

Club Robotique

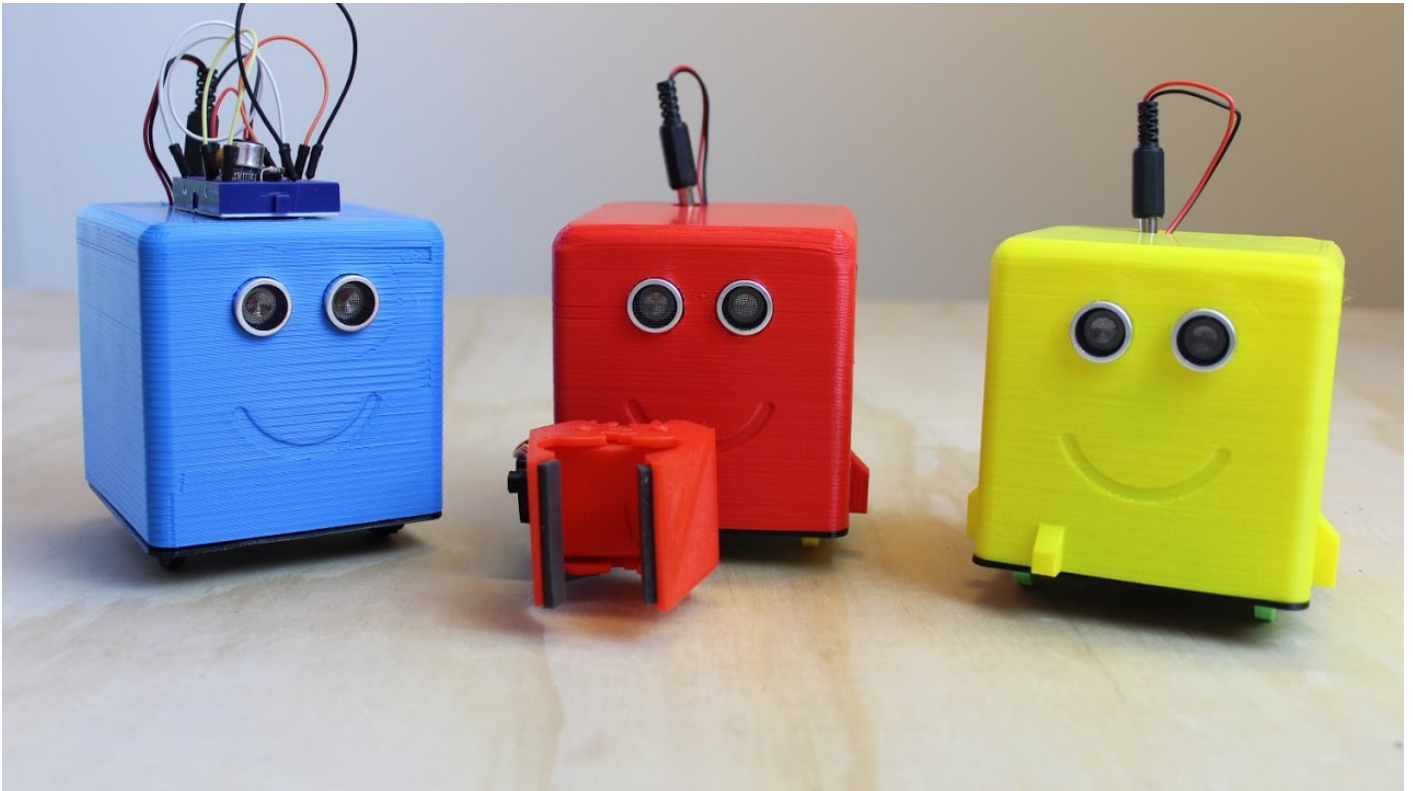
Cycle 1 - LittleBot

- Description du projet
- Phase 1 - Conception - 3 x 1H30
- Phase 2 - Câblage et programmation du LittleBot - 4 x 1h30
- Phase 3 - Assemblage et réctification - 2 x 1h30
- Fiche Séance 2 - 1H30
- Qu'est ce qu'un servo-moteur ?
- Piloter un servomoteur avec Mblock
- Fiche Séance 3 - 1H30
- Qu'est ce qu'un capteur à ultrasons ?
- Câblage et programmation du LittleBot
- Apprendre à utiliser TinkerCad (classe)
- Programme MBLOCK

Description du projet

Introduction

Pour le premier cycle, nous allons créer un robot mobile très simple appelé LittleBot.



Pour ce robot, nous utiliserons la programmation par bloc de type Scratch via l'environnement de développement MBlock ainsi que le logiciel de modélisation 3D TinkerCAD.

Durant ce cycle, nous apprendrons à utiliser une imprimante 3D mais aussi des composants pour de l'électronique.

Nous découvrirons les principes fonctionnement et la mise en œuvre d'un capteur ultrason ainsi que de servomoteurs.

Le programme, le câblage et l'assemblage sera donné pour la réalisation de ce projet.

L'objectif final de ce cycle est de faire avancer le robot, le faire tourner et de lui faire éviter des objets.

Description du cycle

- Compétences Animateur
- Prérequis participants
 - Age 10-15
 - Notions de base en électricité et circuit électronique
 - Notions de base en géométrie
 - Des bases en Langage de programmation par bloc "scratch" est un plus
- Logiciels :
 - Programmation : MBlock (scratch) et Arduino IDE (C)
 - Modélisation CAO : TinkerCAD, ONSHAPE, FreeCAD
 - Préparation impression 3D (slicer) :
- Compétences et technologies :
 - Programmer une Carte microcontrôleur Arduino depuis MBlock et Arduino IDE
 - Stocker dans une variable la valeur d'un Capteur de distance ultrason HC-SR04
 - Piloter un Servomoteur en donnant la valeur de sa vitesse de rotation
 - Télécharger le modèle 3D d'une pièce sur internet
 - Modifier le modèle 3D d'une pièce avec TinkerCAD
 - Préparer un fichier d'impression 3D avec ...
 - Lancement d'une impression 3D sur l'imprimante ...
 -

Description du déroulé

11 séances :

- 4 séances Conception électronique et câblage : Arduino et MBlock
 - Introduction Arduino et MBlock
 - Actionneur : servomoteur (déplacement)
 - Capteur : ultrason (distance)
 - Couplage capteur moteur
- 4 séances Conception mécanique et assemblage : TinkerCAD et impression 3D
 - 2 séances sur TinkerCAD
 - 2 séances sur l'impression 3D
- 3 séances d'assemblage et de programmation : Arduino IDE
 - Assemblage du LittleBot
 - Programmation de la cinématique du véhicule
 - Algorithmie pour stratégie d'évitement d'obstacle

Matériel

Projet inspiré de : <https://www.thingiverse.com/thing:2417739>

- 1 x Arduino Nano ou compatible ([seeeduino](#), funduino,...) : ~10€

- 1 x câble USB C : ~1€
- 1 x [Sensor Shield](#) pour Arduino Nano : ~3€
- 2 x servomoteur à rotation 360° ([DM-S0090D-R 9g/0.08s/1.6kg.cm](#)) : ~2€
- 1 x Module ultrason ([HC-SR04](#)) : ~2€
- 2 x élastique pour les roues (diamètre Xmm)

Une imprimante 3D pour imprimer les pièces suivantes (~5€ de filament) :

- Châssis :
- Capot :
- Roues avec rainure pour insertion élastique :

Phase 1 - Conception - 3 x 1H30

Conception 3D avec TinkerCAD

Prérequis participant :

- Travail sur ordinateur
- Manipulation avec la souris (clic droit, clic gauche, sélection,...)

Compétences Animateur :

Compétences techniques et soft skills :

- Pratique du logiciel TinkerCAD ou autre logiciel de CAO (Solidworks, Fusion360,...)
- Animation

Matériels nécessaires

logiciel, électronique, mécanique, outils, matériaux, code...

- 12 PCs (1 par participant)
 - Connexion et navigateur internet (Firefox)
 - Session invité ou nominative

Préparation :

Matériel par participant sur un poste PC en début de séance :

- PC allumé
- Codes d'accès session Windows/Linux
- Création d'une activité avec le tutoriel animateur TinkerCAD
- Lien d'invitation à l'activité
- Numérotation des PCs pour que les élèves sachent quel compte utiliser
- Temps de préparation : 5min

Documentation / Tutoriels :

- Tutoriel animateur TinkerCAD - création d'une activité
- Tutoriel élève TinkerCAD - modélisation d'une pièce

Déroulement de la séance

- Consignes : Sécurité, précautions matériel :
 - Travail individuel
- Phases et méthodes d'animation
 1. Présentation du club et du cycle 1 (10min)
 2. Tutoriel TinkerCAD (15min)
 3. Présentation de la phase de conception (5min)
 4. Présentation du cahier des charges pour la conception
 1. Contraintes sur les dimensions
 5. Charges

Conclusion / Rangement / Démontage :

- Rangement en fin de séance
 - Débrancher et ranger les composants
 - Chaque participant vérifie la boîte du voisin (check-list)
 - Remettre ordinateur dans l'état initial
- Programme de la prochaine séance

Phase 2 - Câblage et programmation du LittleBot - 4 x 1h30

Prérequis Participant

- Travail sur ordinateur
- Manipulation avec la souris (clic droit, clic gauche, sélection,...)
- Manipulation du clavier
- Notions d'électricité
- Notions de programmation par bloc (scratch)

Prérequis Animateur

Compétences techniques

- Pratique du logiciel Arduino IDE, connaissance en programmation C++
- Gestion des drivers sous Windows ou Expérience avec Ubuntu / Linux Mint
- Circuits et branchements électrique

Soft skills

- Animation
- Gestion de la Motivation et de l'attention

Matériels nécessaires

logiciel, électronique, mécanique, outils, matériaux, code...

- 12 PCs (1 par participant)
 - Windows ou Linux
 - Connexion et navigateur internet (Firefox)
 - Session invité ou nominative

- Logiciel Arduino IDE pré-installé
- Composants électroniques
 - Carte Arduino Nano (x12)
 - Shield (extension Arduino Nano) (x12)
 - Capteur Ultrasons HRC-SR04 (x12)
 - Servo-Moteur DM-S0090D (x24)
 - Câble Dupont (x48 - 4/participants)
- Programmes du LittleBot et des composants

Préparation

Matériel par participant sur un poste PC en début de séance :

- PC allumé
- Codes d'accès session Windows/Linux
- Création d'une activité avec le tutoriel animateur Arduino IDE
- **Vérifier que le téléversement de programme sur la carte fonctionne depuis Arduino IDE**
- Temps de préparation : 5min

Documentation / Tutoriels :

- Tutoriel animateur Arduino IDE- ~~création d'une activité~~
- Tutoriel élève Arduino IDE- ~~modélisation d'une pièce~~
- [Tutoriel Arduino Nano](#)
- [Tutoriel Capteur Ultrason](#)
- [Tutoriel Servo moteur](#)

Exemple de déroulement pédagogique

Exemple indicatif de déroulé par séance

Déroulé effectué en 2023-2024 au club robotique de l'IUT de Haguenau (1H30/séance, collégiens 6ème-3ème) :

- Séance 1 : Câblage et programmation d'une LED sur ? [Arduino IDE](#) ou [MBlock](#) ?
- Séance 2 : Principe physique, Câblage et programmation d'un capteur ultrason
- Séance 3 : Principe physique, câblage, et programmation d'un servomoteur
- Séance 4 : Câblage complet du LittleBot
- Séance 5 : Programmation du LittleBot

Exemple de déroulé pour la séance 1

- Consignes : Sécurité, précautions matériel :
 - Travail individuel
- Phases et méthodes d'animation
 1. Mise en contexte de la séance précédente(10min)
 2. Tutoriel Arduino IDE (15min)
 3. Présentation de la phase de programmation (5min)
 4. Présentation du cahier des charges pour la programmation
 1. Contraintes sur les alimentations (5v ou 3.3v)
 2. Quelles pins sont à utiliser pour les capteurs ultrasons ?
 3. Comment brancher un servo-moteur ?
 4. Contrainte du sens de rotation des servo-moteurs
 5. Rendre le changement de sens du robot aléatoire

Déroulement pédagogique complet

Pour la découverte du câblage et de la programmation, nous allons découvrir petit-à-petit les différents composants électroniques, logiciels et programmes informatiques (code) nécessaire au fonctionnement du LittleBot. Commençons pas la découverte de la carte microcontrôleur "Arduino Nano" et sa carte de développement "Sensor Shield v3".

Tutoriel : Qu'est-ce que l'"Arduino"

Maintenant que nous sommes experts en microcontrôleur, voyons comment l'utiliser pour de la robotique.

Tutoriel : Qu'est-ce qu'un robot ?

Un robot est composé de capteurs et d'actionneurs. Nous allons d'abord découvrir et apprendre à utiliser un capteur de distance :

Tutoriel : Qu'est-ce qu'un capteur ultrason ?

Ce capteur peut être utilisé pour modifier le mouvement du robot qui est actionné par des servomoteurs :

Tutoriel : Qu'est-ce qu'un servomoteur ?

Maintenant que nous savons capter notre environnement et actionner des moteurs, nous allons pouvoir programmer le mouvement des moteurs en fonction de ce que le capteur reçoit :

Tutoriel : Câblage et programmation du LittleBot

Conclusion / Rangement / Démontage :

- Rangement en fin de séance
 - Débrancher et ranger les composants
 - Chaque participant vérifie la boîte du voisin (check-list)
 - Remettre ordinateur dans l'état initial
- Programme de la prochaine séance

Phase 3 - Assemblage et r  ctification - 2 x 1h30

Pr  requis participant :

- Travail manuel et sur ordinateur
- Manipulation avec la souris (clic droit, clic gauche, s  lection,...)
- Manipulation du clavier

Comp  tences Animateur :

Comp  tences techniques et soft skills :

- Pratique du logiciel Arduino IDE, connaissance en programmation C++
- Animation
- Comp  tences manuelles

Mat  riels n  cessaires

logiciel,   lectronique, m  canique, outils, mat  riaux, code...

- 12 PCs (1 par participant)
 - Connexion et navigateur internet (Firefox)
 - Session invit   ou nominative
 - Logiciel Arduino IDE pr  -install  
- Composants   lectroniques
 - Carte Arduino Nano (x12)
 - Shield (extension Arduino Nano) (x12)
 - Capteur Ultrasons HRC-SR04 (x12)
 - Servo-Moteur DM-S0090D (x24)
 - Câble Dupont (x48 - 4/participants)
- Programme du LittleBot
- Ch  ssis et capot imprim  s en 3D (x12)

Pr  paration :

Mat  riel par participant sur un poste PC en d  but de s  ance :

- PC allum  

- Codes d'accès session Windows/Linux
- Création d'une activité avec le tutoriel animateur Arduino IDE
- Matériels électroniques et corps du LittleBot désassemblé
- Temps de préparation : 5min

Documentation / Tutoriels :

- [Tutoriel Arduino Nano](#)
- [Tutoriel Capteur Ultrason](#)
- [Tutoriel Servo moteur](#)
- [Tutoriel Câblage et Programmation](#)

Déroulement de la séance

- Consignes : Sécurité, précautions matériel :
 - Travail individuel

Lors de notre première séance, nous allons câbler et assembler le robot. Tout d'abord, nous allons câbler notre capteur ultrasons à notre shield. Par la suite, nous allons assembler le robot dans son châssis avec le tutoriel d'assemblage.

Puis nous allons brancher nos servo-moteurs à notre Shield en passant par les emplacements prévu sur notre châssis.

Pour se faire, nous allons utiliser le tutoriel de [Câblage et programmation](#) ainsi que le tutoriel d'assemblage du LittleBot. **TUTO EN COURS DE RÉDACTION**

Puis nous allons programmer notre robot pour vérifier son fonctionnement via le tutoriel de [Câblage et programmation](#).

Une fois ceci fait nous assemblons complètement notre robot. Il ne manque plus qu'à le brancher et le faire tourner !

Lors de notre seconde séance nous allons rectifier / améliorer notre robot.

Avec le code que nous avons donné à notre carte, le robot ne tourne que dans un sens lors de la rencontre d'un obstacle.

Nous allons donc améliorer le programme pour rendre la direction aléatoire. Pour se faire nous allons utiliser le tutoriel Amélioration du LittleBot.

De plus notre LittleBot étant branché par USB à notre ordinateur, il ne pourrait pas aller bien loin. Nous lui installerons alors une batterie qui se représente ici par une pile. Voici comment la brancher. **TUTO EN COURS DE RÉDACTION.**

A la fin de ces 2 séances, notre LittleBot est totalement autonome et peu se déplacer pendant des heures. Félicitation !

Fiche Séance 2 - 1H30

Introduction aux servo-moteurs

Fiche Animateur

Prérequis participant :

- Travail sur ordinateur
-

Compétences Animateur :

Compétences techniques et soft skills :

- Expérience avec les composants de base Arduino
- Bases de la Programmation Arduino IDE
- Animation

Matériels nécessaires

logiciel, électronique, mécanique, outils, matériaux, code...

- 12 PCs (1 par participant)
 - MBlock et Arduino IDE installés
 - configuration port série et modèle carte
- arduino Nano + Sensor shield + servomoteur

Préparation :

Matériel par participant sur un poste PC en début de séance :

- Rassembler le matériel électronique
- PC allumé avec MBlock et arduino IDE ouvert
- Temps de préparation : 1H

Documentation :

- Tutoriel introduction "Qu'est ce qu'un servo-moteur ?"
- Tutoriel branchement
- Check-list composants présents dans la boîte. Pour vérification en début et en fin de séance.

Fiche participant

Objectifs

Compétences techniques :

- Prendre en main servo et Logiciel
- Faire tourner un servo moteur

Savoir-être, compétences transversales :

- Travail individuel
- Lire un tutoriel détaillé

Déroulement de la séance

- Consignes : Sécurité, précautions matériel :
 - Travail individuel
 - Ne pas plier les pattes des composants plus que nécessaire : risque de casse
 - Tout est fragile
- Phases et méthodes d'animation
 1. Présentation de l'activité du jour (5min)
 2. Présentation Différent servo (15min)
 3. Présentation Servo moteur - PWM - fonctionnement méca (15min)
 4. Branchement servo à la carte (5min)
 5. Programmer et faire tourner le servo (30min)

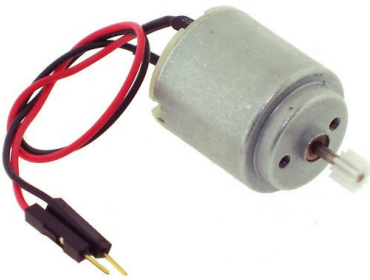
Conclusion / Rangement / Démontage :

- Rangement en fin de séance
 - Débrancher et ranger les composants
 - Chaque participant vérifie la boîte du voisin (check-list)
 - Remettre ordinateur dans l'état initial
- Programme de la prochaine séance

Qu'est ce qu'un servo-moteur ?

I Introduction

Lorsque vous créez un projet qui intègre des éléments robotique vous arrivez forcément à devoir faire un choix de moteur pour automatiser votre création. Plusieurs solutions existent, comme par exemple des moteurs à courant continu:



Ceux-ci ont l'avantage d'être très simple d'utilisation. Il suffit de connecter leurs deux fil d'alimentation à une batterie et ils se mettent à tourner.

Une autre solution serait d'utiliser des moteurs pas à pas :



Ces moteurs sont très pratique pour déplacer quelque-chose à un endroit précis, du fait que vous pouvez précisément le positionner à un angle donné. Par contre leur utilisation n'est pas aussi triviale que pour un moteur à courant continu.

Ce que je vous propose dans ce tutoriel c'est de découvrir les Servomoteurs. il en existe deux type :

- A contrôle d'angle
- A contrôle de vitesse de rotation

Les premiers ne tournent pas en continu, mais en général entre 0° et 180° et vous pouvez contrôler leur angle de rotation. Très pratique quand vous voulez faire un bras robotisé, piloter l'orientation des roues avant d'une voiture, ou bien contrôler un petit mécanisme. Les Servomoteurs suivants sont très pratiques lorsque vous souhaitez faire tourner un objet en contrôlant sa vitesse, par exemple des roues! :

tiptopboards.com/169-thickbox_mini-servo-moteur-sg90-9g-mod_c3_a9lisme-arduino.jpg

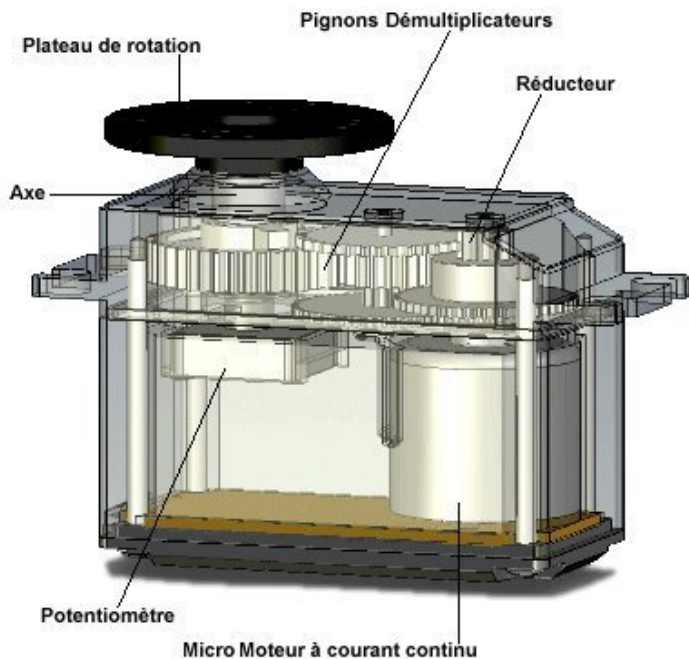
Ce sont des Servomoteurs à contrôle d'angle sur 180° et le tutoriel va donc porter sur ce type de moteur.

II Fonctionnement

<https://fr.wikipedia.org/wiki/Servomoteur> --> fonctionnement

Les Servomoteurs intègrent au sein d'un même boîtier un moteur à courant continu, un potentiomètre, un réducteur et un circuit de contrôle. L'idée est que la valeur d'angle est mesurée grâce au potentiomètre et le circuit de contrôle fait tourner le moteur et corrige l'orientation. Voilà une image qui donne une idée du fonctionnement interne:

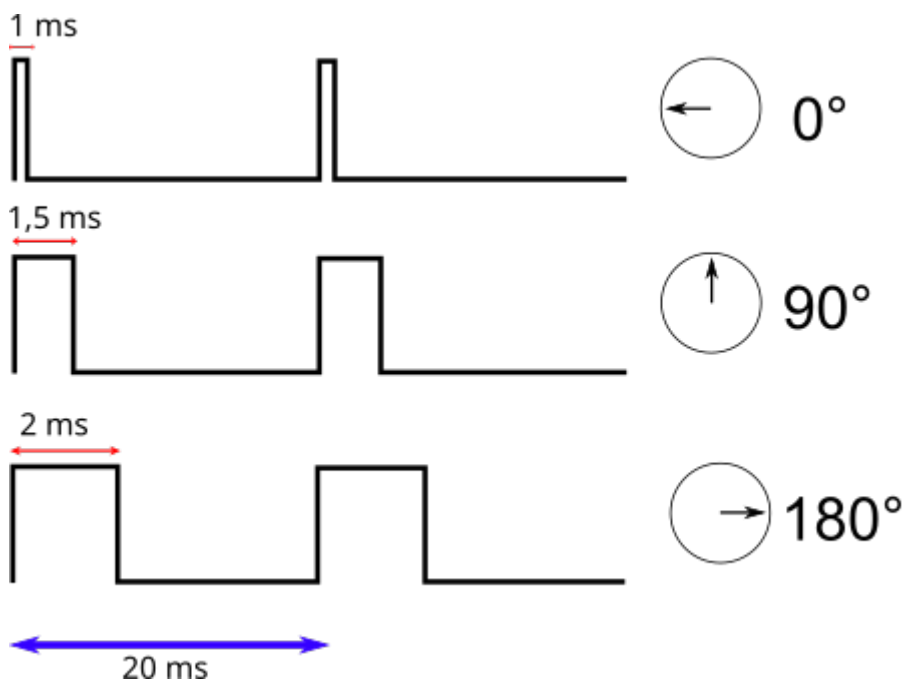
upload.wikimedia.org/wikipedia/commons/d/d3/micro_servo.jpg



Eric G

Vous l'avez sûrement remarqué, le Servomoteur a trois fils. Le fil marron correspond à la masse, le fil rouge au 5 Volts et le fil orange à l'envoi de données. C'est par le fil orange que nous allons envoyer le signal pour la commande de l'angle voulu au Servomoteur.

L'instruction par le fil orange s'envoie sous la forme d'un signal PWM (Pulse Width modulation=Modulation en largeur d'impulsion). Le principe est que l'envoi d'instruction se fait par un signal électrique qui passe de façon régulière et rapide (30-50Hz ou 300Hz) de 0 à 5 Volts. La valeur de l'angle voulu est définie par le rapport entre le temps où le signal est à 5 Volts et le temps où celui-ci est à 0 Volt. Par exemple pour un angle de 0° , on envoie 5V pendant 1ms puis 0V pendant 19ms : Le signal est à 5V pendant 5% du temps (1ms/20ms). Une image vous donnera une meilleure idée:

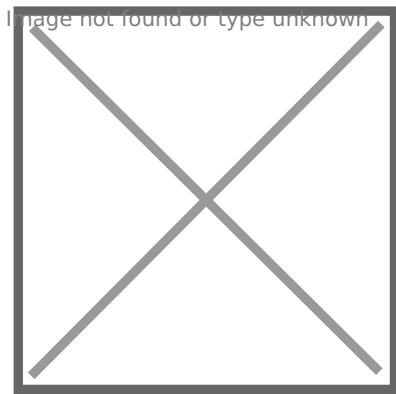


Source : <https://upload.wikimedia.org/wikipedia/commons/thumb/f/f6/TiemposServo.svg/220px-TiemposServo.svg.png>

III Mise en pratique

Pour consolider votre compréhension nous allons mettre en pratique avec un Arduino. L'Arduino permet de générer des signaux PWM avec la fonction `AnalogWrite()` mais nous ne pouvons pas l'utiliser avec un Servomoteur, car sa fréquence est trop élevée (environ 500Hz). Pour ce faire nous allons plutôt utiliser la librairie `servo.h` qui est disponible de base avec le programme Arduino.

Connectez d'abord un Servomoteur à l'Arduino comme sur l'image:



et on utilisera pour l'exemple le code suivant

```
#include <Servo.h>

Servo myservo;  // On crée un objet MyServo par lequel on envoie les instructions au servo

void setup()
{
  pinMode(6, OUTPUT);
  myservo.attach(6);  // On associe notre objet au Pin connecté au fil de données du servo
}

void loop()
{
  myservo.write(0);    //donne l'ordre de mettre le Servo à son angle minimum
  delay(1000);
  myservo.write(255);  //donne l'ordre de mettre le Servo à son angle maximum
  delay(1000);
}
```

Si tout se passe bien, vous devriez voir le Servomoteur tourner de 180° toutes les secondes. Après ça, je vous conseille de vous amuser en modifiant les valeurs et/ou le code afin de vous familiariser avec son utilisation.

Vous savez maintenant le minimum nécessaire pour correctement utiliser un Servomoteur.

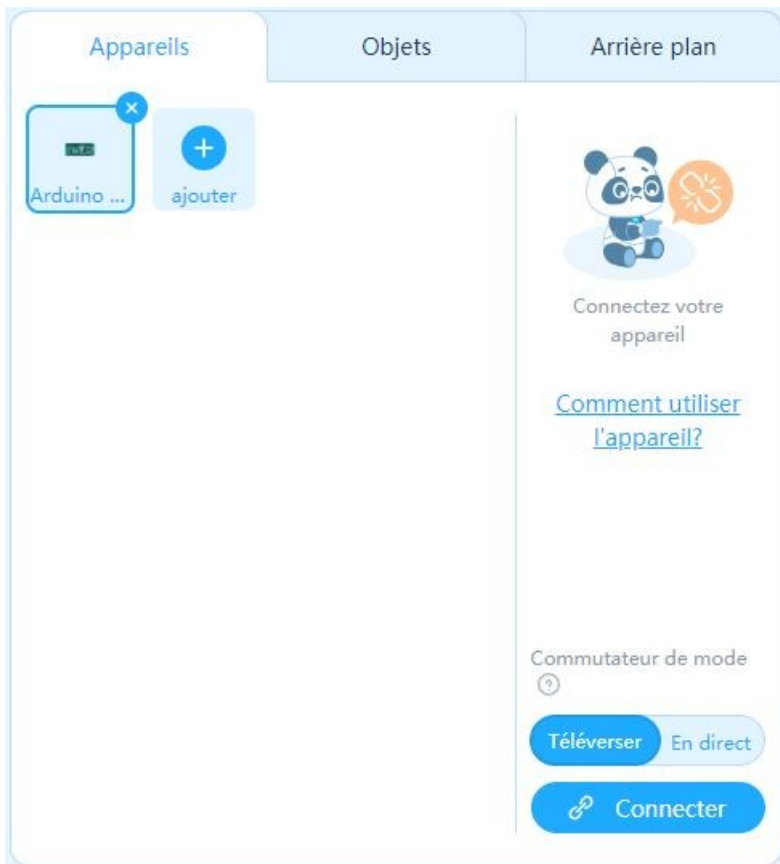
Piloter un servomoteur avec Mblock

I/ Introduction
















Pour ce faire nous allons utiliser Mblock.

Après avoir lancé le logiciel, nous allons ajouter la carte que nous utilisons :

Cliquez sur "Ajouter".



Puis sur la carte qui nous intéresse, ici c'est l'Arduino Nano.

 <p>microbit Développeurs: mBlock</p>	 <p>Arduino Uno Développeurs: Ablock</p>	 <p>Arduino Mega2560 Développeurs: Ablock</p>	 <p>Arduino Micro Développeurs: Ablock</p>	 <p>Arduino Yun Développeurs: Ablock</p>
 <p>Arduino Leonardo Développeurs: Ablock</p>	 <p>OrangeBoard Développeurs: woojh</p>	 <p>Arduino Nano Développeurs: Ablock</p>	 <p>cerebro Développeurs: epsilon11101</p>	 <p>Robot BrickOne Développeurs: duongthelonqnv</p>
				

Devenez un développeur de mBlock pour débloquer plus de potentiel.

Annuler

OK

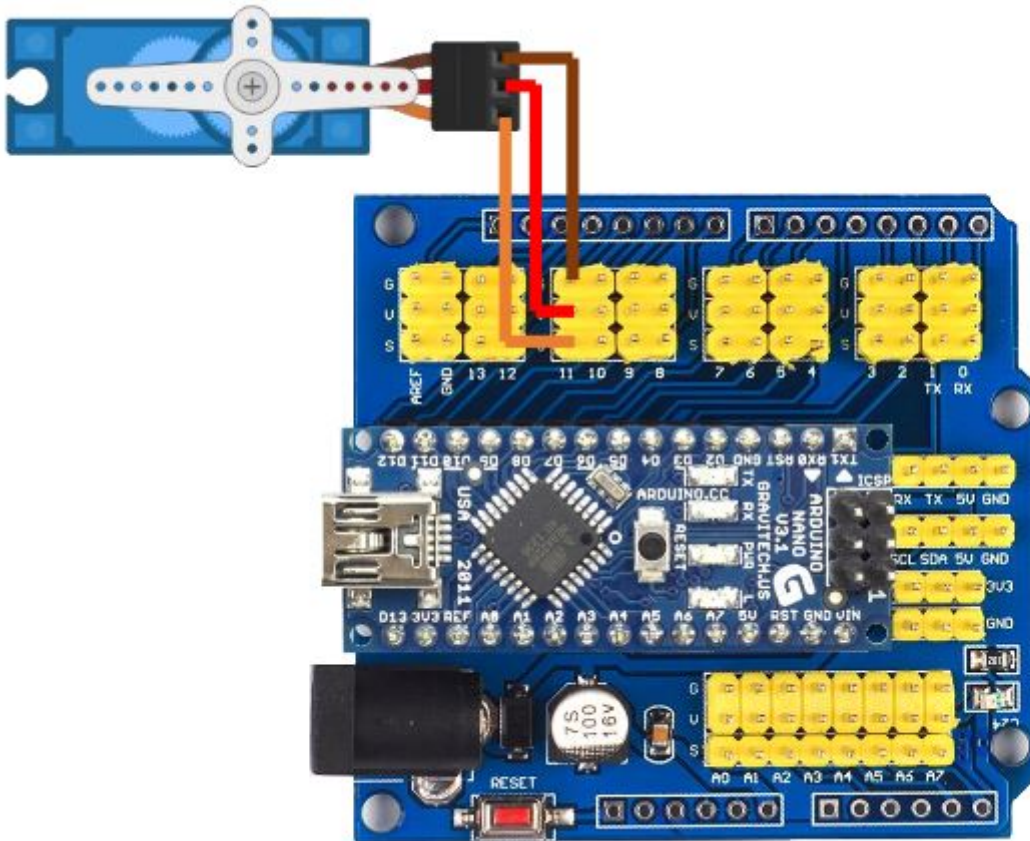


Puis cliquez sur connecter, cette fenêtre s'ouvrira.

Cochez "Afficher tous les appareils disponibles", puis sélectionnez le port COM de votre carte.

Votre carte Arduino est maintenant connectée au logiciel.

II/ Branchement du servo



III/ Code

lorsque Arduino démarre

pour toujours

∞ régler la sortie du port PWM 11 sur 180

attendre 1 secs

∞ régler la sortie du port PWM 11 sur 0

attendre 1 secs

Fiche Séance 3 - 1H30

Introduction au capteur à ultrasons

Fiche animateur

Prérequis participant :

- Travail sur ordinateur
-

Compétences animateur :

Compétences techniques et soft skills :

- Expérience avec les composants de base Arduino
- Bases de la programmation Arduino IDE
- Animation

Matériels nécessaires

logiciel, électronique, mécanique, outils, matériaux, code...

- 12 PCs (1 par participant)
 - MBlock et Arduino IDE installés
 - configuration port série et modèle carte
- Capteur à ultrasons (HC-SR04) + carte Sensor shield + arduino Nano + X câbles dupont

Préparation :

Matériel par participant sur un poste PC en début de séance :

- Rassembler le matériel électronique
- PC allumé avec MBlock et Arduino IDE
- Temps de préparation : 1H

Documentation :

- Tutoriel introduction "[Qu'est ce qu'un capteur à ultrasons ?](#)"
- Check-list composants présents dans la boîte. Pour vérification en début et en fin de séance.

Fiche participant

Objectifs

Compétences techniques :

- Introduction générale, objectifs du club et du cycle 1
- Prendre en main capteur ultrasons
- Faire tourner un servo selon une distance

Savoir-être, compétences transversales :

- Travail individuel
- Lire un tutoriel détaillé

Déroulement de la séance

- Consignes : Sécurité, précautions matériel :
 - Travail individuel
 - Ne pas plier les pattes des composants plus que nécessaire : risque de casse
 - Tout est fragile
- Phases et méthodes d'animation
 1. Présentation de l'activité du jour
 2. Présentation Capteur ultrason
 3. Branchement servo +
 4. [capteur ultrasons](#)
 5. Programmation Mblock et Arduino

Conclusion / Rangement / Démontage :

- Rangement en fin de séance
 - Débrancher et ranger les composants
 - Chaque participant vérifie la boîte du voisin (check-list)
 - Remettre ordinateur dans l'état initial
- Programme de la prochaine séance

Qu'est ce qu'un capteur à ultrasons ?



HC-SR04 est un capteur à ultrasons qui est

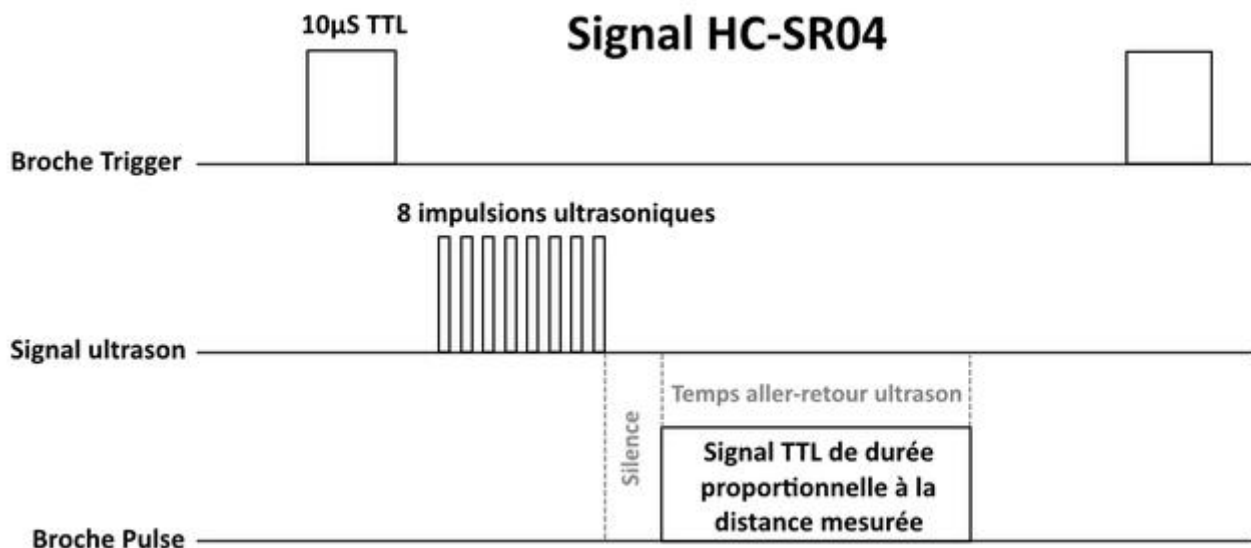
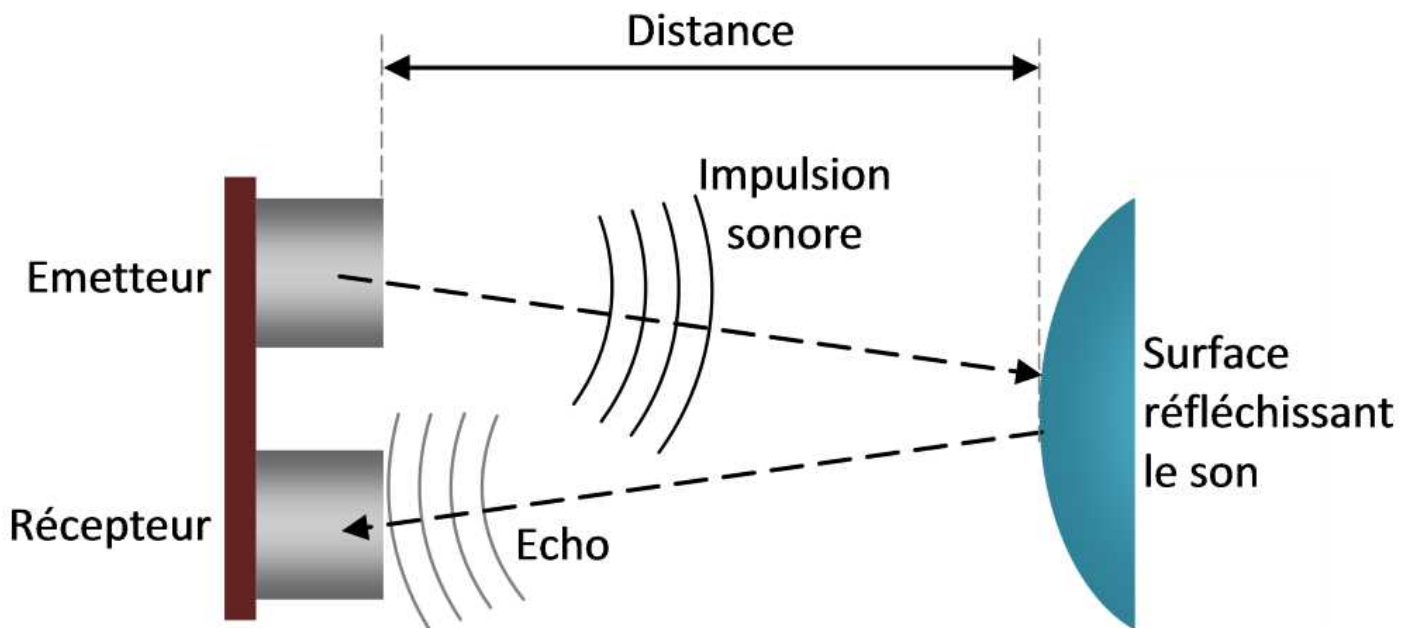
principalement utilisé pour la mesure de distance. En émettant des ondes ultrasonores et en mesurant le temps pris par ces ondes pour rebondir après avoir frappé un objet, le HC-SR04 peut déterminer avec précision la distance à laquelle cet objet se trouve. De par son coût abordable et sa facilité d'intégration avec des plateformes telles qu'Arduino, le HC-SR04 est devenu un choix prisé parmi les amateurs de bricolage et les professionnels.

Principe de fonctionnement

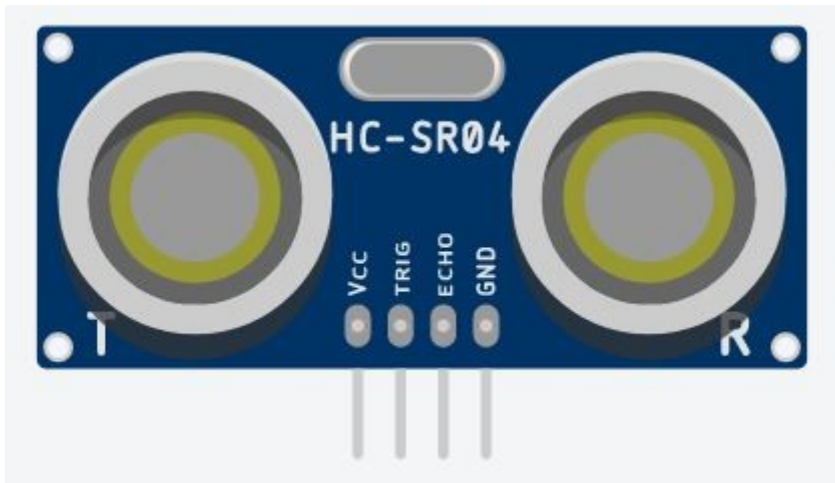
Le fonctionnement d'un capteur à ultrason comme le HC-SR04 est assez simple. Il comporte deux éléments principaux : un émetteur ultrasonore et un récepteur ultrasonore. Voici les étapes clés du fonctionnement du capteur :

1. Lorsque le capteur est alimenté, l'émetteur envoie une série de 8 impulsions ultrasoniques de $10\mu s$ à une fréquence spécifique (généralement de 40 kHz).
2. Lorsque une impulsion sonore atteint un objet, elle rebondit et est renvoyée vers le récepteur ultrasonore comme un écho.

3. Le capteur mesure le temps entre le moment où l'impulsion a été émise et celui où l'écho a été reçu.
4. En utilisant la vitesse connue du son dans l'air (environ 343 m/s ou 34,3 cm/ μ s) et la durée de l'écho mesurée, le capteur calcule la distance jusqu'à l'objet en utilisant la formule : $\text{distance} = (\text{durée de l'écho} / 2) * \text{vitesse du son}$.
5. Le résultat est ensuite converti en une distance numérique et envoyé au Arduino via une sortie numérique



Description du capteur HC-SR04

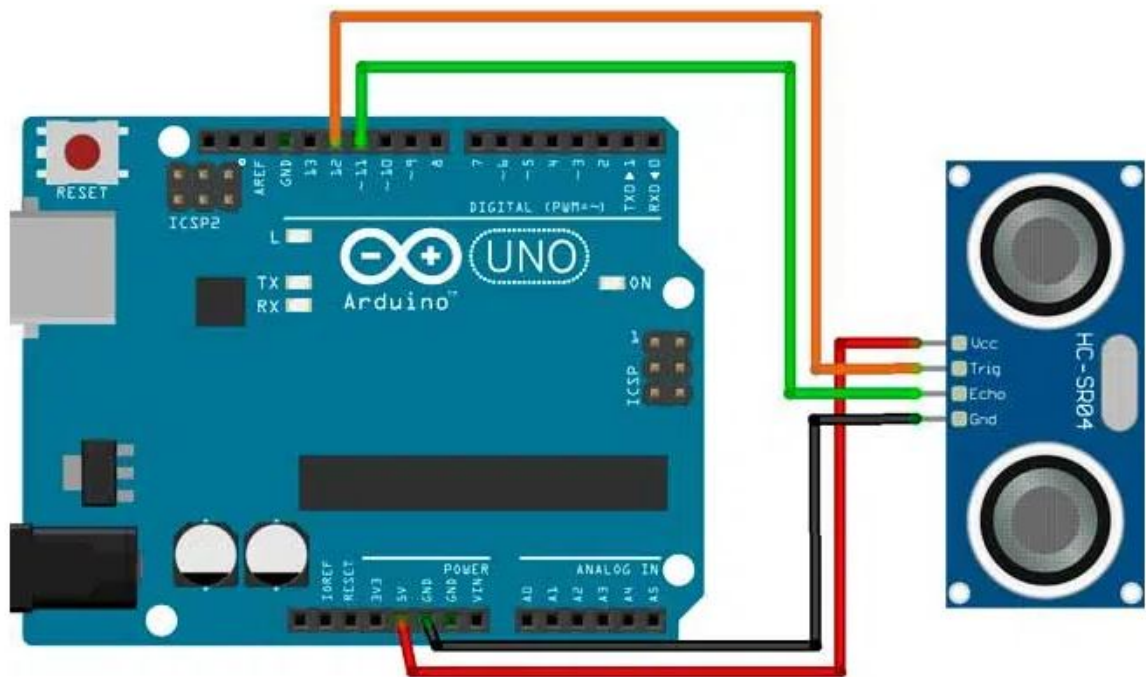


Broche	Description
VCC	Il s'agit de la broche d'alimentation. Elle nécessite généralement une entrée de 5V Courant Continu, la rendant directement compatible avec des cartes comme Arduino.
Trig (Déclenchement)	Cette broche est utilisée pour initier le capteur à émettre une onde ultrasonore. En envoyant une impulsion haute d'au moins 10µs à cette broche, le HC-SR04 émettra une série de 8 impulsions d'ultrasons à 40 kHz.
Echo	Une fois l'onde ultrasonore émise et qu'elle rebondit après avoir frappé un objet, la broche Echo fournit une impulsion de sortie. La largeur de cette impulsion est proportionnelle à la distance de l'objet par rapport au capteur. En mesurant la durée de cette impulsion, Arduino peut déterminer la distance jusqu'à l'objet.
GND (Masse)	Cette broche est connectée à la masse du circuit.

Le câblage :

Pour connecter le capteur HC-SR04 à une carte Arduino, suivez ces étapes :

1. Connectez la broche VCC du HC-SR04 à la broche 5V sur la carte arduino uno. Cela assure que le capteur reçoive la puissance nécessaire pour son fonctionnement.
2. Reliez la broche GND (Masse) du HC-SR04 à l'une des broches de masse (GND) d'Arduino. Cela établit une masse électrique commune entre le capteur et arduino.
3. Connectez la broche Trig du HC-SR04 à une broche numérique d'Arduino, par exemple, la broche D12. Cette broche est responsable de l'envoi d'un signal pour déclencher le capteur afin qu'il émette les ondes ultrasonores.
4. Reliez la broche Echo du HC-SR04 à une autre numérique sur l'Arduino, comme la broche D11. Cette broche détecte l'onde ultrasonore écho après réflexion sur un objet.



Programmer un HC-SR04

Ce code permet de mesurer une distance et de l'afficher sur le serial print du logiciel Arduino IDE.

```
// définition des numéros de broches
```

```
const int trigPin = 12;
```

```
const int echoPin = 11; // définition des variables

long duration;

int distance;

void setup()

{

  pinMode(trigPin, OUTPUT); // Définit le trigPin comme sortie

  pinMode(echoPin, INPUT); // Définit le echoPin comme entrée

  Serial.begin(9600); // Commence la communication série

}

void loop()

{

  // Efface le trigPin

  digitalWrite(trigPin, LOW);

  delayMicroseconds(2); // Met le trigPin à l'état HIGH pendant 10 microsecondes

  digitalWrite(trigPin, HIGH);

  delayMicroseconds(10);

  digitalWrite(trigPin, LOW); // Lit le echoPin, renvoie le temps de trajet de l'onde sonore en
microsecondes

  duration = pulseIn(echoPin, HIGH); // Calcul de la distance

  distance = duration * 0.034 / 2; // La vitesse du son est d'environ 0.034 cm par microseconde
```

```
Serial.print("Distance: "); // Affiche la distance sur le moniteur série

Serial.println(distance);

}
```

Source :

<https://www.moussasoft.com/hc-sr04-capteur-ultrason-avec-arduino>

<https://www.carnetdumaker.net/articles/mesurer-une-distance-avec-un-capteur-ultrason-hc-sr04-et-une-carte-arduino-genuino/>

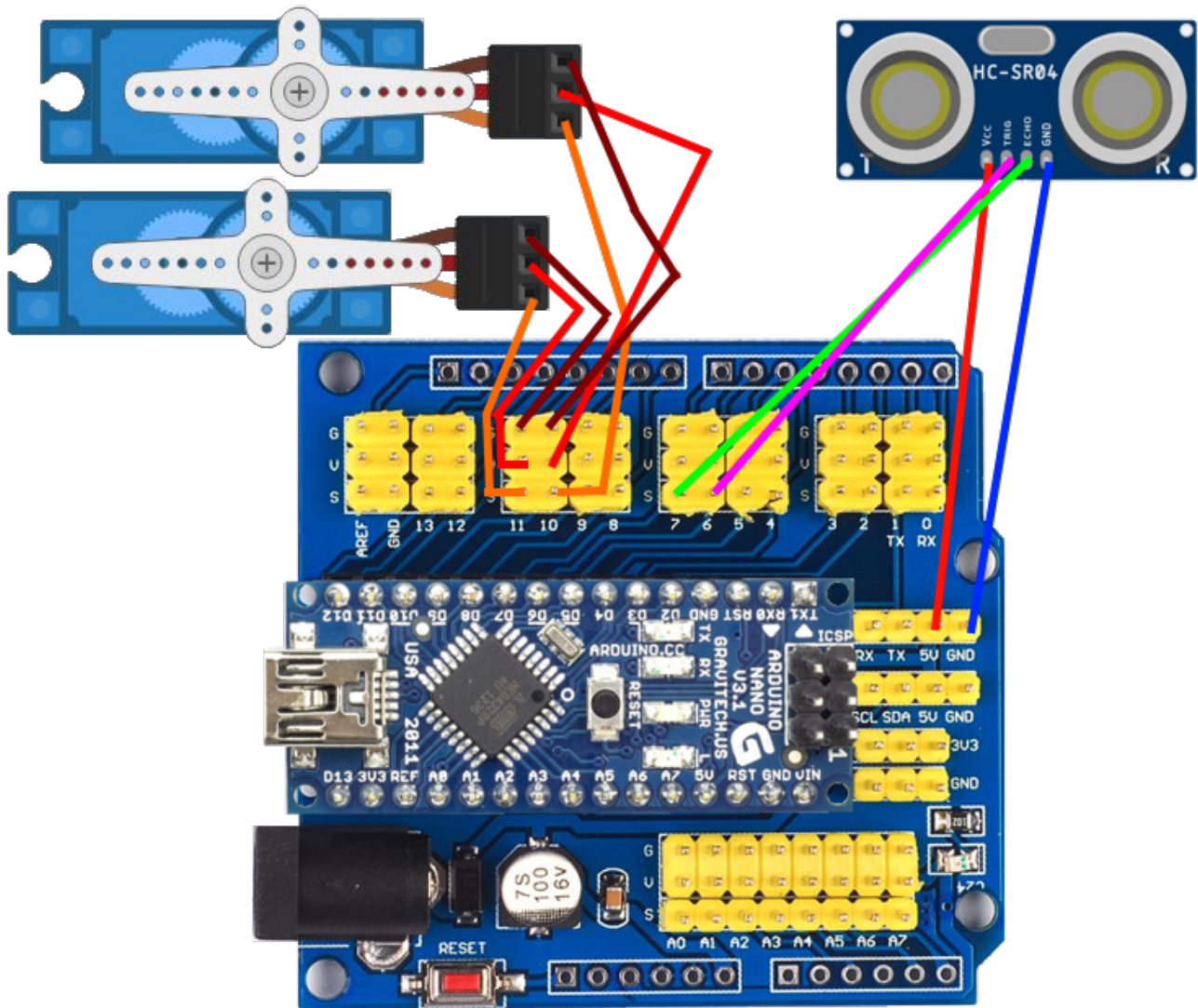
Câblage et programmation du LittleBot

Le câblage

Pour le câblage et la programmation du LittleBot, il nous faut :

- Un Arduino Nano (ou équivalent)
- Un Sensor Shield
- Un capteur à ultrason (HC-SR04)
- 2 Servomoteur / Moteur (DM-S0090D)

Le branchement se présente comme ceci :



Transl

Translator

1. Tout d'abord nous branchons l'Arduino Nano sur notre Sensor Shield. Attention, il y a un sens. Le port de charge doit être sur l'extérieur de votre Shield.
2. Nous allons à présent brancher notre capteur à ultrason :
 - VCC sur une pin 5V.
 - Trig sur la pin 6.
 - Echo sur la pin 7.

- GND sur une pin GND

3. Nous allons brancher nos Servomoteur, les câbles de nos servo sont tous reliés à un raccord. Celui ci ne peut être branché que dans un seul sens. Nous brancherons donc un servo sur la pin 10 et un servo sur la pin 11.

- Le fil marron sur la pin G
- Le fil rouge sur la pin V
- Le fil orange sur la pin S

Ainsi le servo qui est sur la pin 10 sera notre roue droite et le Servo sur la pin 11 sera notre roue gauche.

Votre câblage est terminé.

Passons maintenant à la programmation.

Le programme

Ici nous décomposerons notre programme pour bien l'écrire.

Tout d'abord, nous déclarons la librairie et les servo que nous utiliserons :

```
#include <Servo.h>
#define trigPin 6
#define echoPin 7
Servo servo1;
Servo servo2;
```

Puis nous déclarons sur quelles pins sont branchés notre capteur et nos servo :

```
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  servo1.attach(11);
  servo2.attach(10);
}
```

Rentrons dans le vif du sujet :

```
void loop() {
  long duration, distance;      // Nous déclarons notre variable que nous retrouverons plus tard
  digitalWrite(trigPin, LOW);  // Ici notre capteur à ultrason est en "position 0"
```

```

delayMicroseconds(2);          // Pendant 2 Microsecondes
digitalWrite(trigPin, HIGH);    //Ici notre capteur à ultrason est "activé"
delayMicroseconds(10);         //Pendant 10 Microsecondes
digitalWrite(trigPin, LOW);     //Puis nous le retournons en position "0"
duration = pulseIn(echoPin, HIGH); // Nous déclarons notre variable "duration" qui est la
durée du trajet du son.
distance = (duration*0.034) / 2; // Nous déclarons notre variables "distance" par la durée
multiplié par la vitesse du son le tout divisé par 2.
if (distance < 20) {            // Nos déplacement commence ici, "Si la distance est
inférieur à 20cm alors..."
    servo1.writeMicroseconds(1000); //Servo Gauche tourne à l'envers
    servo2.writeMicroseconds(2000); //Servo Droit tourne à l'envers
    delay (2000); // pendant 2 sec
    servo1.writeMicroseconds(1000); //Servo Gauche tourne à l'envers
    servo2.writeMicroseconds(1500); //Arrêt du Servo Droit
    delay (2000); // pendant 2 sec
}

else {                          //Sinon...
    servo1.writeMicroseconds(2000); //Servo Gauche tourne
    servo2.writeMicroseconds(1000); //Servo Droit tourne
    delay (2000); // pendant 2 sec
}
}

```

Puis nous assemblons le tout, voici à quoi cela devrait ressembler :

```

#include <Servo.h>
#define trigPin 6
#define echoPin 7
Servo servo1;
Servo servo2;

void setup() {
    pinMode(trigPin, OUTPUT);
    pinMode(echoPin, INPUT);
    servo1.attach(11);
    servo2.attach(10);
}

```

```
void loop() {  
  long duration, distance;  
  digitalWrite(trigPin, LOW);  
  delayMicroseconds(2);  
  digitalWrite(trigPin, HIGH);  
  delayMicroseconds(10);  
  digitalWrite(trigPin, LOW);  
  duration = pulseIn(echoPin, HIGH);  
  distance = (duration*0.034) / 2;  
  if (distance < 20) {  
    servo1.writeMicroseconds(1000);  
    servo2.writeMicroseconds(2000);  
    delay (2000);  
    servo1.writeMicroseconds(1000);  
    servo2.writeMicroseconds(1500);  
    delay (2000);  
  }  
  
  else {  
    servo1.writeMicroseconds(2000);  
    servo2.writeMicroseconds(1000);  
    delay (2000);  
  }  
}
```

Apprendre à utiliser Tinkercad (classe)

Utiliser Tinkercad

Tinkercad est un "logiciel" de modélisation 3D simple d'utilisation.

Il permet aussi de programmer et de créer des câblages avec des Arduino numérique.

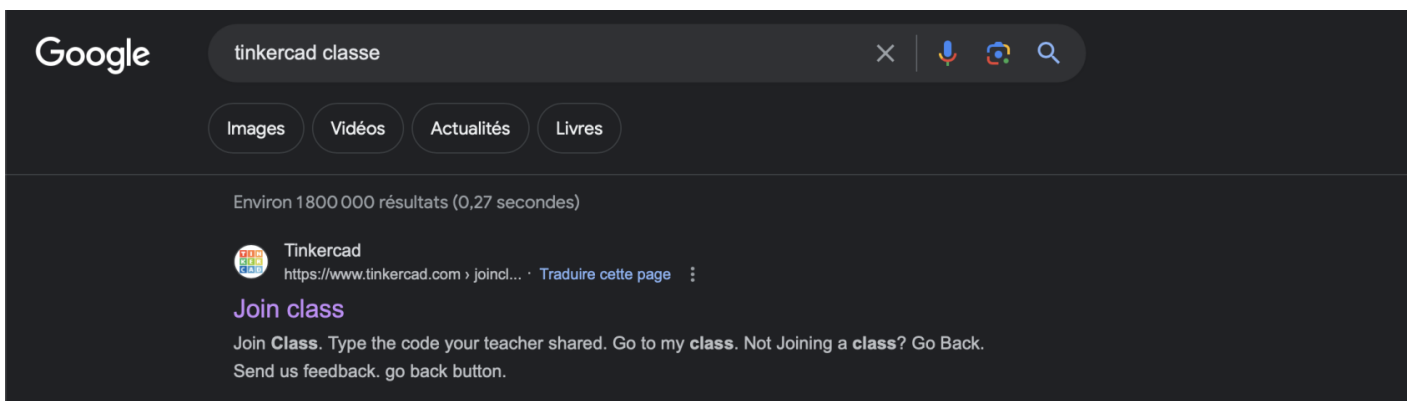
Tout d'abord il vous faudra, avec un compte enseignant, créer une classe sur le site web :

<https://www.tinkercad.com/>

Dans ce tuto nous allons apprendre les bases de ce logiciel avec la fonction perçage.

Voici les étapes pour que les utilisateurs puisse se connecter à la classe.

Rejoindre une classe



La première étapes sera d'aller sur le site web <https://www.tinkercad.com/joinclass>.

Puis d'entrer le code de classe fourni par le professeur.

<×

Rejoindre une classe

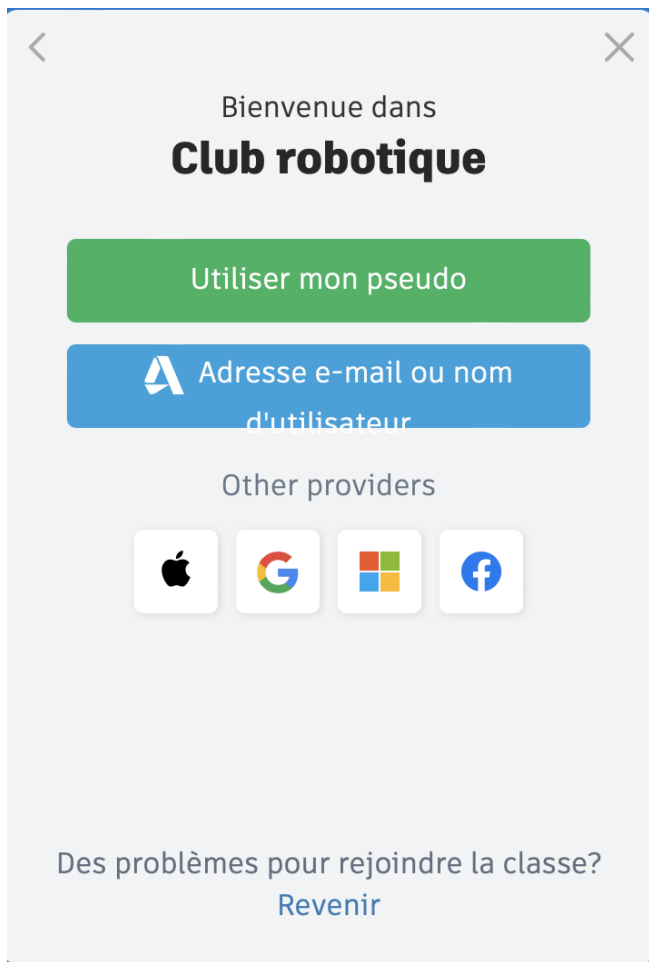
Entre le code fourni par ton professeur

59K 63V QVD

Accéder à ma classe

Des problèmes pour rejoindre la classe?

[Revenir](#)



Nous allons utilisé un pseudo prédéfini par le professeur.



Bienvenue dans
Club robotique

Quel est ton pseudo?

eleve(numéro)

Et voilà!

Des problèmes pour rejoindre la classe?

[Revenir](#)

Programme MBLOCK

