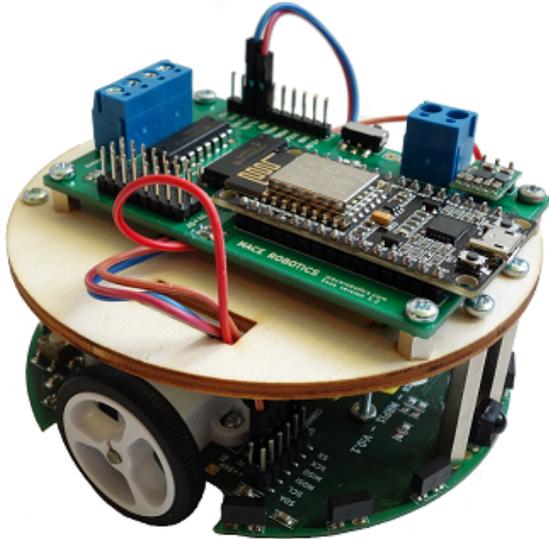


## Initiation à la programmation de robots

### Présentation du robot utilisé



Notre robot, nommé MRduino, est une petite machine disposant de deux roues montées chacune sur un moteur. Il comporte, en outre, divers équipements sur deux cartes électroniques, des capteurs qui permettent de mesurer des distances, des LED que l'on peut allumer ou éteindre, un haut-parleur etc.

La carte électronique supérieure joue le rôle de « cerveau » du robot, la carte inférieure exécute les ordres qu'elle reçoit de l'autre carte et lui transmet les informations qu'elle reçoit des capteurs.

### Principe de programmation

Nous « écrivons » des programmes sur un ordinateur, un Raspberry Pi -mais on peut utiliser n'importe quel ordinateur- et nous les transférons à la carte supérieure du robot.

Il faut donc un logiciel pour écrire les programmes. Ce sera Blockly. Il faut aussi un logiciel pour faire le transport des programmes de l'ordinateur au robot. Et là, ce sera une application appelée Arduino IDE (ou IDE Arduino).

### Lancer les applications sur l'ordinateur

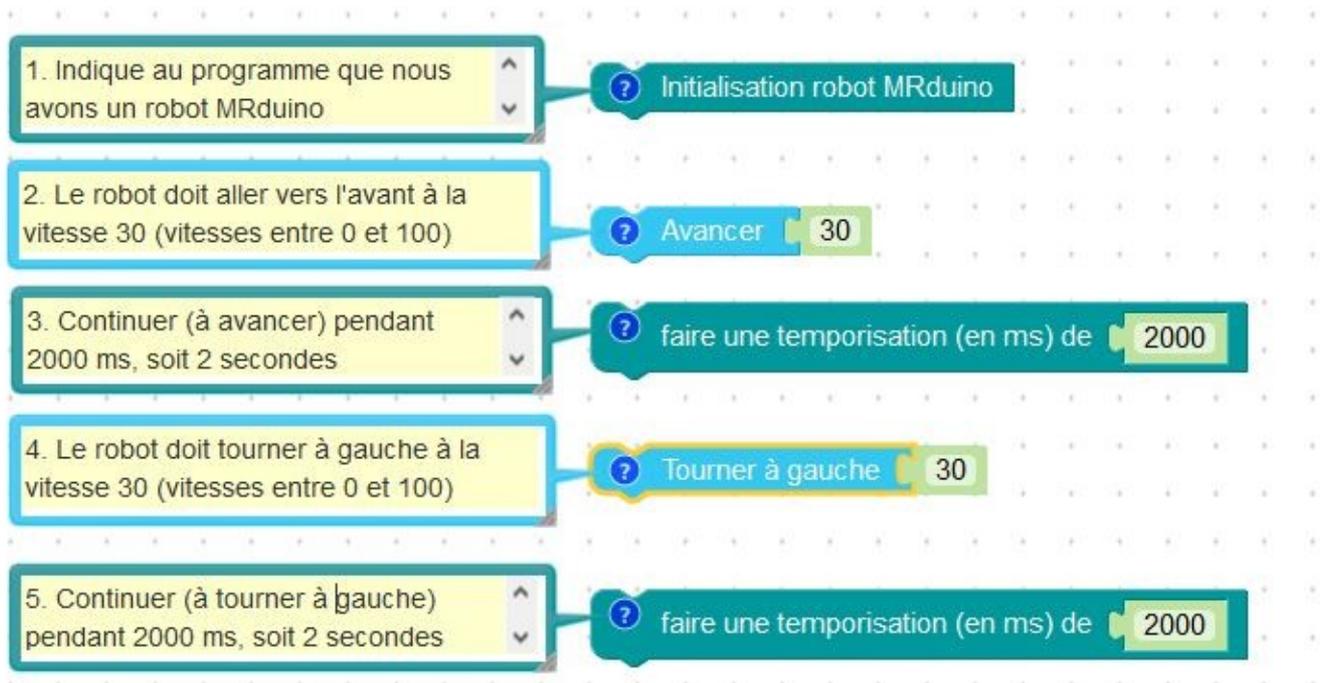
Lancer les deux applications Blockly et Arduino IDE comme indiqué par les animateurs. En principe ces deux logiciels sont configurés correctement.

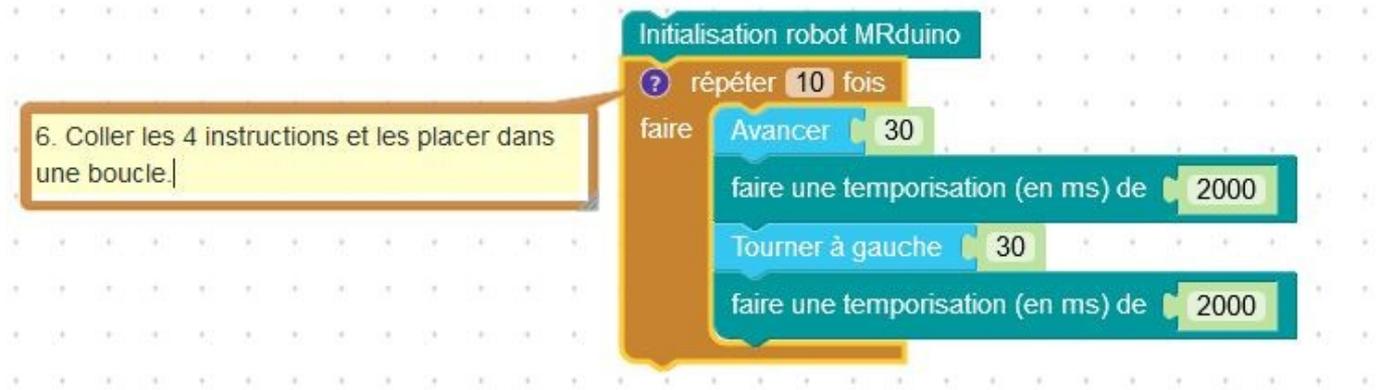
## Un premier programme en langage Blockly

Comme son nom l'indique, Blockly permet de réaliser des programmes à partir de blocs comme Scratch que nous avons vu dans la séquence 1.

Tout d'abord, ouvrir la famille de blocs « Mrduino Robot » et choisir le bloc « Initialisation robot MRduino » . Glisser-déposer ce bloc sur la zone blanche.

Suivre ensuite les indications numérotées sur les images suivantes.





Voici ce que l'on doit avoir quand ce premier exercice est terminé.

Cliquer ensuite sur « code arduino ».

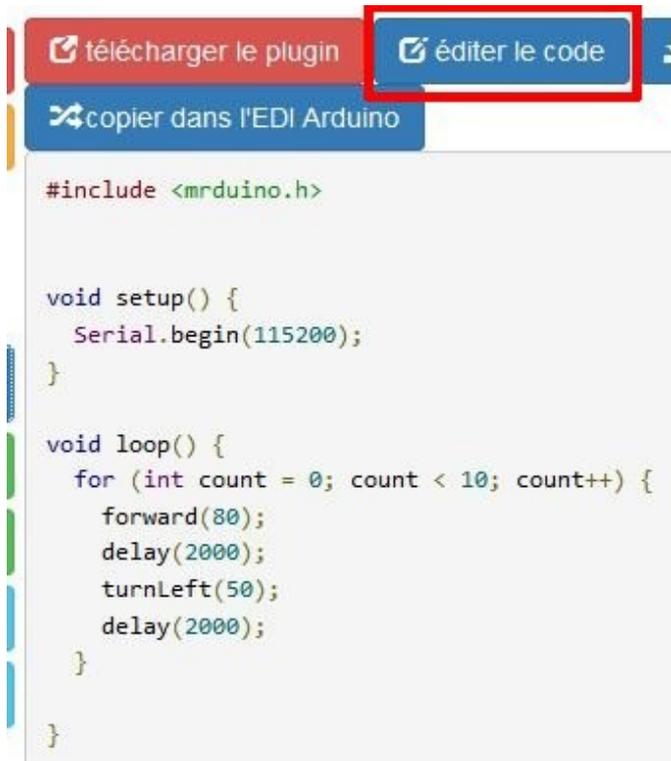


Noter ici que l'on peut passer facilement des blocs au code et inversement avec les deux « boutons » « code arduino » et « blocs ».

Notre code est toujours dans Blockly. Il faut le transférer dans Arduino IDE.

## Téléverser le programme dans le robot

Il s'agit maintenant de transférer notre programme dans la « mémoire » du robot. Ici, c'est ce que l'on appelle « téléverser » le programme



### édition du code Arduino

```
#include <mrduino.h>

void setup() {
  Serial.begin(115200);
}

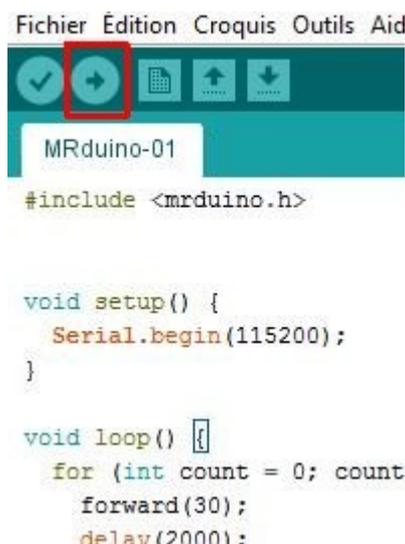
void loop() {
  for (int count = 0; count < 10; count++) {
    forward(80);
    delay(2000);
    turnLeft(50);
    delay(2000);
  }
}
```

Les blocs ont été traduits en « code », forme de langage plus proche de ce que peut comprendre le « cerveau » du robot.

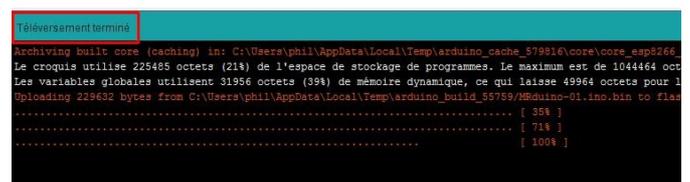
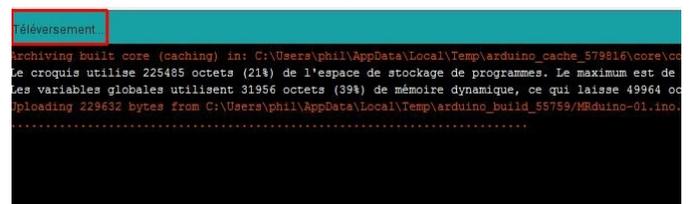
Maintenant, dans Blockly, cliquer sur « éditer le code »

Nous voyons ici le « code » dans une nouvelle fenêtre qui vient de s'ouvrir.

Ce code doit être copié dans le logiciel « Arduino IDE ». Sélectionnez-le entièrement (Ctrl-A) puis copiez-le dans le presse-papier (Ctrl-C)



Dans la zone de texte de l'IDE Arduino, faites un « coller » par Ctrl-V. Le code généré avec Blockly doit maintenant apparaître dans la fenêtre de l'IDE Arduino. Il ne reste qu'à le téléverser dans le robot.



Connecter le robot à l'ordinateur (câble USB sur la carte électronique supérieure) puis cliquez sur l'icône de téléversement. Après sauvegarde du fichier (la première fois, un nom est demandé), le code est « compilé » puis téléversé dans le robot.

## Démarrer le robot

Nous pouvons maintenant procéder au premier test de fonctionnement du robot.

Il suffit de le débrancher pour de mettre les deux interrupteurs sur « on ». Les LED vertes doivent être allumées.

Si tout fonctionne bien, le robot doit avancer tout droit, puis tourner à gauche et recommencer indéfiniment.

## Améliorations du programme

```
void loop() {  
  for (int count = 0; count < 10; count++) {  
    forward(80);  
    delay(4000);  
    turnLeft(50);  
    delay(2000);  
  }  
}
```

**Paramètres à modifier**

Modifier les paramètres des différentes instructions, soit dans Blockly, soit directement dans l'IDE Arduino. Recommencer le téléversement (interrupteurs du robots sur « off ») et tester maintenant le programme modifié.

## Un programme analogue à celui de la séquence 1 : les polygones.

The image shows a Blockly script for drawing a polygon. The code starts with an initialization block for the MRduino robot. It sets a variable 'n' to 3 and calculates the angle as 360 degrees divided by 'n'. The program then activates control, turns on LED 2, and enters a loop that repeats 'n' times. Inside the loop, it moves forward 100 units, toggles LED 1 and LED 2, waits 1000 ms, turns right by the calculated angle, toggles LED 1 and LED 2 again, and waits another 1000 ms.

## Un programme bien plus ambitieux utilisant les capteurs de distance.

Nous écrivons ici un programme qui permet au robot de se déplacer à l'intérieur d'une enceinte fermée sans toucher aux parois. Quand il approche d'un obstacle, paroi ou autre, les capteurs lui envoient l'information et il tourne ou recule pour choisir une autre direction de progression.

En somme, il fait comme vous même quand vous approchez d'un mur !

```

Initialisation robot MRduino
répéter 10 fois
faire
  mettre la variable Dist1 à Capteur proximité 1
  mettre la variable Dist2 à Capteur proximité 2
  mettre la variable Dist3 à Capteur proximité 3
  mettre la variable Dist4 à Capteur proximité 4
  mettre la variable Dist5 à Capteur proximité 5
  mettre la variable Dist6 à Capteur proximité 6
  si Dist3 > 600 et Dist4 > 600
  alors
    reculer 40
    faire une temporisation (en ms) de 200
    Tourner à droite 30
    faire une temporisation (en ms) de 500
  sinon si Dist1 > 300
  alors
    Tourner à droite 30
    faire une temporisation (en ms) de 100
  
```

The image displays two segments of code blocks for a robot's navigation logic. The top segment contains four conditional blocks, each starting with 'sinon si' (if not) and followed by an 'alors' (then) block. Each 'sinon si' block checks a distance variable (Dist2, Dist3, Dist4, Dist5) against a threshold value. If the condition is met, the 'alors' block performs a turn (right or left) and a 100ms delay. The bottom segment starts with a 'sinon si' block checking 'Dist6' against 300. If true, it turns left and delays 100ms. If false ('sinon'), it moves forward by 70 units and delays 100ms.

```
sinon si [Dist2] > 500
alors [Tourner à droite] 30
    [faire une temporisation (en ms) de] 100
sinon si [Dist3] > 600
alors [Tourner à droite] 40
    [faire une temporisation (en ms) de] 100
sinon si [Dist4] > 600
alors [Tourner à gauche] 40
    [faire une temporisation (en ms) de] 100
sinon si [Dist5] > 500
alors [Tourner à gauche] 30
    [faire une temporisation (en ms) de] 100

sinon si [Dist6] > 300
alors [Tourner à gauche] 30
    [faire une temporisation (en ms) de] 100
sinon [Avancer] 70
    [faire une temporisation (en ms) de] 100
```