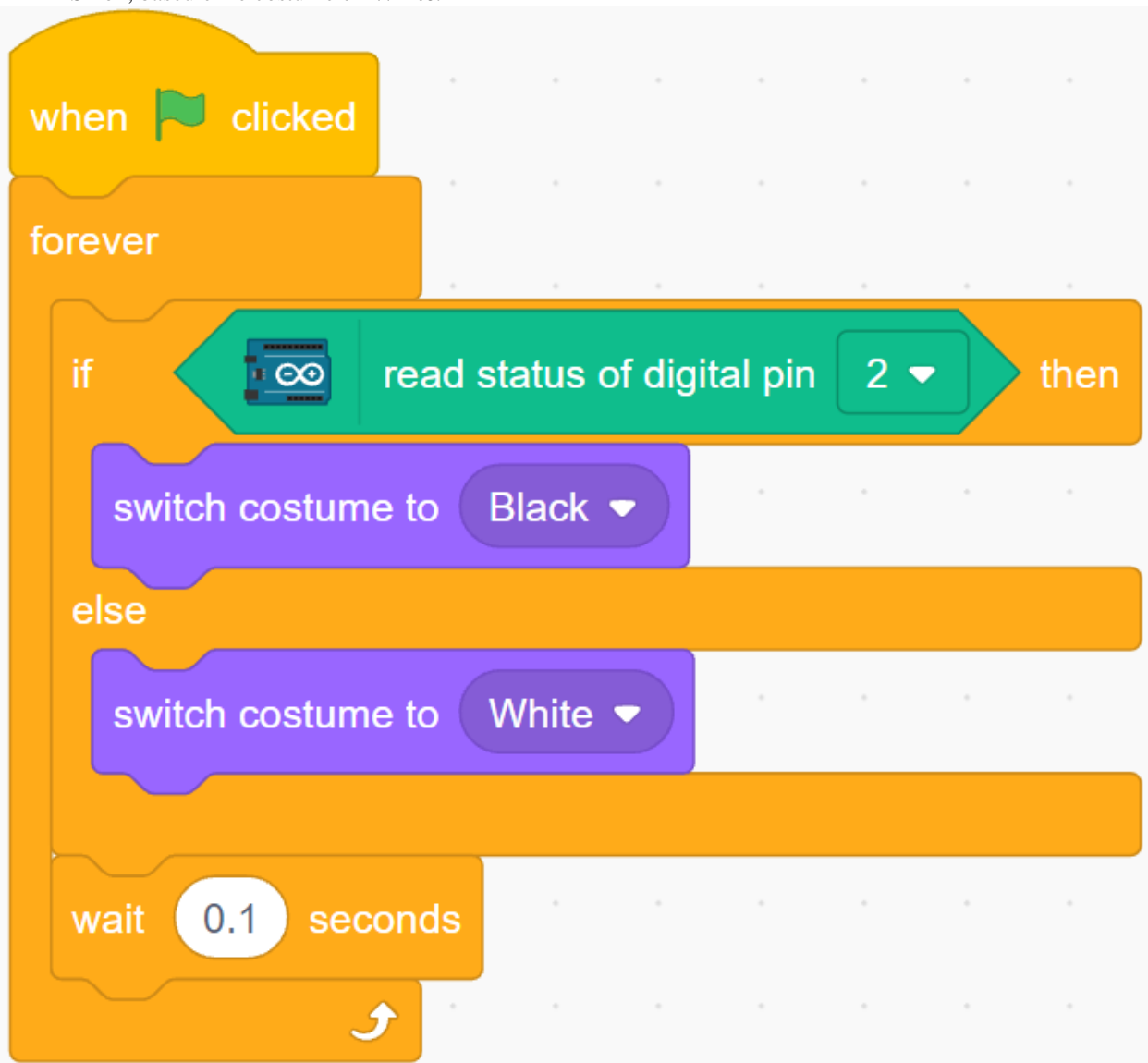


#### 4. Script pour le sprite Boîte Carrée

Retournez à la page **Blocks** et programmez le sprite **Square Box**.

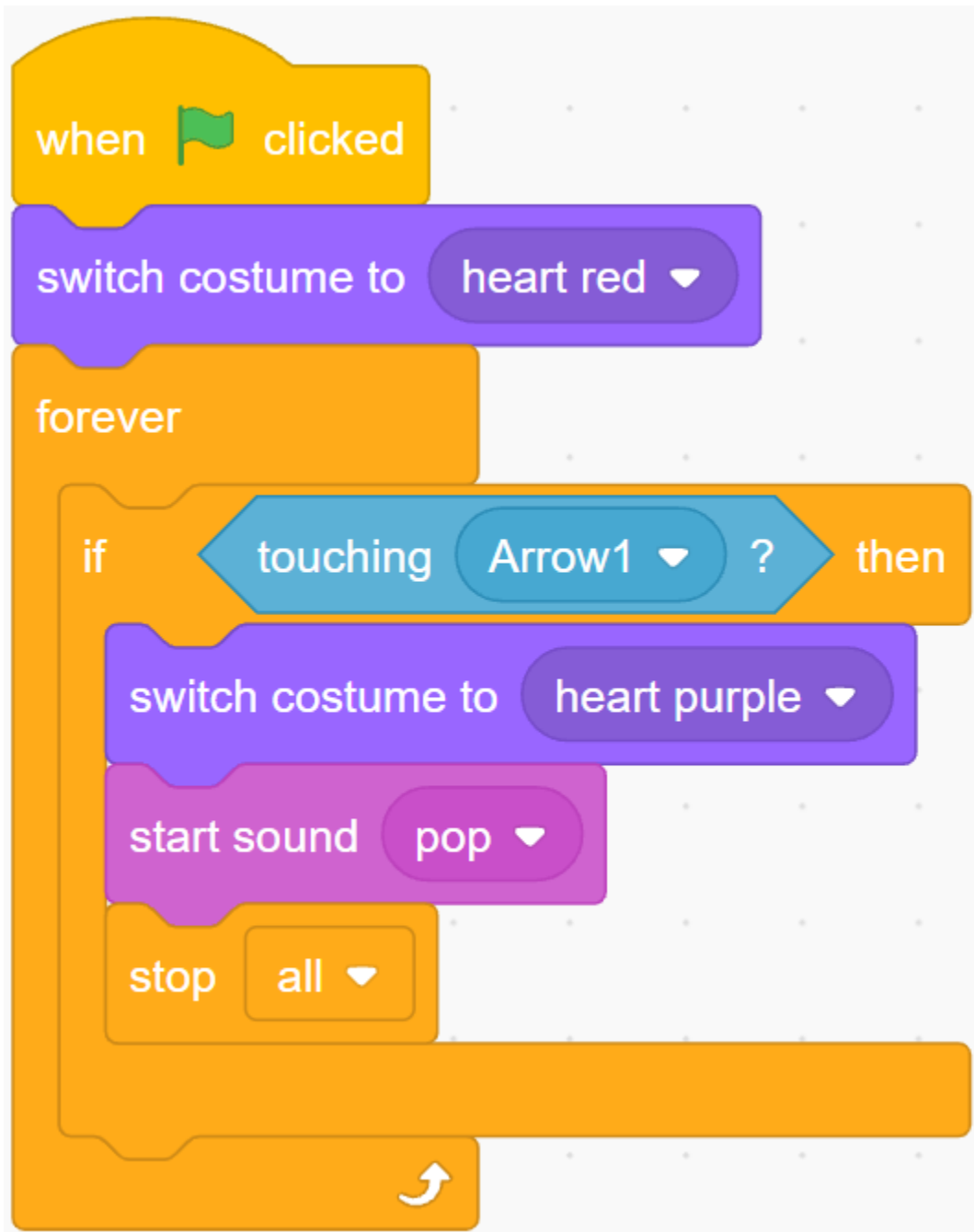


- Ainsi, lorsque la valeur de la broche numérique 2 (module de Suivi de Ligne) est 1 (ligne noire détectée), alors changez le costume pour **Black**.
- Sinon, basculez le costume en **White**.



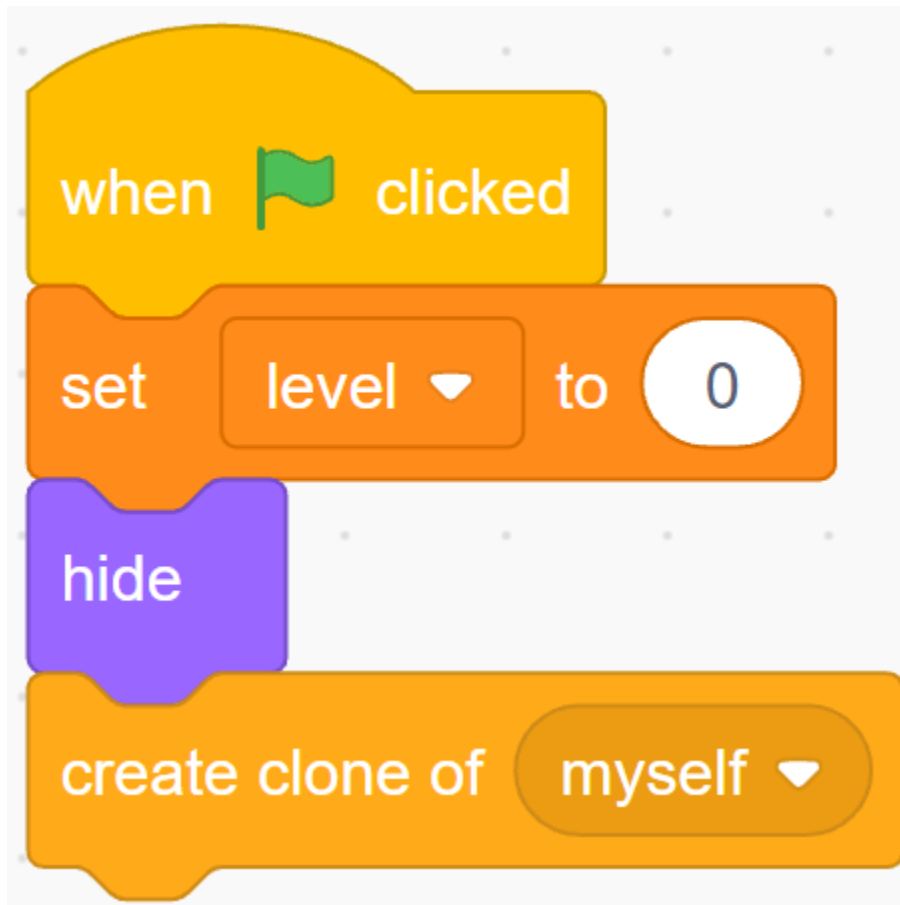
### 5. Script pour le sprite Cœur

Le sprite **Heart** est protégé à l'intérieur de **Square Box**, et par défaut a un costume rouge. Lorsque le sprite Flèche1 le touche, la partie se termine.



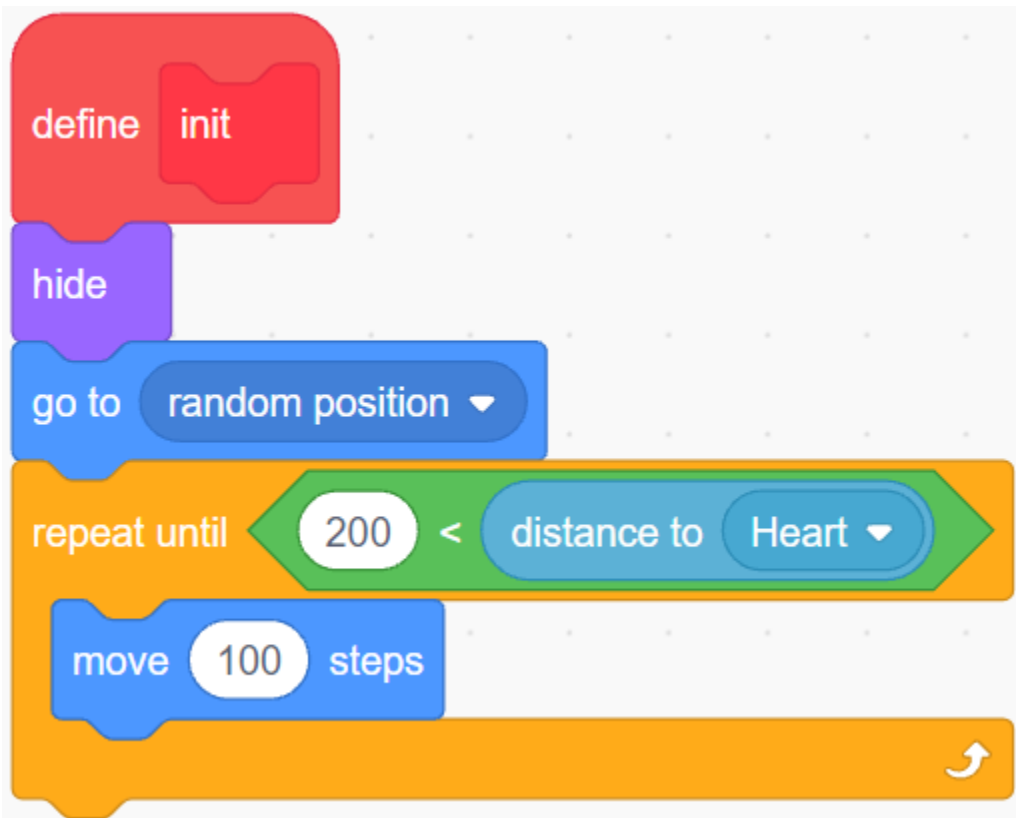
#### 6. Script pour le sprite Flèche1

Faites que le sprite **Arrow1** se cache et crée un clone lorsque le drapeau vert est cliqué.

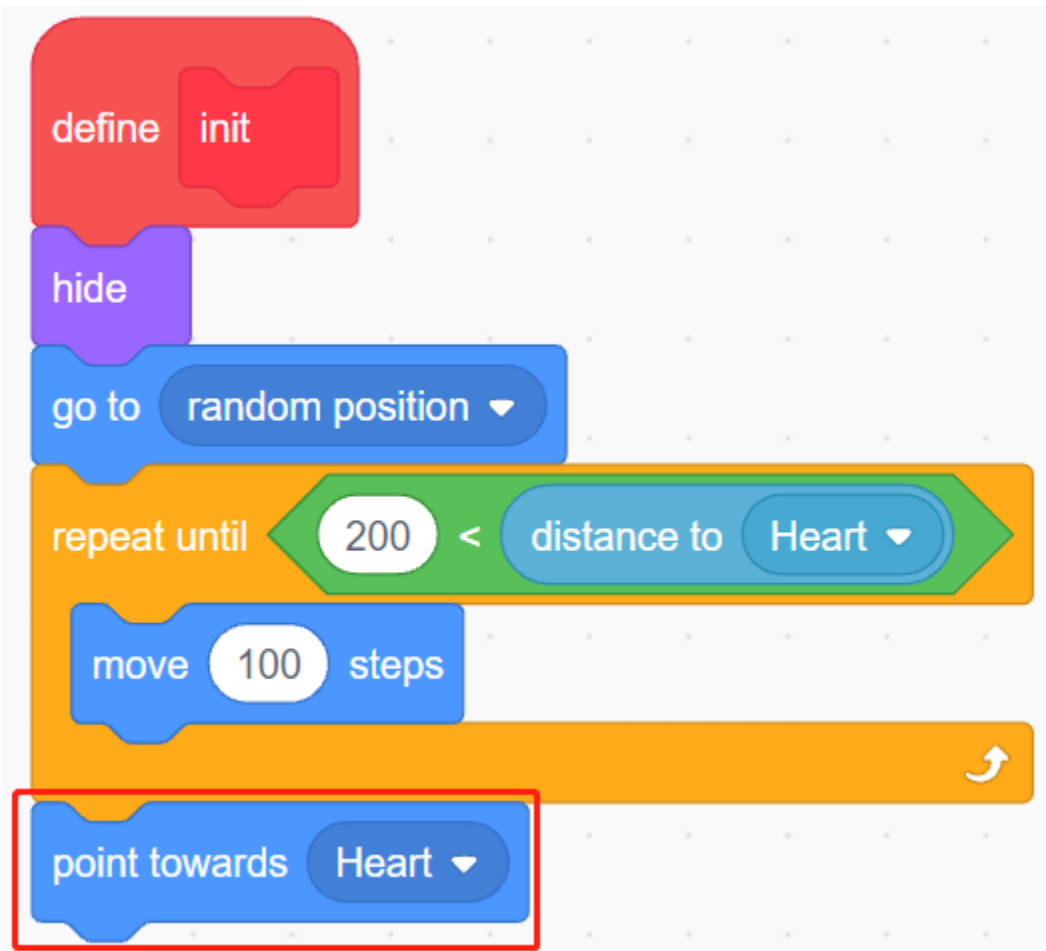


Créez un bloc [init] pour initialiser la position, l'orientation et la couleur du sprite **Arrow1**.

Il apparaît à un endroit aléatoire, et si la distance entre lui et le sprite **Heart** est inférieure à 200, il se déplace vers l'extérieur jusqu'à ce que la distance soit supérieure à 200.

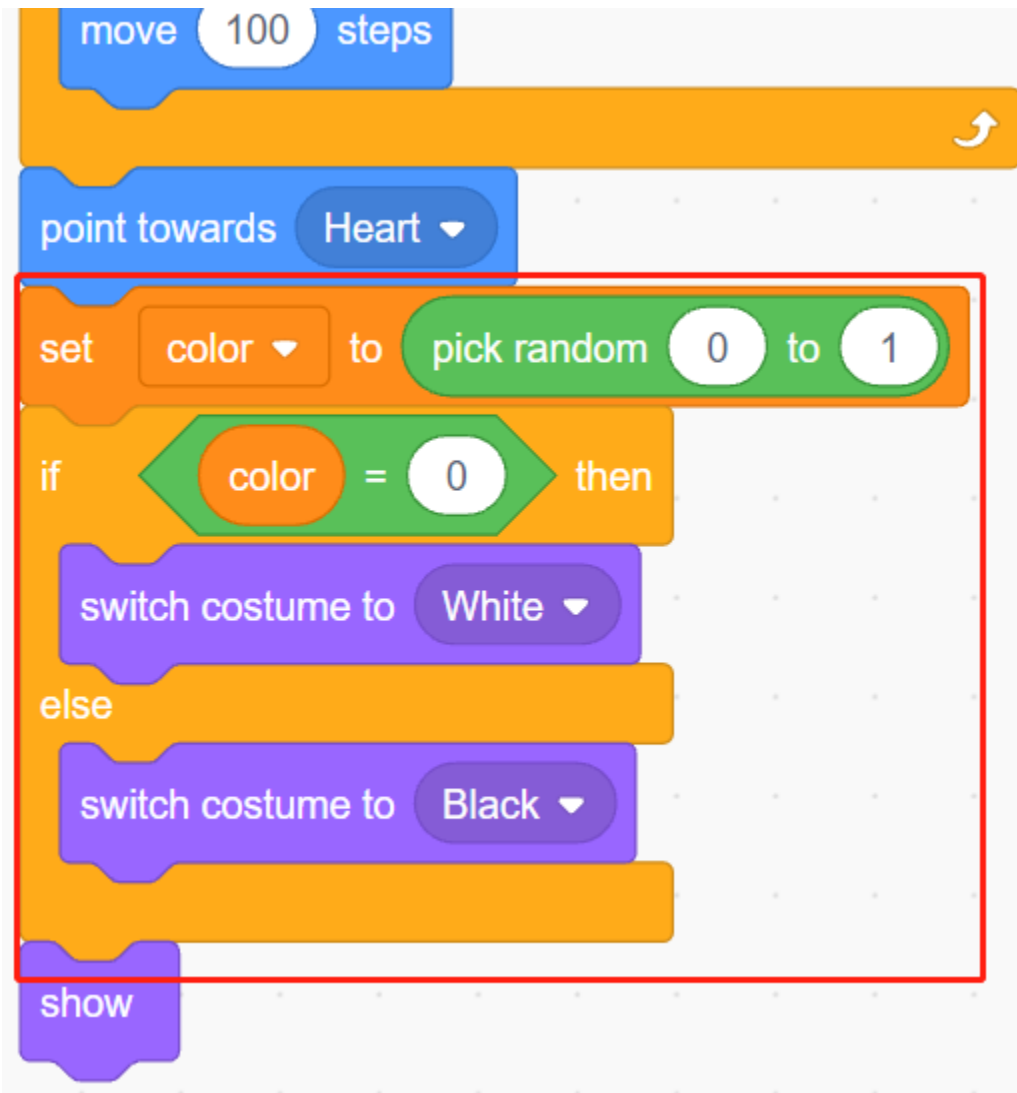


Réglez sa direction face au sprite **Heart**.

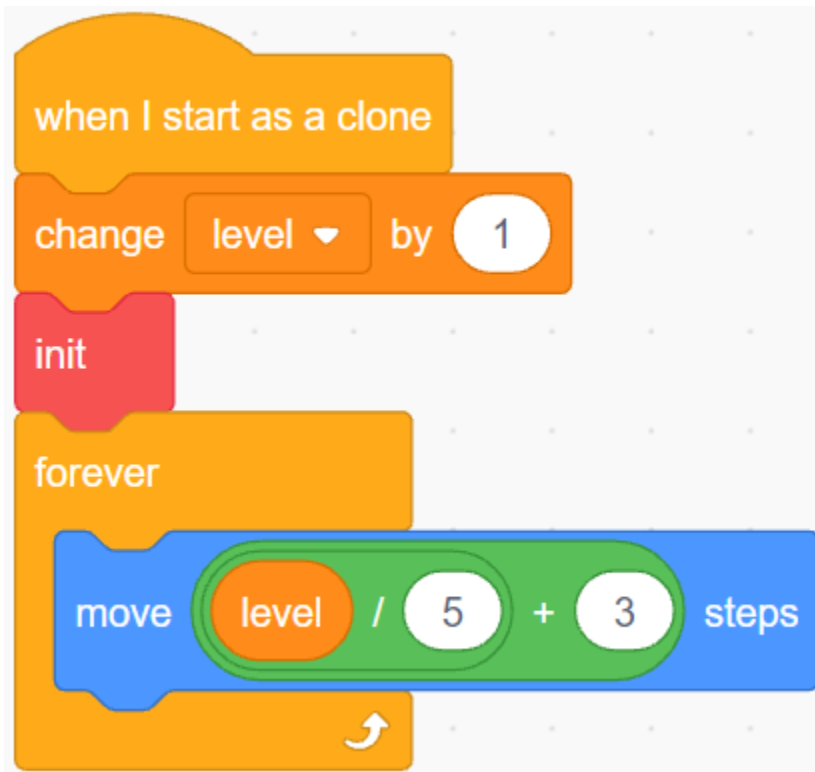


Faites alterner sa couleur aléatoirement entre noir/blanc.

- Si la variable couleur est 0, basculez le costume en **White**.
- Si la variable couleur est 1, basculez le costume en **Black**.

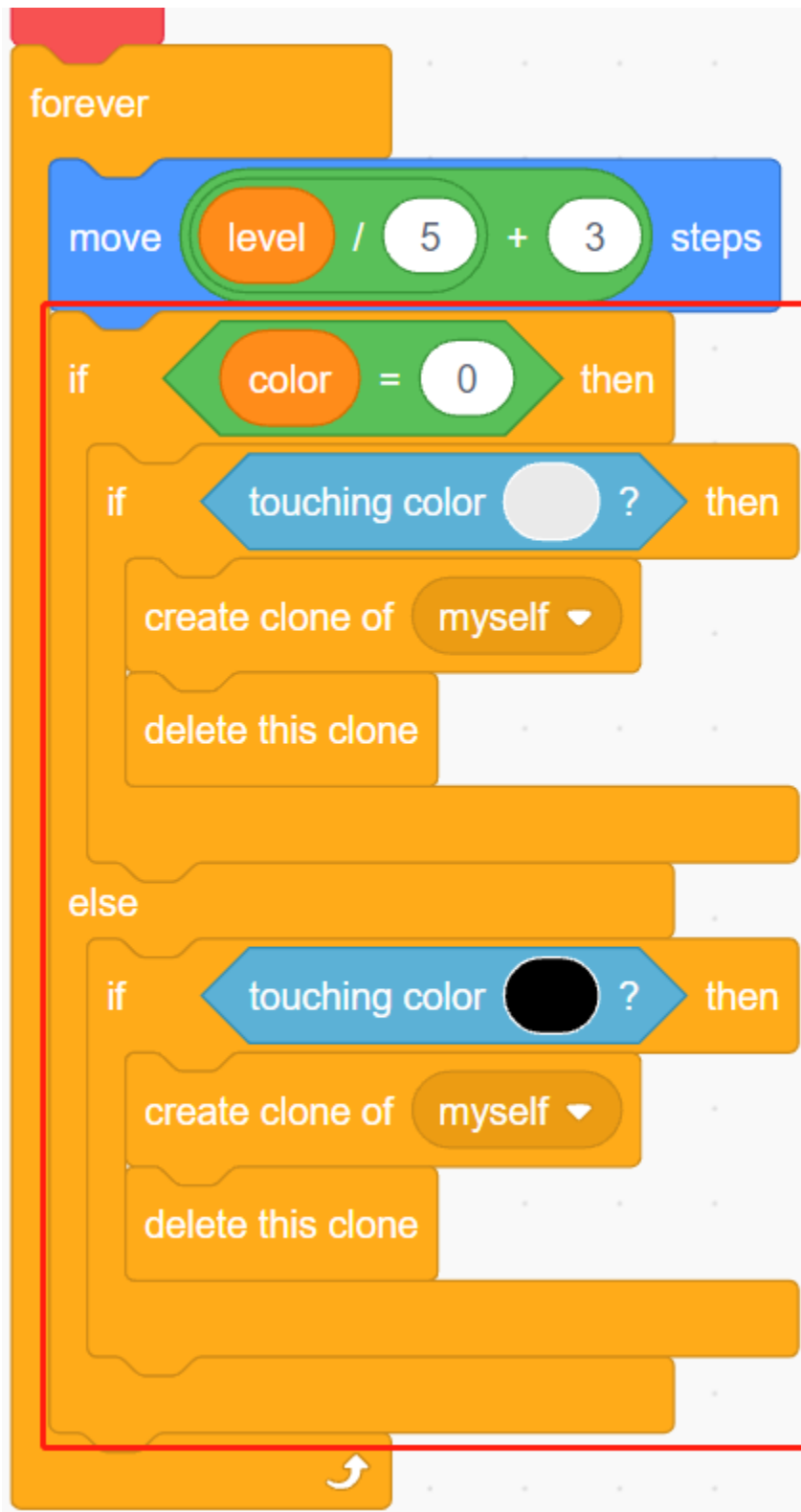


Maintenant, laissez-le commencer à bouger, il se déplacera plus vite au fur et à mesure que la valeur de la variable **level** augmente.



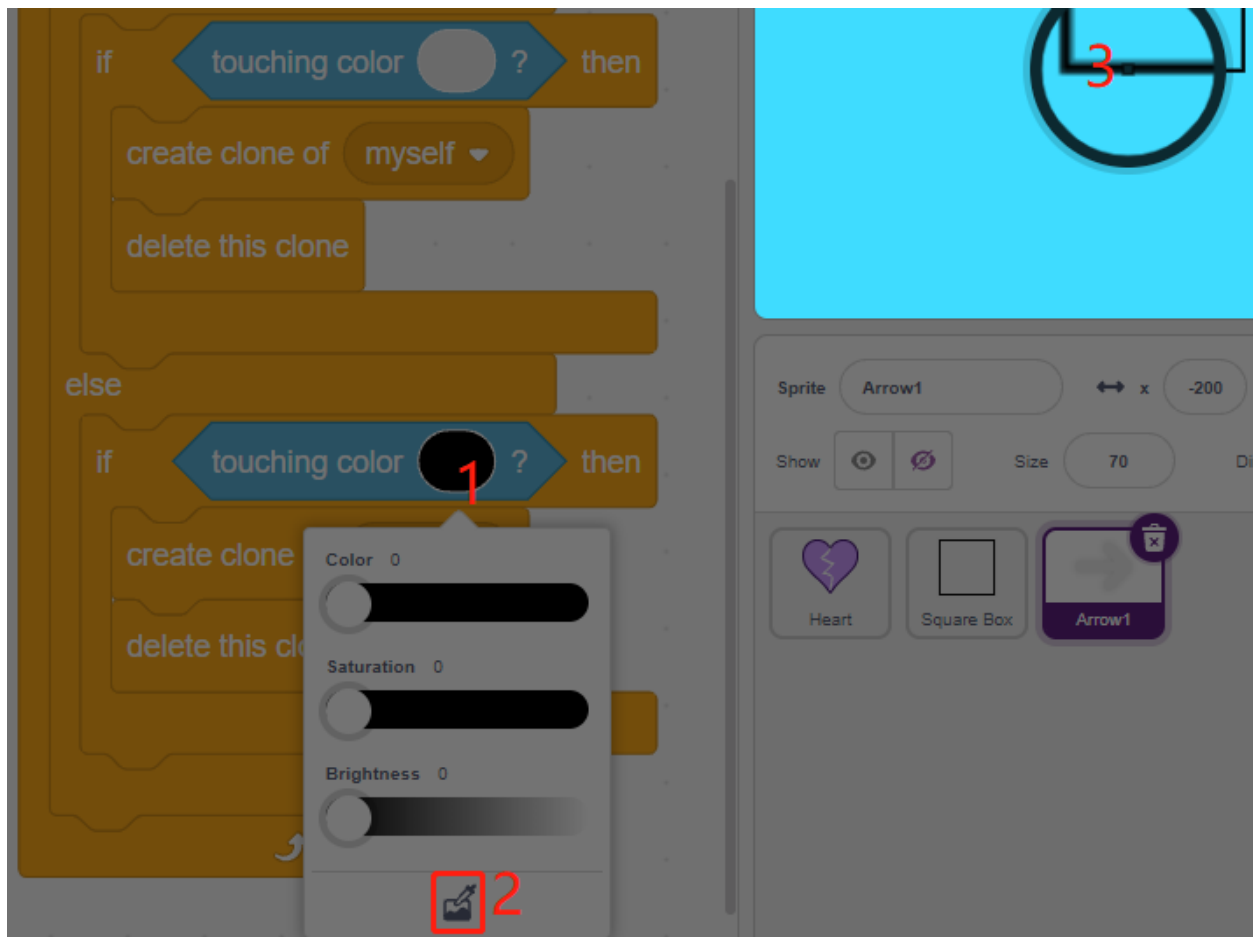
Maintenant, réglez son effet de collision avec le sprite **Square Box**.

- Si le sprite **Arrow1** et le sprite **Square Box** ont la même couleur (qui sera modifiée selon la valeur du module de Suivi de Ligne), soit noir soit blanc, un nouveau clone est créé et la partie continue.
- Si leurs couleurs ne correspondent pas, le sprite **Arrow1** continue de se déplacer et la partie se termine lorsqu'il heurte le sprite **Heart**.



**Note :** Les deux blocs [touch color()] doivent prendre en compte les costumes noir/blanc de Boîte Carrée séparément.

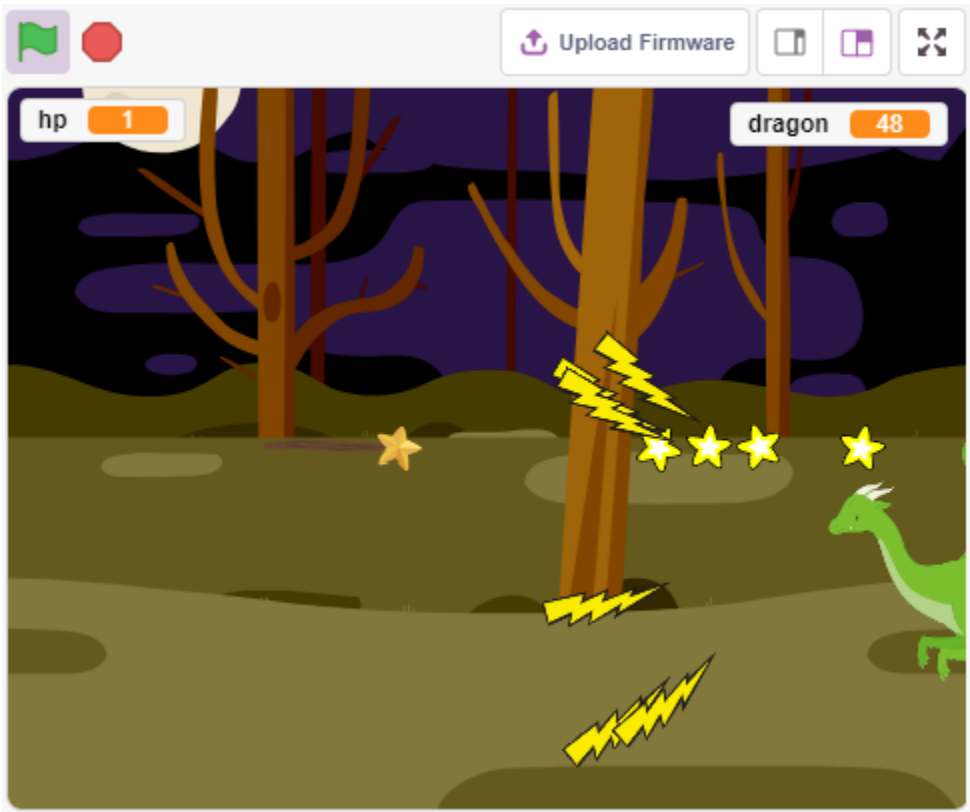




## 8.25 2.22 JEU - Tueur de Dragon

Ici, nous utilisons un joystick pour jouer à un jeu de chasse au dragon.

En cliquant sur le bouton vert, le dragon flottera de haut en bas sur le côté droit et crachera du feu par intermittence. Vous devez utiliser le joystick pour contrôler le mouvement de la baguette magique et lancer des attaques d'étoiles sur le dragon, tout en évitant les flammes qu'il projette, et finalement le vaincre.



8.25.1 Vous Apprendrez

Pour ce projet, nous aurons besoin des composants suivants.  
Il est certainement pratique d’acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Fils de Cavalier</i>	
<i>Module de Joystick</i>	-

### 8.25.2 Construire le Circuit

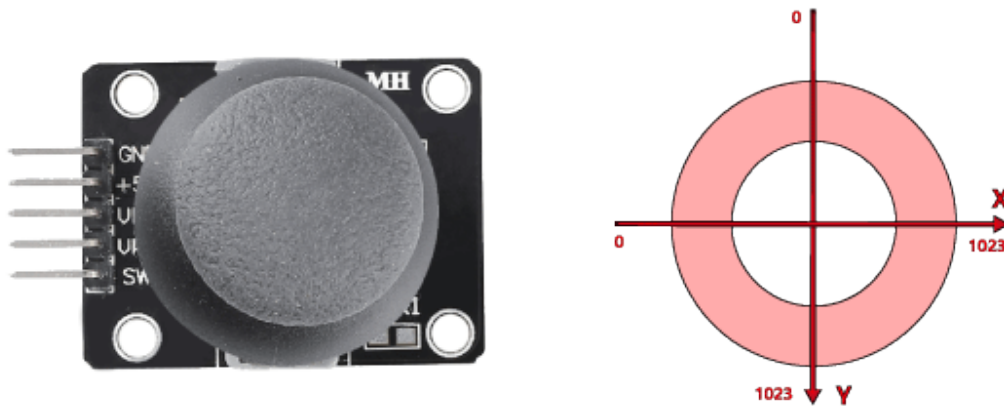
Un joystick est un dispositif d'entrée composé d'un bâton qui pivote sur une base et transmet son angle ou sa direction au dispositif qu'il contrôle. Les joysticks sont souvent utilisés pour contrôler des jeux vidéo et des robots.

Pour communiquer une gamme complète de mouvements à l'ordinateur, un joystick doit mesurer la position du bâton sur deux axes – l'axe X (de gauche à droite) et l'axe Y (de haut en bas).

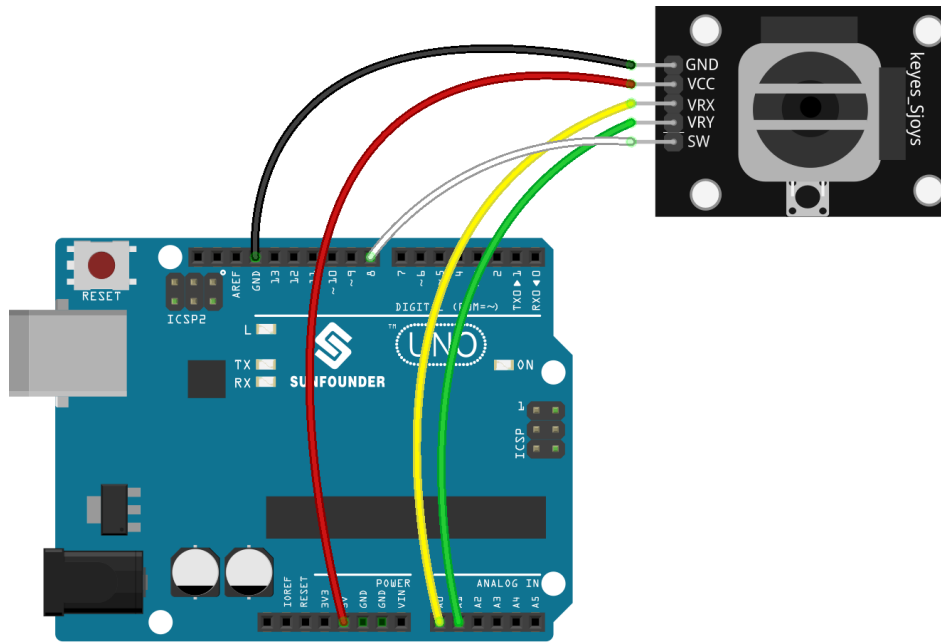
Les coordonnées de mouvement du joystick sont montrées dans la figure suivante.

**Note :**

- La coordonnée x va de gauche à droite, la plage est de 0 à 1023.
- La coordonnée y va de haut en bas, la plage est de 0 à 1023.



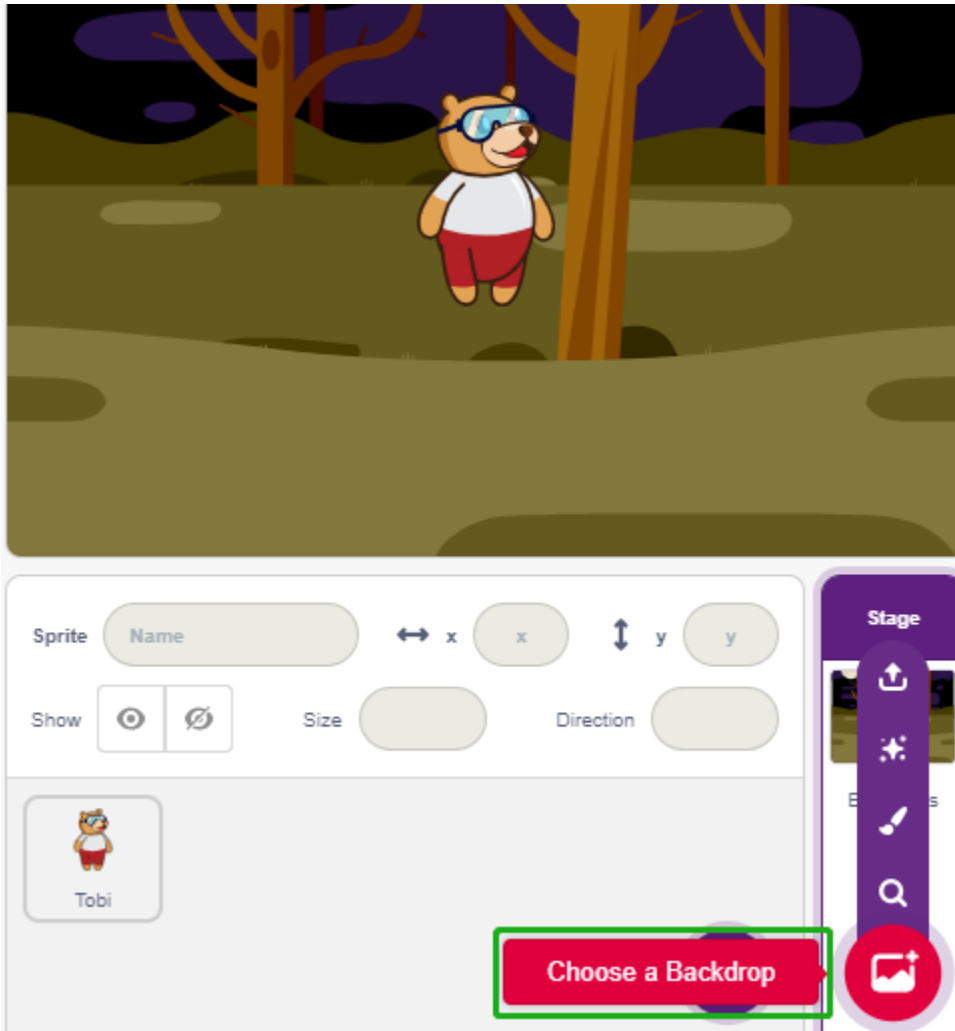
Maintenant, construisez le circuit selon le schéma suivant.



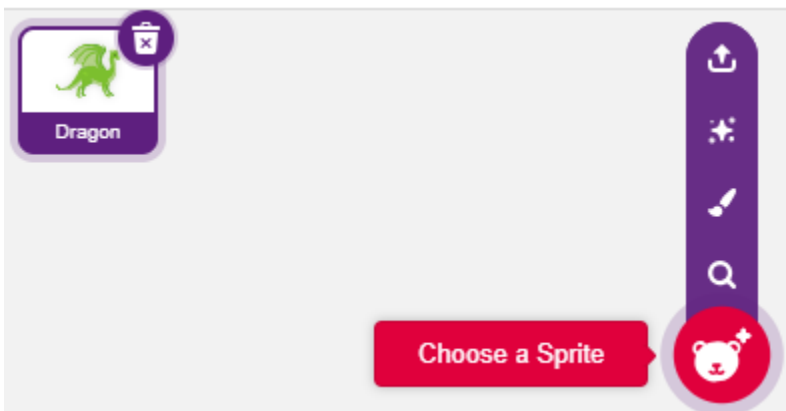
### 8.25.3 Programmation

#### 1. Dragon

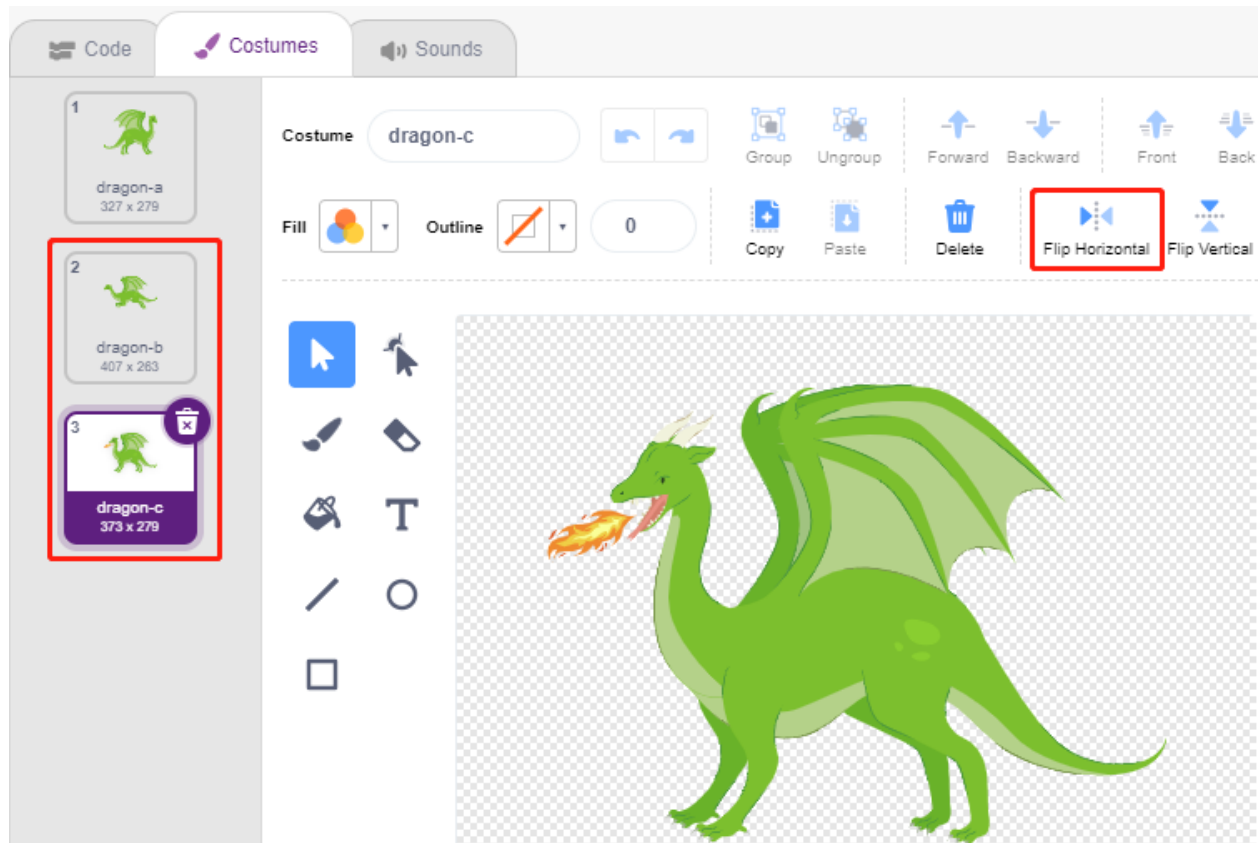
Ajout du décor **Woods** via le bouton **Choose a Backdrop**.



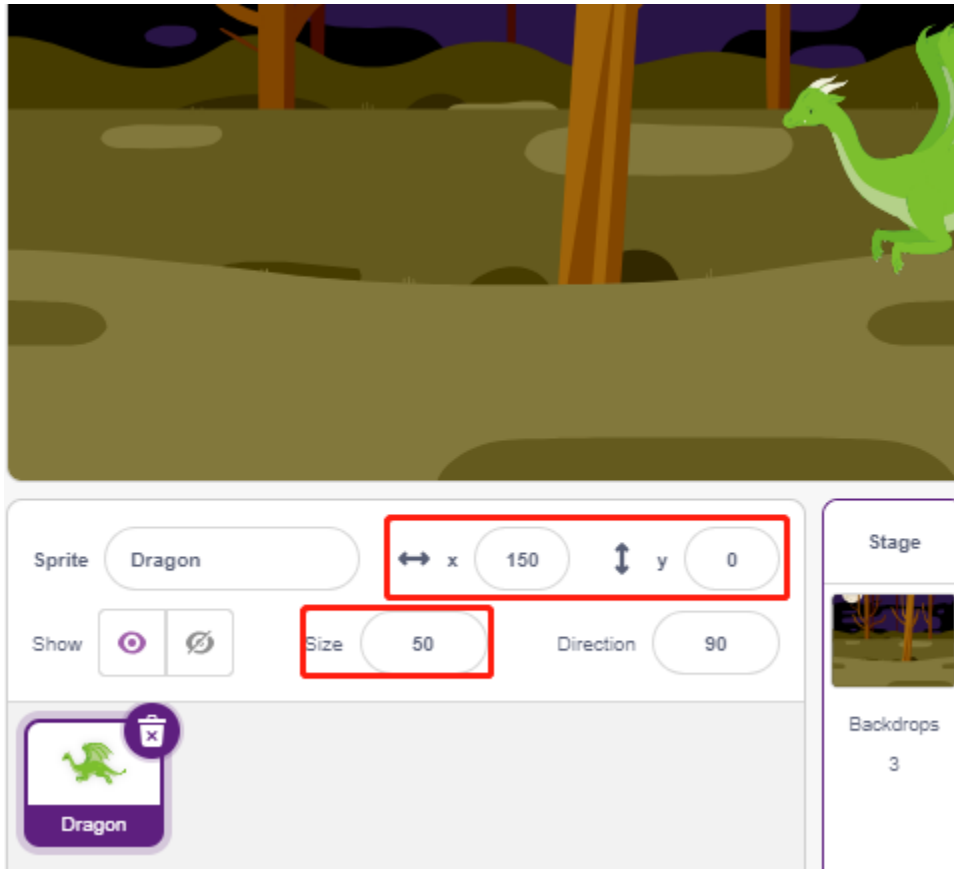
— Supprimez le sprite par défaut et ajoutez le sprite **Dragon**.



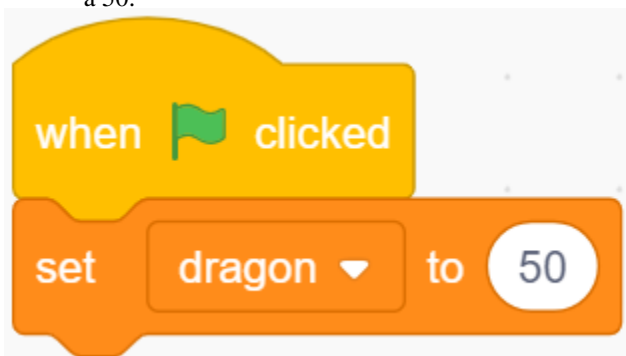
— Allez à la page **Costumes** et retournez horizontalement les costumes dragon-b et dragon-c.



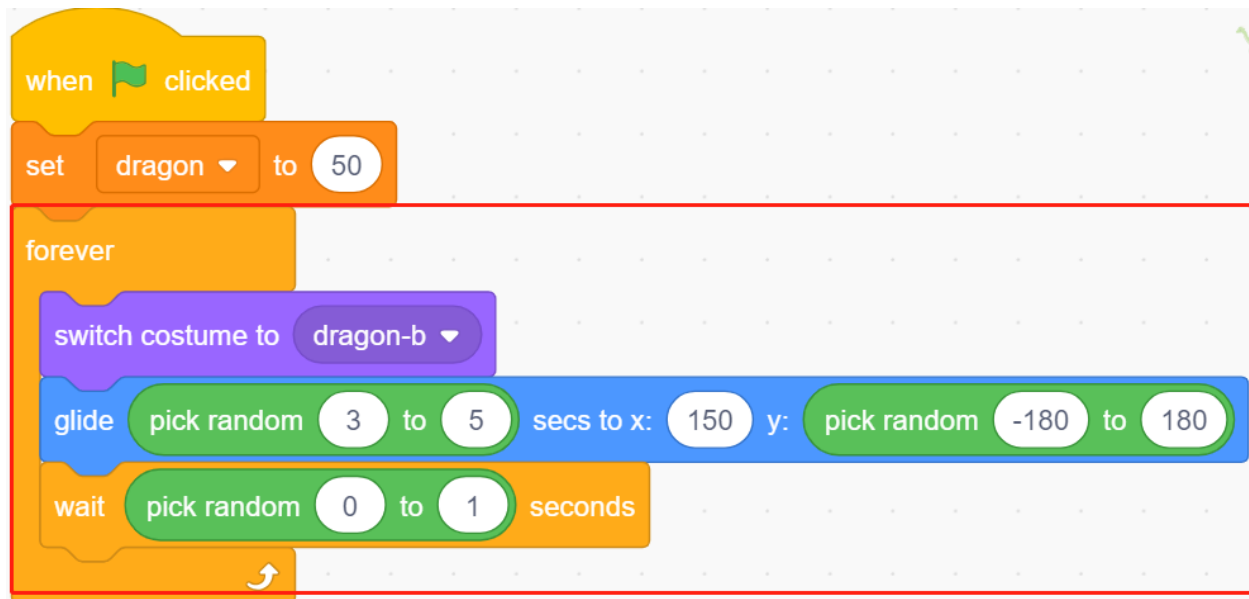
— Réglez la taille à 50%.



- Créez maintenant une variable - **dragon** pour enregistrer les points de vie du dragon, et réglez la valeur initiale à 50.

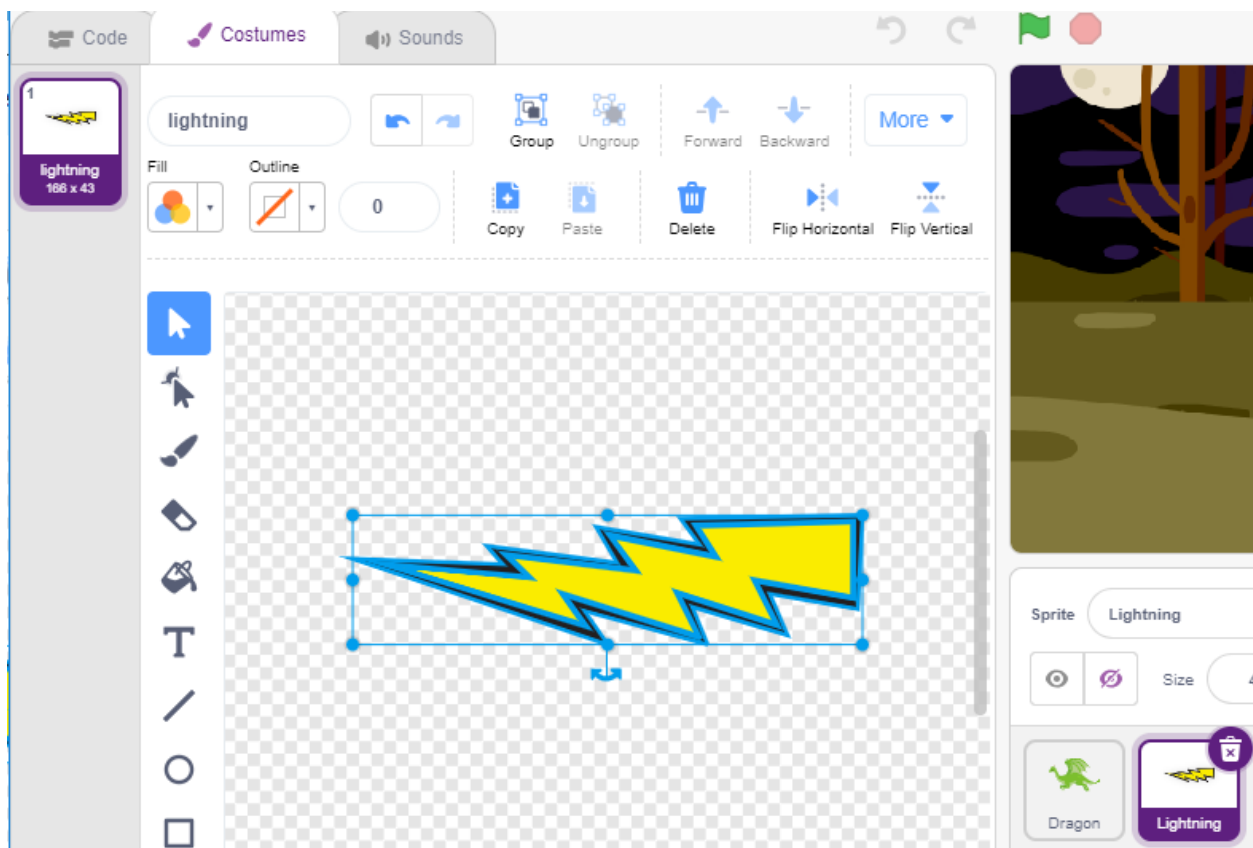


- Ensuite, changez le costume du sprite pour **dragon-b** et faites bouger le sprite **Dragon** de haut en bas dans une certaine plage.



- Ajoutez un sprite **Lightning** comme le feu soufflé par le sprite **Dragon**. Vous devez le tourner de 90° dans le sens des aiguilles d'une montre sur la page Costumes, ceci pour faire bouger le sprite **Lightning** dans la bonne direction.

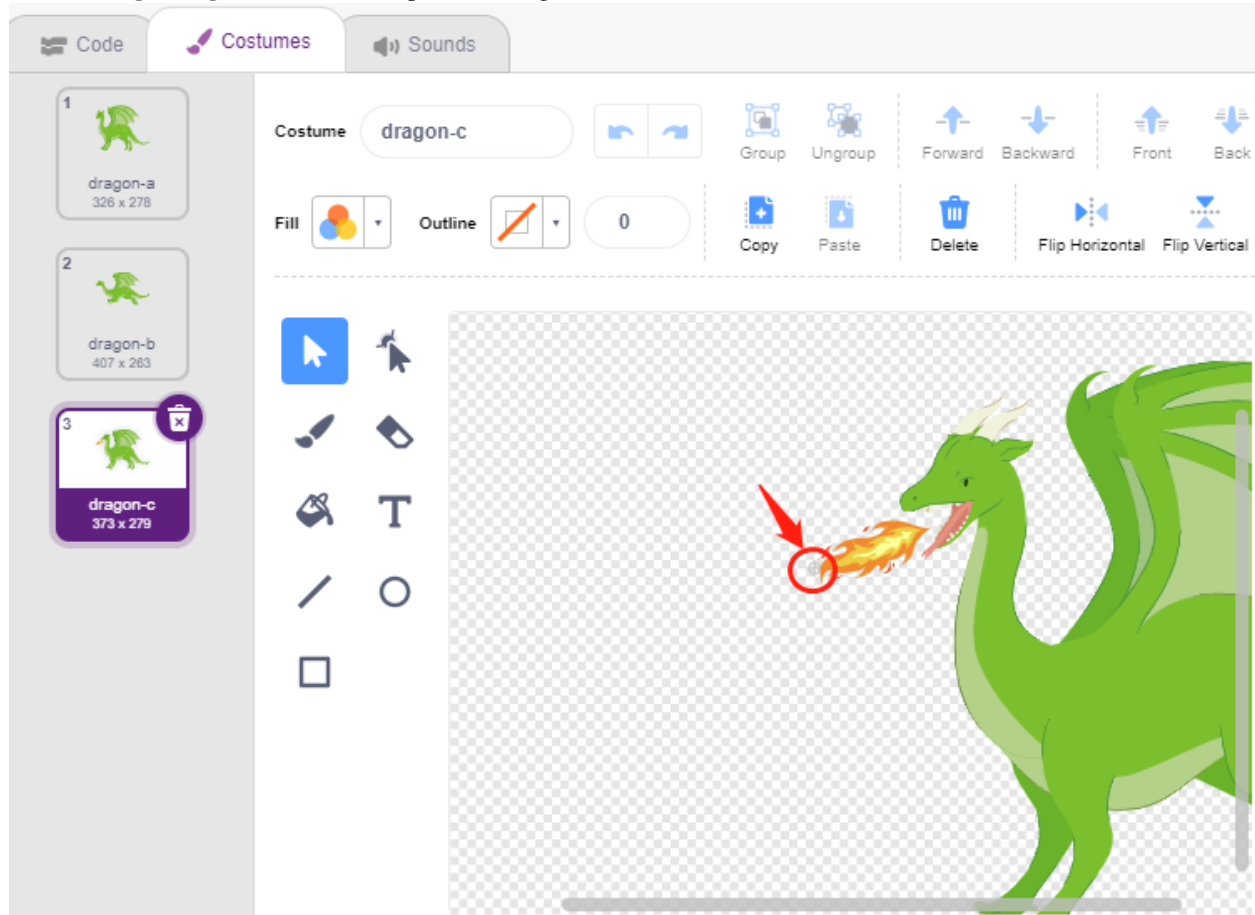
**Note :** Lors de l'ajustement du costume du sprite **Lightning**, vous pouvez le déplacer hors du centre, ce qui doit être évité ! Le point central doit être exactement au milieu du sprite !



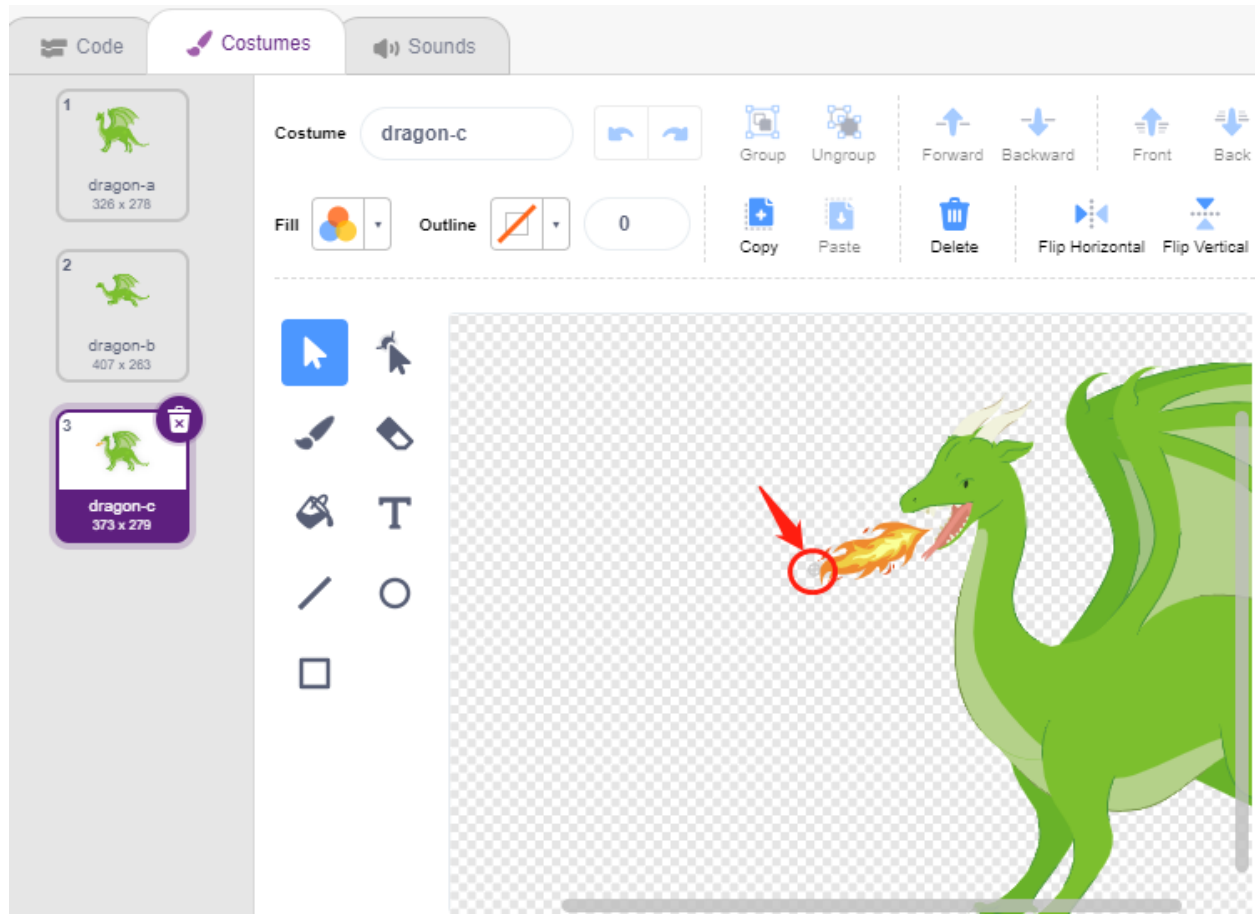
- Ensuite, ajustez le costume **dragon-c** du sprite **Dragon** pour que son point central soit à la queue du feu.



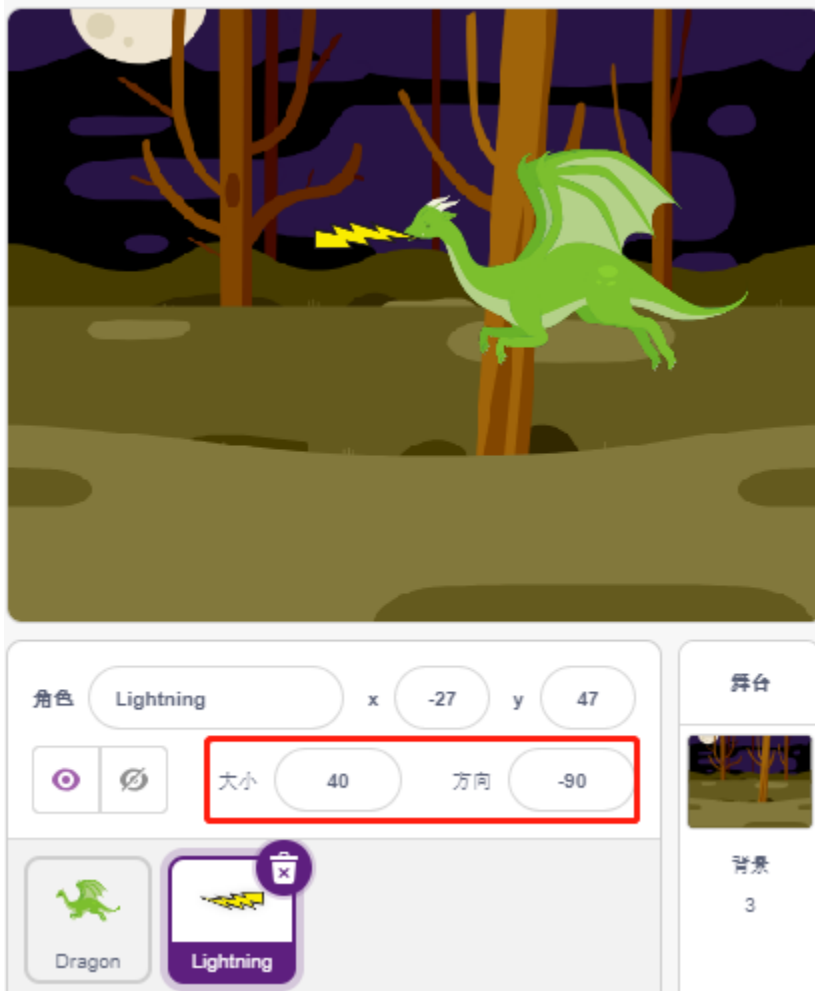
Cela permettra d'aligner correctement les positions du sprite **Dragon** et du sprite **Lightning**, et d'éviter que l'**Lightning** ne soit lancé des pieds du dragon.



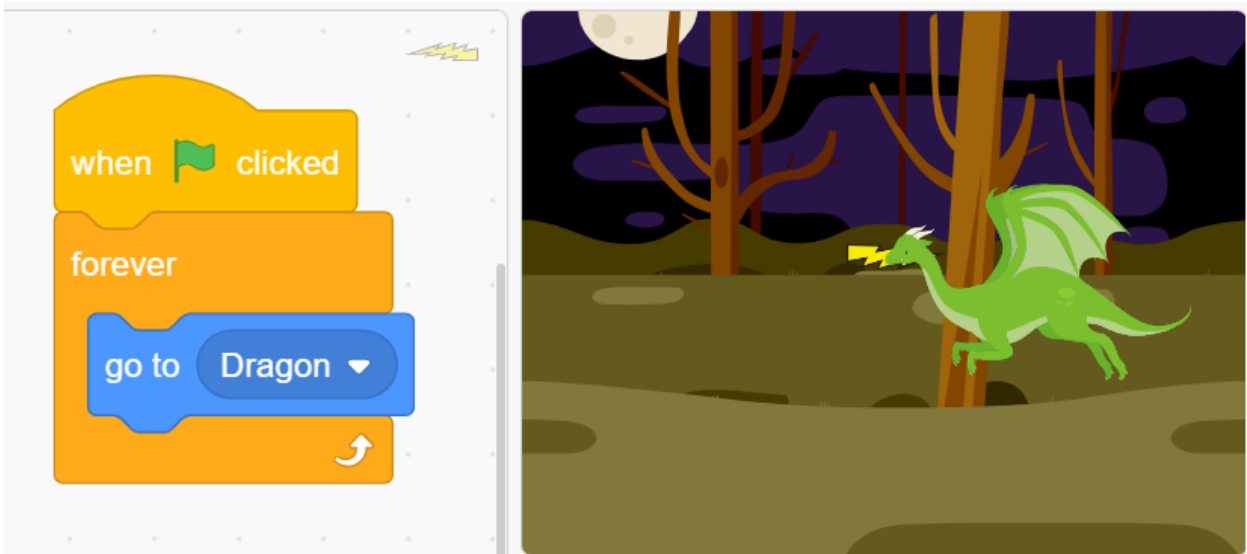
— De même, **dragon-b** doit faire coïncider la tête du dragon avec le point central.



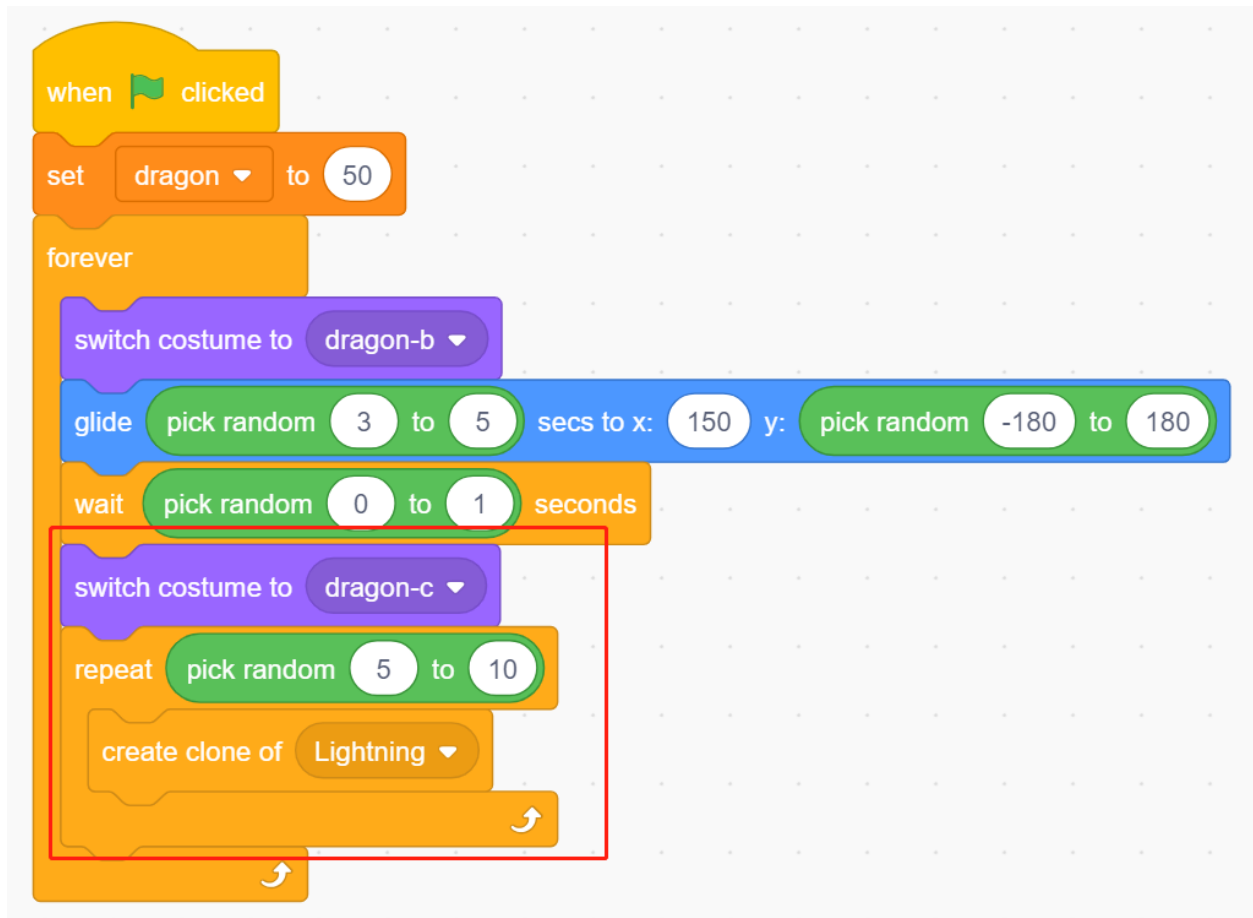
— Ajustez la taille et l'orientation du sprite **Lightning** pour rendre l'image plus harmonieuse.



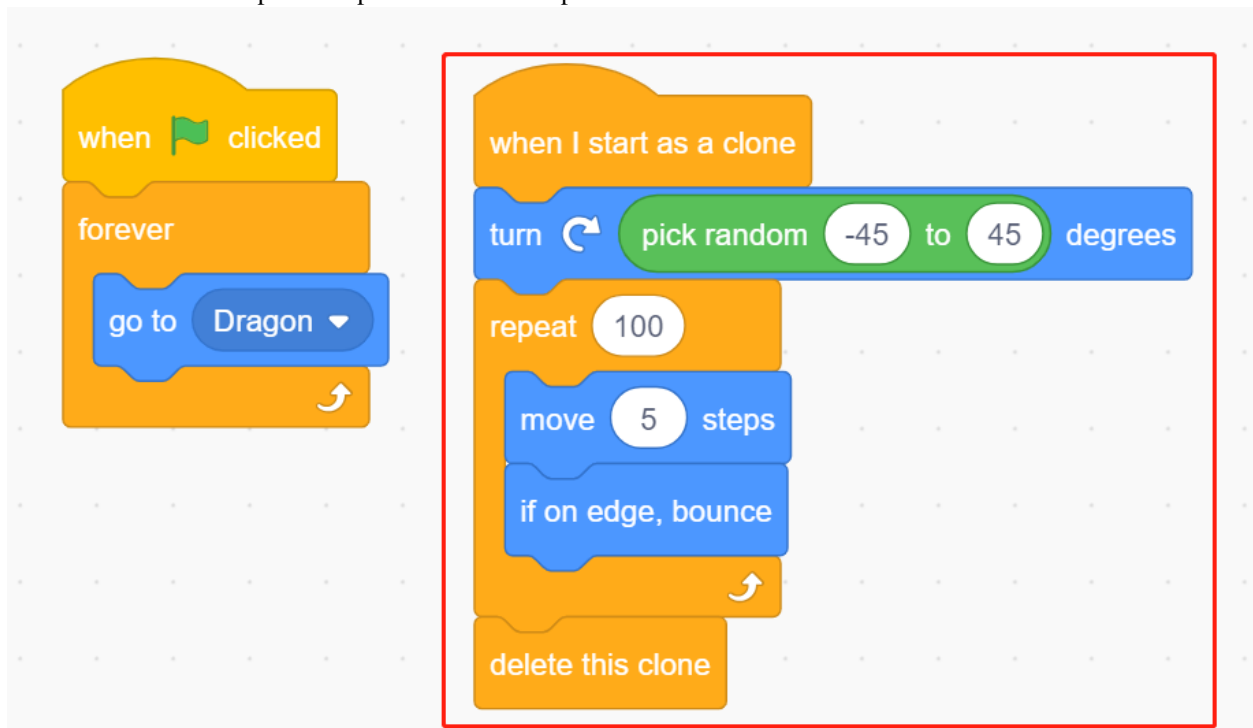
- Programmez maintenant le sprite **Lightning**. C'est simple, faites-le suivre le sprite **Dragon** en permanence. À ce stade, cliquez sur le drapeau vert et vous verrez le **Dragon** se déplacer avec un éclair dans la bouche.



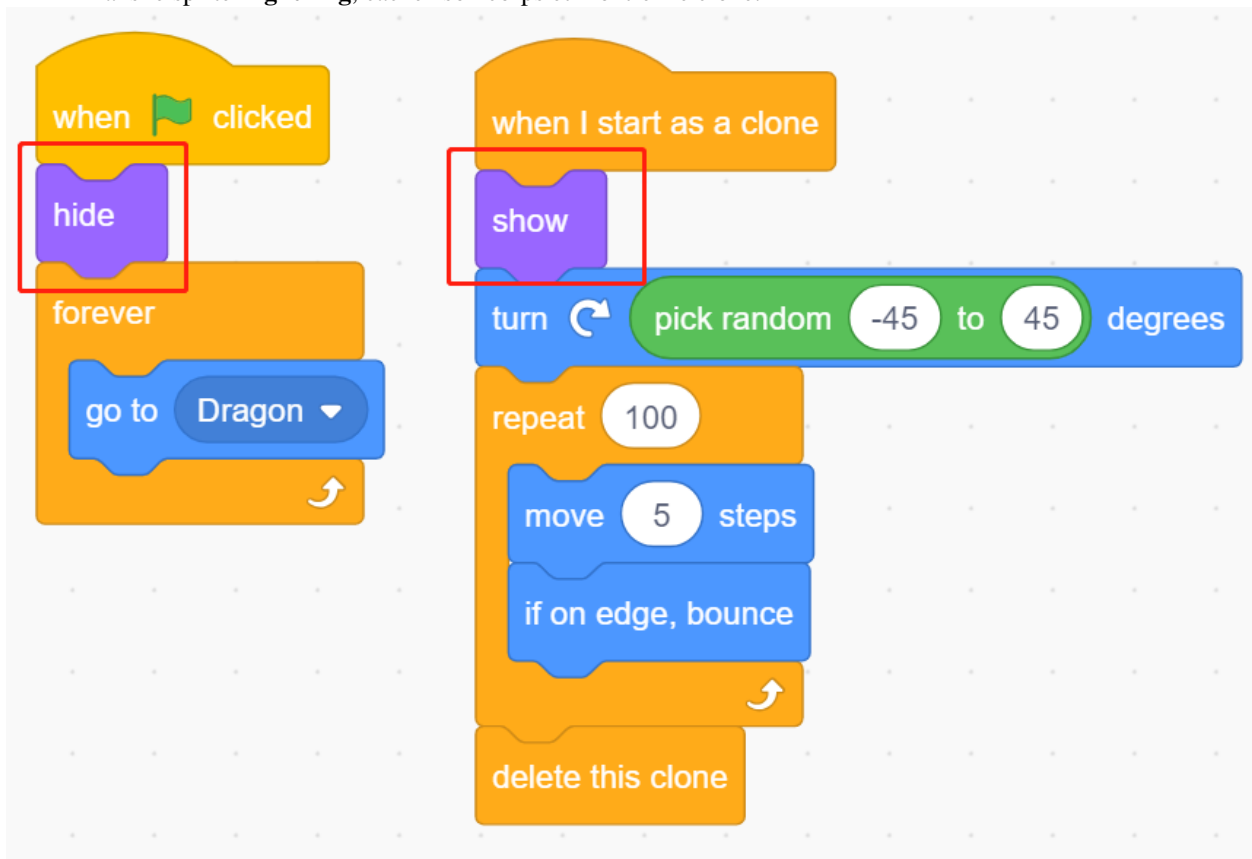
- Revenez au sprite **Dragon**, maintenant faites-le cracher du feu, en veillant à ce que le feu dans sa bouche ne soit pas projeté, mais à créer un clone pour le sprite **Lightning**.



- Cliquez sur le sprite **Lightning** et laissez le clone de **Lightning** être projeté sous un angle aléatoire, il rebondira sur le mur et disparaîtra après un certain temps.



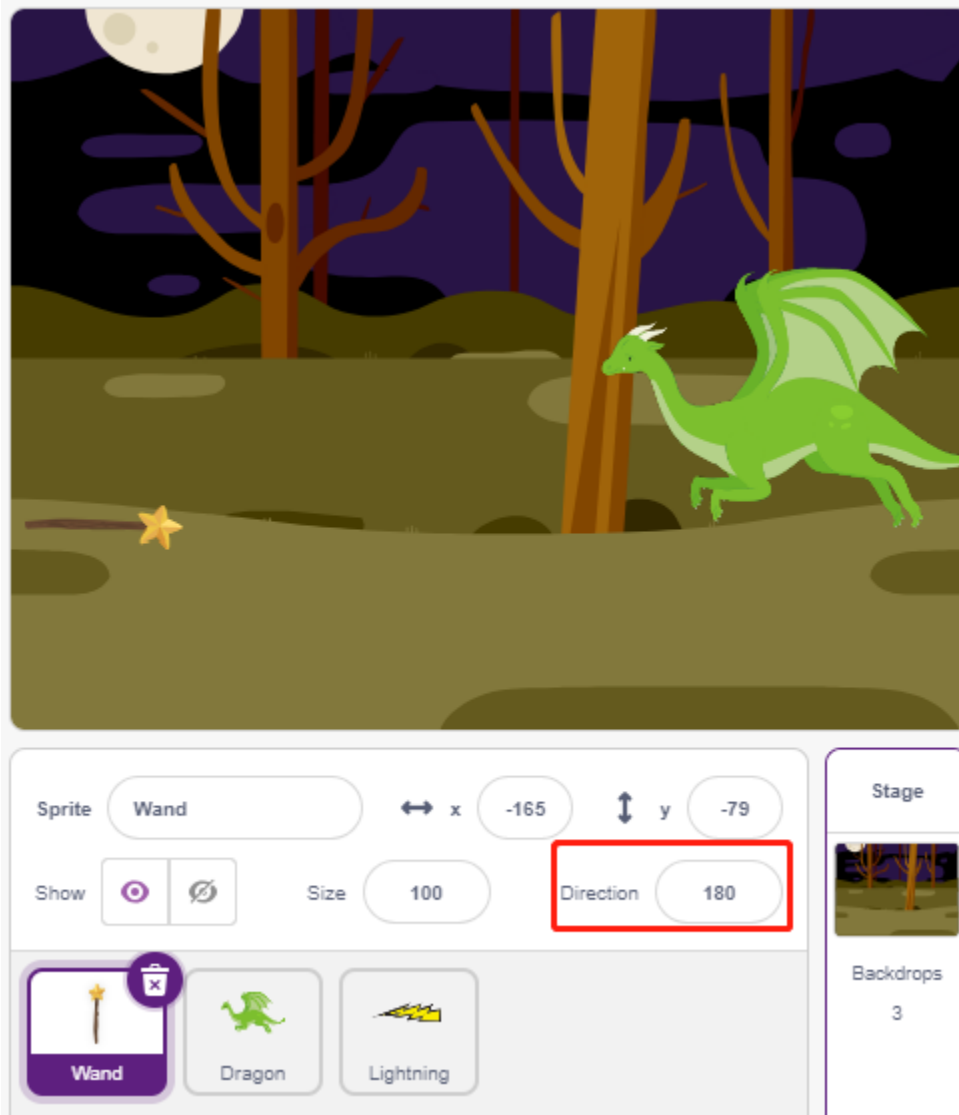
— Dans le sprite **Lightning**, cachez son corps et montrez le clone.



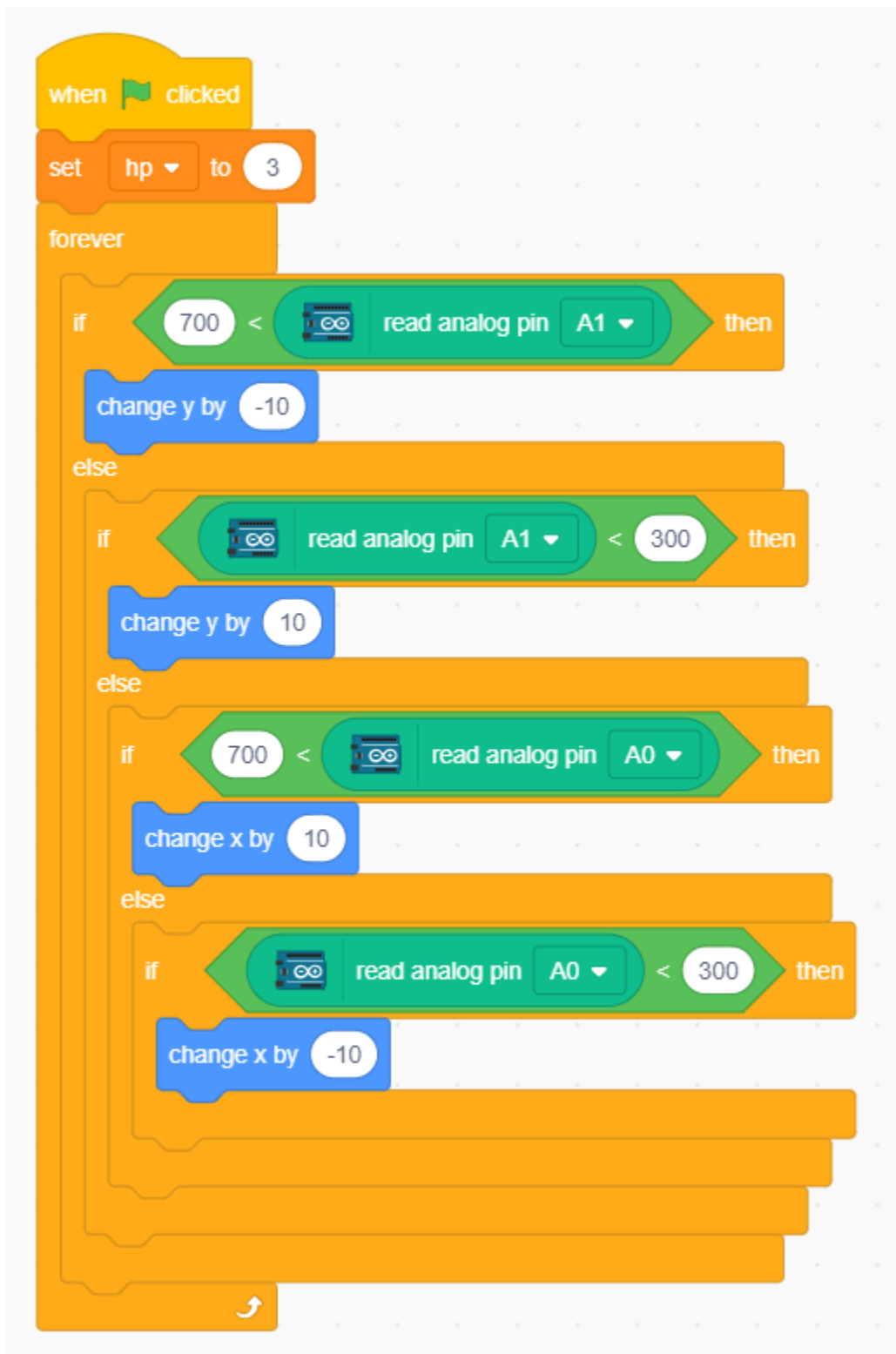
Maintenant, le dragon peut se déplacer de haut en bas et cracher du feu.

## 2. Baguette Magique

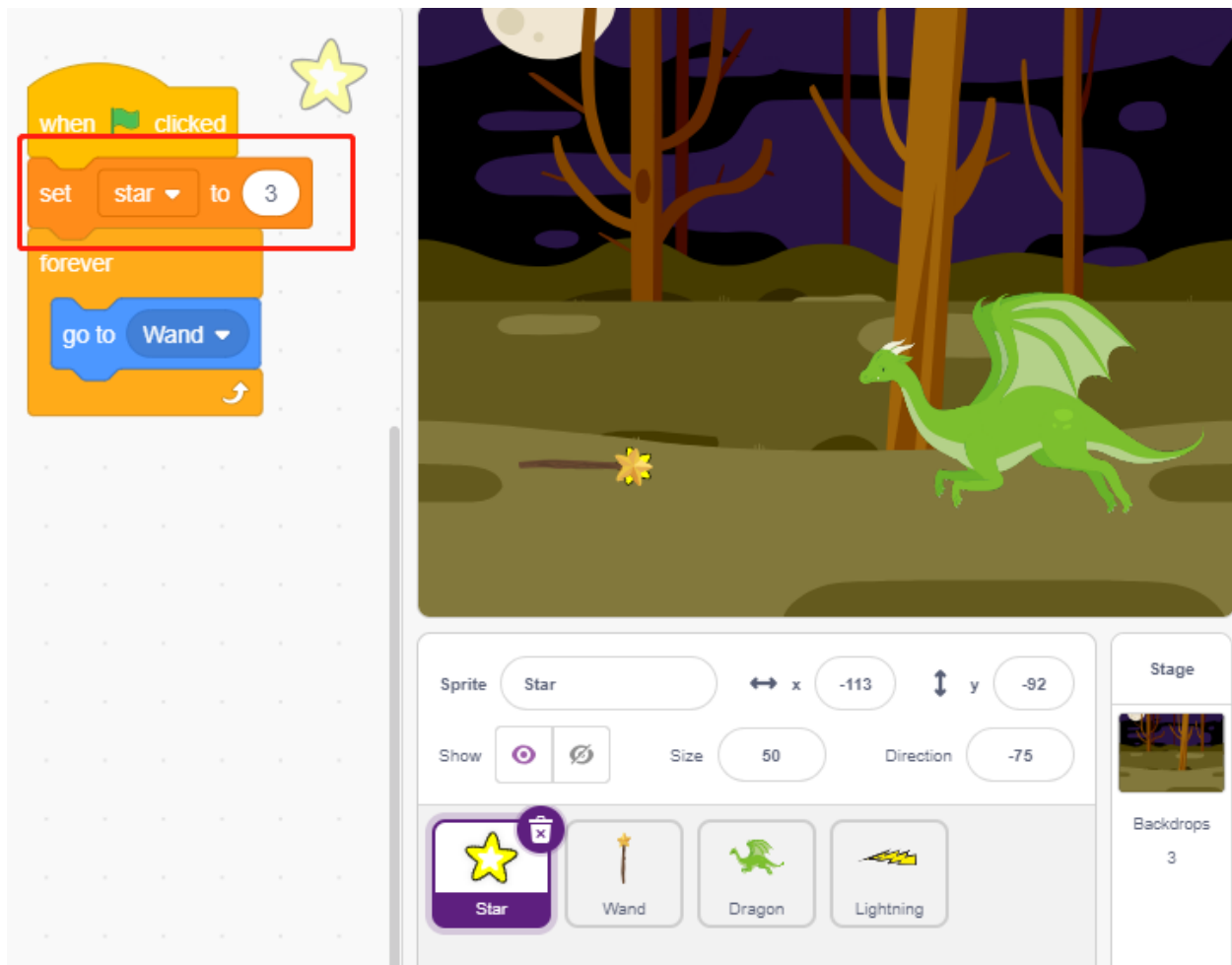
— Créez un sprite **Wand** et tournez sa direction à 180 pour pointer vers la droite.



- Créez maintenant une variable **hp** pour enregistrer sa valeur de vie, initialement réglée sur 3. Puis lisez la valeur du Joystick, qui est utilisée pour contrôler le mouvement de la baguette.

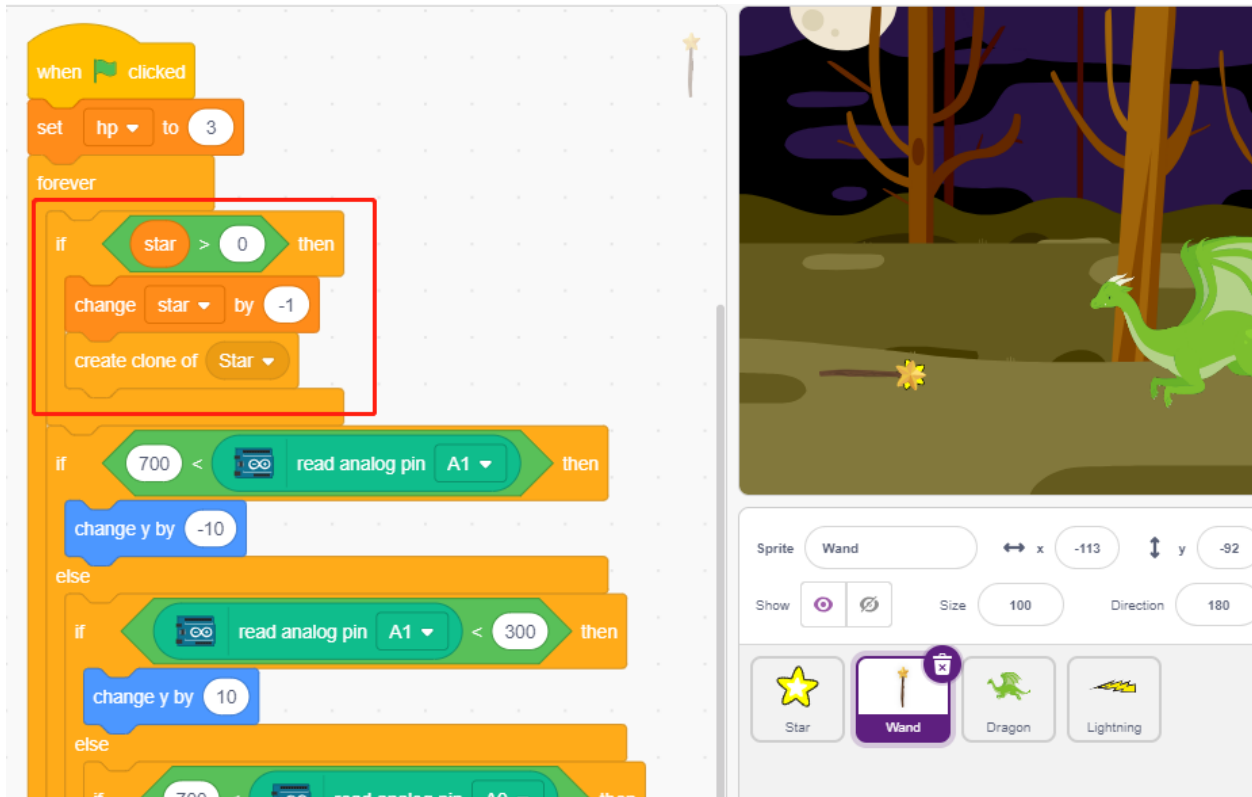


- Le dragon a des éclairs, et la baguette qui les écrase a sa « balle magique » ! Créez un sprite **Star**, redimensionnez-le, et programmez-le pour qu'il suive toujours le sprite **Wand**, et limitez le nombre d'étoiles à trois.

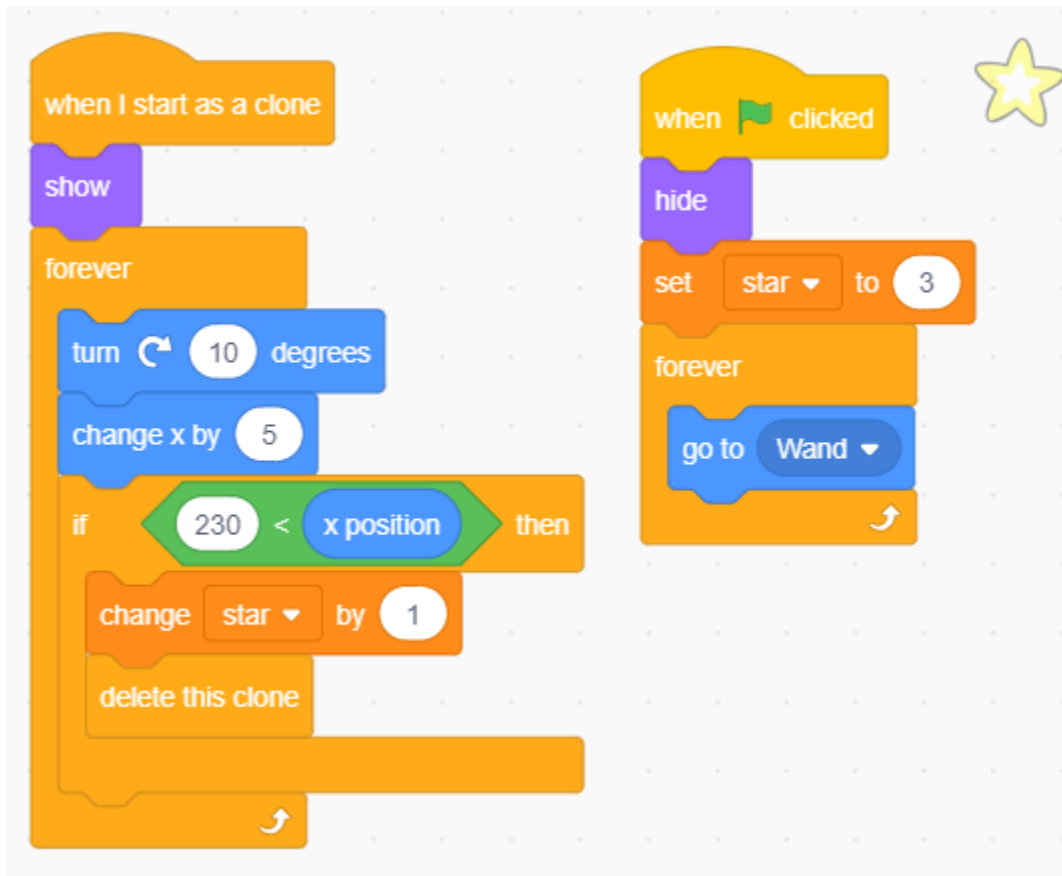


- Faites tirer des étoiles automatiquement par le sprite **Wand**. Le sprite **Wand** tire des étoiles de la même manière que le dragon crache du feu – en créant des clones.





- Revenez au sprite **Star** et programmez son clone pour qu'il tourne et tire vers la droite, disparaisse après avoir dépassé la scène et restaure le nombre d'étoiles. Comme pour le sprite **Lightning**, cachez le corps et montrez le clone.



Maintenant, nous avons une baguette qui tire des balles d'étoiles.

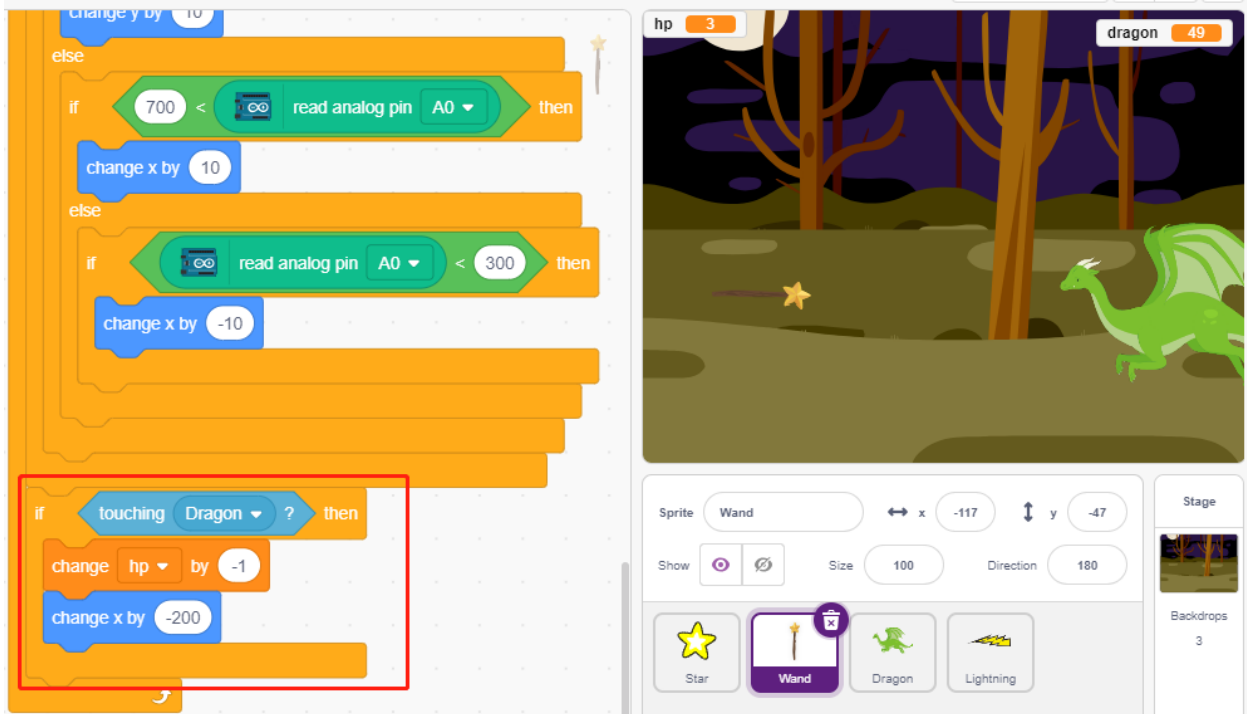
### 3. Combat !

La baguette et le dragon sont actuellement encore en désaccord l'un avec l'autre, et nous allons les faire combattre. Le dragon est fort, et la baguette est le brave homme qui part en croisade contre le dragon. L'interaction entre eux se compose des parties suivantes.

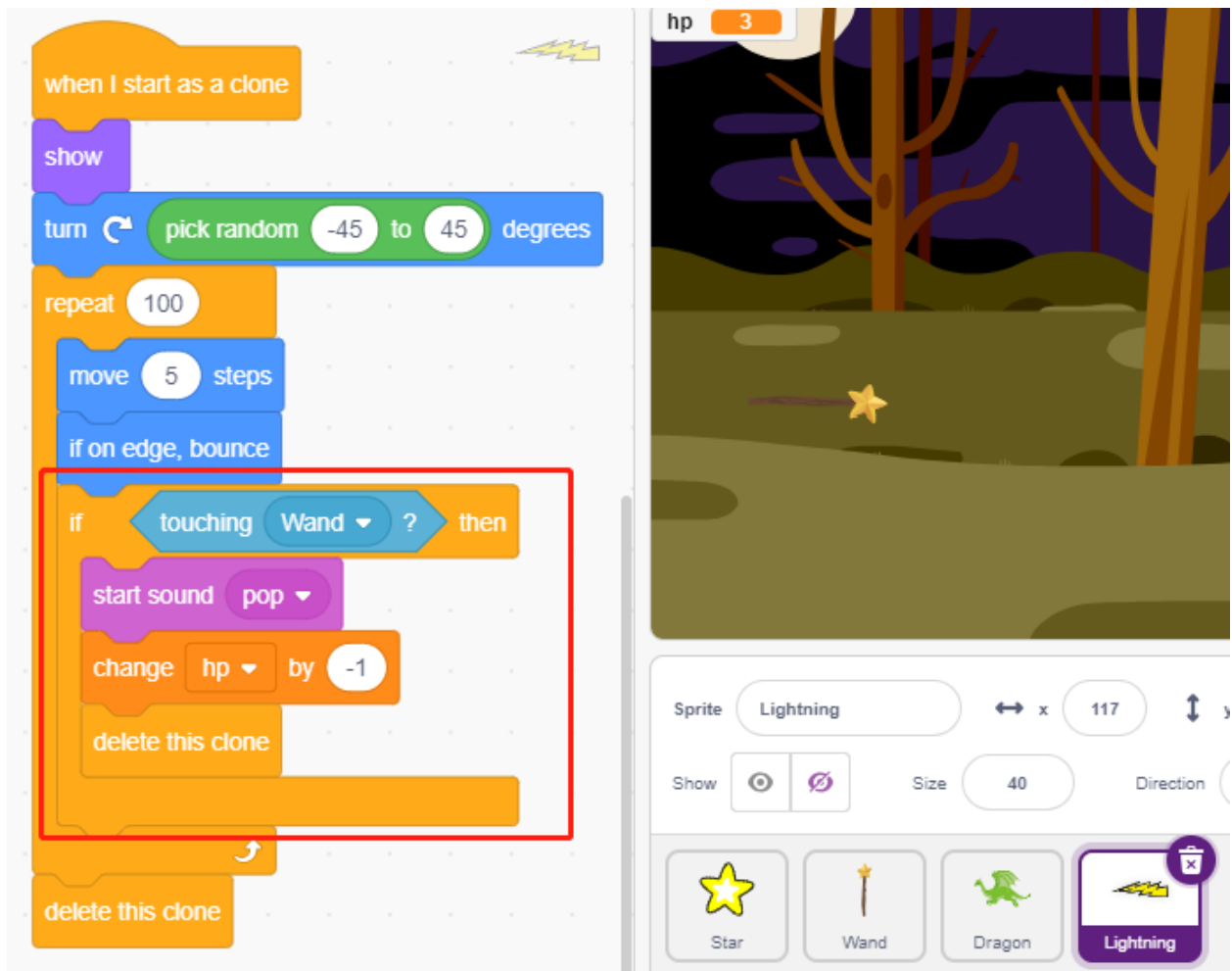
1. si la baguette touche le dragon, elle sera repoussée et perdra des points de vie.
2. si un éclair frappe la baguette, la baguette perdra des points de vie.
3. si la balle d'étoile touche le dragon, le dragon perdra des points de vie.

Une fois cela résolu, passons à la modification des scripts pour chaque sprite.

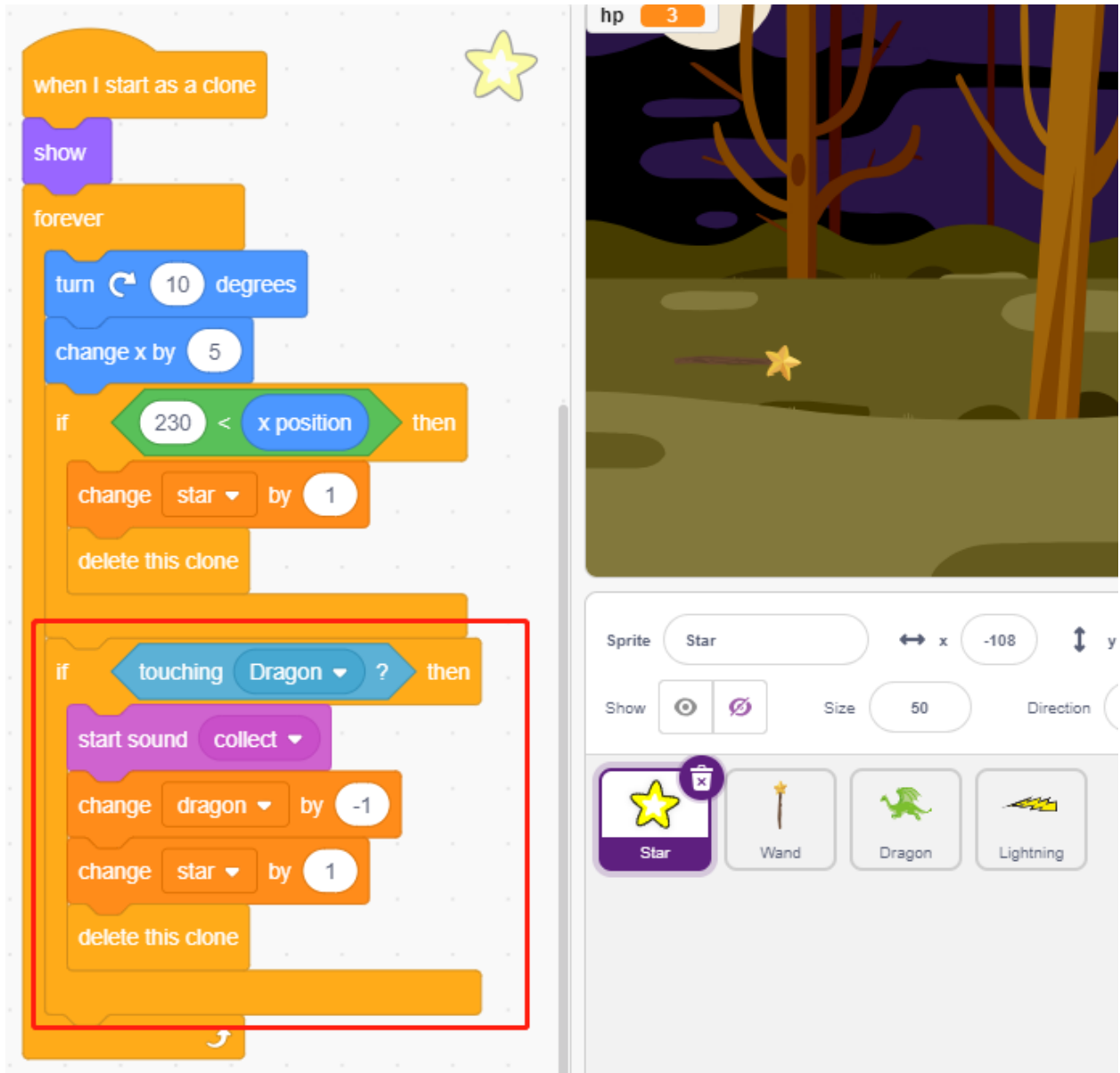
- Si la **Wand** touche le **Dragon**, elle sera repoussée et perdra des points de vie.



- Si **Lightning** (un clone du sprite **Lightning**) touche le sprite **Wand**, il produira un son de pop et disparaîtra, et la **Wand** perdra des points de vie.



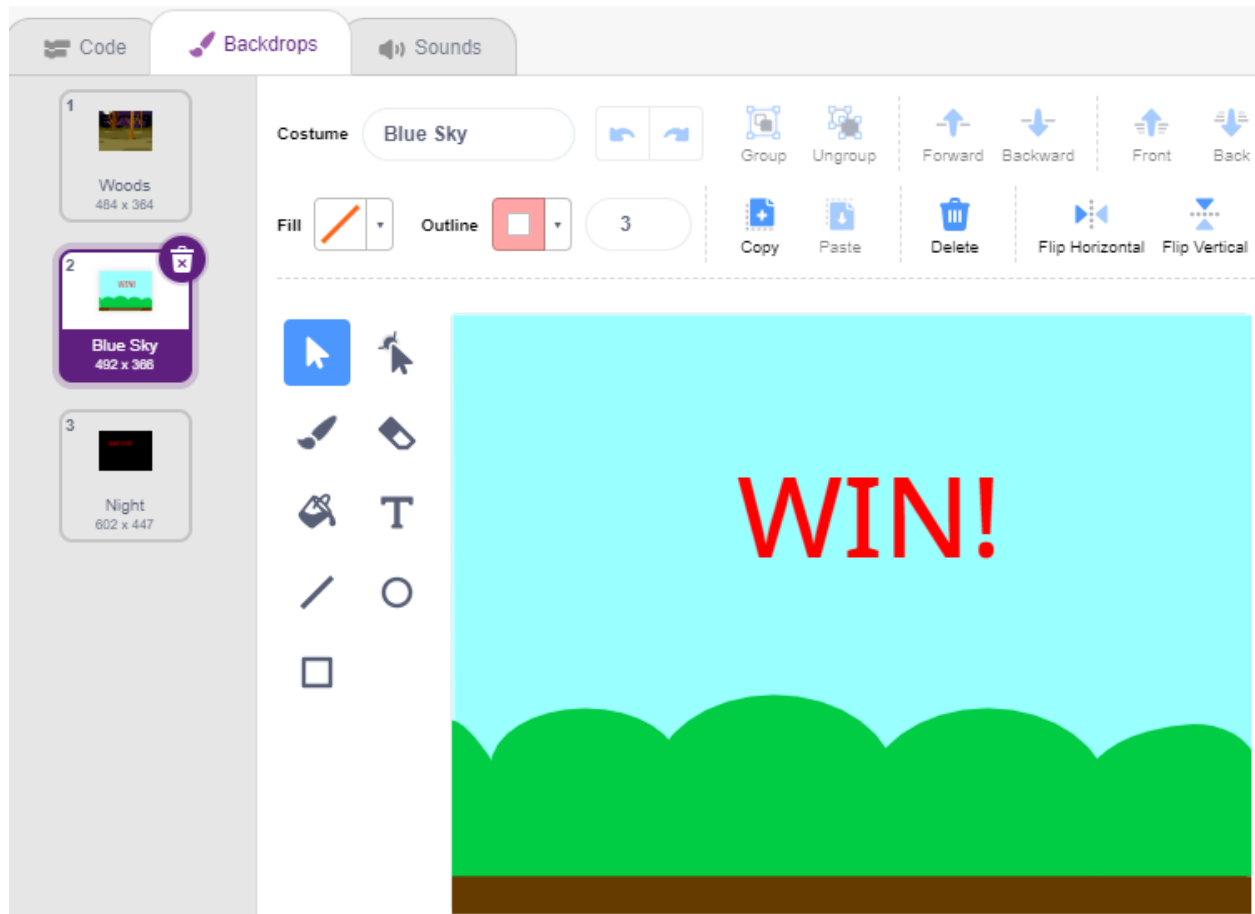
- Si une **Star** (clone du sprite **Star**) touche le **Dragon**, elle émettra un son de collecte et disparaîtra, tout en restaurant le nombre d'**Star**, et le **Dragon** perdra des points de vie.



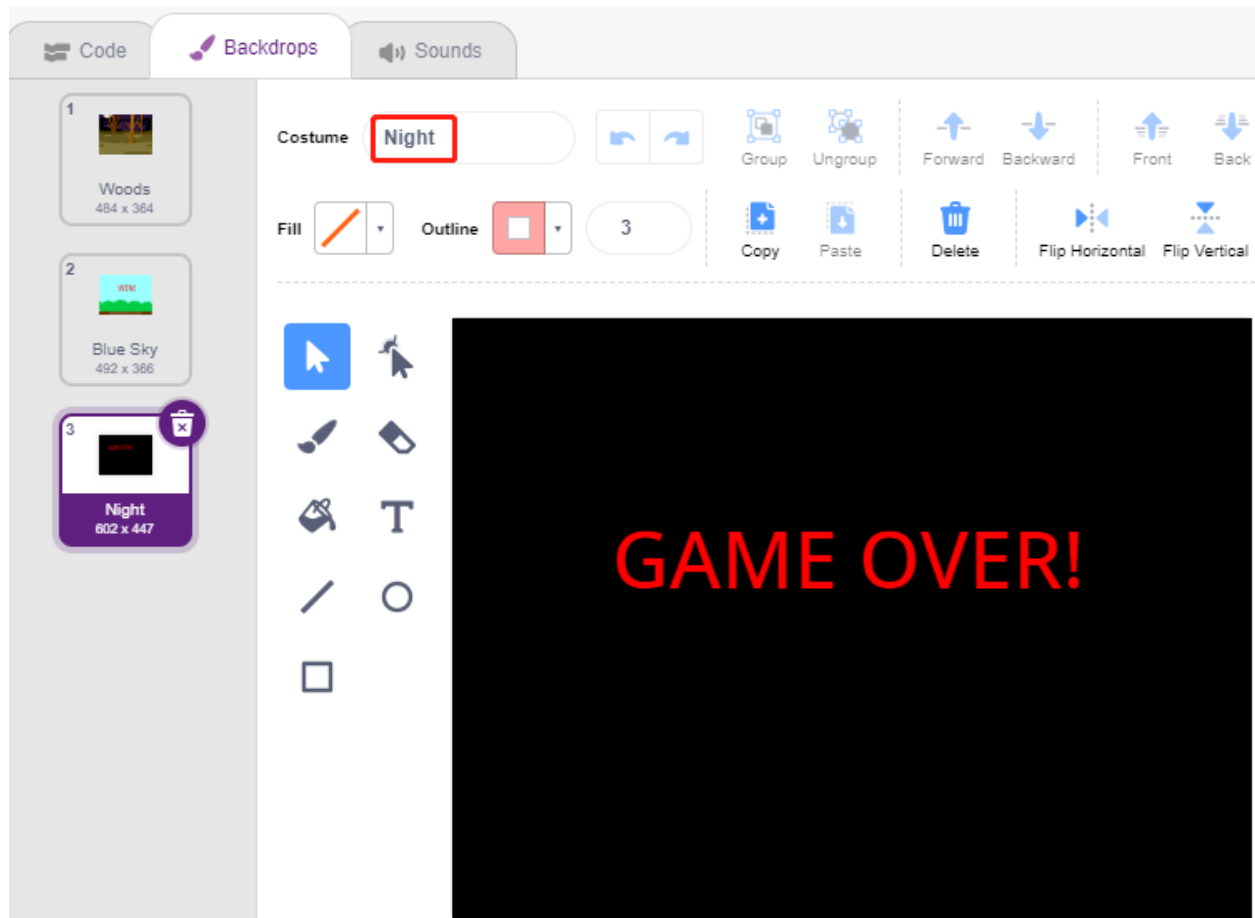
#### 4. scène

La bataille entre la **Wand** et le **Dragon** se terminera finalement par des gagnants et des perdants, que nous représentons avec la scène.

- Ajoutez l'arrière-plan **Blue Sky**, et écrivez le mot « VICTOIRE ! » dessus pour représenter que le dragon a été vaincu et l'aube est arrivée.



— Et modifiez l’arrière-plan vierge comme suit, pour représenter que le jeu a échoué et tout sera dans l’obscurité.



- Écrivez maintenant un script pour changer ces arrière-plans, lorsque le drapeau vert est cliqué, passez à l'arrière-plan **Woods** ; si le point de vie du dragon est inférieur à 1, alors le jeu réussit et changez l'arrière-plan pour **Blue Sky** ; si le point de valeur de vie de la **Wand** est inférieur à 1, alors changez pour l'arrière-plan **Night** et le jeu échoue.



### 3. Jouez avec la voiture Scratch

Les projets suivants sont écrits dans l'ordre de difficulté de programmation, il est recommandé de les lire dans l'ordre. Dans chaque projet, il y a des étapes très détaillées pour vous apprendre à construire le circuit et comment le programmer étape par étape pour obtenir le résultat final.

Bien sûr, vous pouvez également ouvrir le script directement pour l'exécuter, mais vous devez vous assurer d'avoir téléchargé le matériel pertinent de [github](#).

Une fois le téléchargement terminé, décompressez-le. Référez-vous à [Mode Téléchargement](#) pour exécuter individuellement les scripts directement.



## 8.26 3.1 Tester la Voiture

Ici, vous apprendrez à écrire des scripts pour faire avancer la voiture, mais vous devez vous référer à *Projets de Voiture* pour assembler la voiture et pour obtenir une compréhension de base de celle-ci.

Mais avant de commencer le projet, vous devez connaître les étapes pour utiliser PictoBlox en *Mode Téléchargement*.

### 8.26.1 Composants Requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

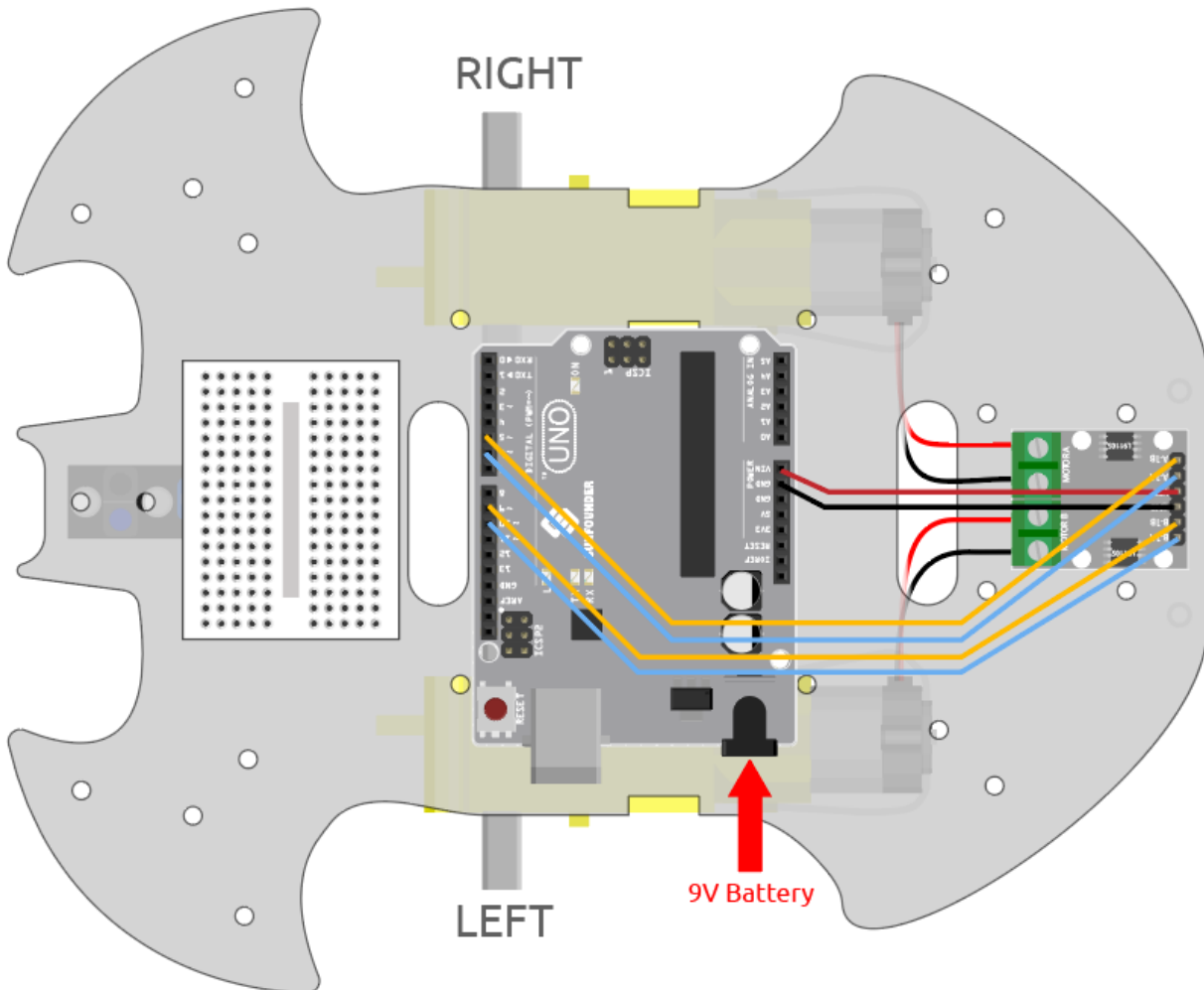
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-

### 8.26.2 Construire le Circuit

Le module de pilote de moteur L9110 est un module de pilote de moteur haute puissance pour entraîner des moteurs DC et pas à pas. Le module L9110 peut contrôler jusqu'à 4 moteurs DC, ou 2 moteurs DC avec contrôle de direction et de vitesse.

Connectez les fils entre le module L9110 et la carte R3 selon le schéma ci-dessous.

Module L9110	Carte R3	Moteur
A-1B	5	
A-1A	6	
B-1B(B-2A)	9	
B-1A	10	
OB(B)		Fil noir du moteur droit
OA(B)		Fil rouge du moteur droit
OB(A)		Fil noir du moteur gauche
OA(A)		Fil rouge du moteur gauche

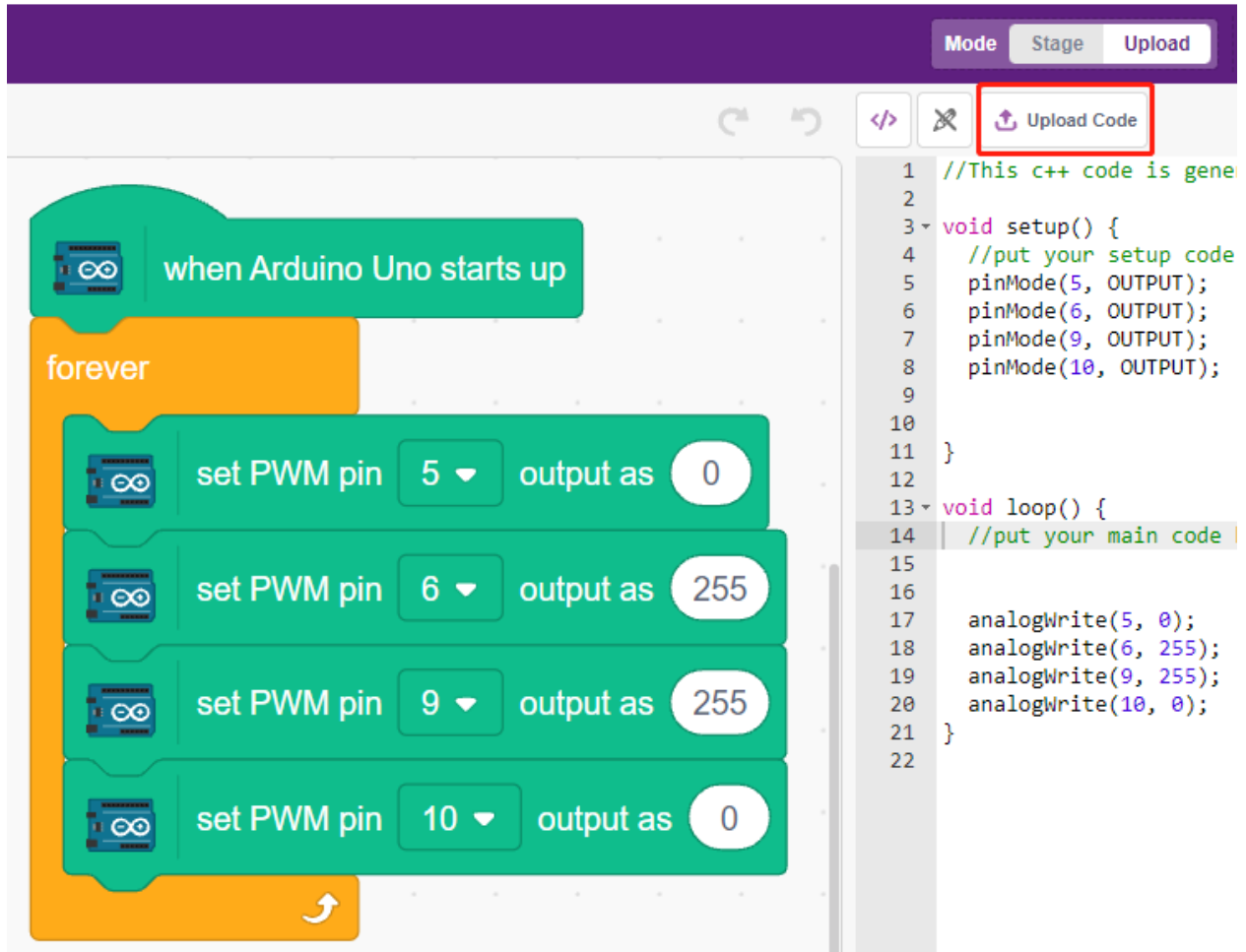


### 8.26.3 Programmation

#### 1. Faire avancer la voiture

Sur la base du câblage ci-dessus, nous savons que les broches 5 et 6 sont utilisées pour contrôler la rotation du moteur droit et les broches 9 et 10 pour celle du moteur gauche. Écrivons maintenant un script pour faire avancer la voiture.

Après avoir sélectionné la carte Arduino Uno, passez en *Mode Téléchargement* et écrivez le script selon le schéma suivant.



Cliquez sur le bouton **Upload Code** pour uploader le code sur la carte R3. Une fois terminé, vous verrez les deux moteurs de la voiture avancer (si vous posez la voiture sur le sol, elle avancera en ligne droite, mais peut-être la voiture ira en courbe car la vitesse des deux moteurs est un peu différente).

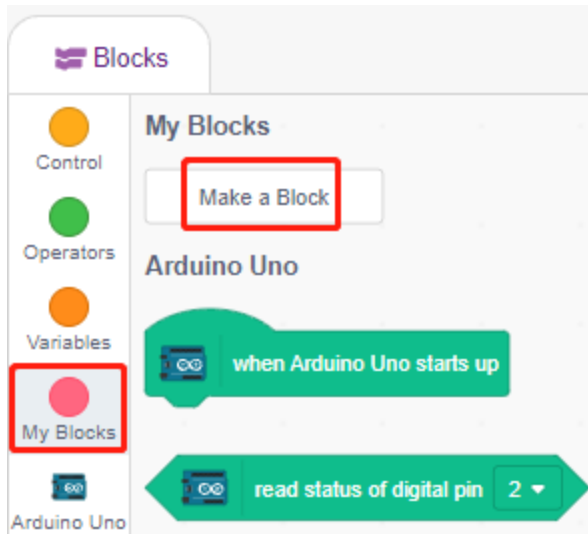
Si les deux ne tournent pas en avant, mais que les situations suivantes se produisent, vous devez réajuster le câblage des deux moteurs.

- Si les deux moteurs tournent en arrière en même temps (le moteur gauche tourne dans le sens des aiguilles d'une montre, le moteur droit tourne dans le sens inverse), inversez le câblage des moteurs gauche et droit en même temps, OA(A) et OB(A) échantent, OA(B) et OB(B) échantent.
- Si le moteur gauche tourne en arrière (rotation dans le sens des aiguilles d'une montre), échangez le câblage de OA(B) et OB(B) du moteur gauche.
- Si le moteur droit tourne en arrière (rotation dans le sens inverse des aiguilles d'une montre), inversez le câblage de OA(A) et OB(A) du moteur droit.

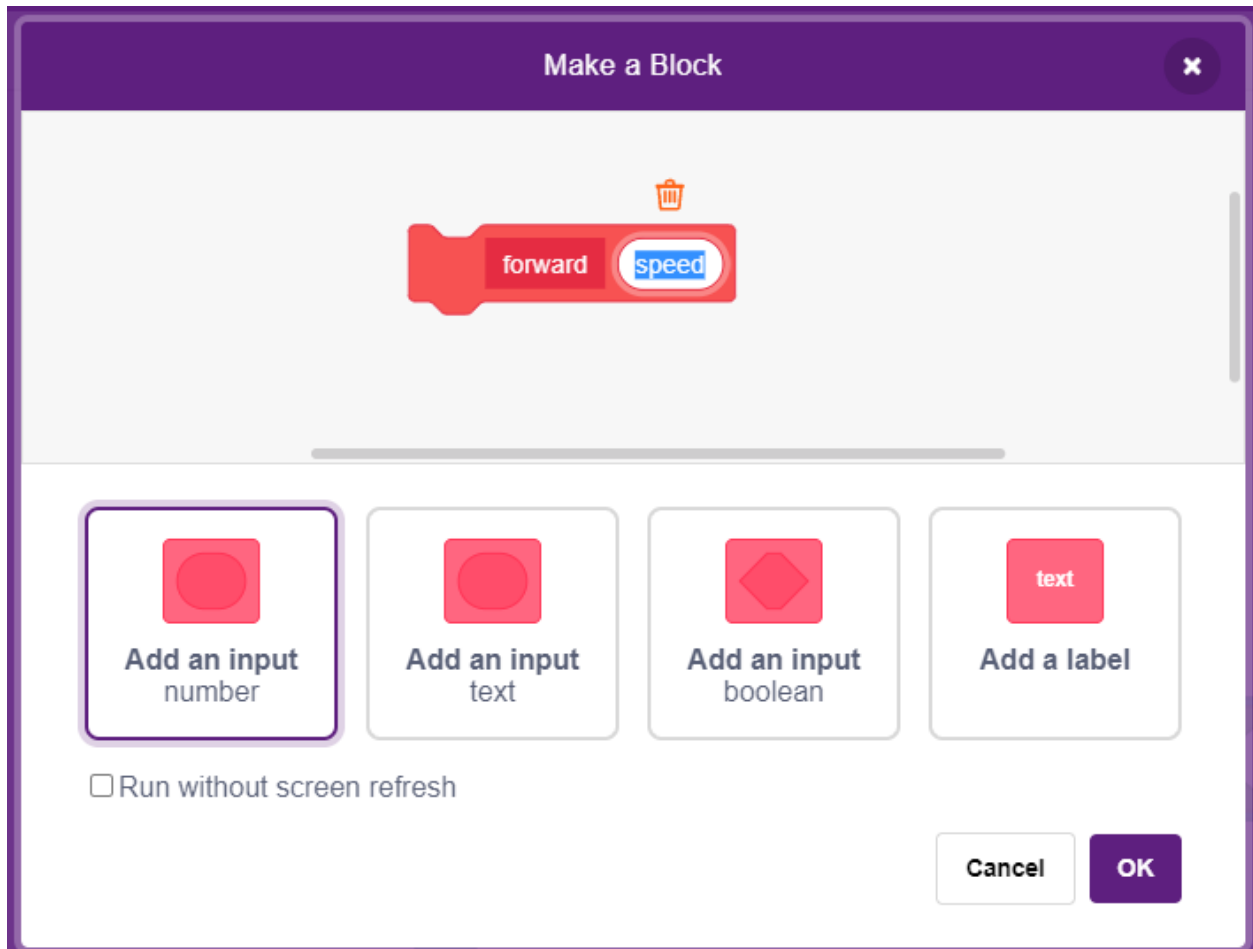
## 2. Création d'un bloc

Pour rendre le script plus propre et facile à utiliser, plaçons ici tous les blocs qui contrôlent le mouvement en avant dans un bloc, et lors de son utilisation, appelez directement ce bloc.

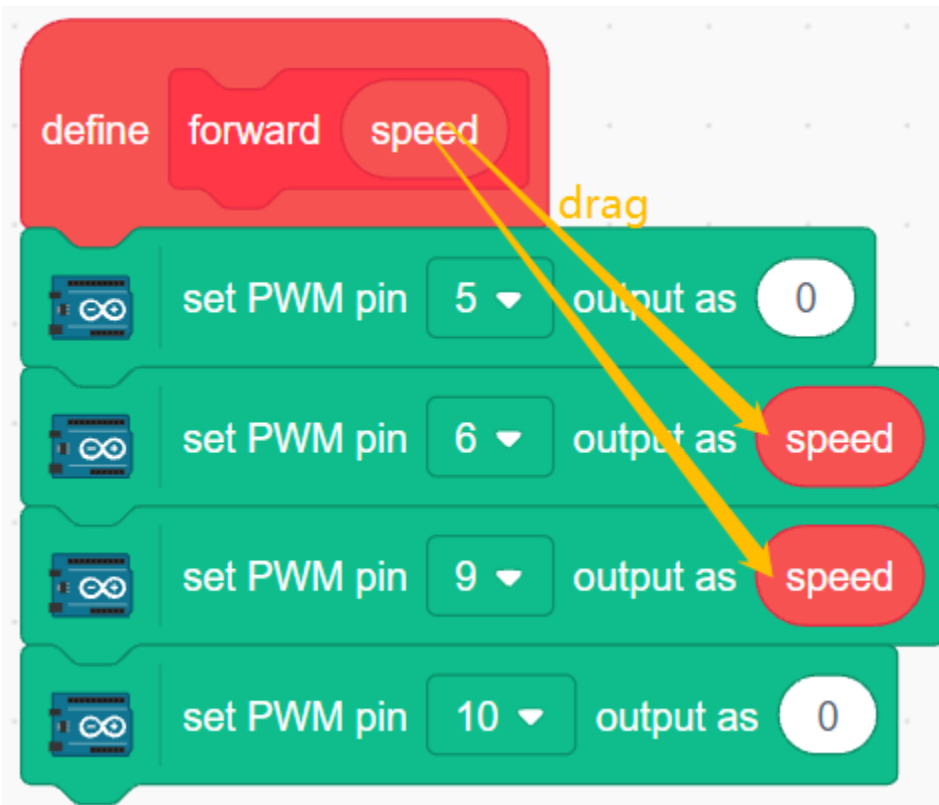
Cliquez sur **Make a Block** dans la palette **My Blocks**.



Entrez le nom du bloc - **forward** et cochez **Add an input**, définissez le nom de l'entrée sur **speed**.

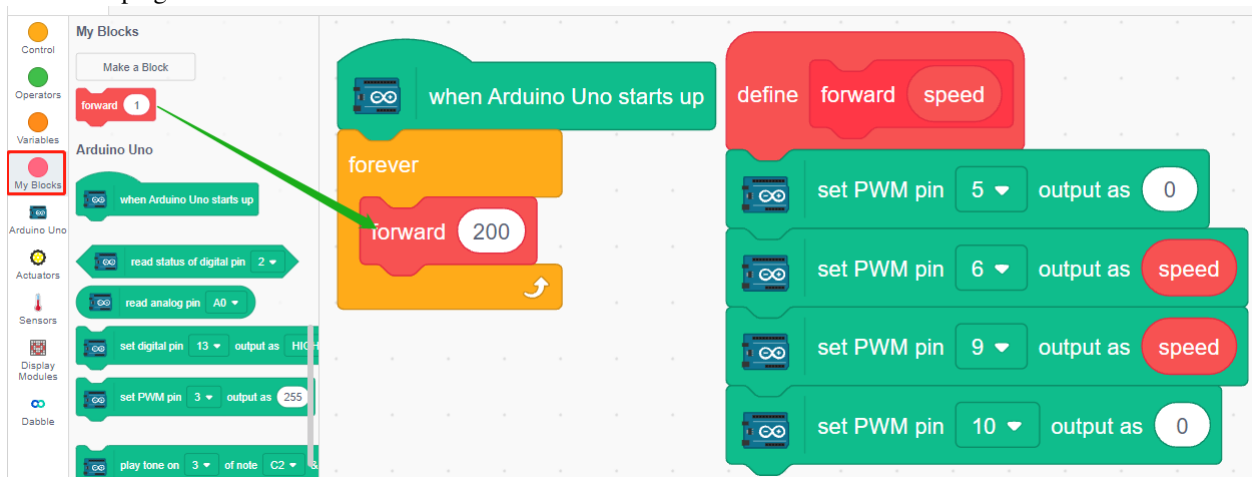


Glissez-déposez les blocs qui contrôlent l'avancement des voitures dans **forward**, notez que vous devez ajouter le paramètre - **speed** aux broches 6 et 9.



Appelez le bloc créé dans le bloc [Forward] - **forward**. En mode Téléversement, le bloc [When Arduino Uno starts up] doit être ajouté au début.

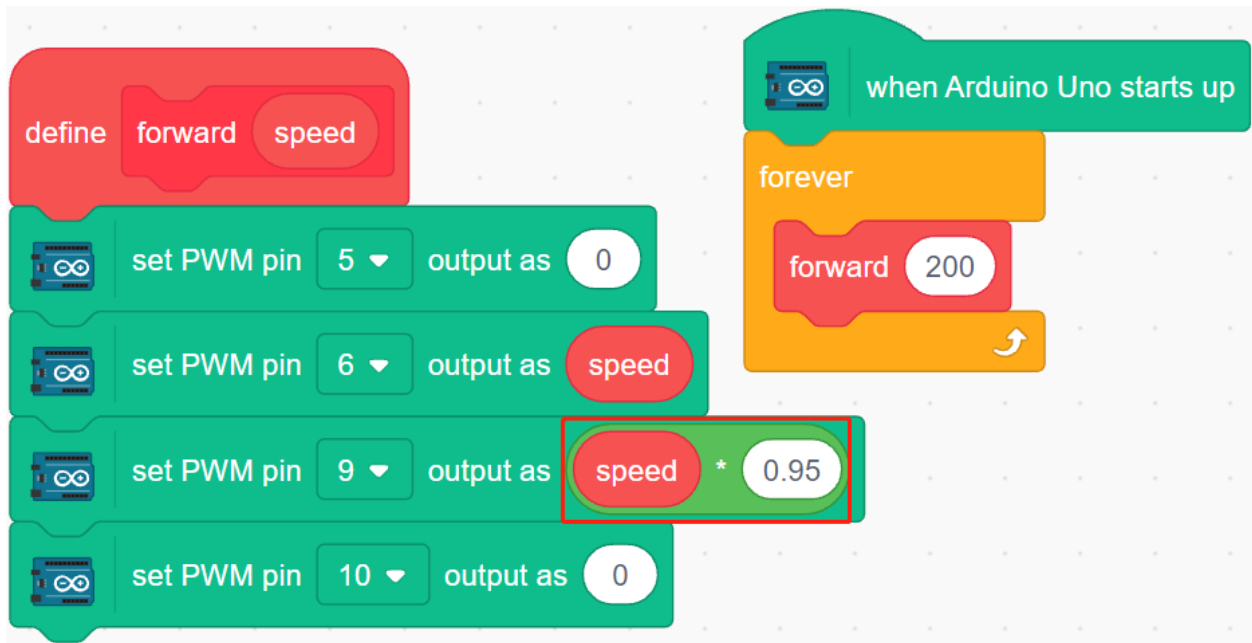
— La plage de vitesse de rotation du moteur est de 100 à 255.



### 3. Ajuster la vitesse des moteurs

Comme il peut y avoir une légère différence dans la vitesse des 2 moteurs, entraînant la voiture à ne pas avancer en ligne droite, nous pouvons donner aux moteurs gauche et droit des vitesses différentes pour garder la voiture avançant le plus possible en ligne droite.

Comme ma voiture avance lentement vers la droite, réduisez ici la vitesse du moteur gauche.



## 8.27 3.2 Mouvement

Ce projet est basé sur [3.1 Tester la Voiture](#) pour faire bouger la voiture dans toutes les directions.

Avant de commencer la programmation, revoyons le principe de fonctionnement du module L9110.

Voici la table de vérité du Moteur B :

B-1A	B-1B(B-2A)	État du Moteur B
1	0	Rotation dans le sens des aiguilles d'une montre
0	1	Rotation dans le sens inverse des aiguilles d'une montre
0	0	Freinage
1	1	Arrêt

Voici la table de vérité du Moteur A :

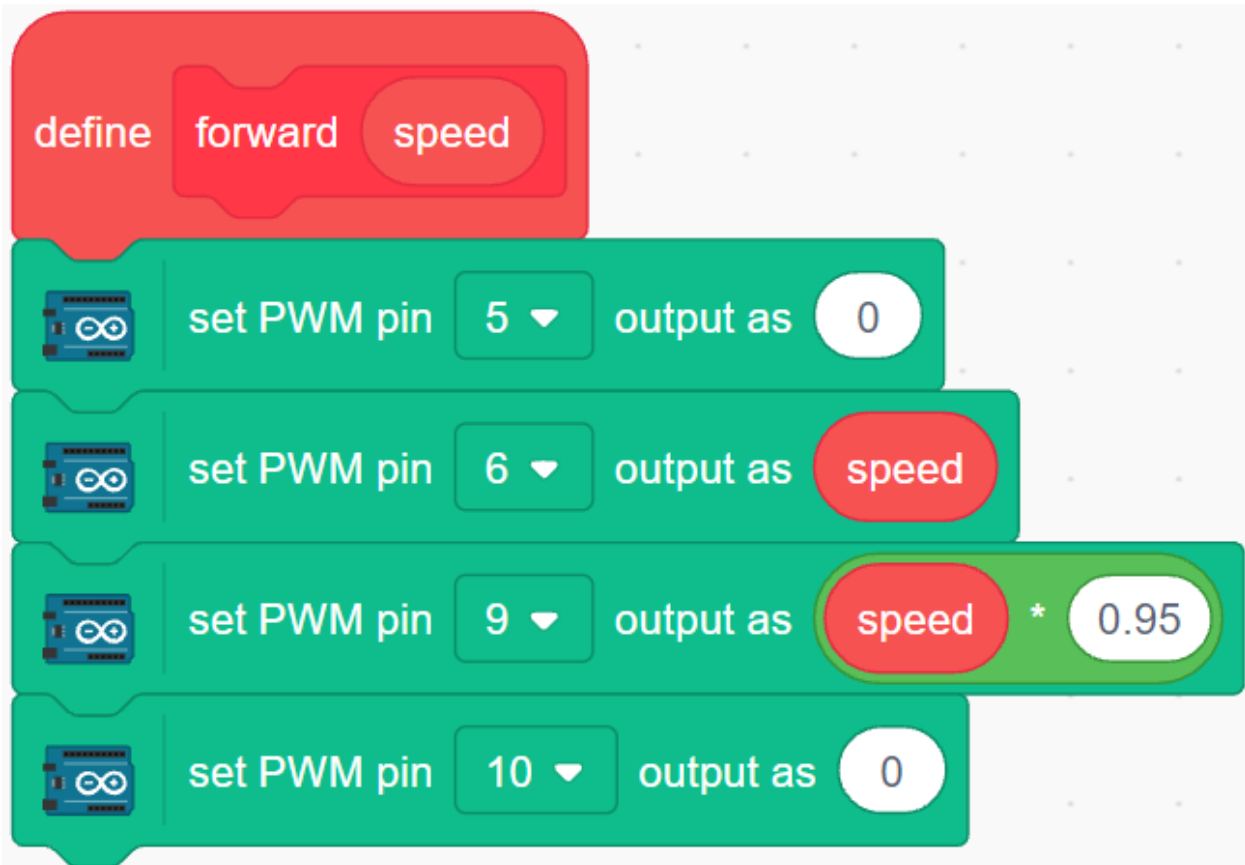
A-1A	A-1B	État du Moteur A
1	0	Rotation dans le sens des aiguilles d'une montre
0	1	Rotation dans le sens inverse des aiguilles d'une montre
0	0	Freinage
1	1	Arrêt

### 8.27.1 Programmation

Créez maintenant des blocs pour faire avancer, reculer, tourner à gauche, tourner à droite et arrêter la voiture respectivement.

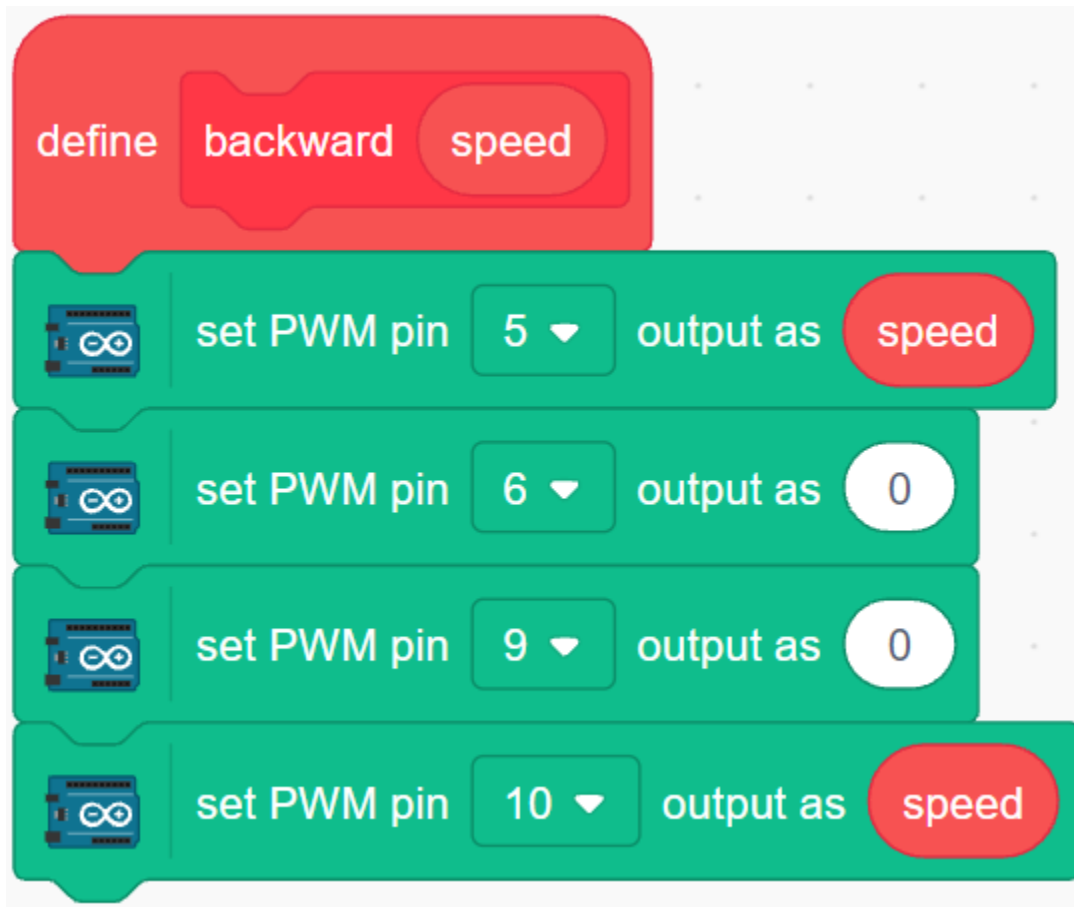
#### 1. Avancer

Le moteur droit tourne dans le sens des aiguilles d'une montre et le moteur gauche dans le sens inverse pour faire avancer la voiture.



#### 2. Reculer

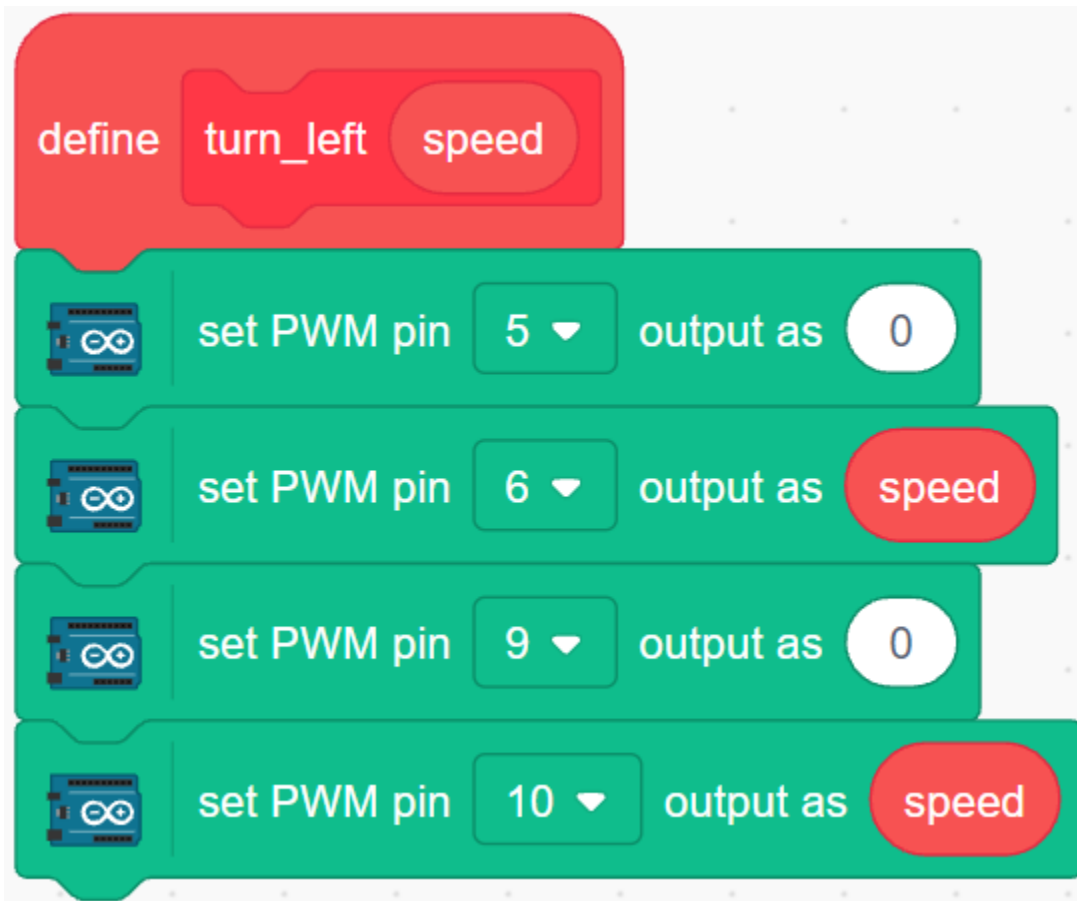
Le recul est juste l'opposé, le moteur droit doit tourner dans le sens inverse des aiguilles d'une montre, le moteur gauche dans le sens des aiguilles d'une montre.



### 3. Tourner à Gauche

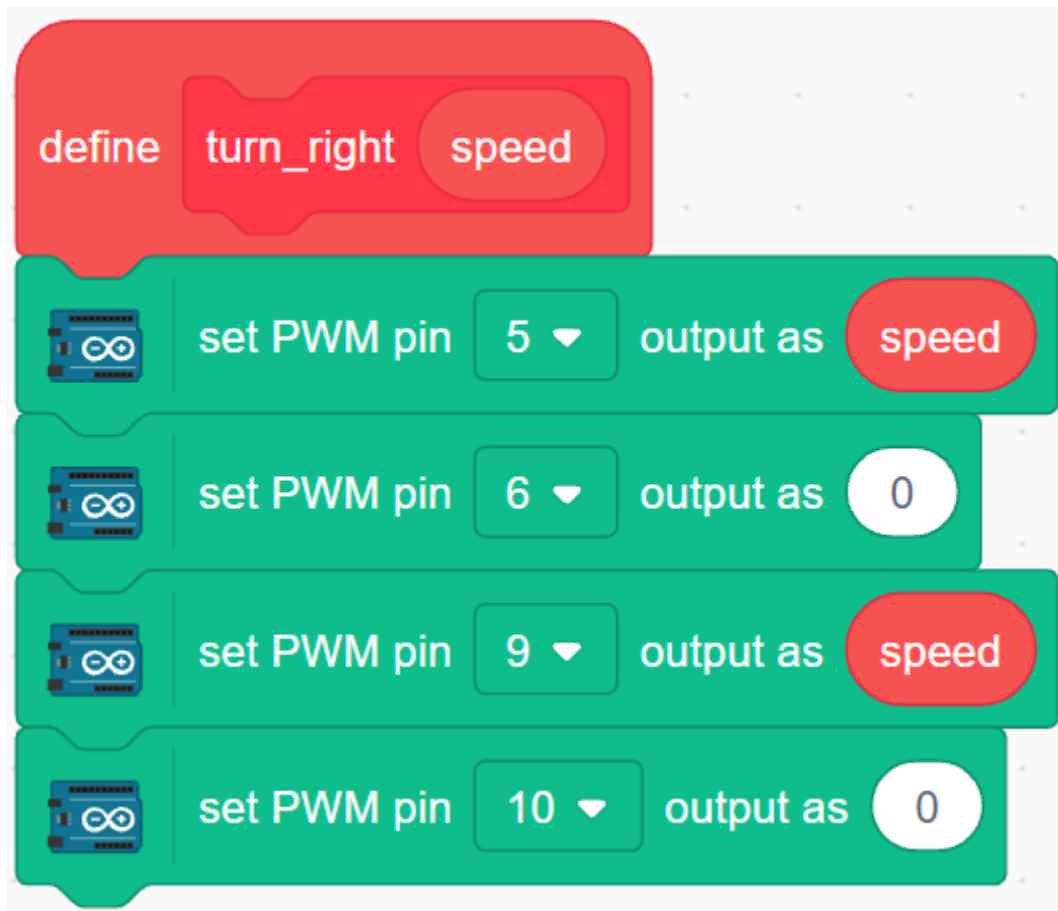
Les moteurs gauche et droit tournent dans le sens des aiguilles d'une montre en même temps pour faire tourner la voiture à gauche.





#### 4. Tourner à Droite

De même, faites tourner les moteurs gauche et droit dans le sens inverse des aiguilles d'une montre pour tourner la voiture à droite.



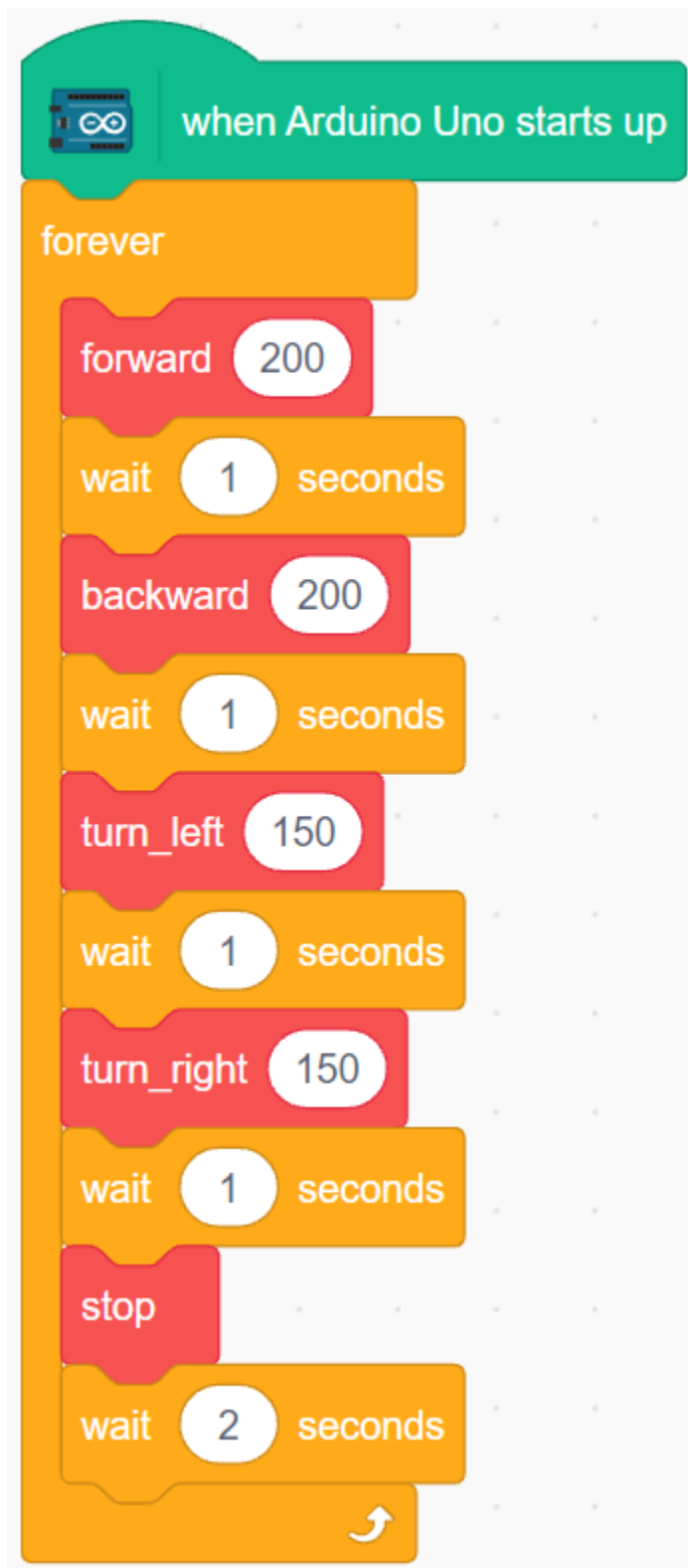
### 5. Arrêt

Arrêtez la voiture en réglant tous les moteurs à 0.



### 6. Faire bouger la voiture

Faites bouger la voiture en avant, en arrière, à gauche et à droite pendant 1 seconde, puis arrêtez. Comme tous les blocs sont placés dans le bloc [Forever], vous verrez que la voiture répète les actions ci-dessus.



## 8.28 3.3 Suivre la ligne

La voiture est équipée d'un module de Suivi de Ligne, qui peut être utilisé pour faire suivre à la voiture une ligne noire.

Avant de commencer le projet, vous devez construire une carte de courbe avec du ruban de ligne noire, la largeur de ligne recommandée est entre 0,8-1,5 cm et l'angle du virage ne doit pas être inférieur à 90 degrés.

### 8.28.1 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

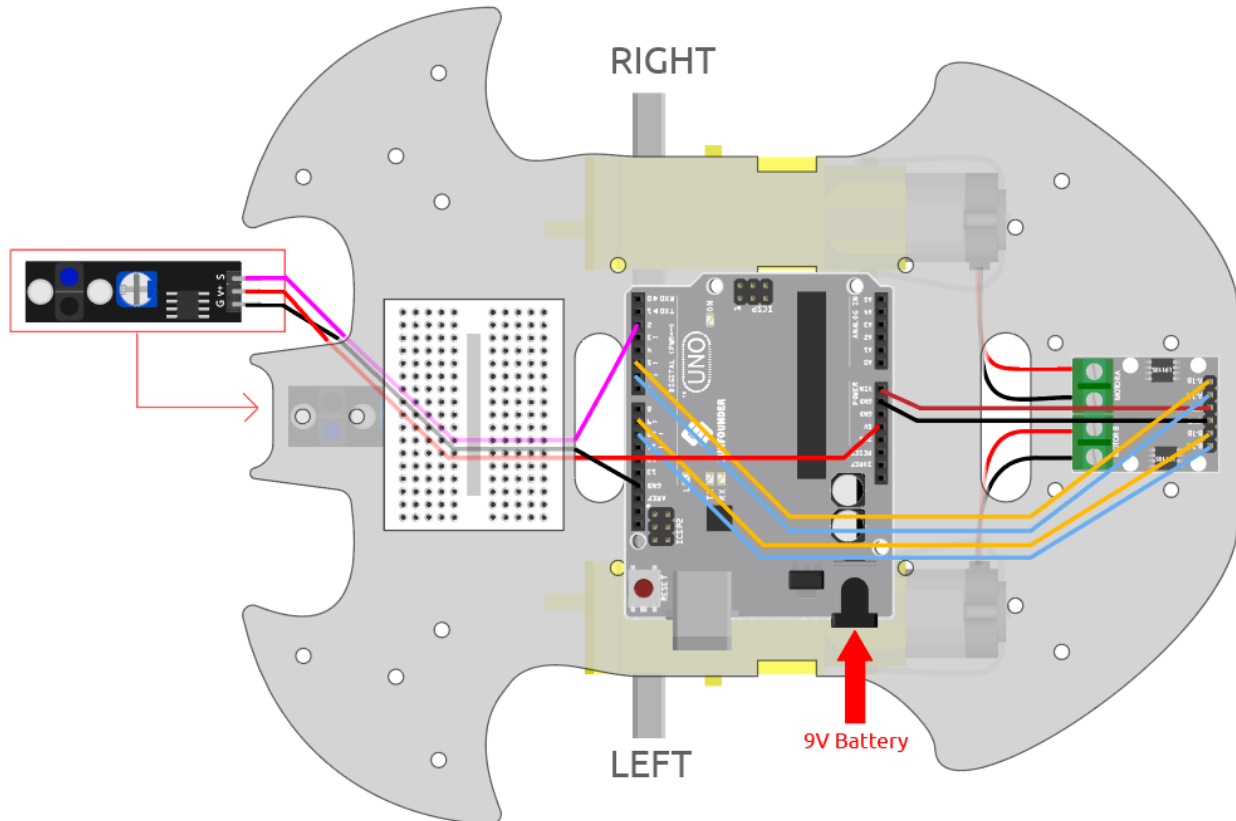
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module de Suivi de Ligne</i>	

### 8.28.2 Construire le Circuit

Il s'agit d'un module numérique de Suivi de Ligne, lorsqu'une ligne noire est détectée, il sort une valeur de 1 ; lorsqu'une ligne blanche est détectée, il sort une valeur de 0. De plus, vous pouvez ajuster sa distance de détection grâce au potentiomètre sur le module.

Construisez le circuit selon le schéma suivant.

Module de Suivi de Ligne	Carte R3
S	2
V+	5V
G	GND



### 8.28.3 Ajuster le Module

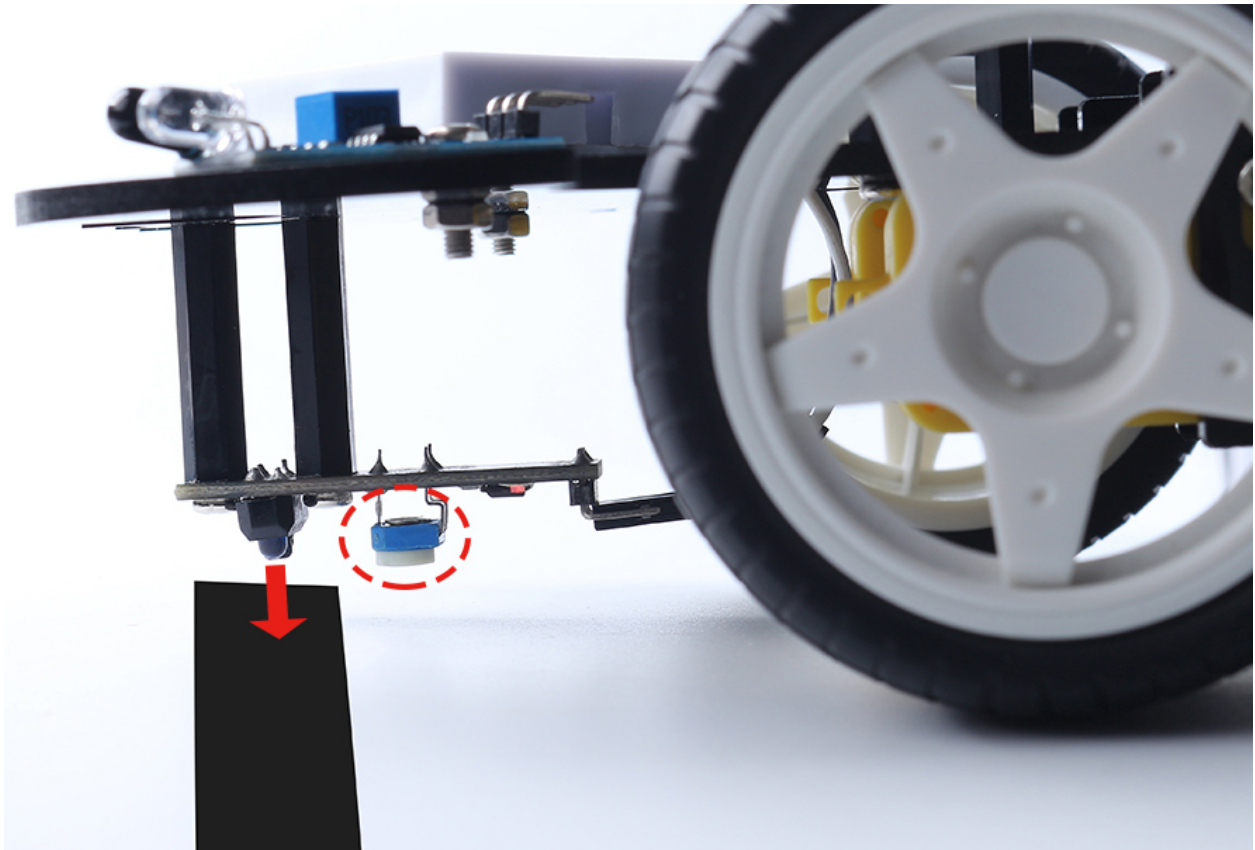
Avant de commencer le projet, vous devez ajuster la sensibilité du module.

Câblez selon le schéma ci-dessus, puis alimentez la carte R3 (soit directement via le câble USB, soit via le câble de la pile bouton 9V), sans télécharger le code.

Collez un ruban électrique noir sur la table et placez la voiture dessus.

Observez la LED de signal sur le module pour vous assurer qu'elle s'allume sur la table blanche et s'éteint sur le ruban noir.

Si ce n'est pas le cas, vous devez ajuster le potentiomètre sur le module, afin qu'il puisse produire l'effet ci-dessus.

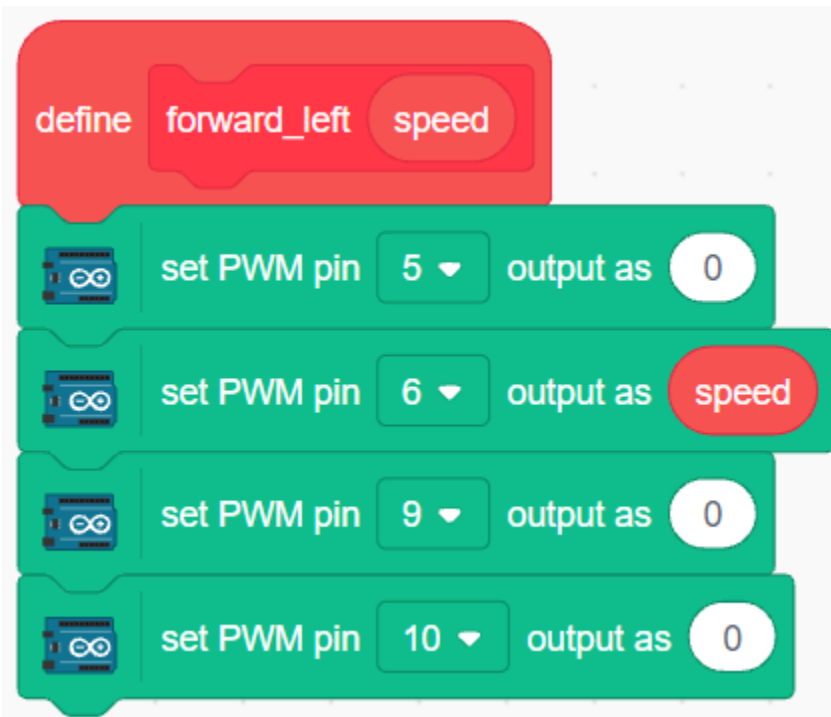


#### 8.28.4 Programmation

Créez maintenant 2 blocs qui permettent à la voiture de se déplacer soit vers l'avant gauche, soit vers l'avant droit.

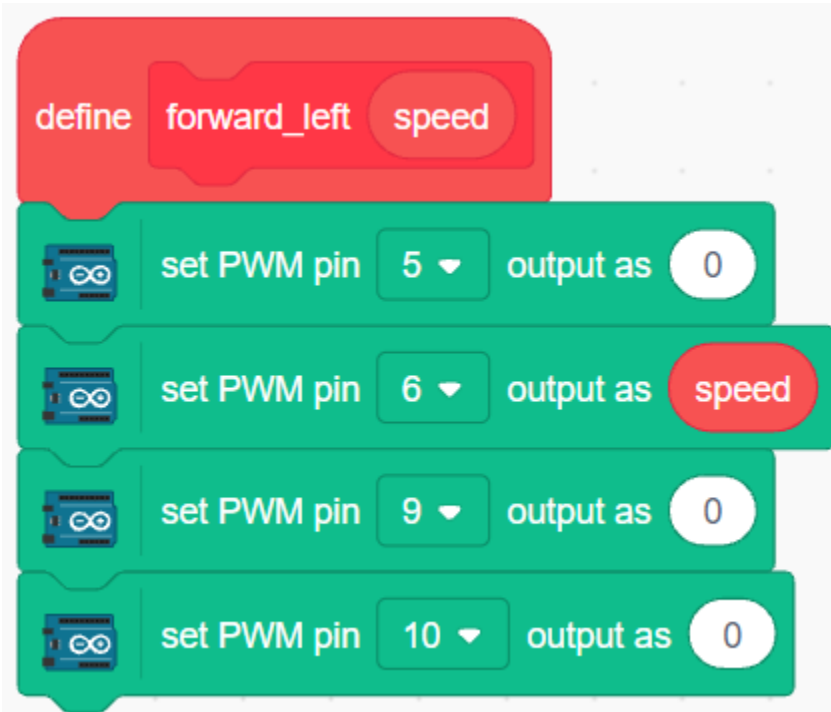
##### 1. Se déplacer vers l'avant gauche

Lorsque le moteur droit tourne dans le sens des aiguilles d'une montre et que le moteur gauche reste immobile, la voiture se déplace légèrement vers l'avant gauche.



## 2. Se déplacer vers l'avant droit

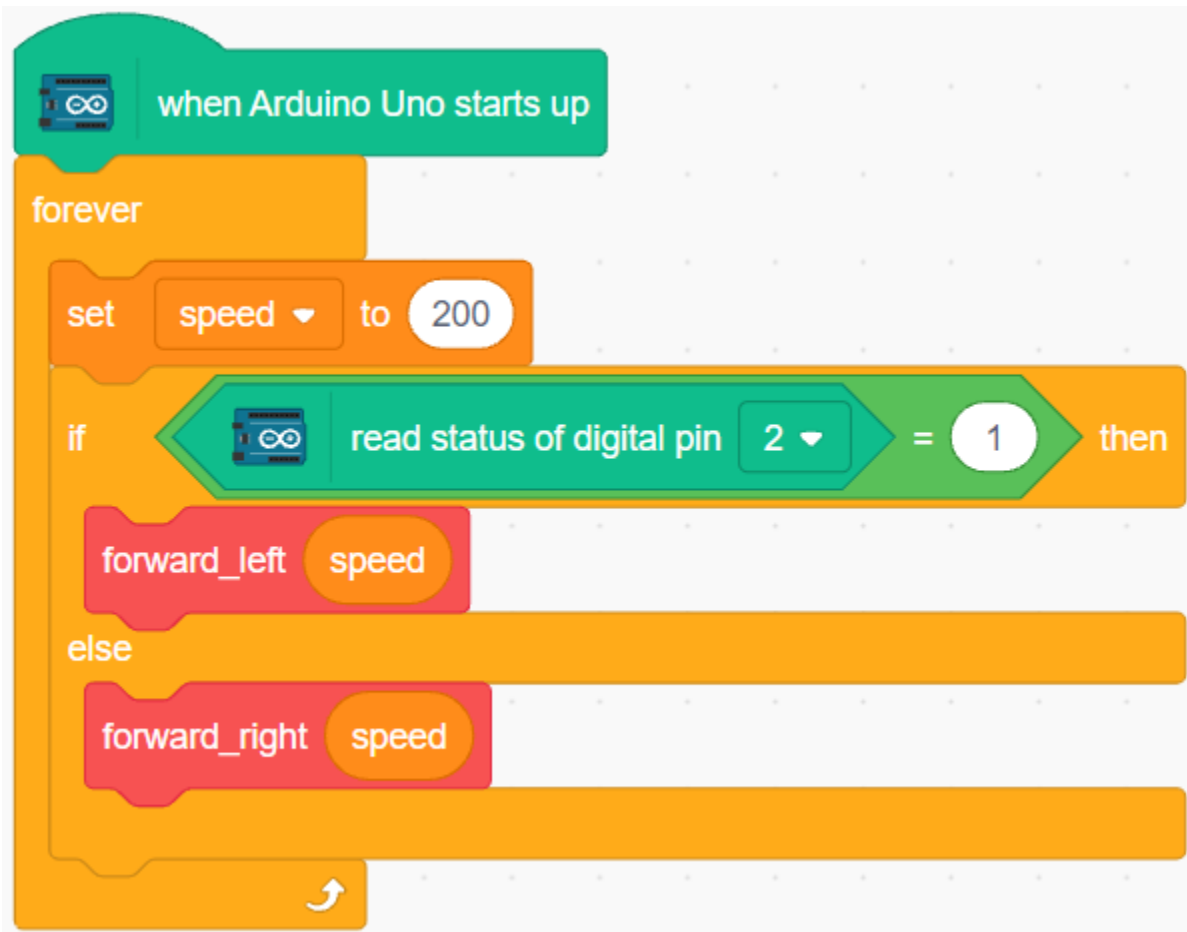
De même, lorsque le moteur gauche tourne dans le sens inverse des aiguilles d'une montre et que le moteur droit ne bouge pas, la voiture se déplace légèrement vers la droite.



## 3. Suivi de Ligne

Lisez la valeur du module de Suivi de Ligne, si c'est 1, cela signifie qu'une ligne noire a été détectée, laissez la voiture se déplacer vers l'avant gauche, sinon avancez vers la droite.





Après avoir téléchargé le code sur la carte R3, alignez le module de Suivi de Ligne sous la voiture avec la ligne noire, et vous verrez la voiture suivre la ligne.

## 8.29 3.4 Suivez Votre Main

Pensez à cette voiture comme à votre animal de compagnie ici, et quand vous lui ferez signe, elle viendra en courant vers vous.

### 8.29.1 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

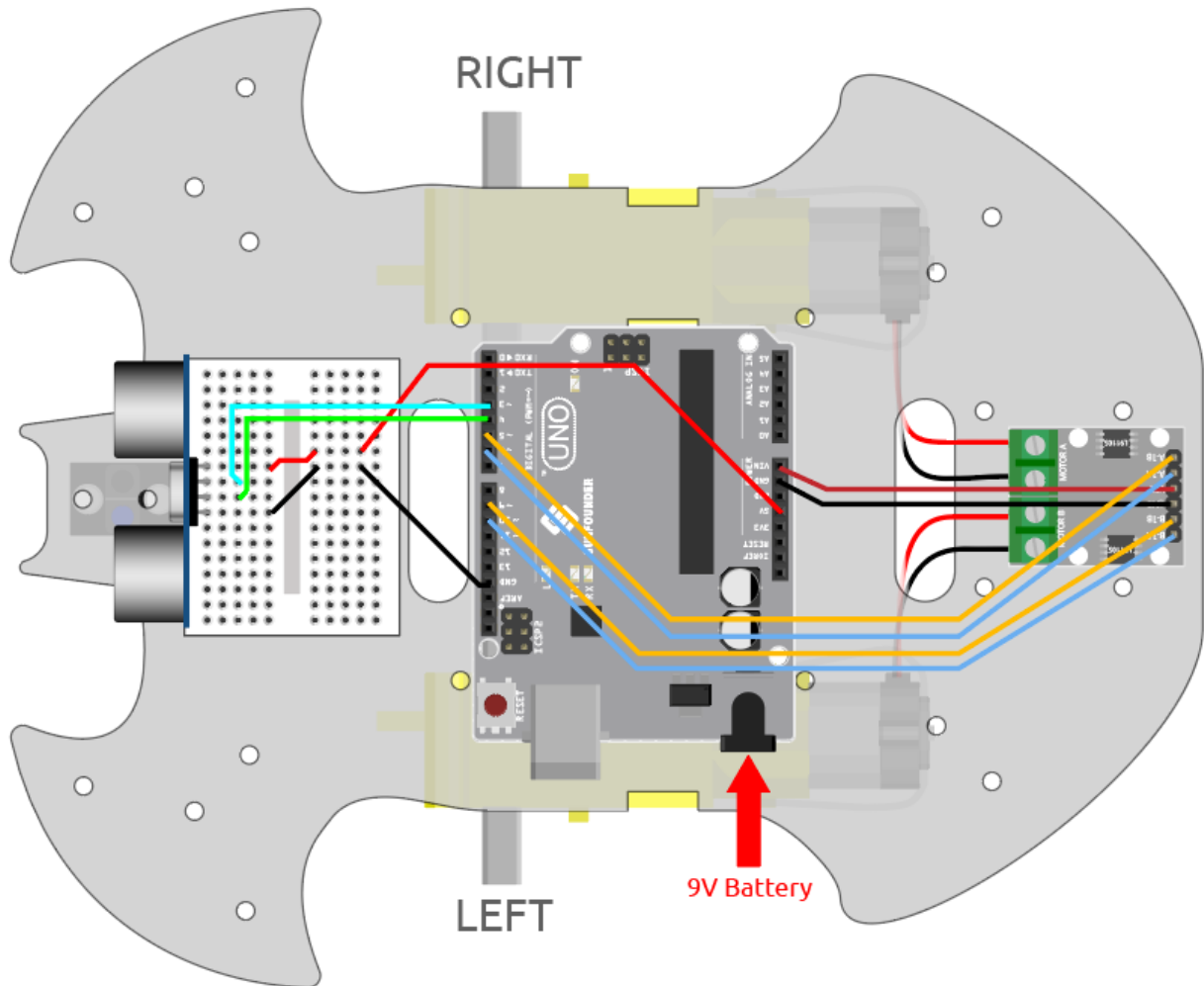
INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module Ultrasonique</i>	

### 8.29.2 Construire le Circuit

Un module de capteur ultrasonique est un instrument qui mesure la distance jusqu'à un objet à l'aide d'ondes sonores ultrasoniques. Il possède deux sondes. L'une envoie des ondes ultrasoniques et l'autre reçoit les ondes et transforme le temps d'envoi et de réception en une distance, détectant ainsi la distance entre l'appareil et un obstacle.

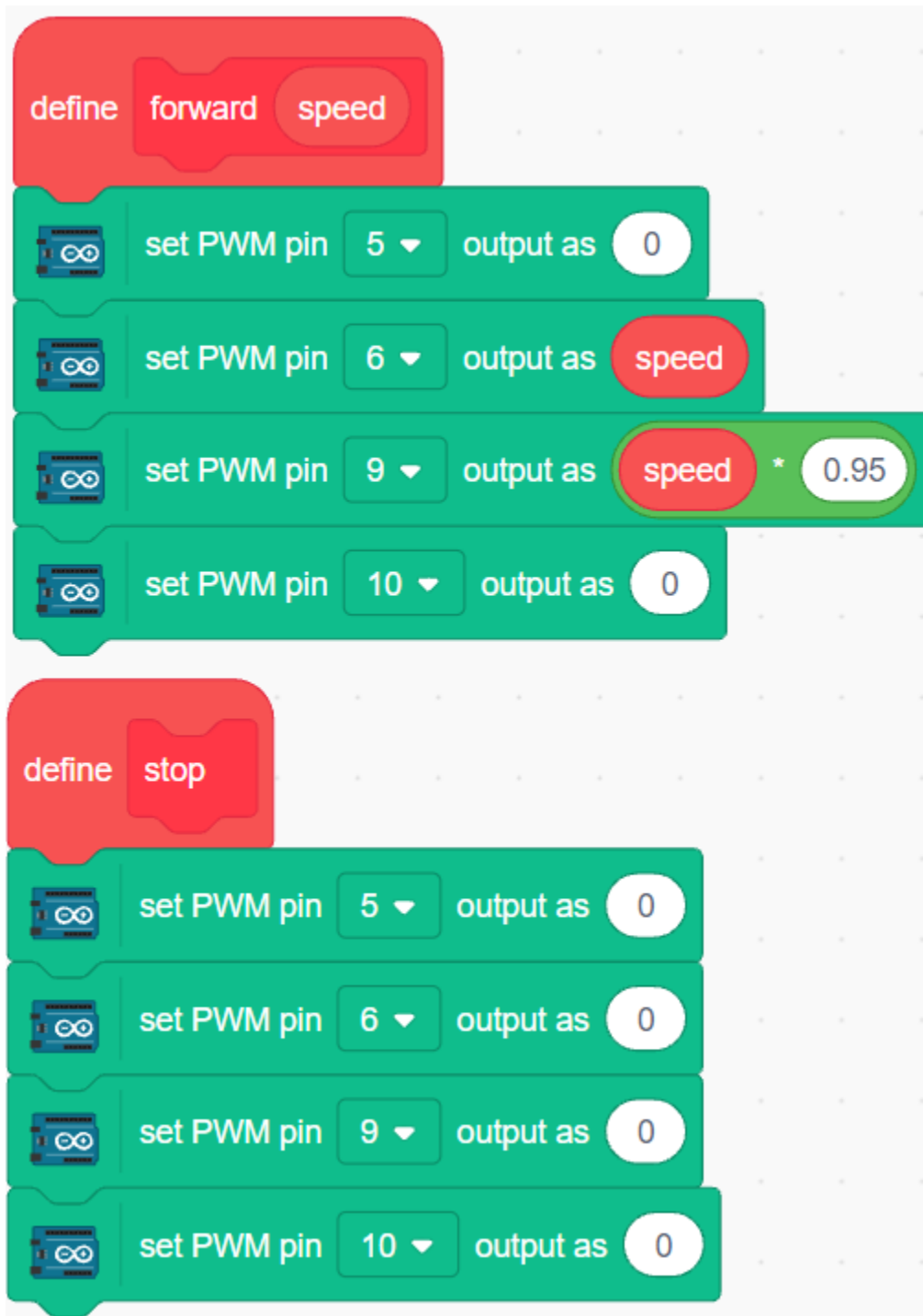
Construisez le circuit selon le schéma suivant.

Module Ultrasonique	Carte R3
Vcc	5V
Trig	3
Echo	4
Gnd	GND

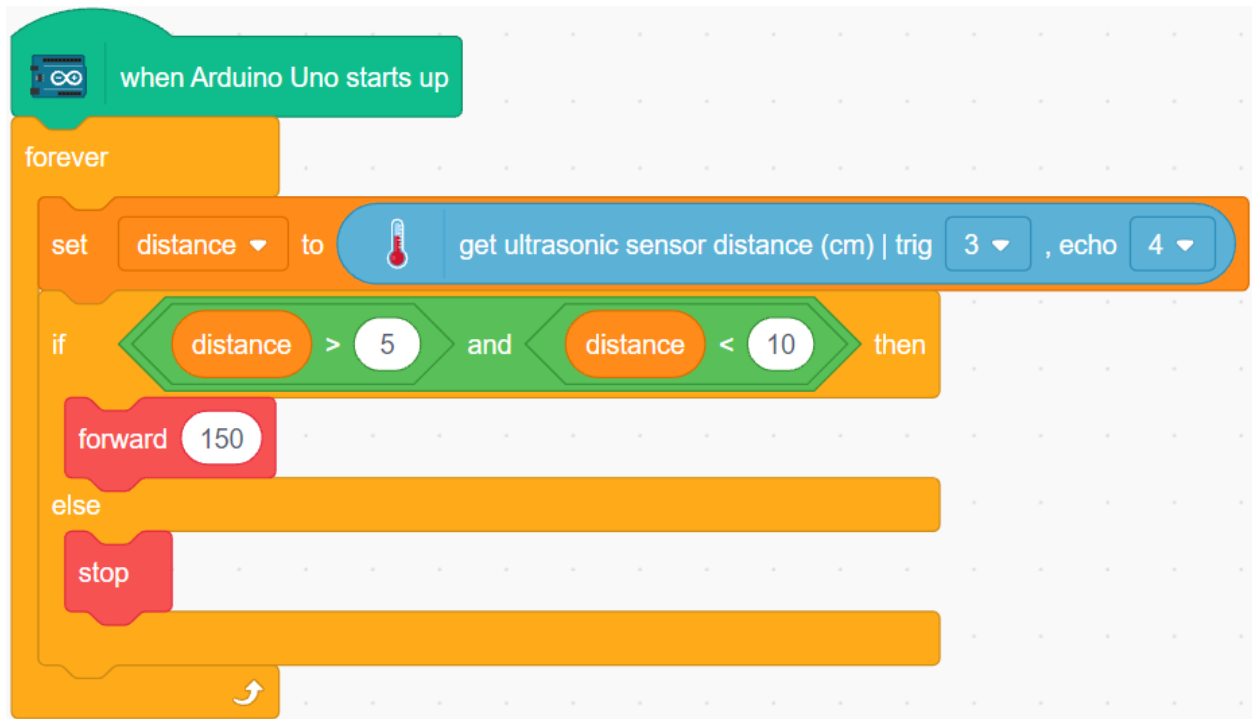


### 8.29.3 Programmation

Créez des blocs pour faire avancer et arrêter la voiture.



Mettez votre main devant la voiture, puis lisez la valeur du module ultrasonique, si la distance détectée de votre main est de 5-10 cm, alors faites avancer la voiture, sinon arrêtez.



## 8.30 3.5 Évitement d'obstacles

Deux modules d'évitement d'obstacles infrarouges sont montés à l'avant de la voiture, pouvant détecter certains obstacles proches.

Dans ce projet, la voiture peut avancer librement, et lorsqu'elle rencontre un obstacle, elle est capable de l'éviter et de continuer à se déplacer dans d'autres directions.

### 8.30.1 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module d'Évitement d'Obstacle</i>	

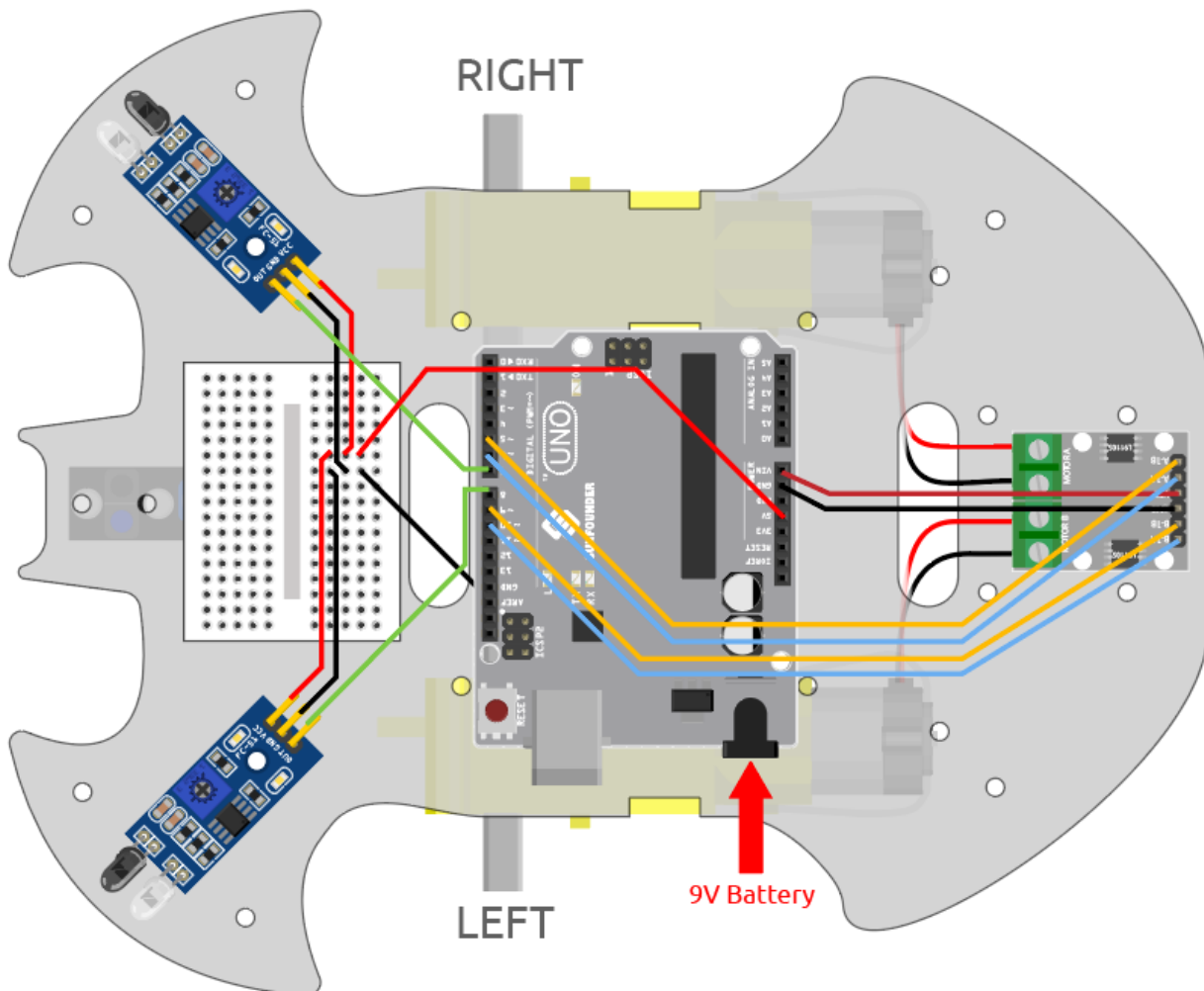
### 8.30.2 Construire le Circuit

Le module d'évitement d'obstacles est un capteur de proximité infrarouge à distance ajustable dont la sortie est normalement haute et basse lorsqu'un obstacle est détecté.

Construisez le circuit selon le schéma ci-dessous.

Module IR Gauche	Carte R3
OUT	8
GND	GND
VCC	5V

Module IR Droit	Carte R3
OUT	7
GND	GND
VCC	5V



### 8.30.3 Ajuster le Module

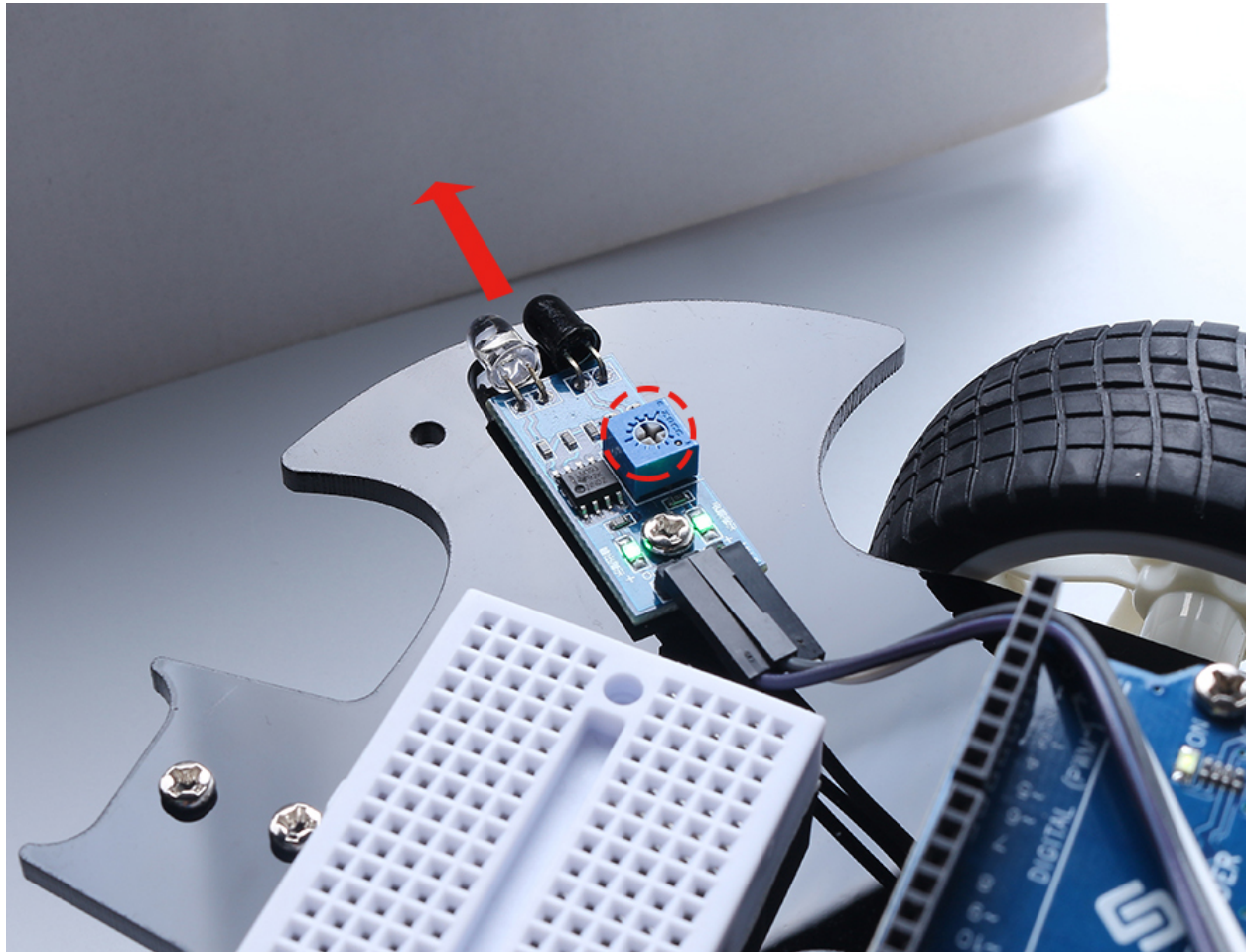
Avant de commencer le projet, vous devez ajuster la distance de détection du module.

Câblez selon le schéma ci-dessus, alimentez la carte R3 (soit en branchant directement le câble USB, soit en clipsant le câble de la pile 9V), sans télécharger le code.

Placez un cahier ou tout autre objet plat à environ 5 cm devant le module d'évitement d'obstacles IR.

Ensuite, utilisez un tournevis pour tourner le potentiomètre sur le module jusqu'à ce que l'indicateur de signal sur le module s'allume juste, afin d'ajuster sa distance de détection maximale de 5 cm.

Suivez la même méthode pour ajuster un autre module infrarouge.



### 8.30.4 Programmation

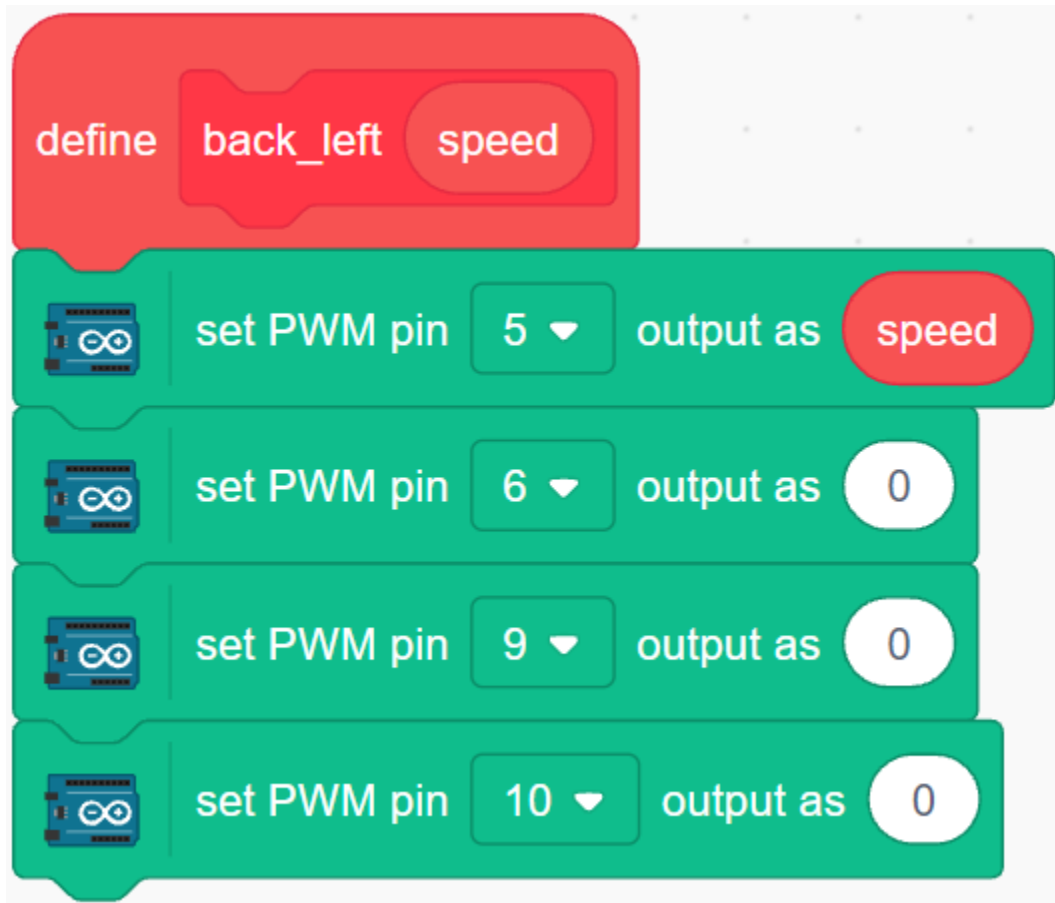
L'effet que nous voulons atteindre.

- Lorsque le module IR gauche détecte un obstacle, la voiture recule vers la gauche
- Lorsque le module IR droit détecte un obstacle, la voiture recule vers la droite.
- Si les deux modules IR détectent l'obstacle, la voiture reculera directement.
- Sinon, la voiture avancera.

Créez maintenant les blocs correspondants.

#### 1. La voiture recule vers la gauche

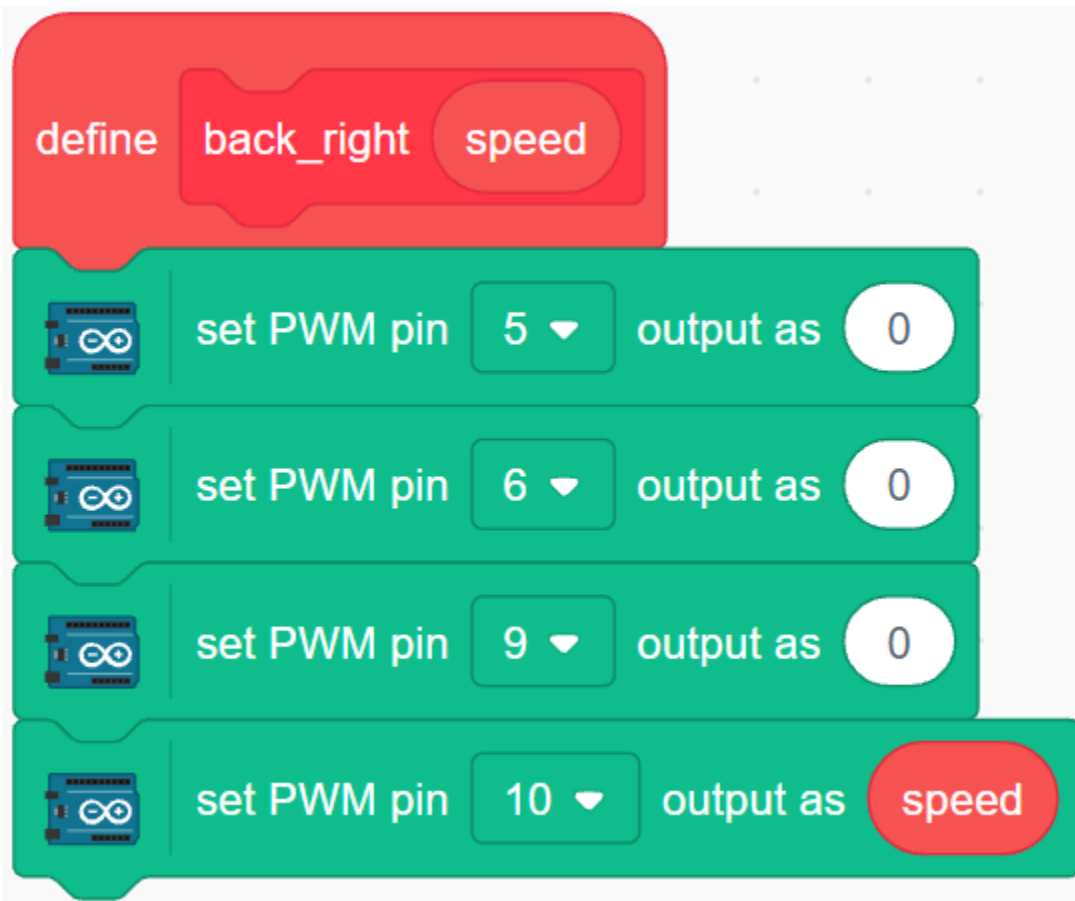
Lorsque le moteur droit tourne dans le sens inverse des aiguilles d'une montre et que le moteur gauche ne tourne pas, la voiture reculera vers la gauche.



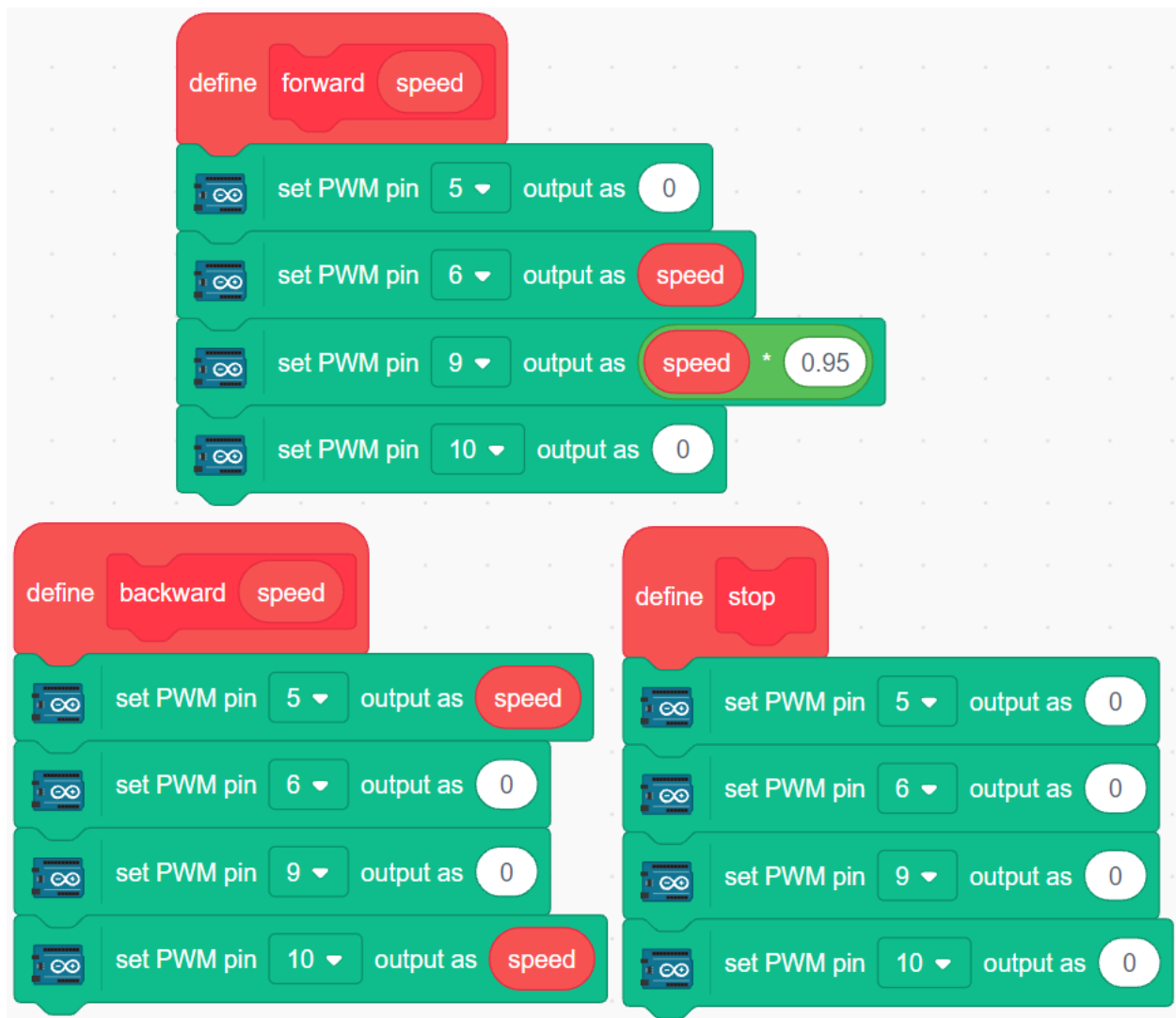
### 2. La voiture recule vers la droite

Lorsque le moteur gauche tourne dans le sens des aiguilles d'une montre et que le moteur droit ne tourne pas, la voiture reculera vers la droite.



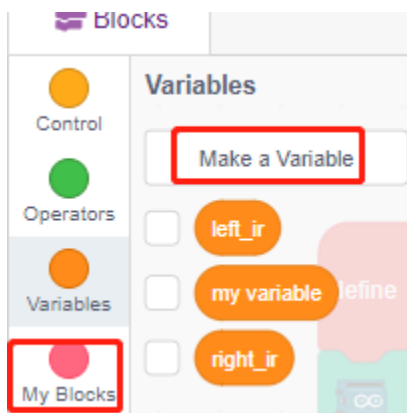


### 3. La voiture avance, recule et s'arrête

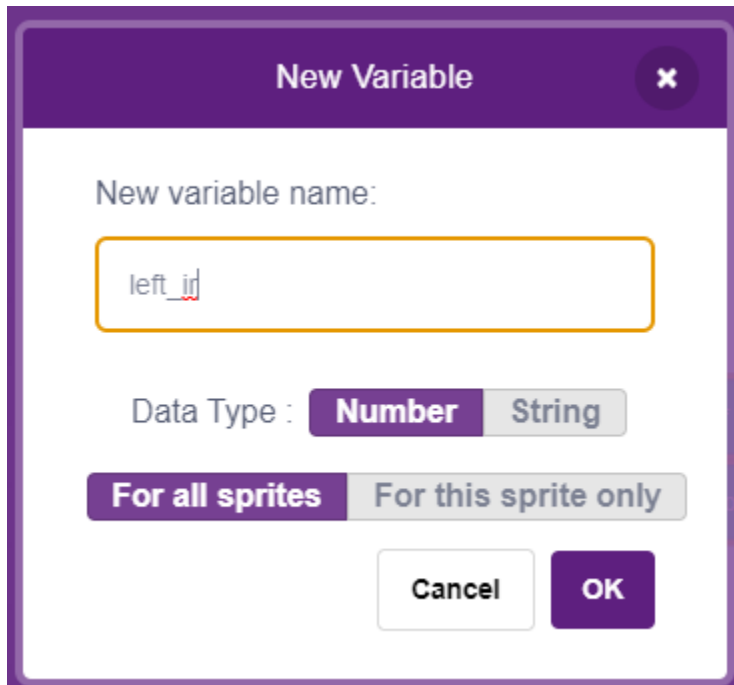


#### 4. Lire les valeurs des 2 modules IR

Cliquez sur **Make a variable** dans la palette **Variables**.



Entrez le nom de la variable et cliquez sur **OK** pour créer une nouvelle variable.

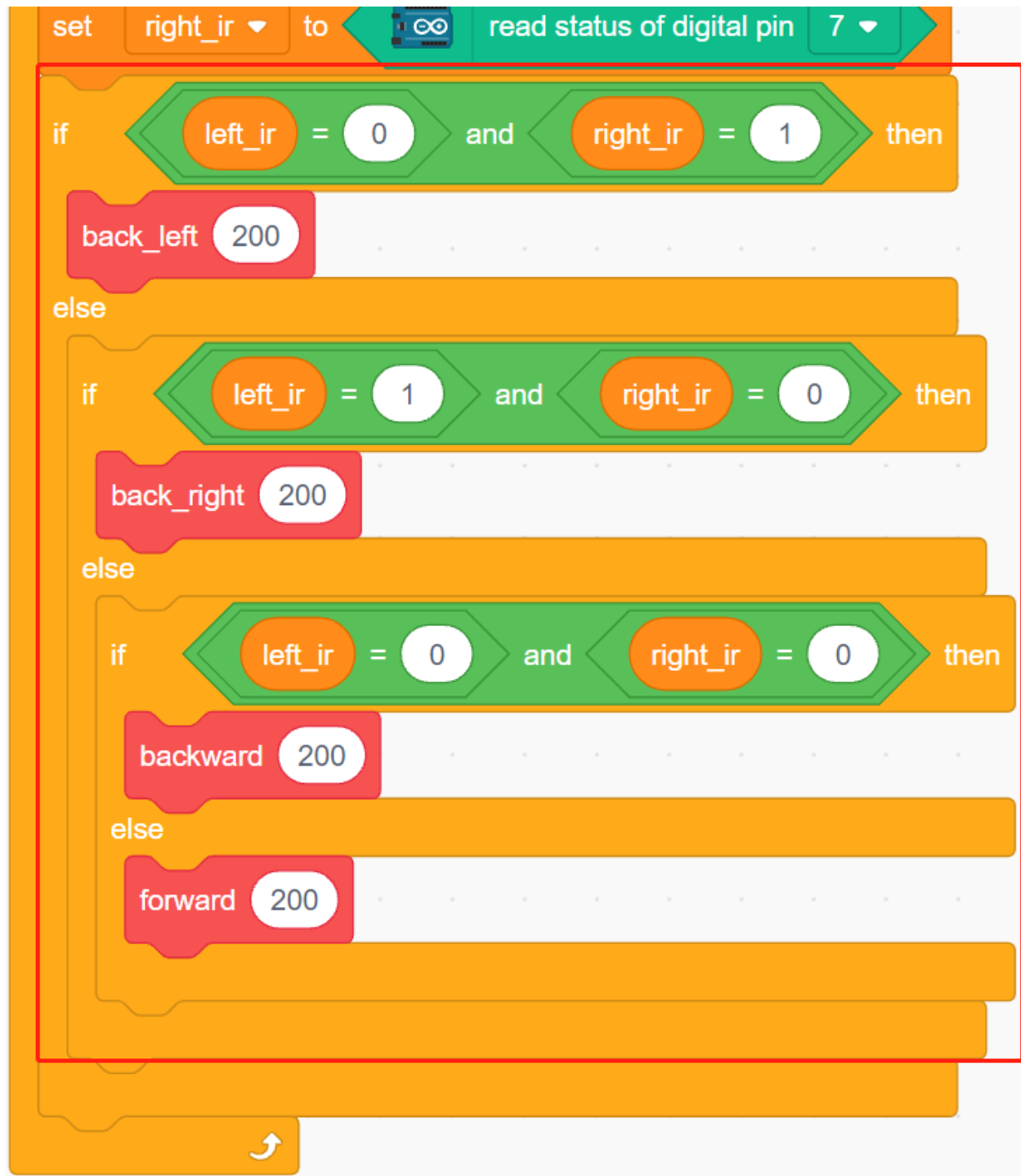


Lisez les valeurs des modules d'évitement d'obstacles IR gauche et droit et stockez-les dans les 2 nouvelles variables.



## 5. Évitement d'obstacles

- Lorsque le module IR gauche est à 0 (obstacle détecté) et que le module IR droit est à 1, faites reculer la voiture vers la gauche.
- Lorsque le module IR droit est à 0 (obstacle détecté), faites reculer la voiture vers la droite.
- Si les 2 modules IR détectent l'obstacle en même temps, la voiture reculera.
- Sinon, la voiture continuera d'avancer.



## 8.31 3.6 Suivez Votre Main 2

Dans le projet *3.4 Suivez Votre Main*, seul le module ultrasonique est utilisé, il ne peut que suivre votre main vers l'avant.

Dans ce projet, nous utilisons 2 modules d'évitement d'obstacles IR en même temps, afin que la voiture puisse suivre votre main à gauche ou à droite.

### 8.31.1 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module Ultrasonique</i>	
<i>Module d'Évitement d'Obstacle</i>	

### 8.31.2 Construire le Circuit

Connectez le module ultrasonique et les deux modules d'évitement d'obstacles IR en même temps.

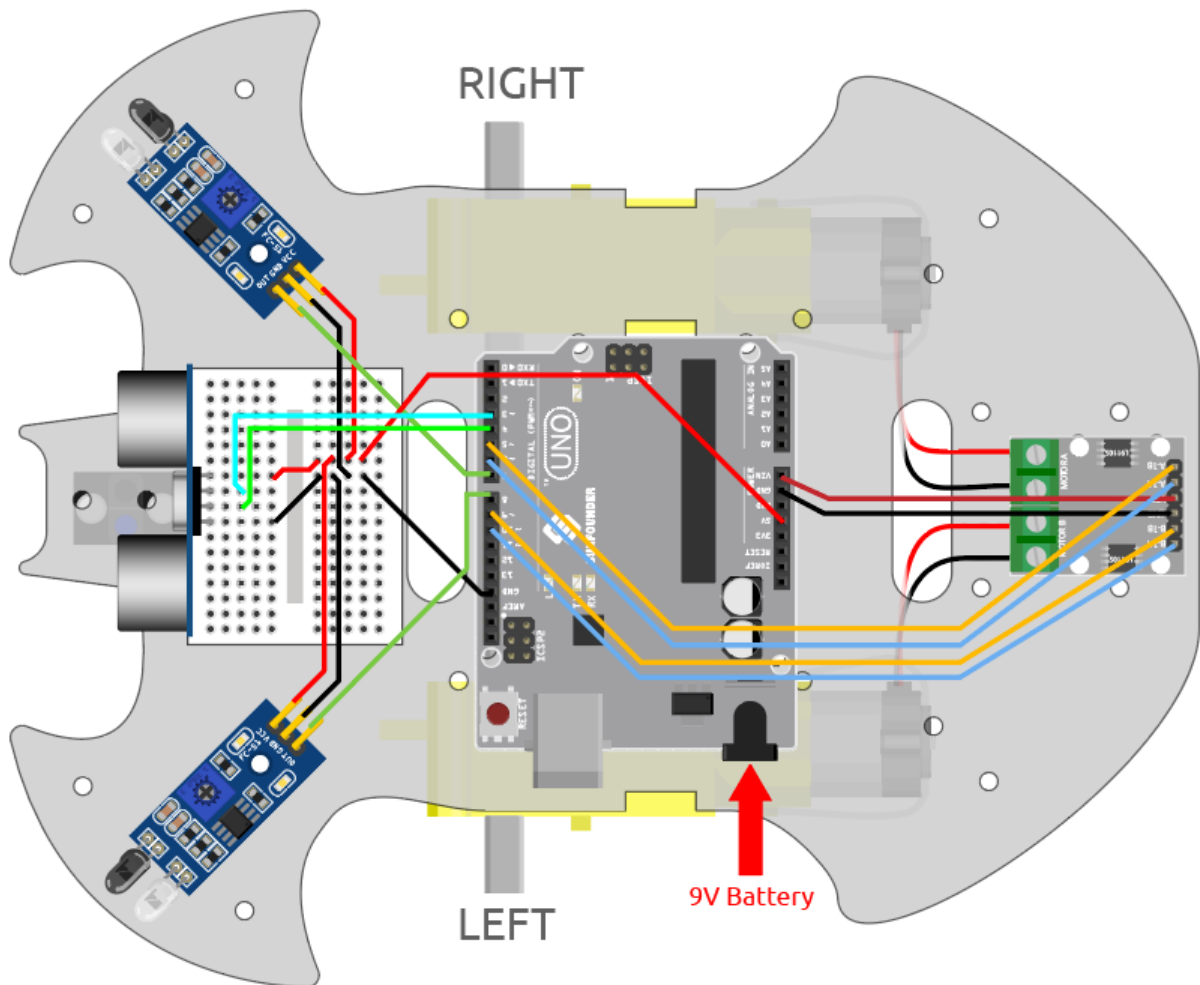
Le câblage entre l'ultrasonique et la carte R3 est le suivant.

Module Ultrasonique	Carte R3
Vcc	5V
Trig	3
Echo	4
Gnd	GND

Le câblage des 2 modules d'évitement d'obstacles IR vers la carte R3 est le suivant.

Module IR Gauche	Carte R3
OUT	8
GND	GND
VCC	5V

Module IR Droit	Carte R3
OUT	7
GND	GND
VCC	5V



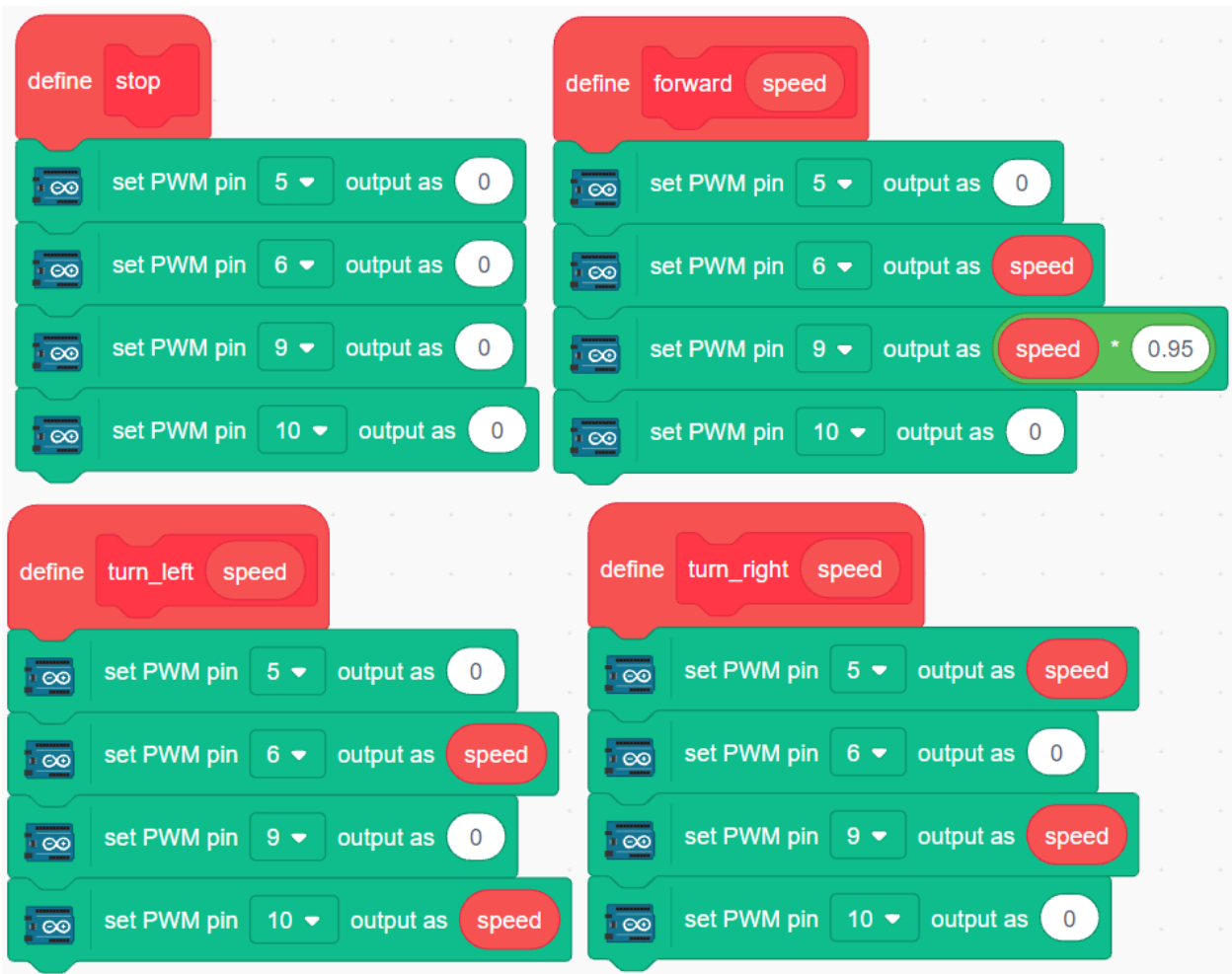
### 8.31.3 Programmation

L'effet à atteindre par ce projet est le suivant

- L'ultrason détecte votre main à environ 5-10 cm devant et laisse la voiture suivre.
- Le module infrarouge à gauche détecte votre main et tourne à gauche.
- Le module IR droit détecte votre main et tourne à droite.

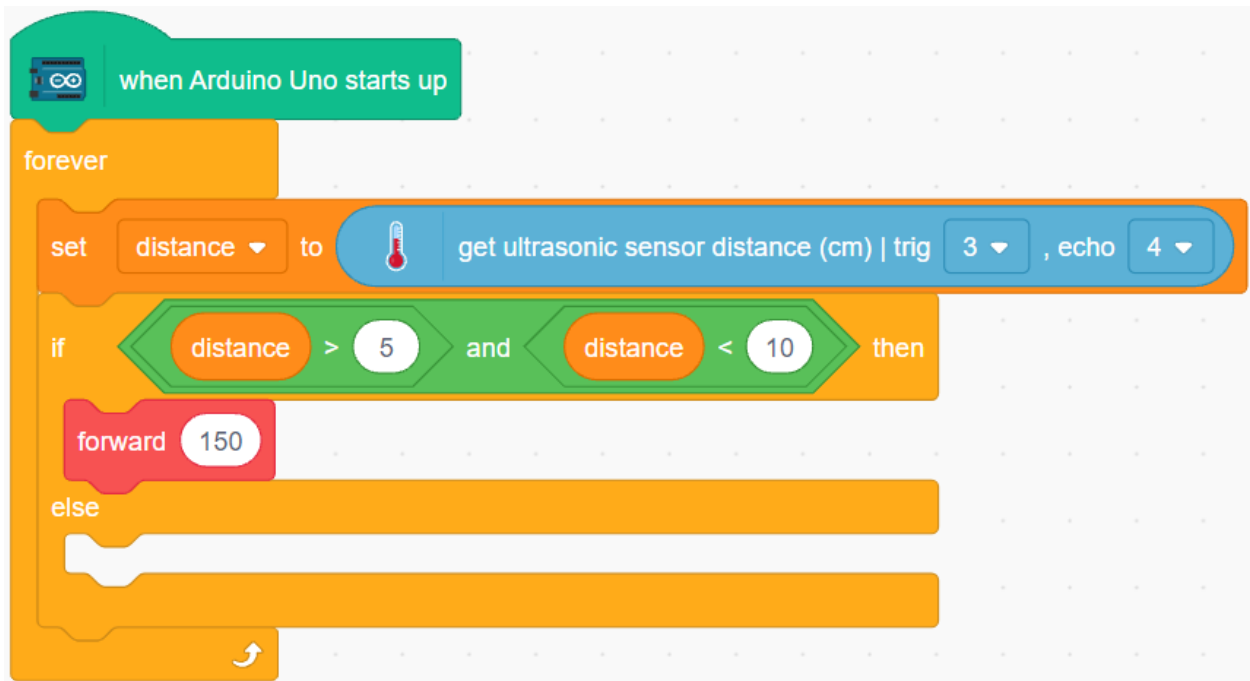
#### 1. Créer un bloc

Créez des blocs qui permettent à la voiture d'avancer, de tourner à gauche, de tourner à droite et de s'arrêter.



## 2. Suivre pour avancer

Lisez la valeur de l'ultrasonique et si votre main est détectée à une distance de 5-10 cm, laissez la voiture suivre.

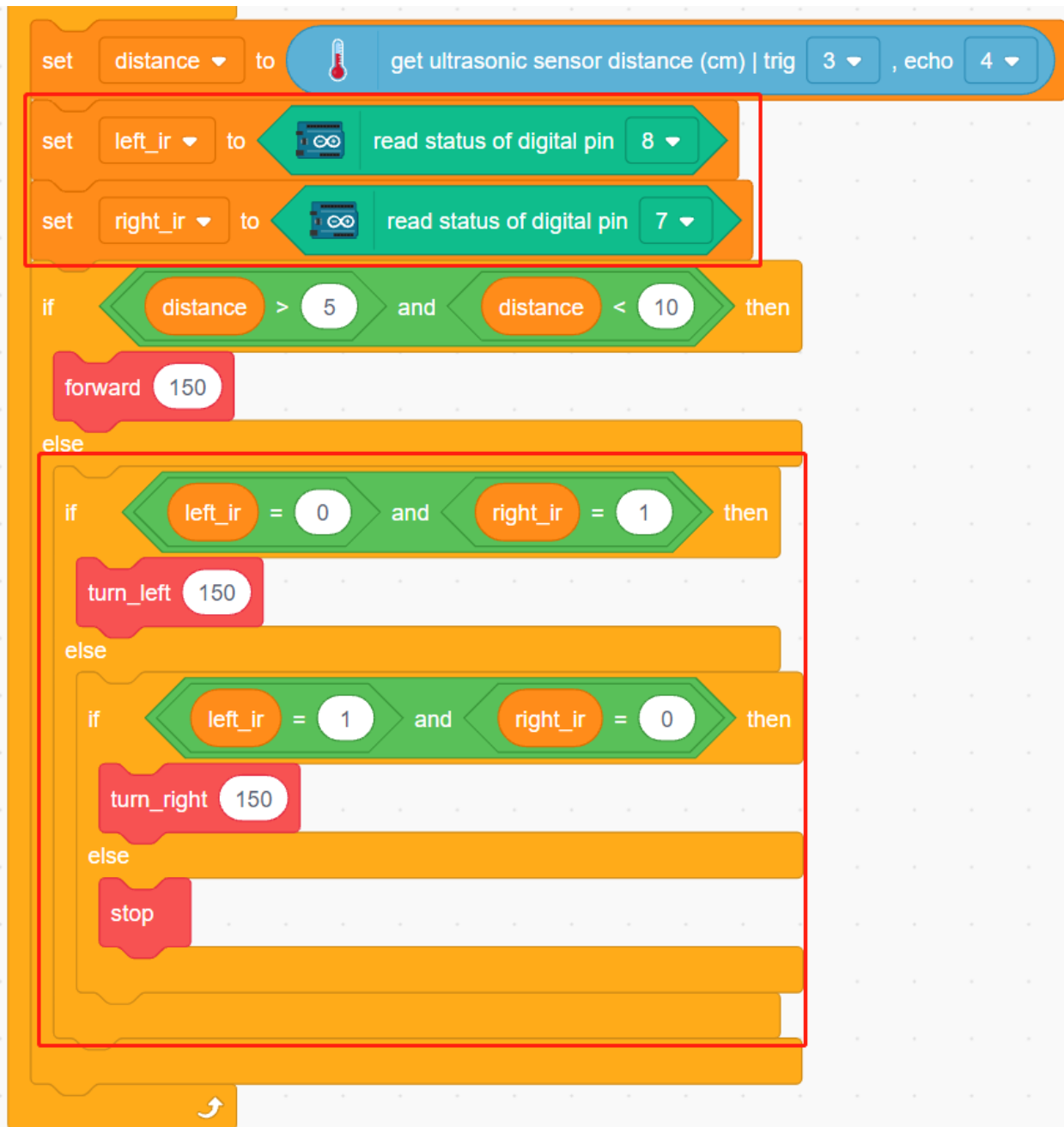


### 3. Suivre pour tourner à gauche et à droite

Lisez les valeurs des modules IR gauche et droit.

- Si le module IR gauche détecte votre main, tournez à gauche.
- Si le module IR droit détecte votre main, tournez à droite.
- Si aucun des modules IR et ultrasonique ne détecte votre main, faites s'arrêter la voiture.





## 8.32 3.7 Évitement d'obstacles 2

Dans le projet 3.5 *Évitement d'obstacles*, seuls 2 modules d'évitement d'obstacles IR sont utilisés pour éviter les obstacles, mais la distance de détection du module d'évitement d'obstacles IR est courte, ce qui peut rendre la voiture trop tardive pour éviter les obstacles.

Dans ce projet, nous ajoutons également un module ultrasonique pour effectuer une détection à longue distance, afin que la voiture puisse détecter les obstacles à une plus grande distance pour prendre une décision.

### 8.32.1 Composants requis

Pour ce projet, nous avons besoin des composants suivants.

Il est certainement pratique d'acheter un kit complet, voici le lien :

Nom	ÉLÉMENTS DANS CE KIT	LIEN
3 in 1 Starter Kit	380+	

Vous pouvez également les acheter séparément via les liens ci-dessous.

INTRODUCTION DES COMPOSANTS	LIEN D'ACHAT
<i>Carte SunFounder R3</i>	
<i>Module de Contrôle Moteur L9110</i>	-
<i>Moteur TT</i>	-
<i>Module Ultrasonique</i>	
<i>Module d'Évitement d'Obstacle</i>	

### 8.32.2 Construire le Circuit

Connectez le module ultrasonique et les 2 modules d'évitement d'obstacles IR en même temps.

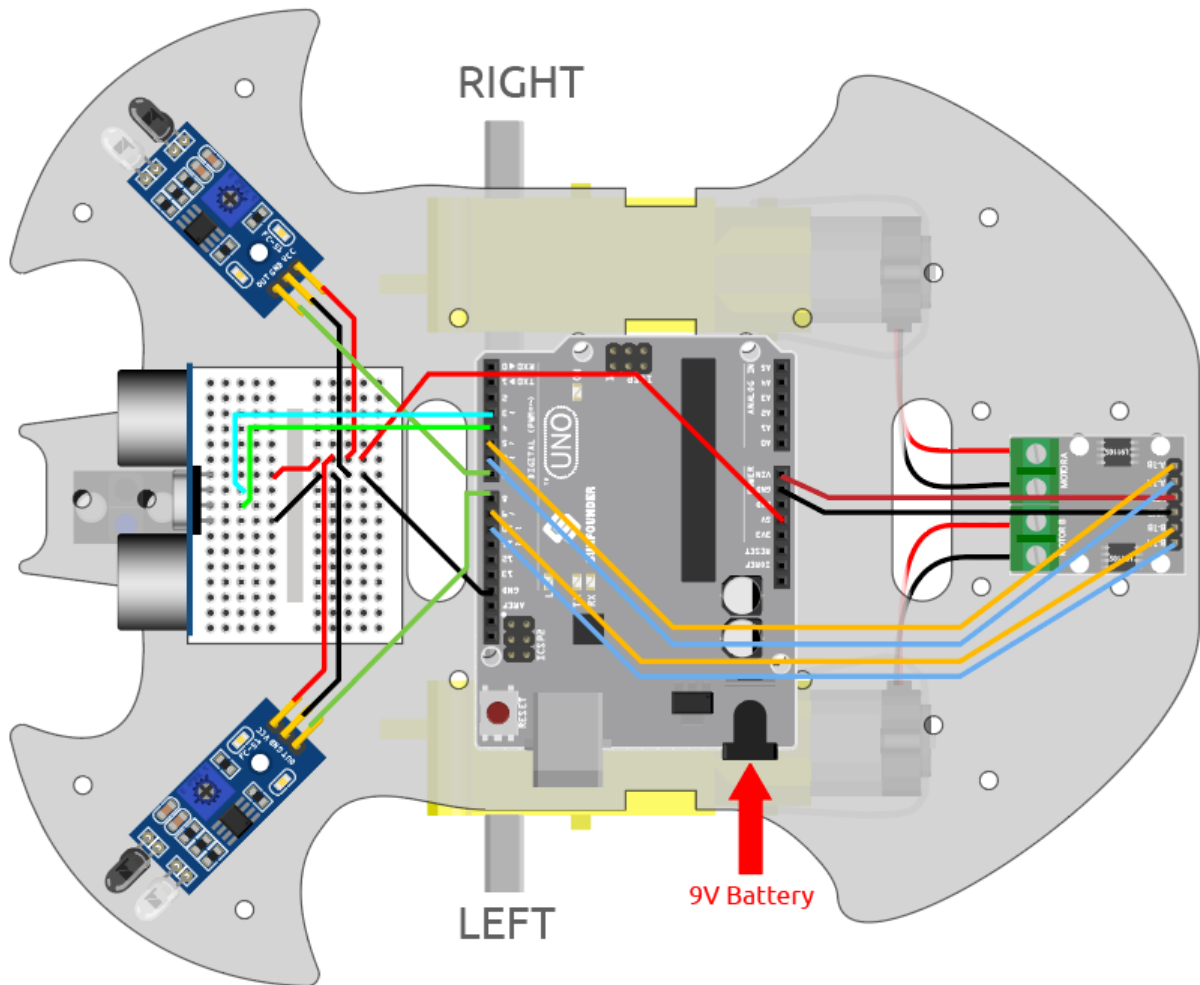
Câblez l'ultrasonique à la carte R3 comme suit.

Module Ultrasonique	Carte R3
Vcc	5V
Trig	3
Echo	4
Gnd	GND

Le câblage des 2 modules d'évitement d'obstacles IR vers la carte R3 est le suivant.

Module IR Gauche	Carte R3
OUT	8
GND	GND
VCC	5V

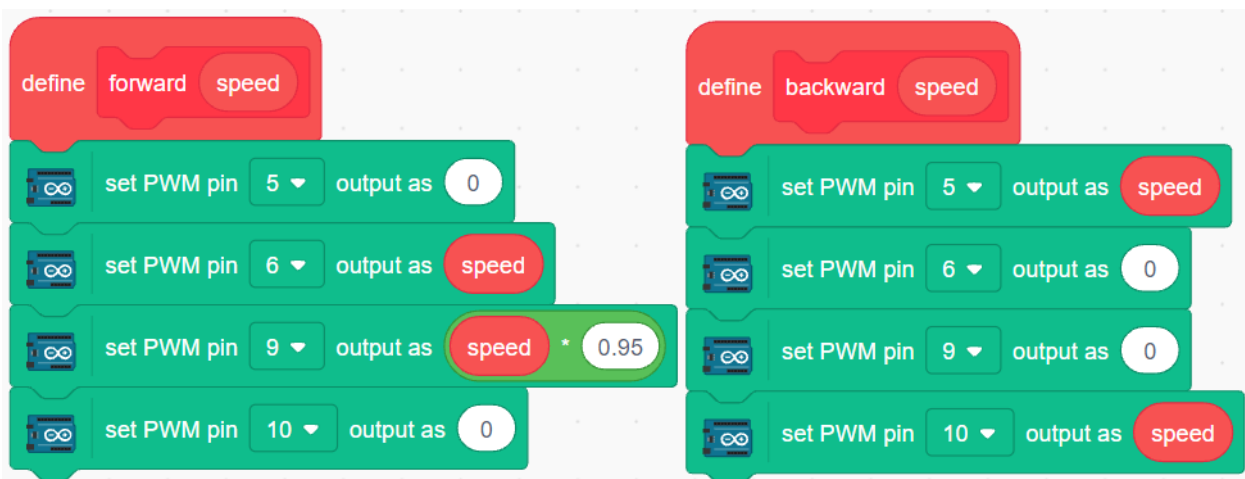
Module IR Droit	Carte R3
OUT	7
GND	GND
VCC	5V



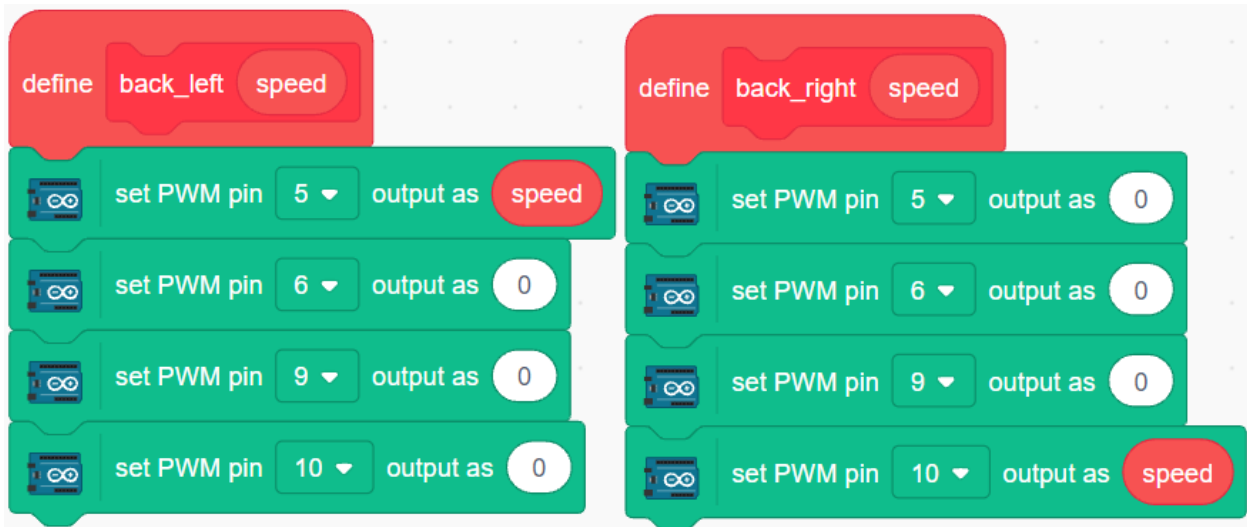
### 8.32.3 Programmation

#### 1. Créer une fonction

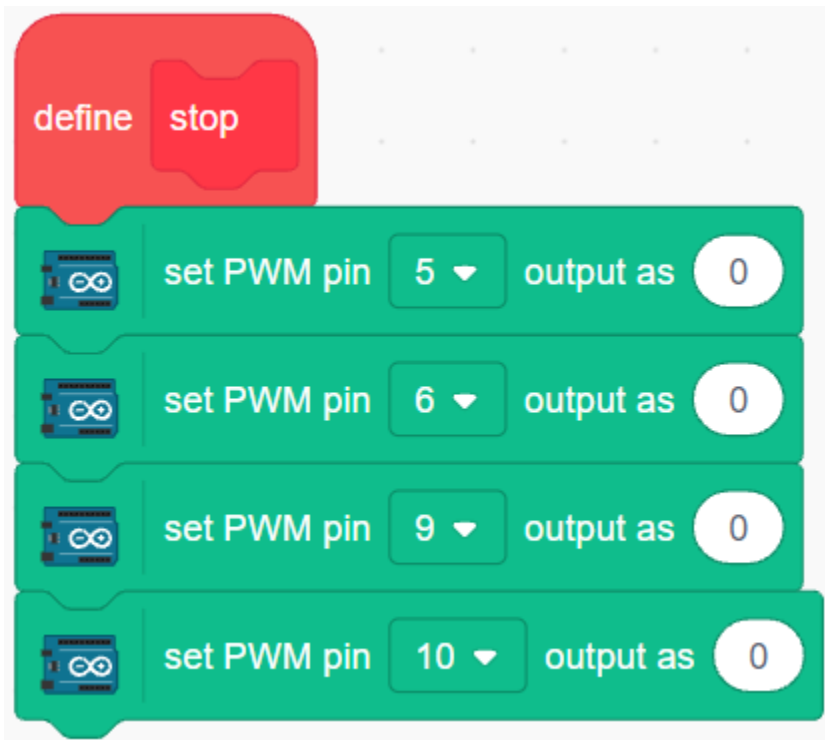
Faites avancer et reculer la voiture.



Faites reculer la voiture vers la gauche et vers la droite.



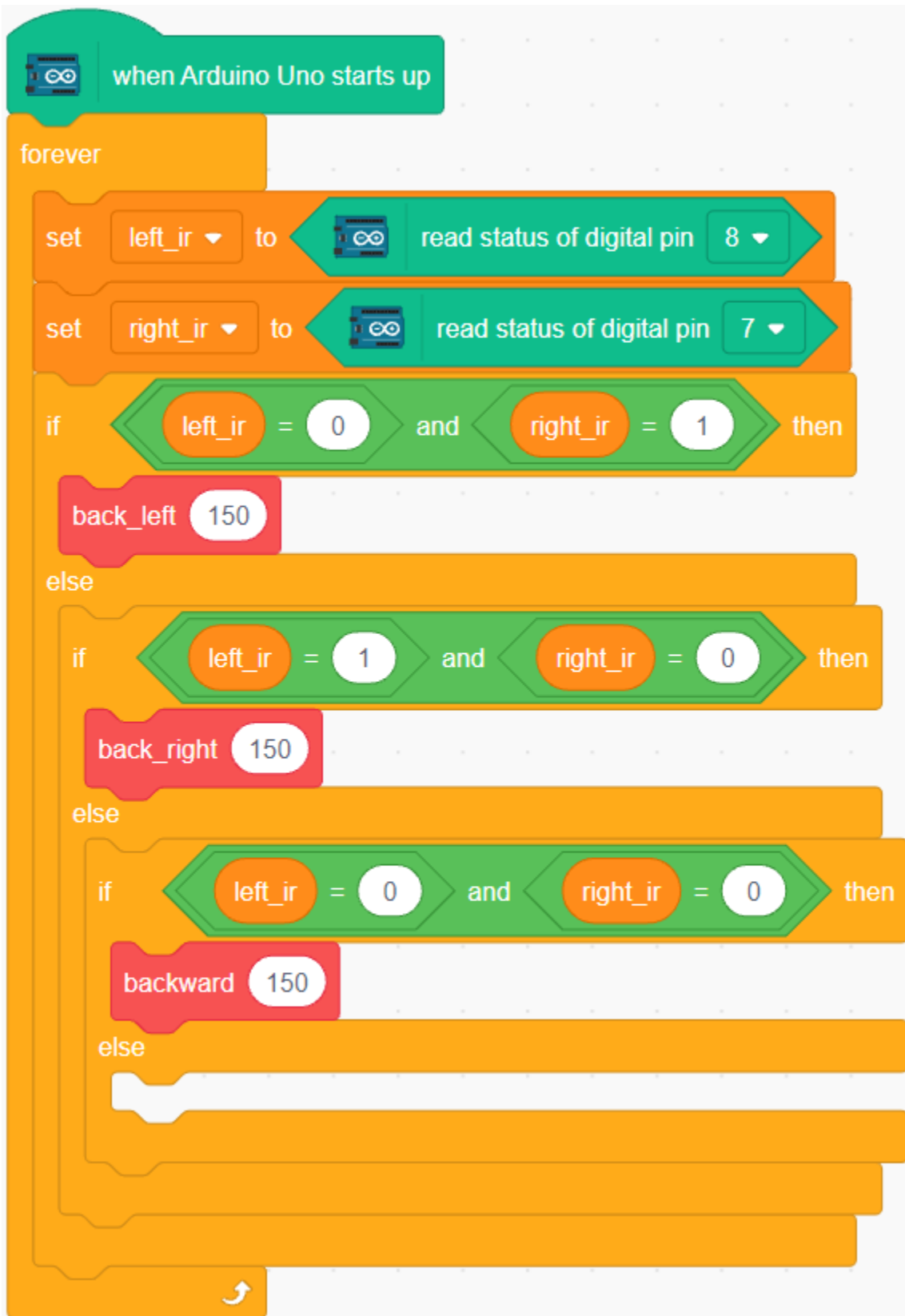
Faites s'arrêter la voiture.



## 2. Évitement d'obstacles d'urgence

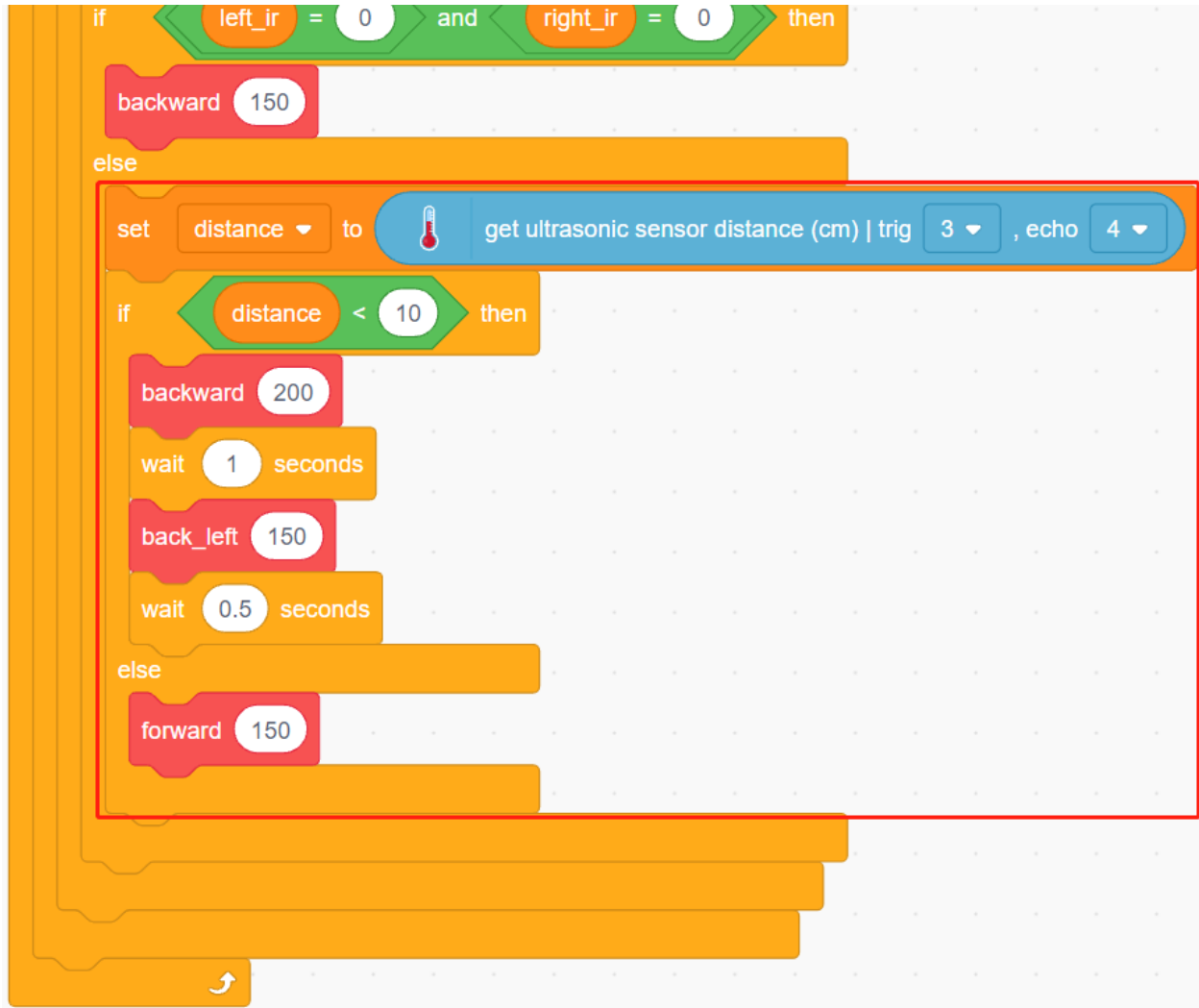
Les 2 modules d'évitement d'obstacles infrarouges sur la voiture sont utilisés pour un évitement d'obstacles d'urgence, détectant les obstacles à courte distance, dans les coins ou les obstacles relativement petits.

- Si le module infrarouge gauche détecte un obstacle, la voiture recule vers la gauche.
- Si le module IR droit détecte un obstacle, la voiture recule vers l'arrière droit.
- Si les 2 modules détectent l'obstacle en même temps, la voiture recule directement.



### 3. Évitement d'obstacles à longue portée

Lisez la valeur du module ultrasonique, lorsque la valeur détectée est inférieure à 10, la voiture recule ; sinon elle continue d'avancer.

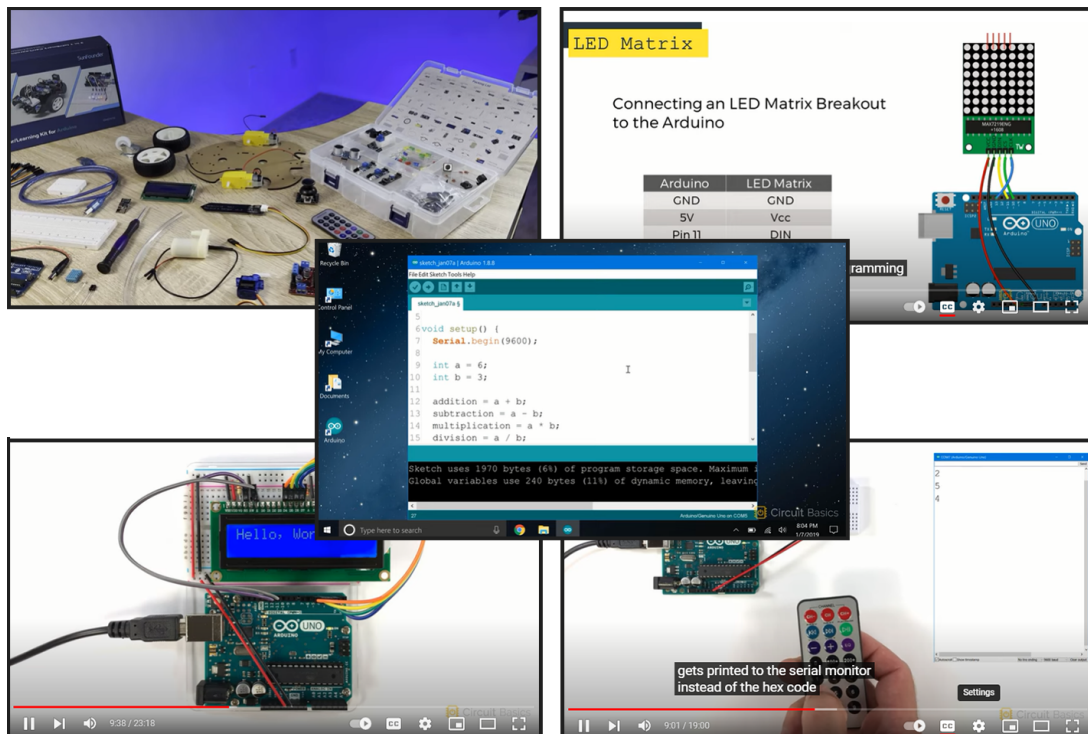


## CHAPITRE 9

### Cours Vidéo

#### — Pour les Sections *Projets de Base*

Si vous trouvez le contenu de la documentation en ligne difficile à comprendre, vous pouvez améliorer votre expérience d'apprentissage en suivant un cours vidéo progressif. Ces vidéos rendront Arduino plus engageant et visuellement attrayant pour vous.

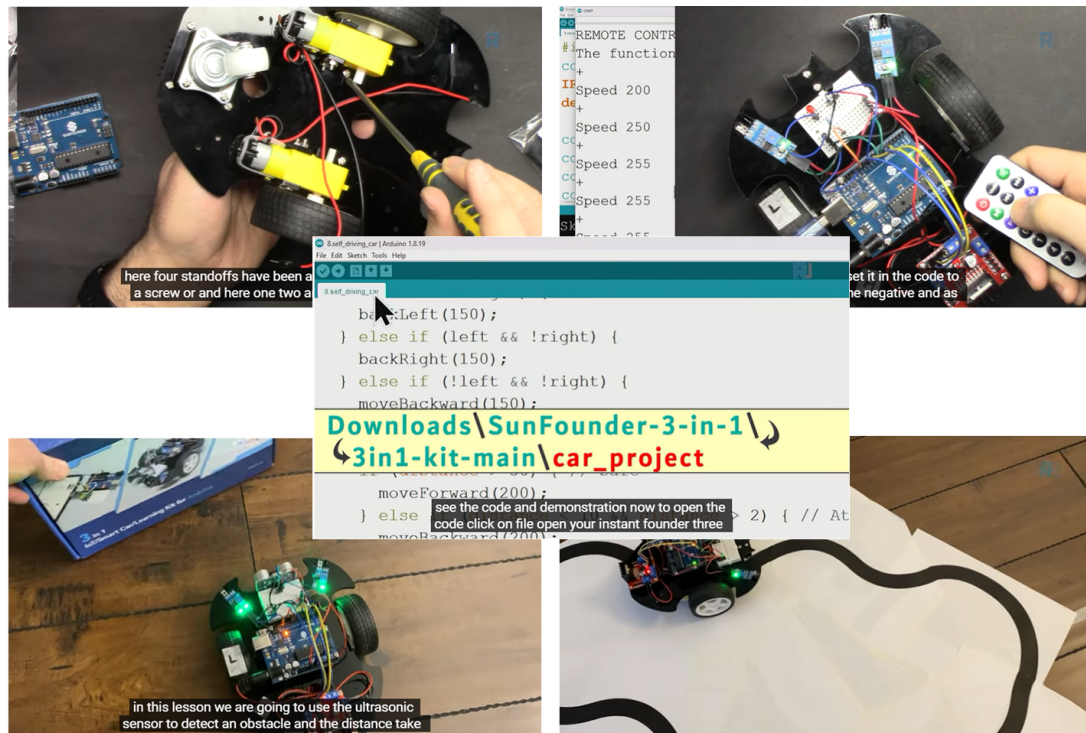


Vous pouvez accéder au cours vidéo à :

Dans ces vidéos, vous découvrirez des expériences fascinantes et des projets présentant les concepts et principes d'Arduino de manière amusante et interactive. En regardant les vidéos et en participant à des activités pratiques, vous vivrez une expérience d'apprentissage passionnante et agréable avec Arduino.

### — Pour les Sections *Projets de Voiture*

Je recommande de regarder les playlists YouTube suivantes pour obtenir plus de conseils et d'expérience pratique : .



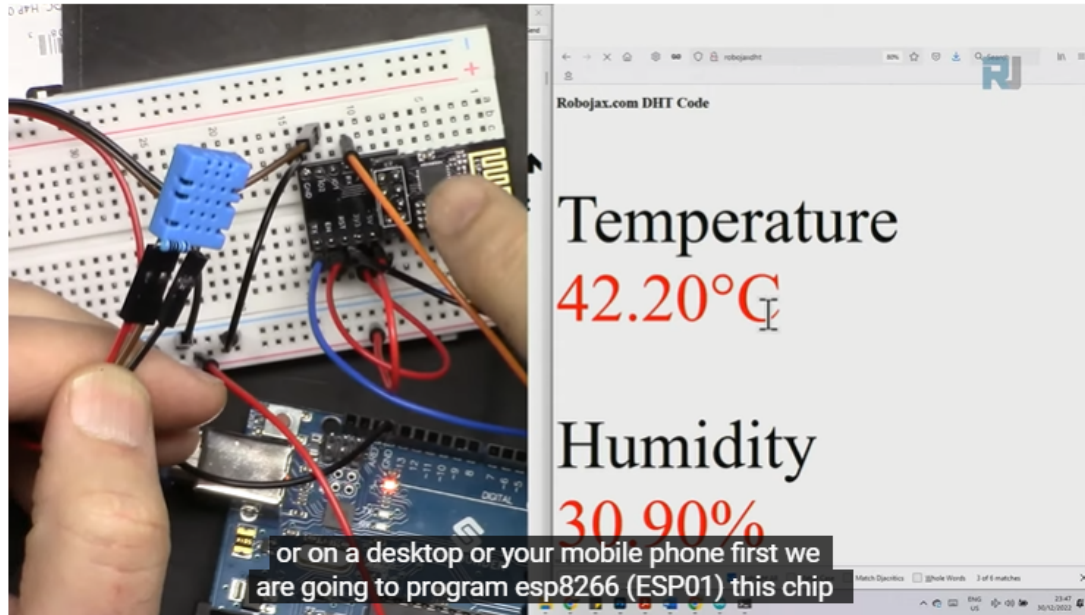
Dans ces vidéos, vous apprendrez les fondamentaux de la robotique et d'Arduino à travers des leçons vidéo captivantes. Étape par étape, vous assemblerez une voiture robot tout en comprenant le fonctionnement des moteurs, des modules d'évitement d'obstacles, des modules de suivi de ligne et des récepteurs infrarouges. Explorez comment la voiture réalise diverses fonctions et libérez votre créativité dans le monde de la robotique et de la technologie.

### — À propos de la Fonction WiFi

Dans la section *Projets IoT* de notre tutoriel en ligne, vous apprendrez à communiquer avec la plateforme IoT Blynk.

Dans , vous serez guidé sur l'écriture d'un serveur web et le téléversement de données de capteurs vers celui-ci. Ce tutoriel vous apprendra à établir une connexion entre votre projet Arduino et un serveur web en utilisant le WiFi.







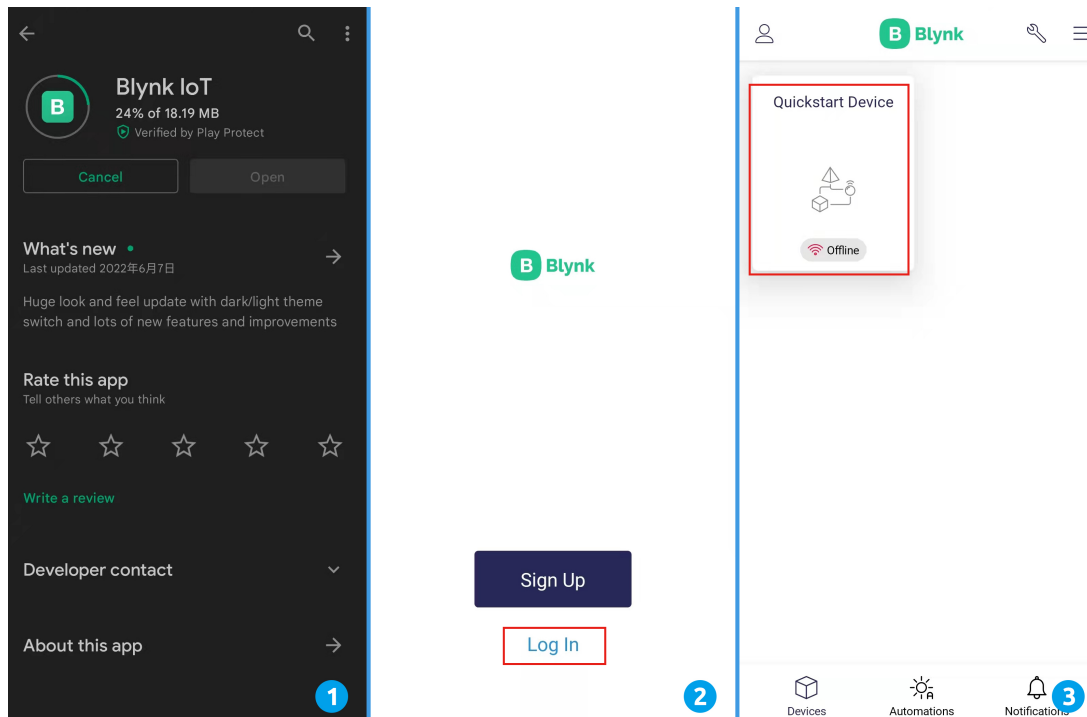
### 10.1 Comment utiliser Blynk sur un appareil mobile ?

---

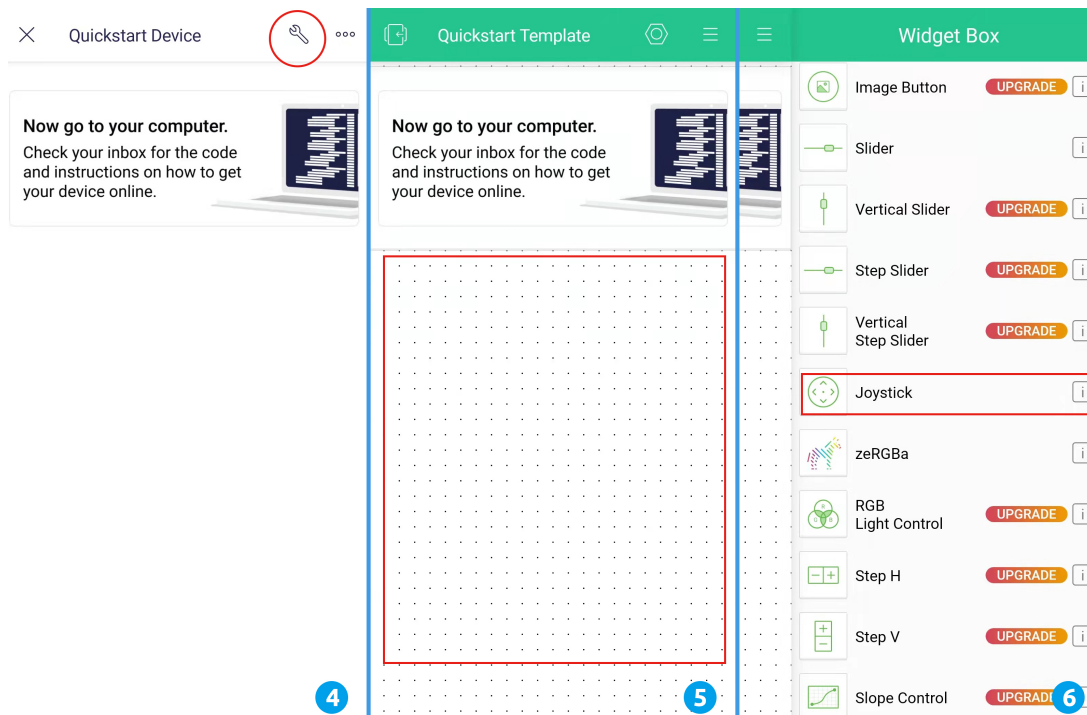
**Note :** Comme les flux de données ne peuvent être créés que dans Blynk sur le web, vous devrez vous référer à différents projets pour créer des flux de données sur le web, puis suivre le tutoriel ci-dessous pour créer des widgets dans Blynk sur votre appareil mobile.

---

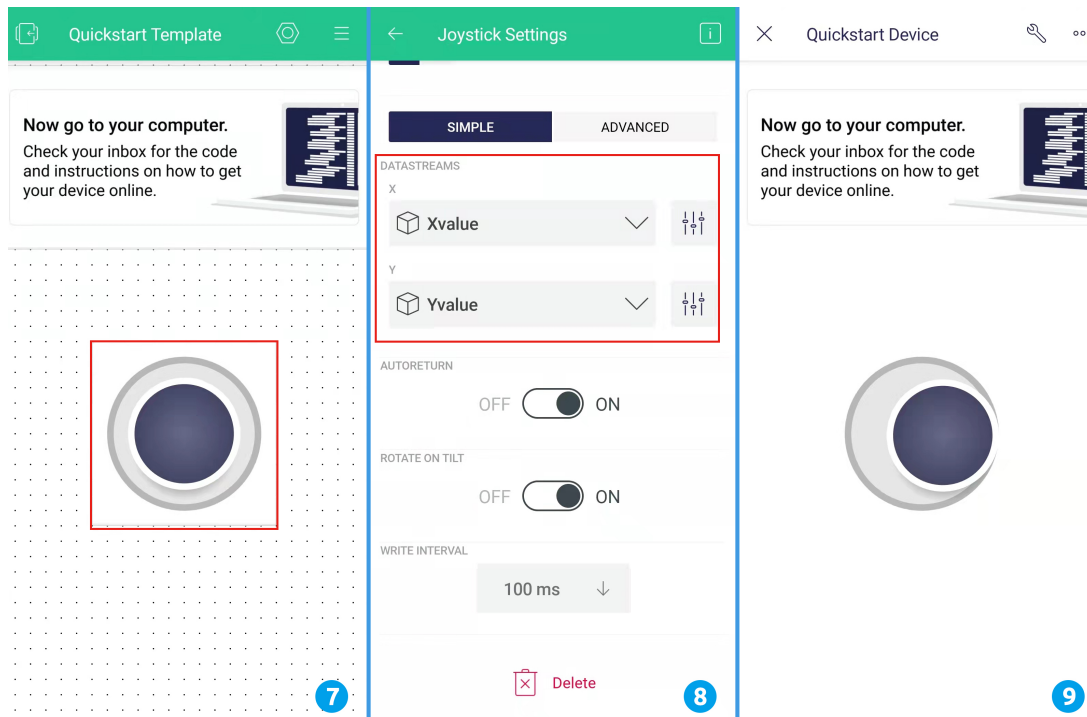
1. Ouvrez Google Play ou l'APP Store sur votre appareil mobile et recherchez « Blynk IoT » (pas Blynk(legacy)) pour télécharger.
2. Après avoir ouvert l'APP, connectez-vous, ce compte doit être le même que celui utilisé sur le client web.
3. Ensuite, allez dans **Dashboard** (si vous n'en avez pas, créez-en un) et vous verrez que le **Dashboard** pour mobile et web sont indépendants l'un de l'autre.



4. Cliquez sur l'icône **Edit**.
5. Cliquez sur la zone vide.
6. Choisissez le même widget que sur la page web, comme sélectionner un widget **Joystick**.



7. Maintenant, vous verrez apparaître un widget **Joystick** dans la zone vide, cliquez dessus.
8. Les paramètres du **Joystick** apparaîtront, sélectionnez les flux de données **Xvalue** et **Yvalue** que vous venez de définir sur la page web. Notez que chaque widget correspond à un flux de données différent dans chaque projet.
9. Revenez à la page du **Dashboard** et vous pourrez utiliser le **Joystick** lorsque vous le souhaitez.

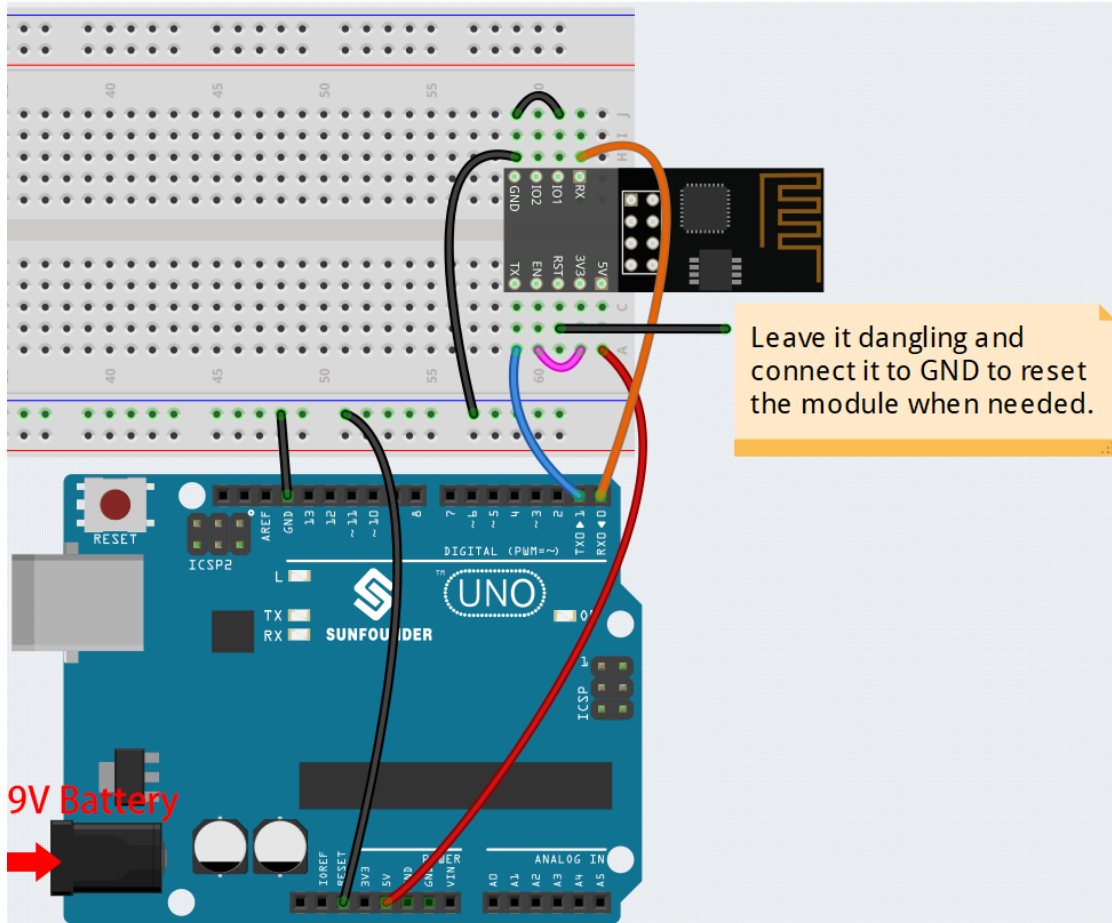


## 10.2 Comment re-graver le firmware pour le module ESP8266 ?

### 10.2.1 Re-graver le Firmware avec R3

#### 1. Construire le circuit

Connectez le module ESP8266 et la carte SunFounder R3.



## 2. Graver le firmware

— Suivez les étapes ci-dessous pour graver le firmware si vous utilisez **Windows**.

1. Téléchargez le firmware et l'outil de gravure.

— Firmware ESP8266

2. Après décompression, vous verrez 4 fichiers.

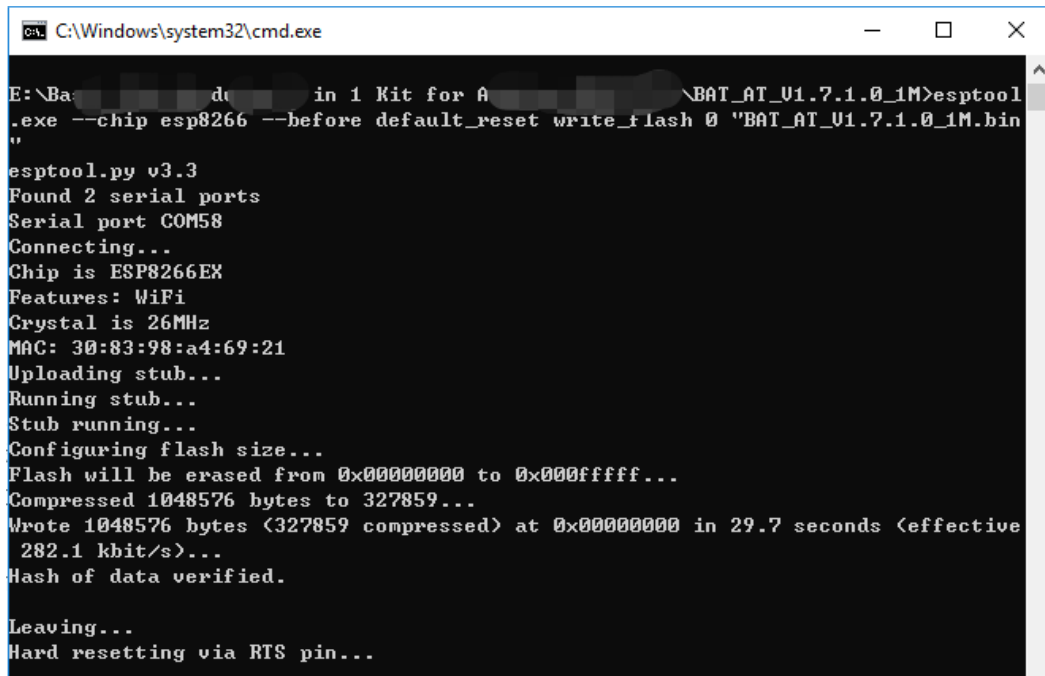
— BAT\_AT\_V1.7.1.0\_1M.bin : Le firmware à graver sur le module ESP8266.

— esptool.exe : C'est un utilitaire en ligne de commande pour Windows.

— install\_r3.bat : C'est le package de commandes pour le système Windows, double-cliquez sur ce fichier pour exécuter toutes les commandes à l'intérieur.

— install\_r4.bat : Identique à install\_r3.bat, mais dédié à la carte UNO R4.

3. Double-cliquez sur install\_r3.bat pour démarrer la gravure du firmware. Si vous voyez l'invite suivante, le firmware a été installé avec succès.



```

C:\Windows\system32\cmd.exe
E:\Ba... du... in 1 Kit for A... \BAT_AT_V1.7.1.0_1M>esptool
.exe --chip esp8266 --before default_reset write_flash 0 "BAT_AT_V1.7.1.0_1M.bin
"
esptool.py v3.3
Found 2 serial ports
Serial port COM58
Connecting...
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: 30:83:98:a4:69:21
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Flash will be erased from 0x00000000 to 0x000fffff...
Compressed 1048576 bytes to 327859...
Wrote 1048576 bytes (327859 compressed) at 0x00000000 in 29.7 seconds (effective
282.1 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...

```

**Note :** Si la gravure échoue, veuillez vérifier les points suivants.




- Réinitialisez le module ESP8266 en insérant le RST sur l'adaptateur ESP8266 à GND puis en le débranchant.
- Vérifiez si le câblage est correct.
- Si l'ordinateur a correctement reconnu votre carte, et assurez-vous que le port n'est pas occupé.
- Rouvrez le fichier install.bat.

— Pour graver le firmware, suivez ces étapes si vous utilisez un système **Mac OS**.

1. Utilisez les commandes suivantes pour installer Esptool. Esptool est un utilitaire basé sur Python, open source et indépendant de la plateforme pour communiquer avec le bootloader ROM dans les puces Espressif.

```
python3 -m pip install --upgrade pip
python3 -m pip install esptool
```

2. Si esptool est correctement installé, il affichera un message tel que [usage : esptool] si vous exécutez `python3 -m esptool`.
3. Téléchargez le firmware.
  - Firmware ESP8266
4. Après décompression, vous verrez 3 fichiers.

 BAT\_AT\_V1.7.1.0\_1M.bin  
 esptool.exe  
 install.bat

- BAT\_AT\_V1.7.1.0\_1M.bin : Le firmware à graver sur le module ESP8266.
  - esptool.exe : C'est un utilitaire en ligne de commande pour Windows.
  - install\_r3.bat : C'est le package de commandes pour le système Windows.
  - install\_r4.bat : Identique à install\_r3.bat, mais dédié à la carte UNO R4.
5. Ouvrez un terminal et utilisez la commande `cd` pour aller dans le dossier de firmware que vous venez de télécharger, puis exécutez la commande suivante pour effacer le firmware existant et re-graver le nouveau firmware.

```
python3 -m esptool --chip esp8266 --before default_reset erase_flash
python3 -m esptool --chip esp8266 --before default_reset write_flash 0
↪ "BAT_AT_V1.7.1.0_1M.bin"
```

6. Si vous voyez l'invite suivante, le firmware a été installé avec succès.

```
MacdeMacBook-Pro:BAT_AT_V1.7.1.0_1M mac$ python3 -m esptool --chip esp8266 --before default_reset w
rite_flash 0 "BAT_AT_V1.7.1.0_1M.bin"
esptool.py v4.3
[Found 3 serial ports
Serial port /dev/cu.usbmodem143401
Connecting....
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: 30:83:98:a4:69:21
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Flash will be erased from 0x00000000 to 0x000fffff...
Compressed 1048576 bytes to 327859...
Wrote 1048576 bytes (327859 compressed) at 0x00000000 in 29.6 seconds (effective 283.0 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
MacdeMacBook-Pro:BAT_AT_V1.7.1.0_1M mac$
```

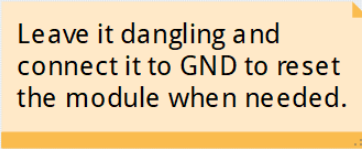
**Note :** Si la gravure échoue, veuillez vérifier les points suivants.

- Réinitialisez le module ESP8266 en insérant le RST sur l'adaptateur ESP8266 à GND puis en le débranchant.
- Vérifiez si le câblage est correct.
- Si l'ordinateur a correctement reconnu votre carte, et assurez-vous que le port n'est pas occupé.
- Rouvrez le fichier install.bat.

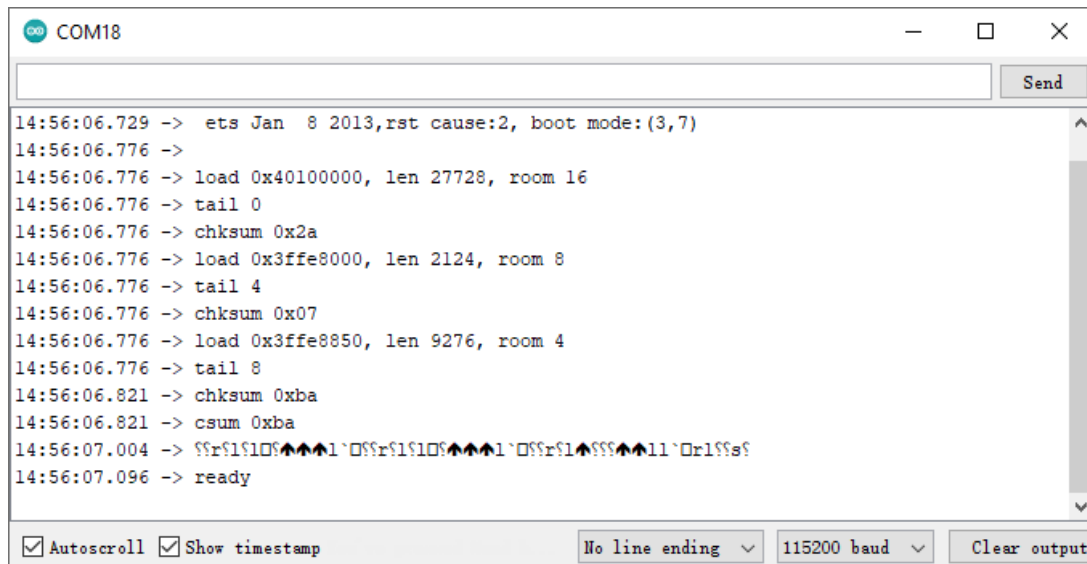
### 3. Test

1. Sur la base du câblage original, connectez IO1 à 3V3.





2. Vous pourrez voir des informations sur le module ESP8266 si vous cliquez sur l'icône de la loupe (Moniteur Série) dans le coin supérieur droit et réglez le débit en bauds sur **115200**.



**Note :**

- Si ready n'apparaît pas, vous pouvez essayer de réinitialiser le module ESP8266 (connectez RST à GND) et rouvrir le Moniteur Série.

3. Cliquez sur **NEWLINE DROPDOWN BOX**, sélectionnez both NL & CR dans l'option déroulante, entrez AT, si cela retourne OK, cela signifie que l'ESP8266 a établi avec succès une connexion avec la carte R3.

The screenshot shows a serial terminal window titled "COM18". The input field contains "AT" followed by a red handwritten number "2". The output area displays several AT commands and their responses:

```

14:56:06.776 -> load 0x40100000, len 27728, room 16
14:56:06.776 -> tail 0
14:56:06.776 -> checksum 0x2a
14:56:06.776 -> load 0x3ffe8000, len 2124, room 8
14:56:06.776 -> tail 4
14:56:06.776 -> checksum 0x07
14:56:06.776 -> load 0x3ffe8850, len 9276, room 4
14:56:06.776 -> tail 8
14:56:06.821 -> checksum 0xba
14:56:06.821 -> csum 0xba
14:56:07.004 -> $$$r$1$10$####1`0$$r$1$10$####1`0$$r$1$10$####1`0r1$$$s$
14:56:07.096 -> ready
15:00:05.119 -> AT
15:00:05.119 -> 
15:00:05.119 -> OK

```

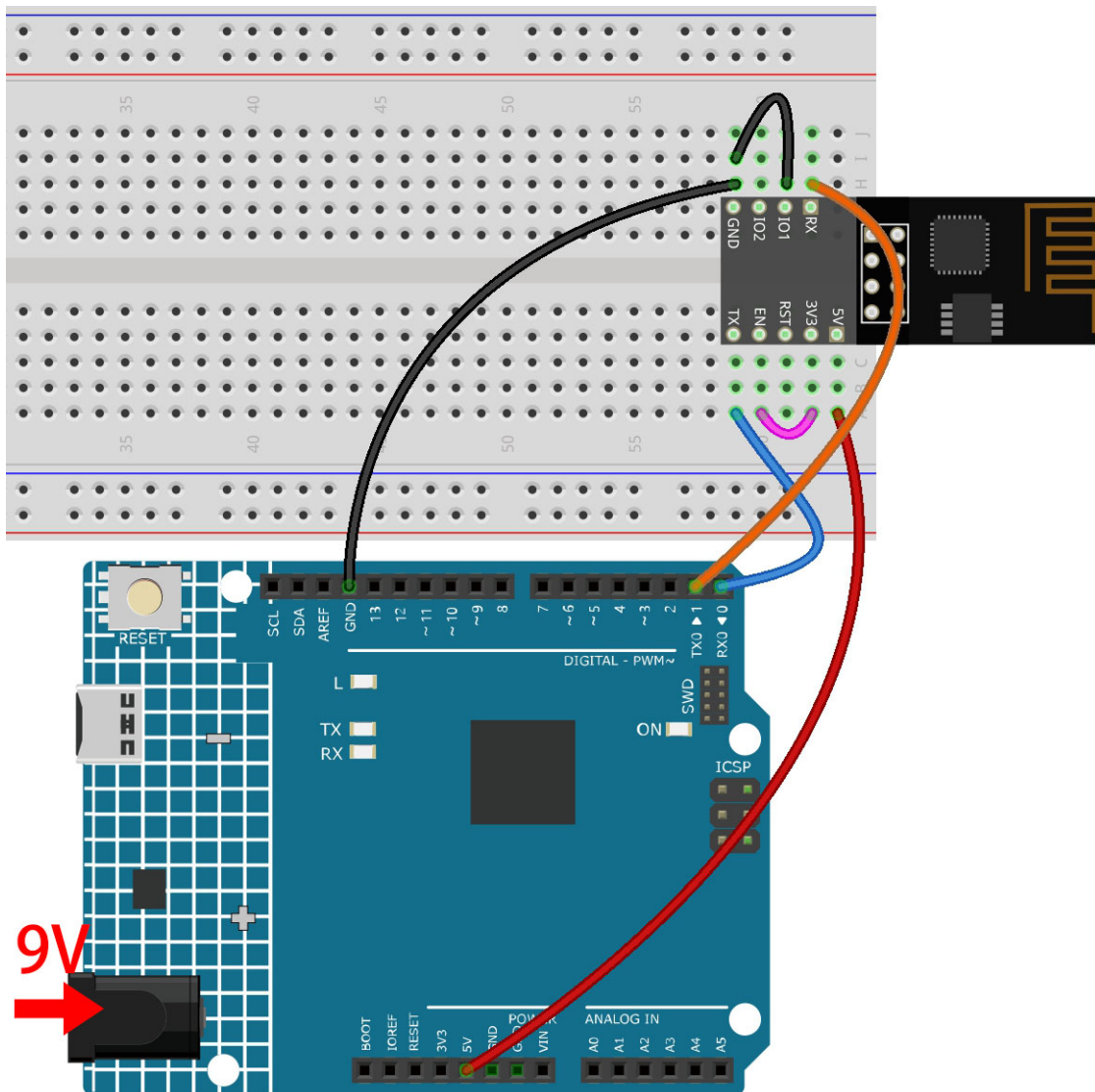
A red box highlights the last three lines of output, with a red handwritten number "4" next to it. At the bottom of the window, there are settings for "Both NL & CR" (indicated by a red handwritten number "1"), "115200 baud", and a "Clear output" button.

Maintenant, vous pouvez continuer à suivre *1.1 Configuration de l'ESP8266* pour régler le mode de travail et le débit en bauds du module ESP8266.

### 10.2.2 Re-graver le Firmware avec R4

## 1. Construire le circuit

Connectez ESP8266 et la carte Arduino UNO R4.



## 2. Téléversez le Code Suivant sur R4

```
void setup() {
  Serial.begin(115200);
  Serial1.begin(115200);
}

void loop() {
  if (Serial.available()) {      // If anything comes in Serial (USB),
    Serial1.write(Serial.read()); // read it and send it out Serial1 (pins 0 & 1)
  }

  if (Serial1.available()) {    // If anything comes in Serial1 (pins 0 & 1)
    Serial.write(Serial1.read()); // read it and send it out Serial (USB)
  }
}
```

## 3. Graver le firmware

- Suivez les étapes ci-dessous pour graver le firmware si vous utilisez **Windows**.
  1. Téléchargez le firmware et l'outil de gravure.

— Firmware ESP8266

2. Après décompression, vous verrez 4 fichiers.

- BAT\_AT\_V1.7.1.0\_1M.bin : Le firmware à graver sur le module ESP8266.
- esptool.exe : C'est un utilitaire en ligne de commande pour Windows.
- install\_r3.bat : C'est le package de commandes pour le système Windows, double-cliquez sur ce fichier pour exécuter toutes les commandes à l'intérieur.
- install\_r4.bat : Identique à install\_r3.bat, mais dédié à la carte UNO R4.

3. Double-cliquez sur install\_r4.bat pour commencer la gravure du firmware. Si vous voyez l'invite suivante, le firmware a été installé avec succès.

```

C:\Windows\system32\cmd.exe

E:\Ba... de ... in 1 Kit for A ... \BAT_AT_V1.7.1.0_1M>esptool
.exe --chip esp8266 --before default_reset write_flash 0 "BAT_AT_V1.7.1.0_1M.bin
"
esptool.py v3.3
Found 2 serial ports
Serial port COM58
Connecting...
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: 30:83:98:a4:69:21
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Flash will be erased from 0x00000000 to 0x000fffff...
Compressed 1048576 bytes to 327859...
Wrote 1048576 bytes (327859 compressed) at 0x00000000 in 29.7 seconds (effective
282.1 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
  
```

**Note :** Si la gravure échoue, veuillez vérifier les points suivants.

- Réinitialisez le module ESP8266 en insérant le RST sur l'adaptateur ESP8266 à GND puis en le débranchant.
- Vérifiez si le câblage est correct.
- Si l'ordinateur a correctement reconnu votre carte, et assurez-vous que le port n'est pas occupé.
- Rouvrez le fichier install.bat.

— Pour graver le firmware, suivez ces étapes si vous utilisez un système **Mac OS**.

1. Utilisez les commandes suivantes pour installer Esptool. Esptool est un utilitaire basé sur Python, open source et indépendant de la plateforme pour communiquer avec le bootloader ROM des puces Espressif.

```
python3 -m pip install --upgrade pip
python3 -m pip install esptool
```

2. Si esptool est correctement installé, il affichera un message tel que [usage : esptool] si vous exécutez `python3 -m esptool`.

3. Téléchargez le firmware.

— Firmware ESP8266

4. Après décompression, vous verrez 4 fichiers.

- BAT\_AT\_V1.7.1.0\_1M.bin : Le firmware à graver sur le module ESP8266.

- `esptool.exe` : C'est un utilitaire en ligne de commande pour Windows.
  - `install_r3.bat` : C'est le package de commandes pour le système Windows.
  - `install_r4.bat` : Identique à `install_r3.bat`, mais dédié à la carte UNO R4.
5. Ouvrez un terminal et utilisez la commande `cd` pour aller dans le dossier de firmware que vous venez de télécharger, puis exécutez la commande suivante pour effacer le firmware existant et re-graver le nouveau firmware.

```
python3 -m esptool --chip esp8266 --before no_reset_no_sync erase_flash
python3 -m esptool --chip esp8266 --before no_reset_no_sync write_flash 0
↪ "BAT_AT_V1.7.1.0_1M.bin"
```

6. Si vous voyez l'invite suivante, le firmware a été installé avec succès.

```
BAT_AT_V1.7.1.0_1M — -bash — 99x22
MacdeMacBook-Pro:BAT_AT_V1.7.1.0_1M mac$ python3 -m esptool --chip esp8266 --before default_reset w
rite_flash 0 "BAT_AT_V1.7.1.0_1M.bin"
esptool.py v4.3
[Found 3 serial ports
Serial port /dev/cu.usbmodem143401
Connecting...
Chip is ESP8266EX
Features: WiFi
Crystal is 26MHz
MAC: 30:83:98:a4:69:21
Uploading stub...
Running stub...
Stub running...
Configuring flash size...
Flash will be erased from 0x00000000 to 0x000fffff...
Compressed 1048576 bytes to 327859...
Wrote 1048576 bytes (327859 compressed) at 0x00000000 in 29.6 seconds (effective 283.0 kbit/s)...
Hash of data verified.

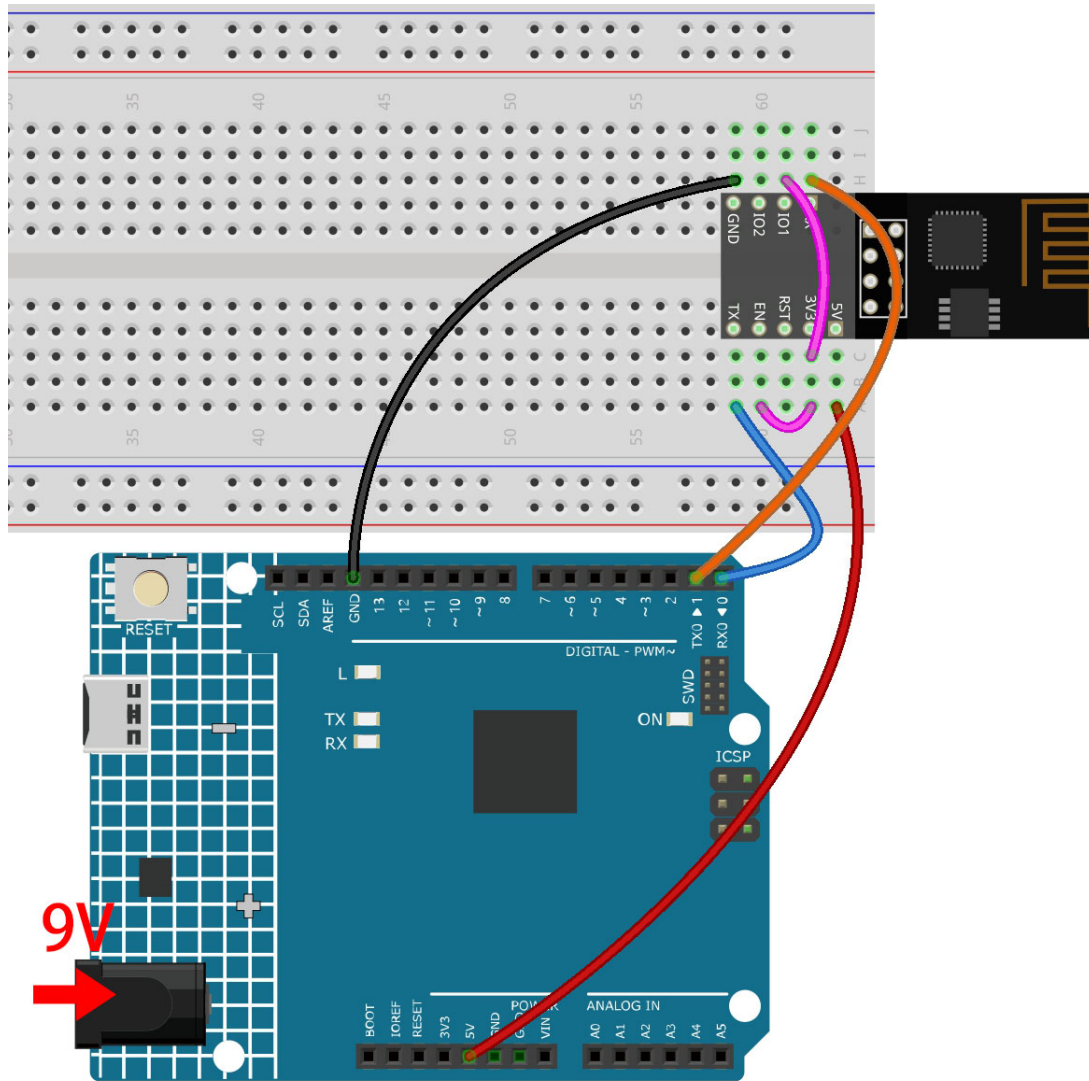
Leaving...
Hard resetting via RTS pin...
MacdeMacBook-Pro:BAT_AT_V1.7.1.0_1M mac$
```

**Note :** Si la gravure échoue, veuillez vérifier les points suivants.

- Réinitialisez le module ESP8266 en insérant le RST sur l'adaptateur ESP8266 à GND puis en le débranchant.
- Vérifiez si le câblage est correct.
- Si l'ordinateur a correctement reconnu votre carte, et assurez-vous que le port n'est pas occupé.
- Rouvrez le fichier `install.bat`.

#### 4. Test

1. Sur la base du câblage original, connectez IO1 à 3V3.



2. Vous pourrez voir des informations sur le module ESP8266 si vous cliquez sur l'icône de la loupe (Moniteur Série) dans le coin supérieur droit et réglez le débit en bauds sur **115200**.

```

COM18
14:56:06.729 -> ets Jan 8 2013, rst cause:2, boot mode:(3,7)
14:56:06.776 ->
14:56:06.776 -> load 0x40100000, len 27728, room 16
14:56:06.776 -> tail 0
14:56:06.776 -> checksum 0x2a
14:56:06.776 -> load 0x3ffe8000, len 2124, room 8
14:56:06.776 -> tail 4
14:56:06.776 -> checksum 0x07
14:56:06.776 -> load 0x3ffe8850, len 9276, room 4
14:56:06.776 -> tail 8
14:56:06.821 -> checksum 0xba
14:56:06.821 -> csum 0xba
14:56:07.004 -> $r$1$10$▲▲▲1`0$$r$1$10$▲▲▲1`0$$r$1$▲$$▲▲11`0r1$$s$
14:56:07.096 -> ready
  
```

☒ Autoscroll 
 ☒ Show timestamp 
 No line ending ▼ 
 115200 baud ▼ 
 Clear output

**Note :**

— Si ready n'apparaît pas, vous pouvez essayer de réinitialiser le module ESP8266 (connectez RST à GND) et rouvrir le Moniteur Série.

3. Cliquez sur **NEWLINE DROPDOWN BOX**, sélectionnez both NL & CR dans l'option déroulante, entrez AT, si cela retourne OK, cela signifie que l'ESP8266 a établi avec succès une connexion avec votre carte.

```

COM18
AT2
14:56:06.776 -> load 0x40100000, len 27728, room 16
14:56:06.776 -> tail 0
14:56:06.776 -> checksum 0x2a
14:56:06.776 -> load 0x3ffe8000, len 2124, room 8
14:56:06.776 -> tail 4
14:56:06.776 -> checksum 0x07
14:56:06.776 -> load 0x3ffe8850, len 9276, room 4
14:56:06.776 -> tail 8
14:56:06.821 -> checksum 0xba
14:56:06.821 -> csum 0xba
14:56:07.004 -> $r$1$10$▲▲▲1`0$$r$1$10$▲▲▲1`0$$r$1$▲$$▲▲11`0r1$$s$
14:56:07.096 -> ready
15:00:05.119 -> AT
15:00:05.119 ->
15:00:05.119 -> OK4
  
```

☒ Autoscroll 
 ☒ Show timestamp 
 1 Both NL & CR ▼ 
 115200 baud ▼ 
 Clear output

Maintenant, vous pouvez continuer à suivre [1.1 Configuration de l'ESP8266](#) pour régler le mode de travail et le débit en bauds du module ESP8266.





# CHAPITRE 11

---

## Remerciements

---

Merci aux évaluateurs qui ont évalué nos produits, aux vétérans qui ont fourni des suggestions pour le tutoriel, et aux utilisateurs qui nous suivent et nous soutiennent. Vos précieuses suggestions sont notre motivation pour fournir de meilleurs produits !

### **Remerciements Particuliers**

- Len Davisson
- Kalen Daniel
- Juan Delacosta

Pouvez-vous prendre un peu de temps pour remplir ce questionnaire ?

---

**Note :** Après avoir soumis le questionnaire, veuillez retourner en haut pour voir les résultats.

---



## CHAPITRE 12

---

### Avis de droit d'auteur

---

Tous les contenus, y compris mais sans s'y limiter, les textes, images et codes de ce manuel sont la propriété de la société SunFounder. Vous ne devez l'utiliser que pour l'étude personnelle, la recherche, le divertissement, ou d'autres fins non commerciales ou à but non lucratif, conformément aux réglementations et lois sur les droits d'auteur, sans porter atteinte aux droits légaux de l'auteur et des détenteurs de droits concernés. Pour toute personne ou organisation utilisant ces contenus à des fins de profit commercial sans autorisation, la société se réserve le droit d'engager des actions en justice.