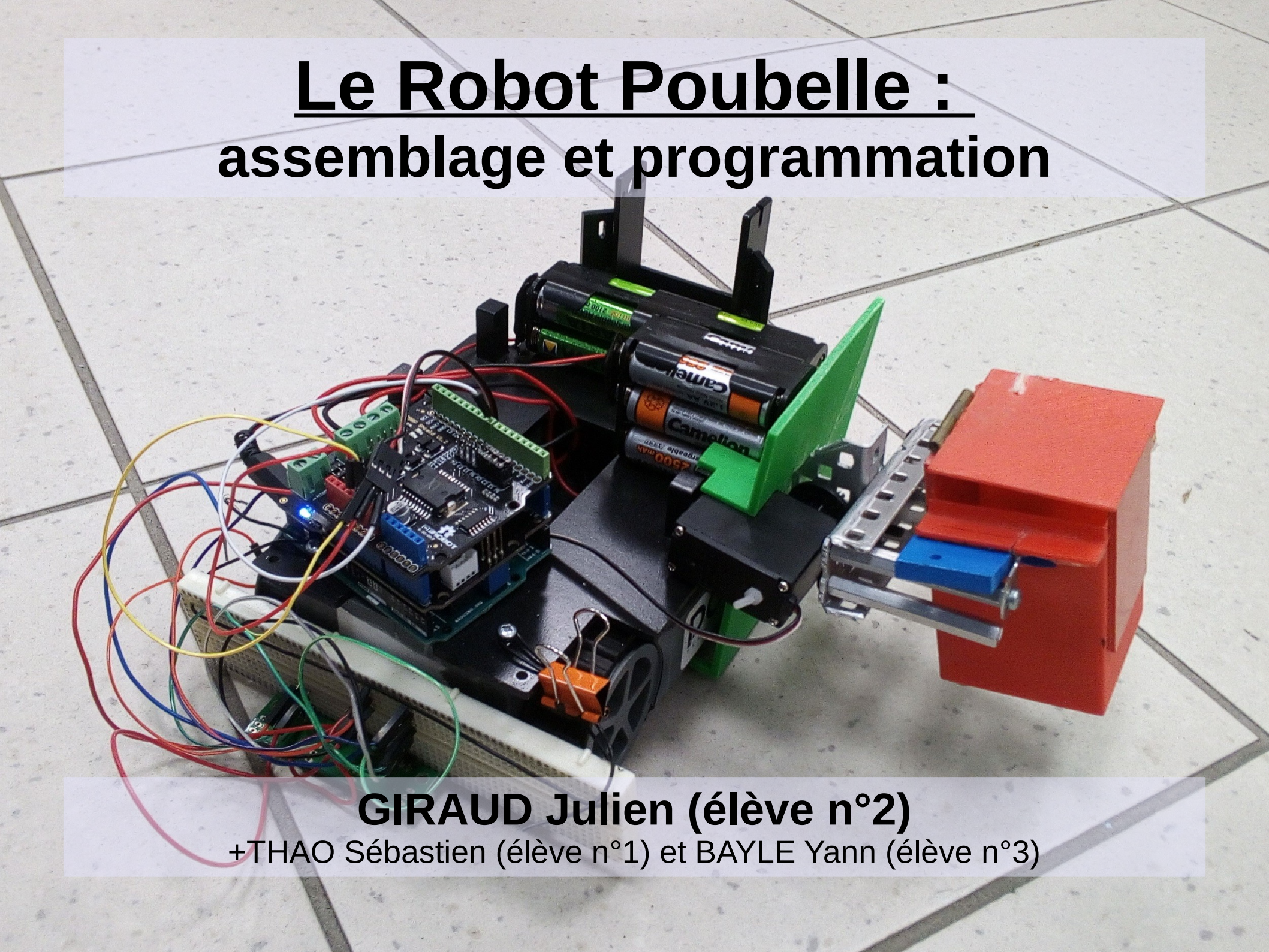
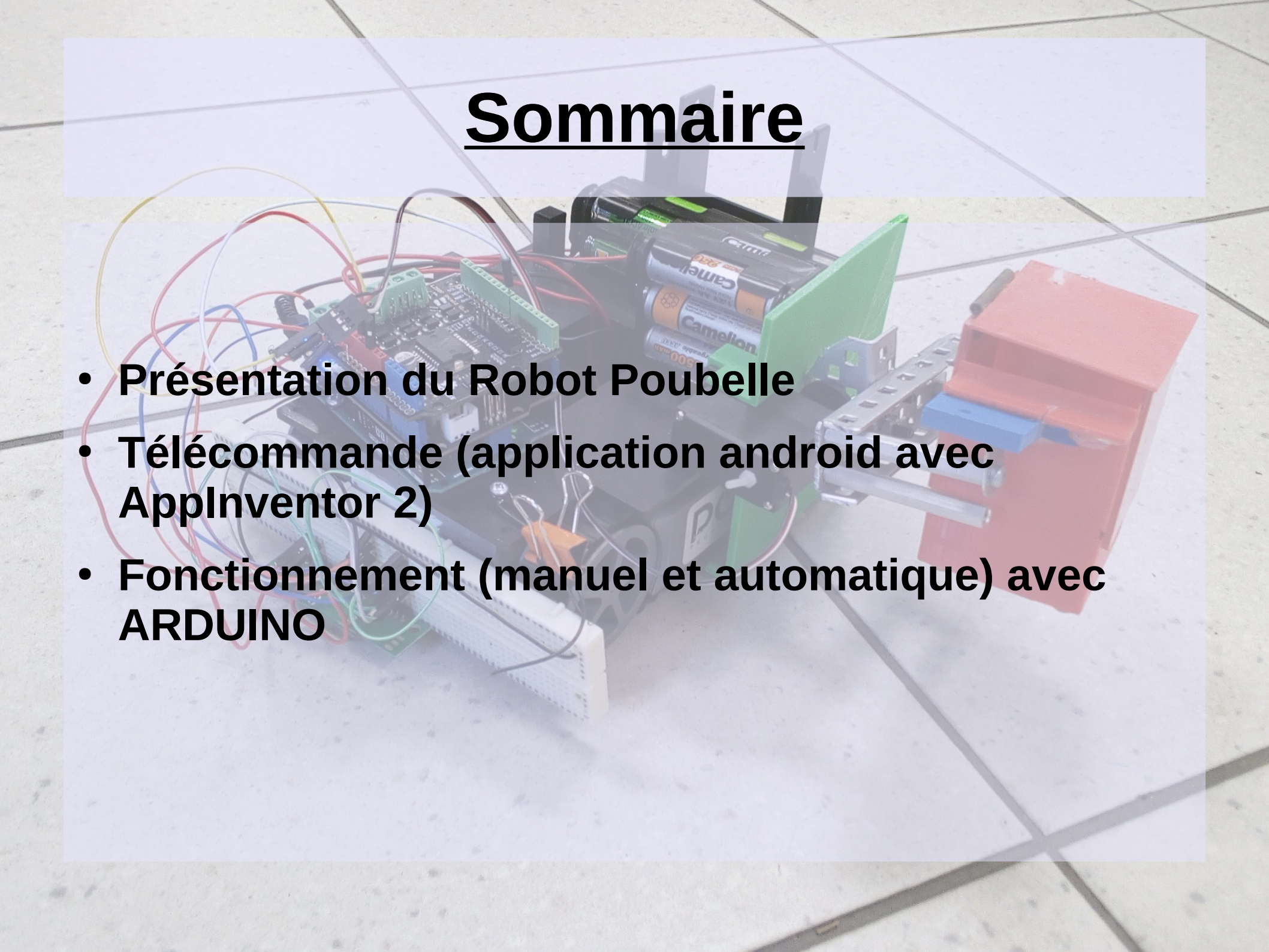
Le Robot Poubelle :
assemblage et programmation



GIRAUD Julien (élève n°2)
+THAO Sébastien (élève n°1) et BAYLE Yann (élève n°3)

Sommaire



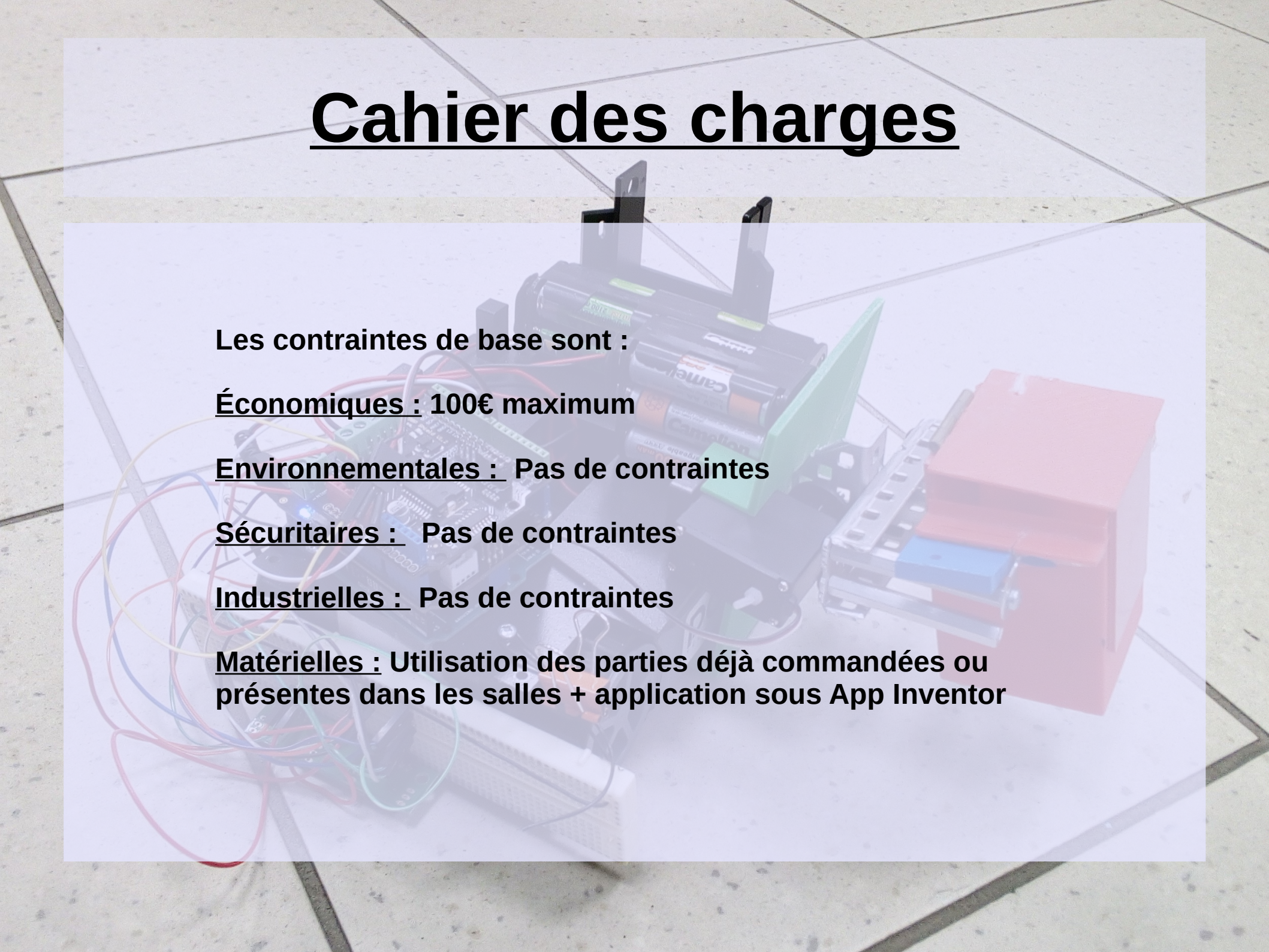
- **Présentation du Robot Poubelle**
- **Télécommande (application android avec AppInventor 2)**
- **Fonctionnement (manuel et automatique) avec ARDUINO**

Sujet

Un gardien de résidence souhaite pouvoir convoyer les poubelles jusqu'à la benne, à l'aide d'un robot suivant un parcours, ou pouvant être commandé manuellement via une application.

Comment permettre au gardien de convoyer les poubelles jusqu'à la benne automatiquement ?

Cahier des charges



Les contraintes de base sont :

Économiques : 100€ maximum

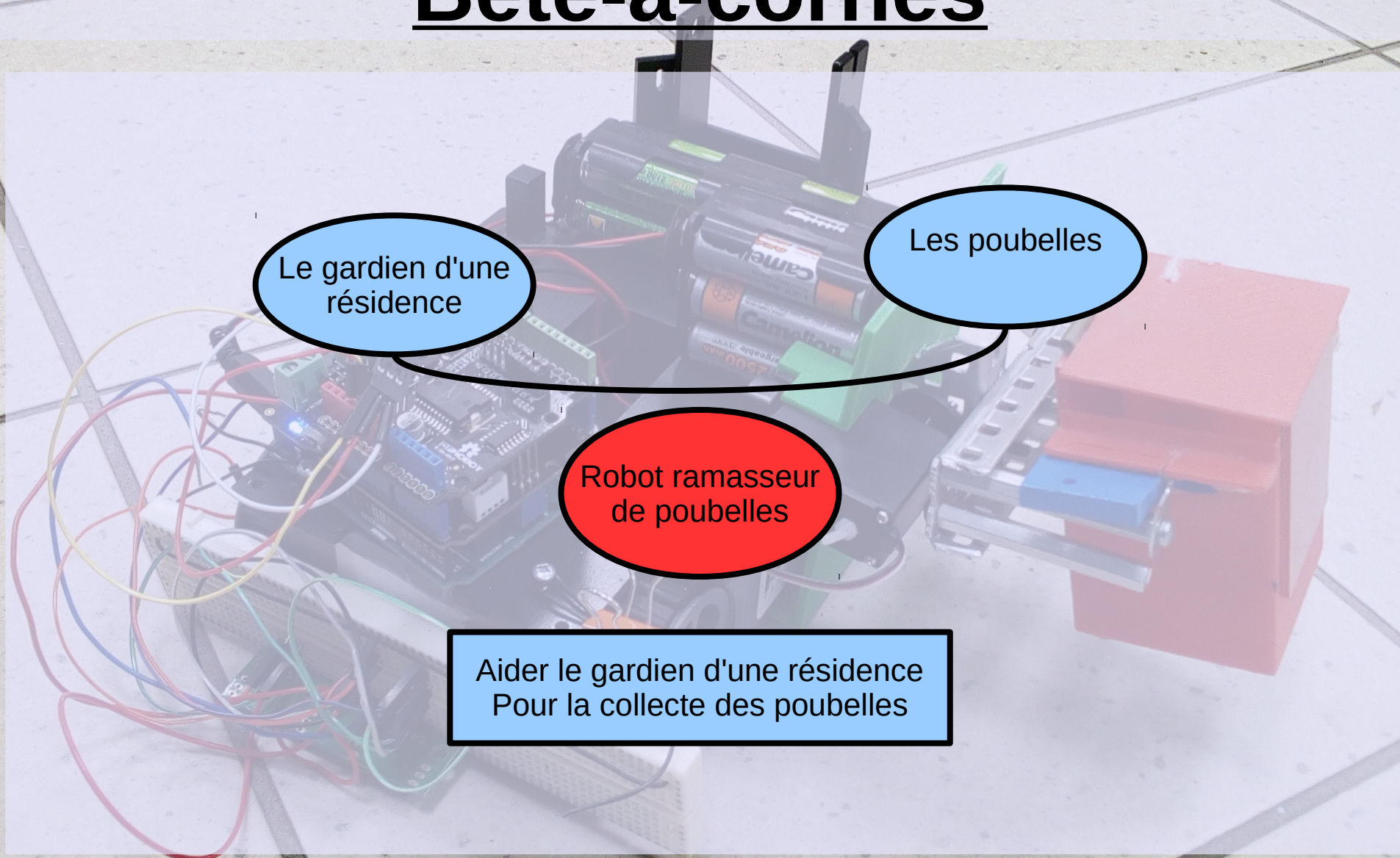
Environnementales : Pas de contraintes

Sécuritaires : Pas de contraintes

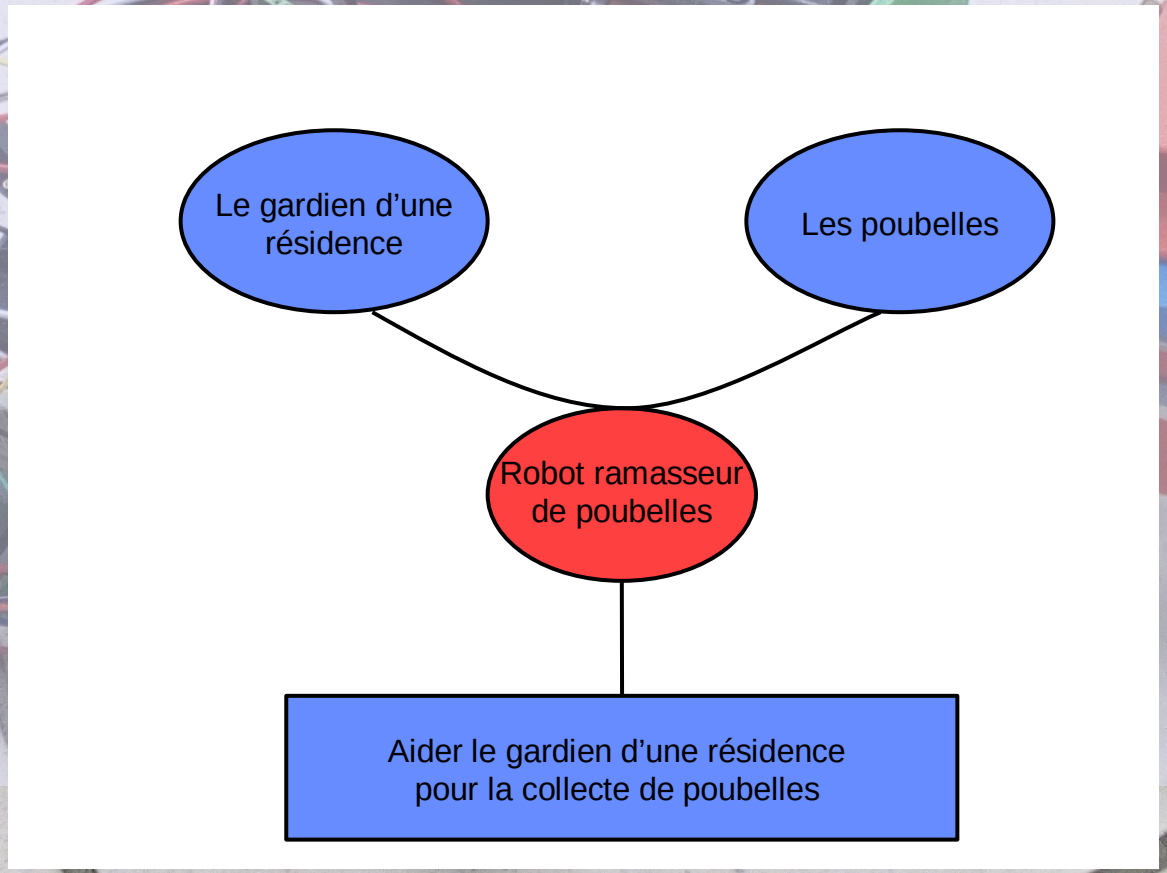
Industrielles : Pas de contraintes

Matérielles : Utilisation des parties déjà commandées ou présentes dans les salles + application sous App Inventor

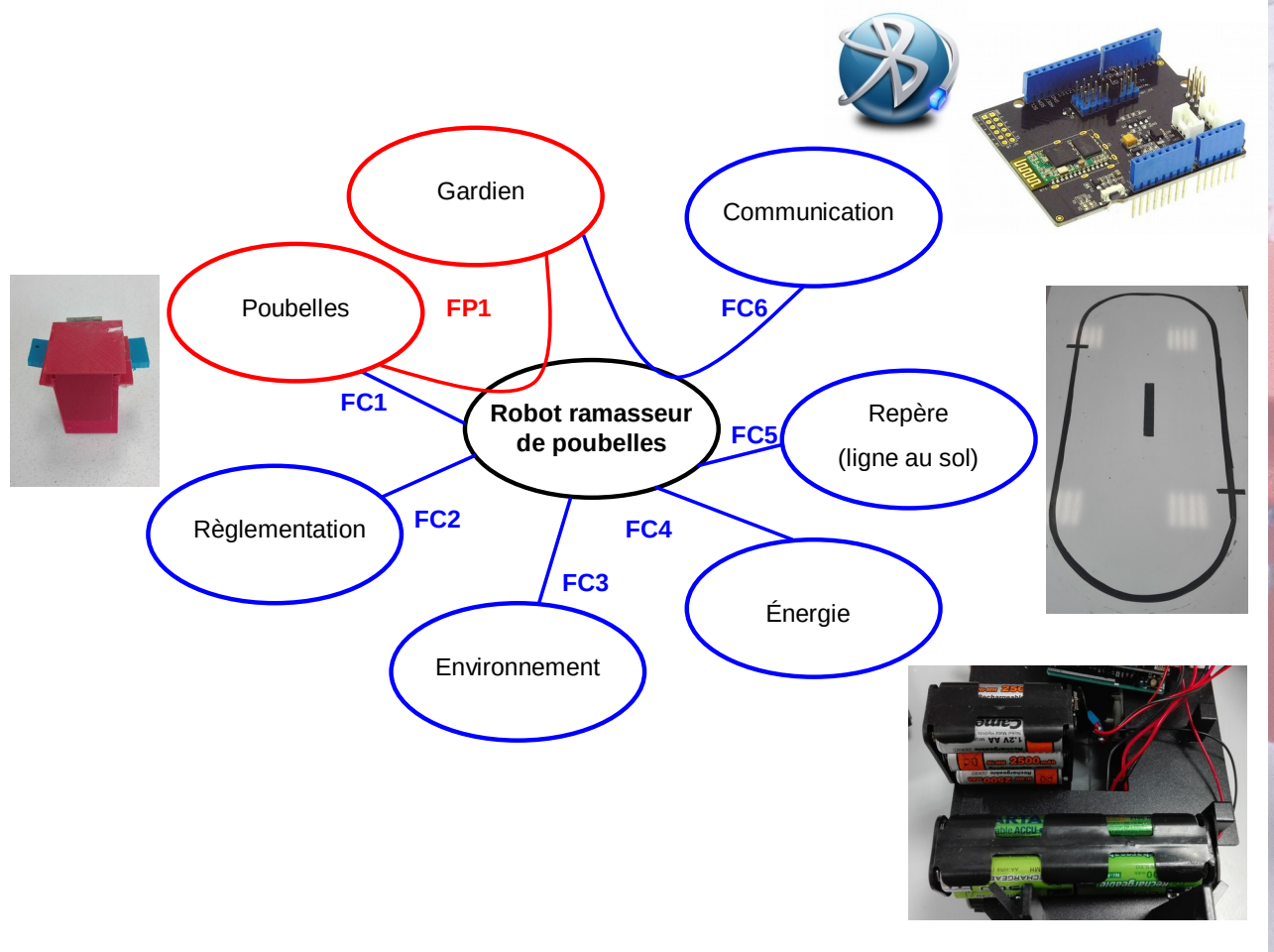
Analyse fonctionnelle: Bête-à-cornes



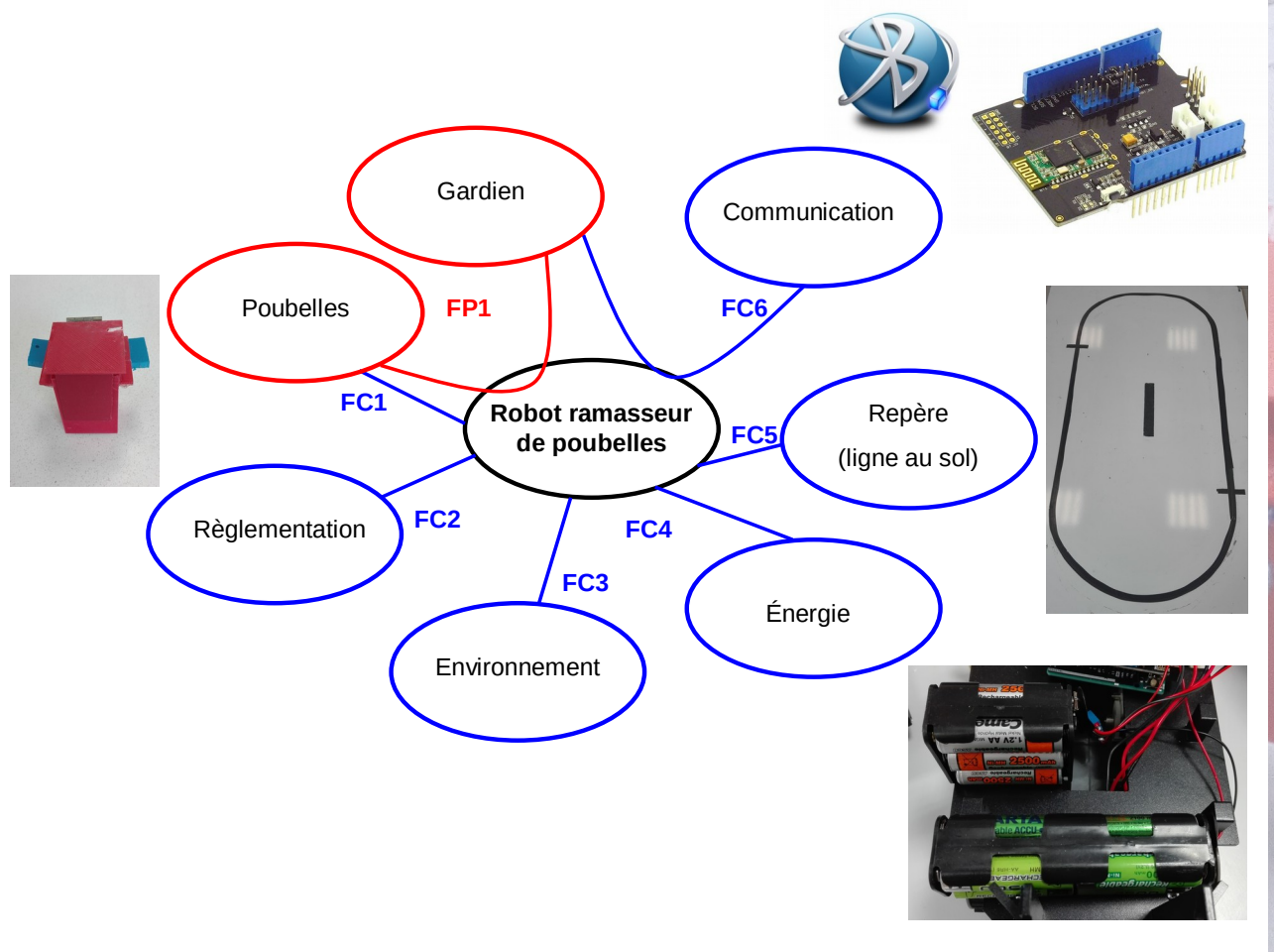
Analyse fonctionnelle: Bête-à-cornes



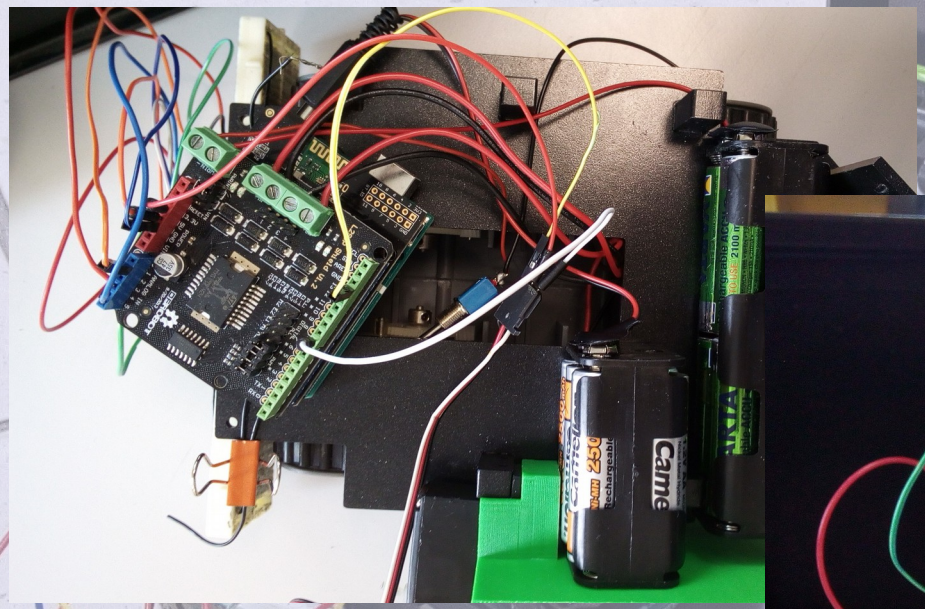
Analyse fonctionnelle: Diagramme pieuvre

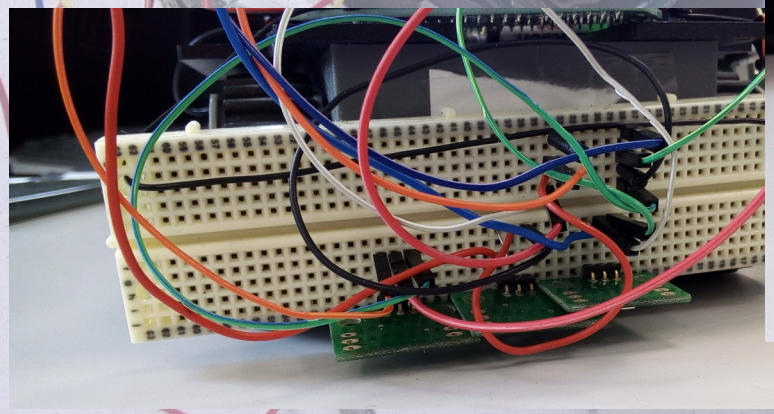


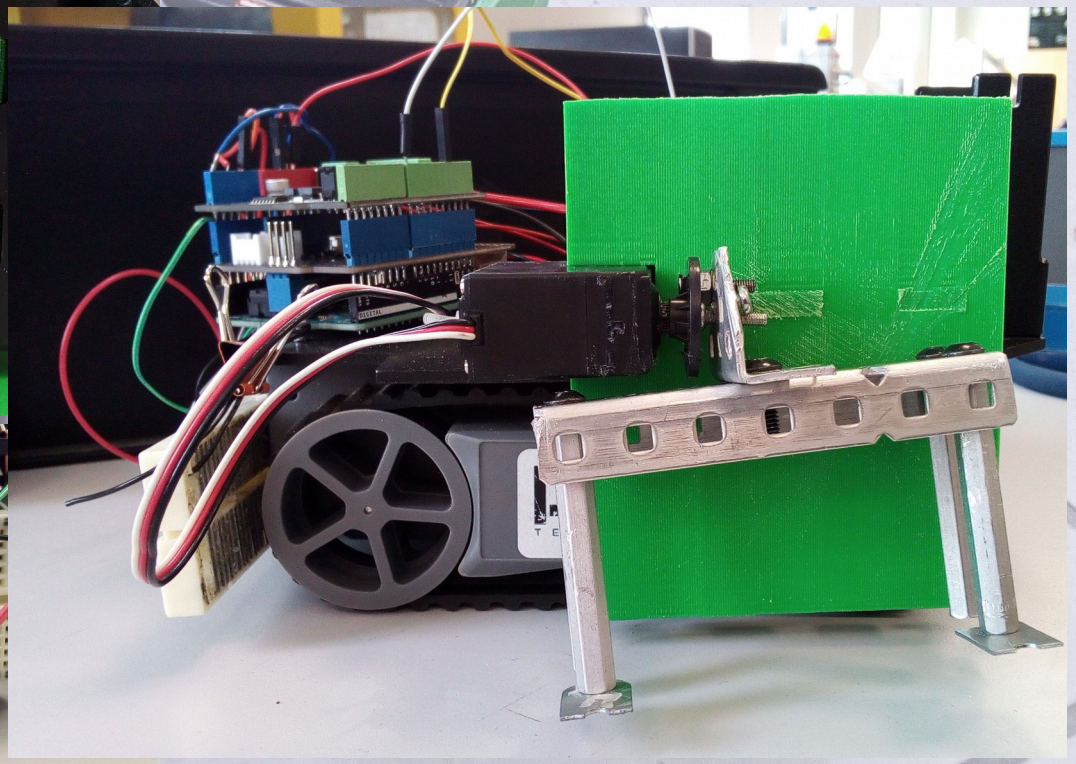
Analyse fonctionnelle: Diagramme pieuvre



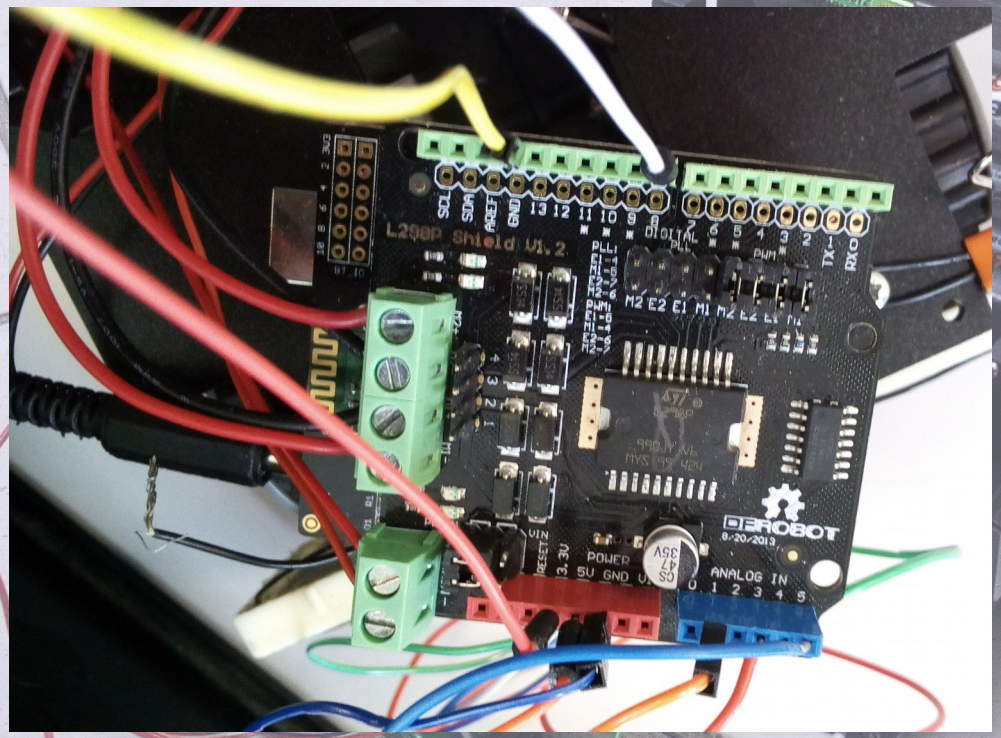
Le Robot

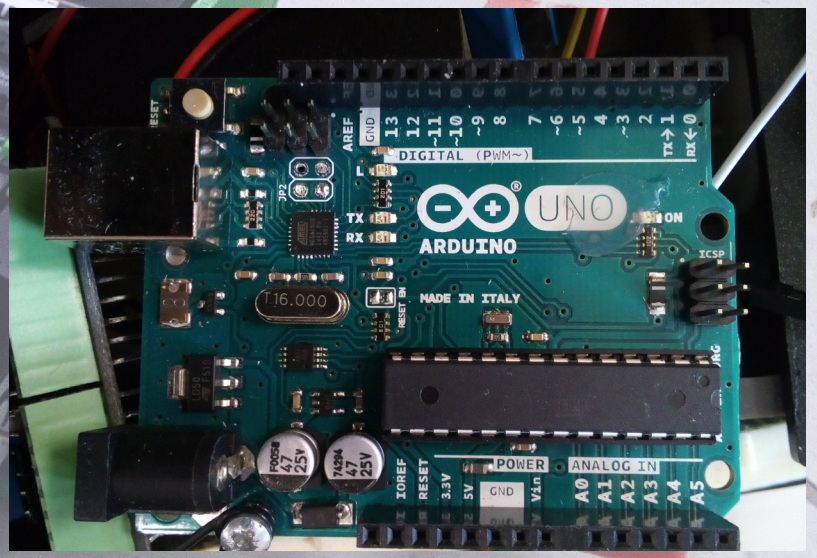




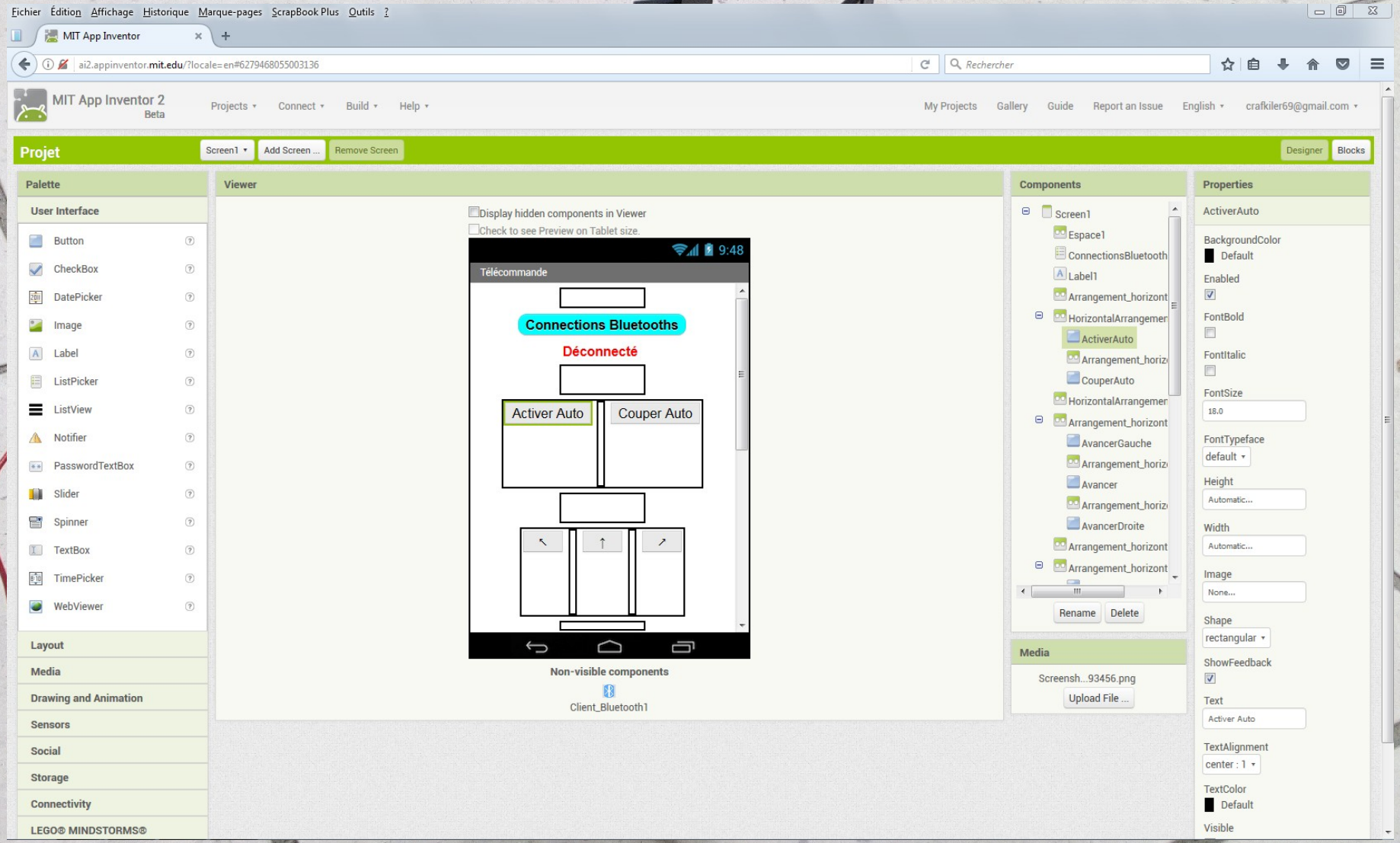


Le Robot

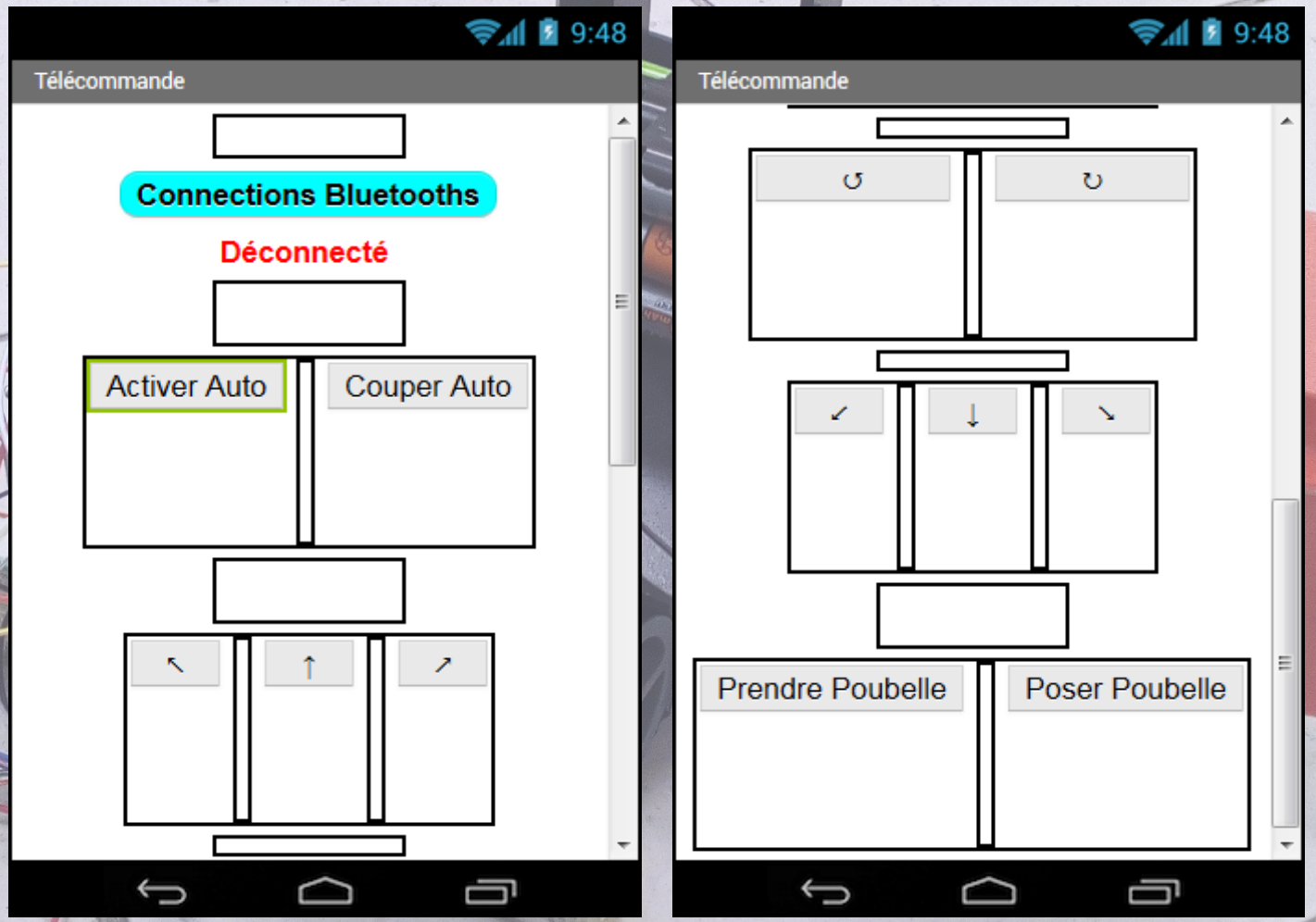

L298P Shield V1.2
DFROBOT


UNO
ARDUINO
MADE IN ITALY
DIGITAL (PWM~)
POWER
ANALOG IN

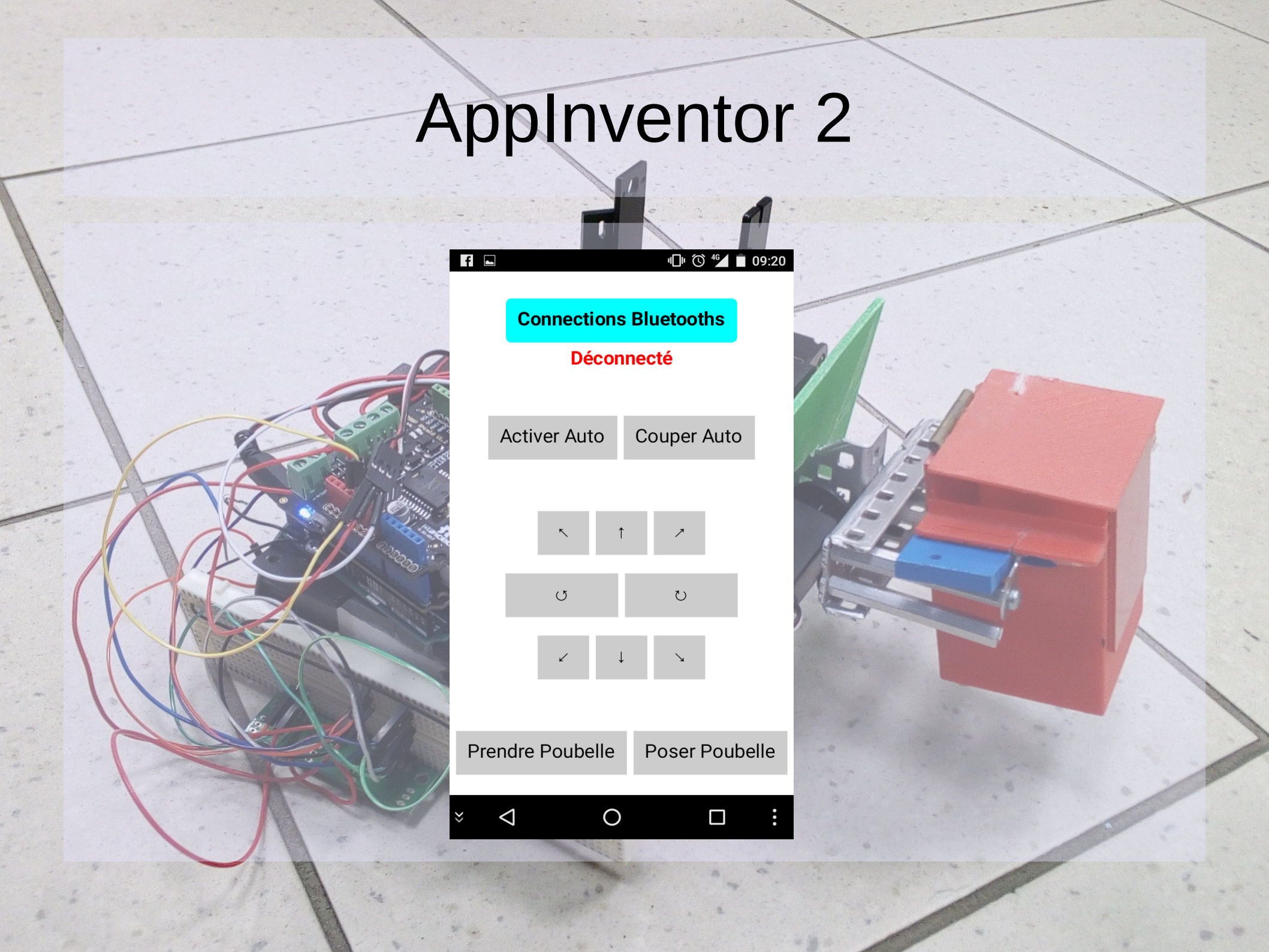
Site AppInventor 2



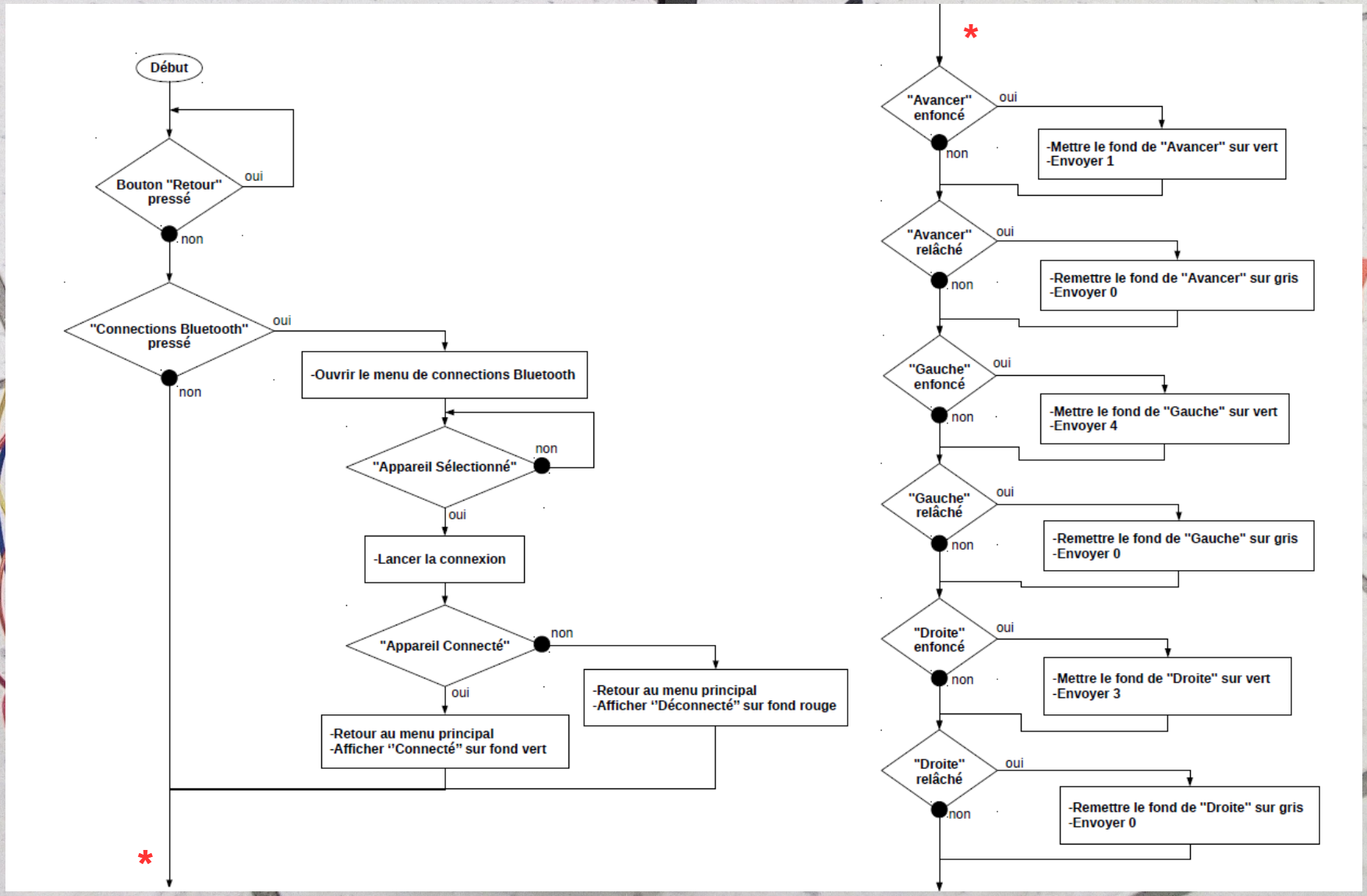
AppInventor 2



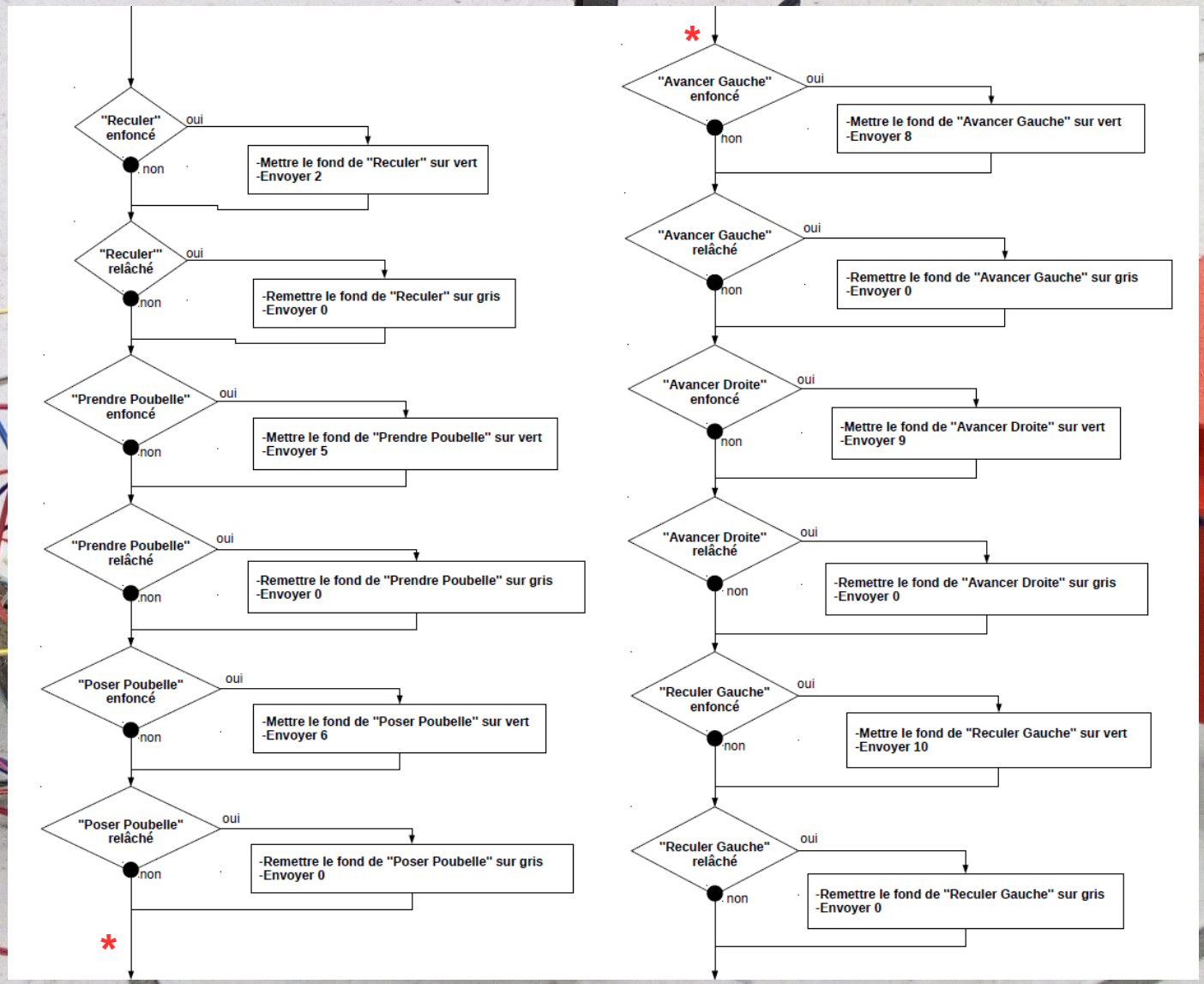
AppInventor 2



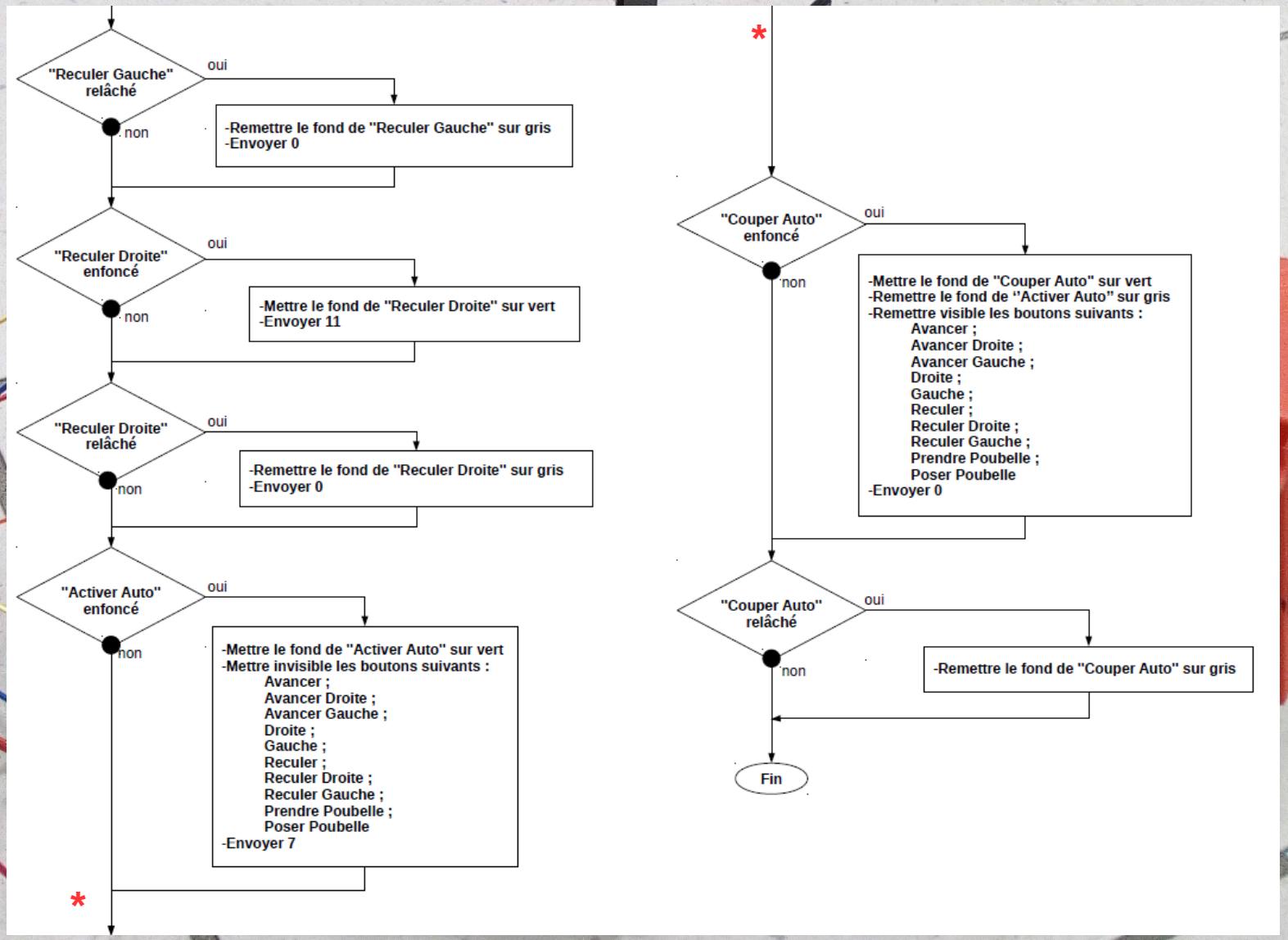
Logigramme AppInventor 2



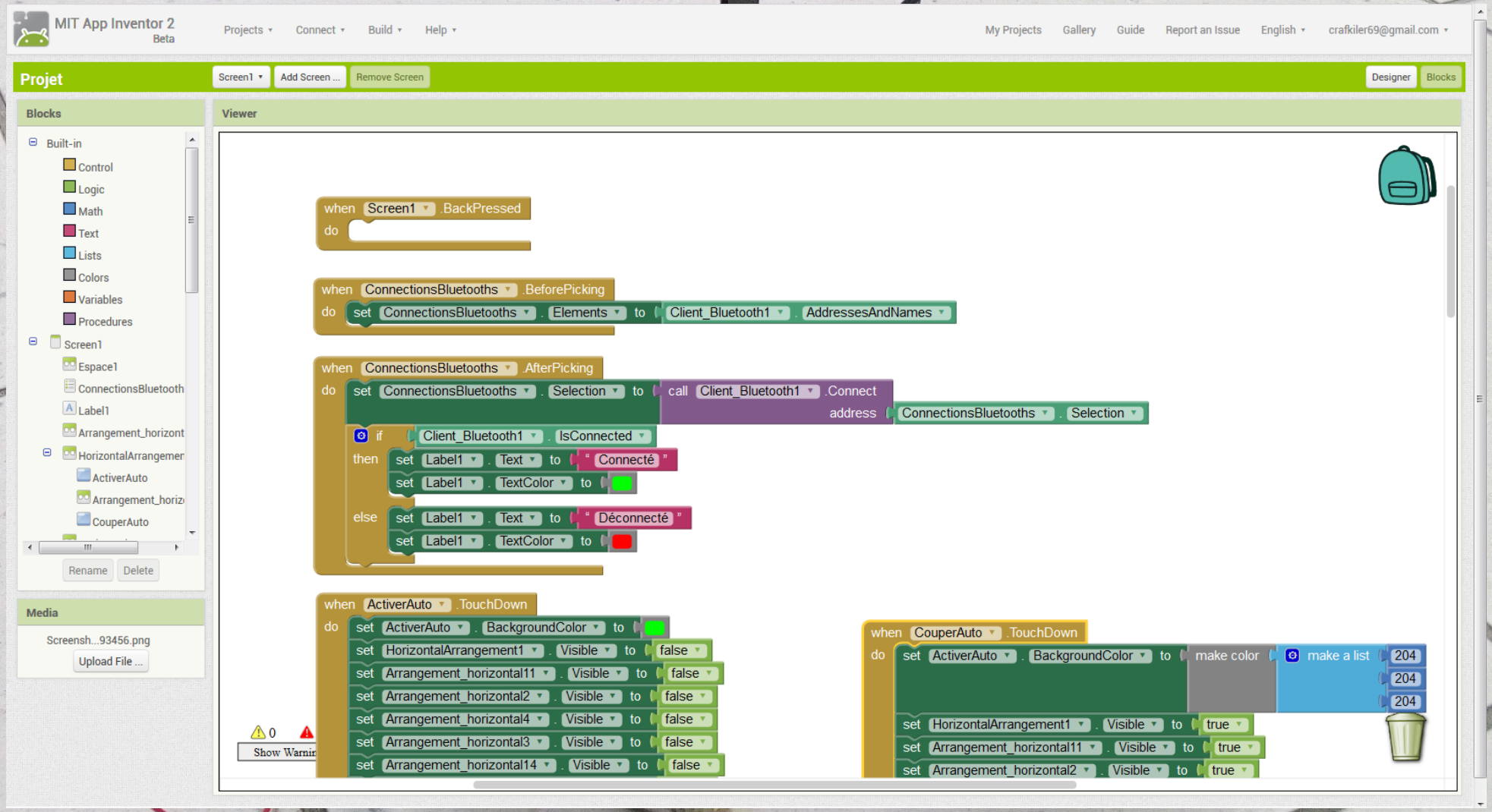
Logigramme AppInventor 2



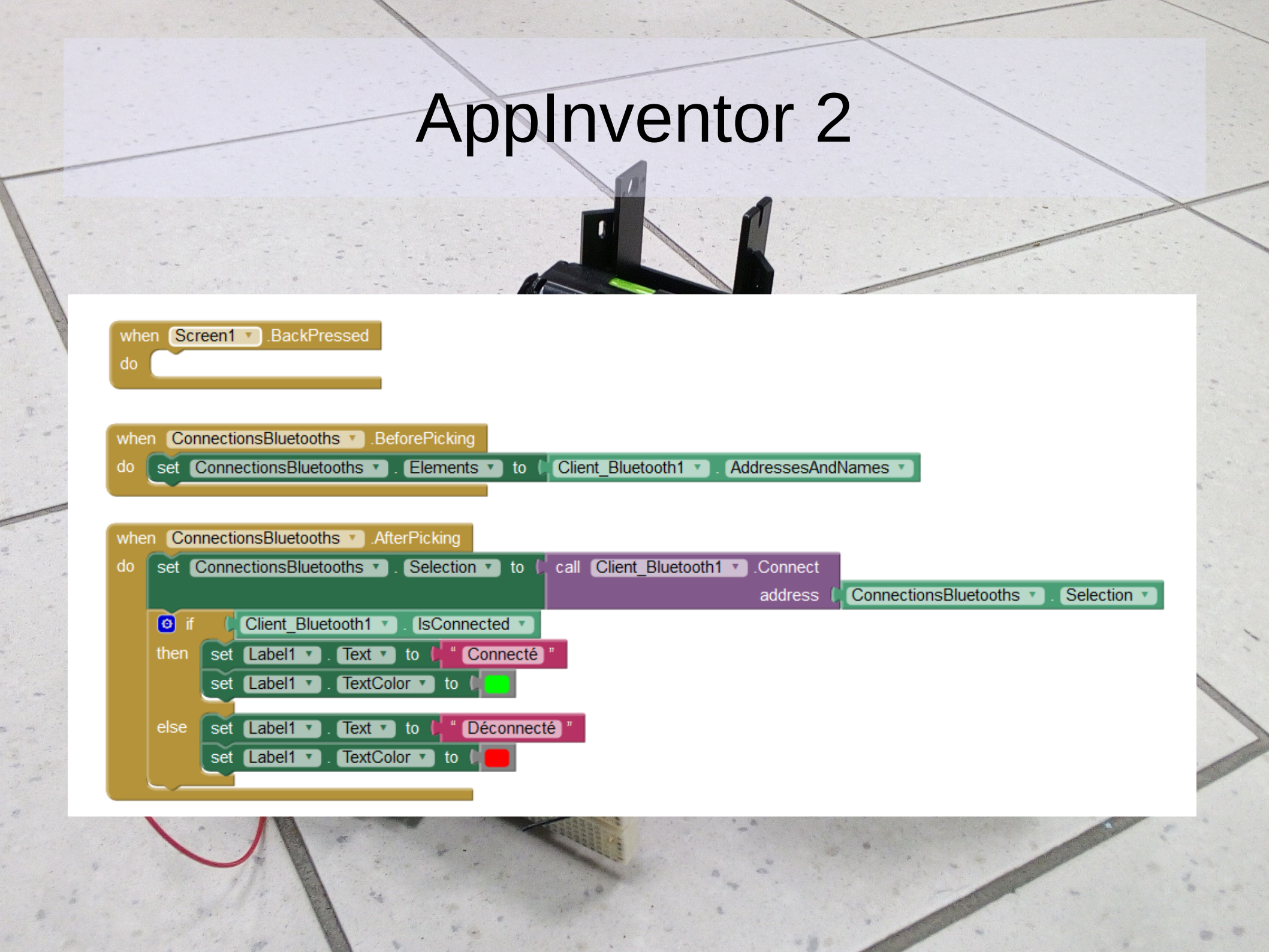
Logigramme AppInventor 2



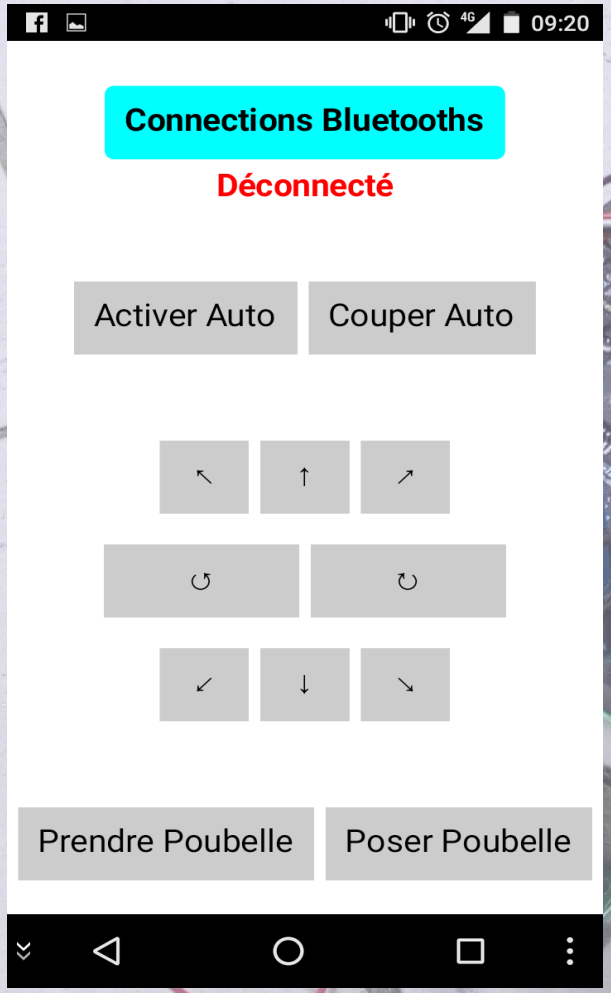
Site AppInventor 2

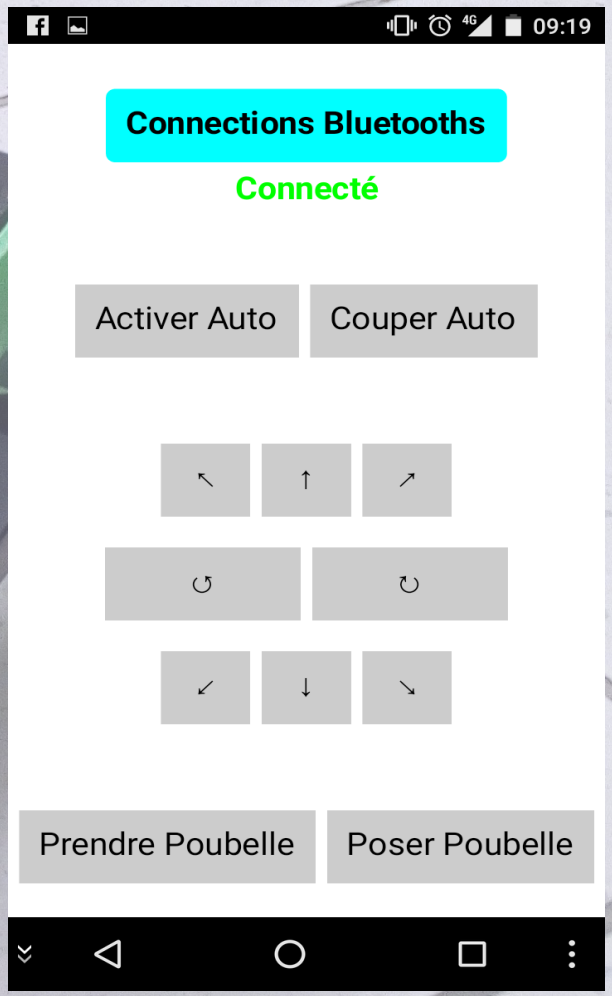


AppInventor 2

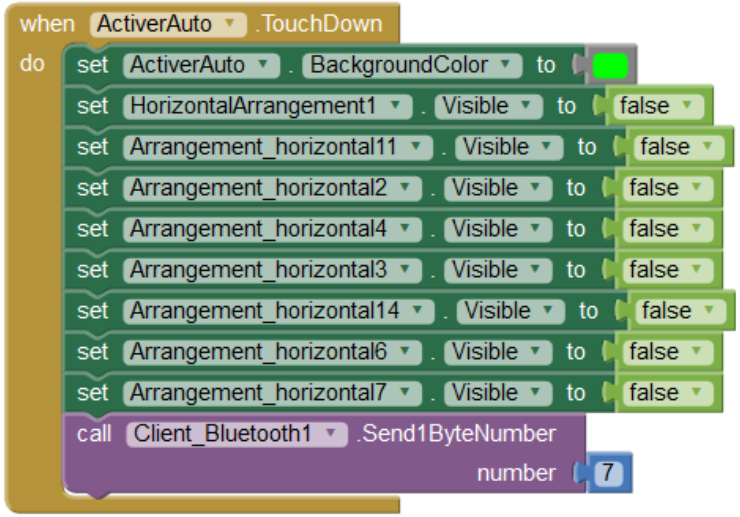


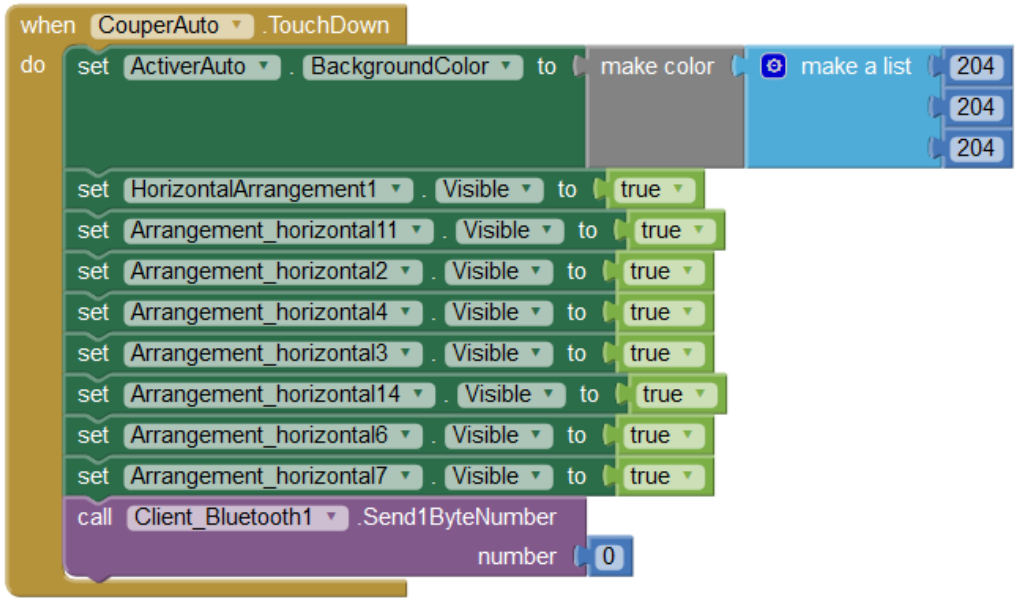
AppInventor 2

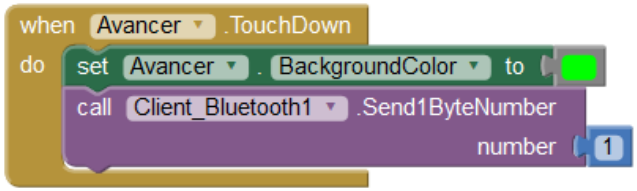


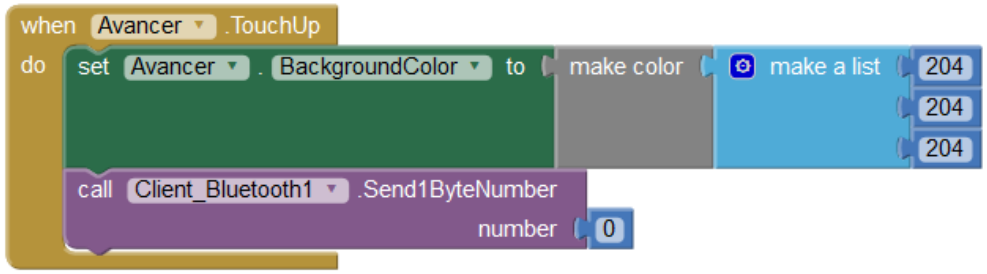


AppInventor 2

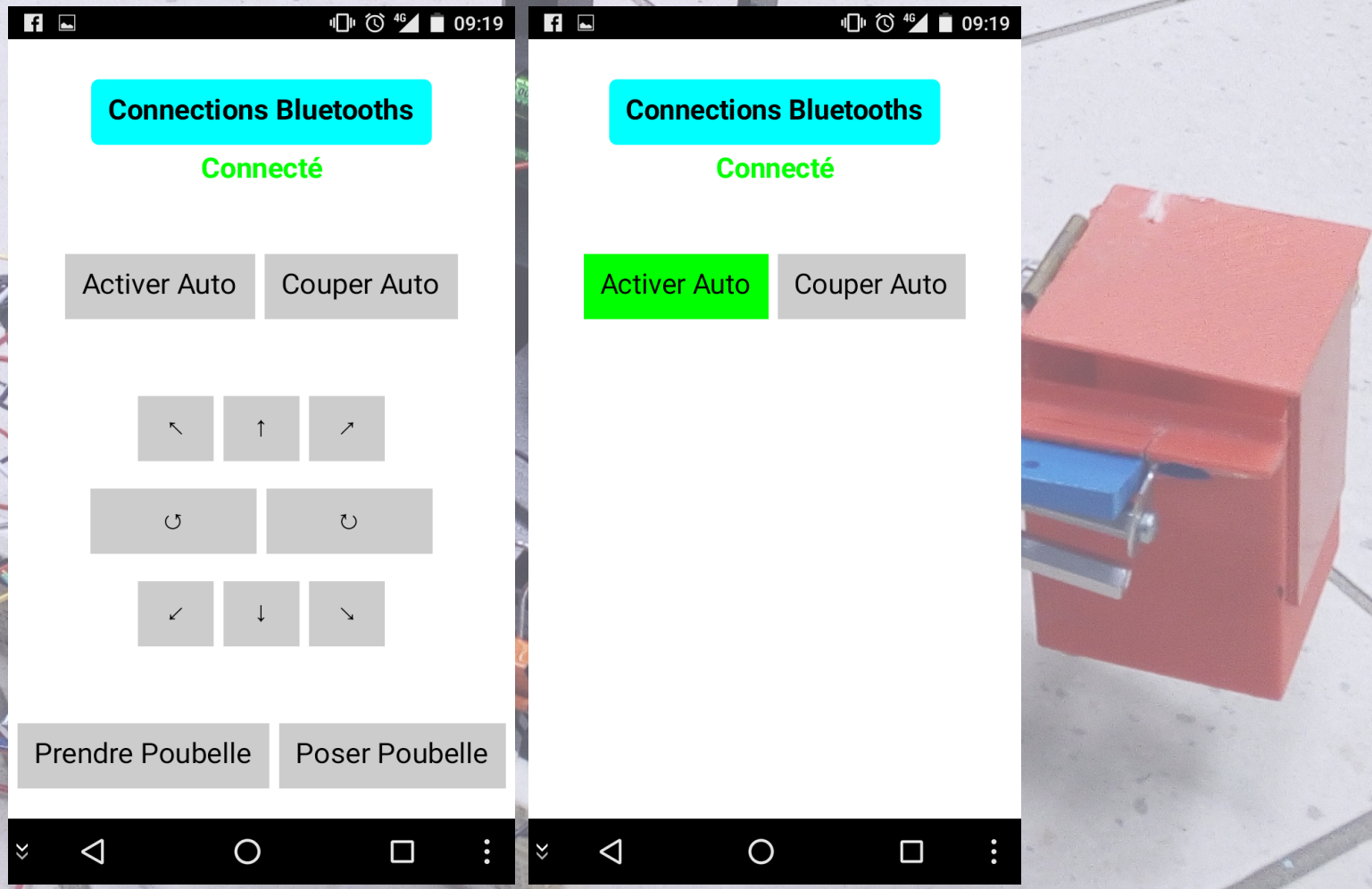




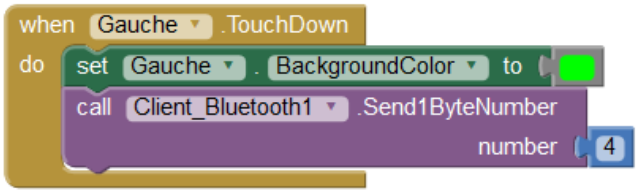


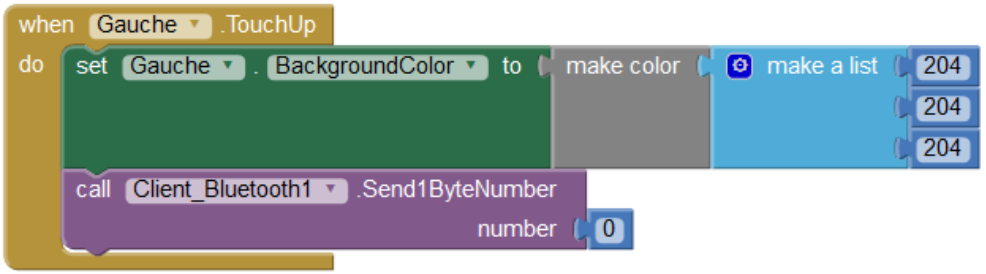


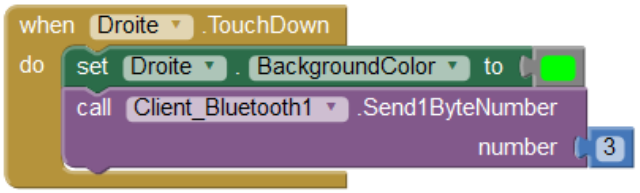
AppInventor 2

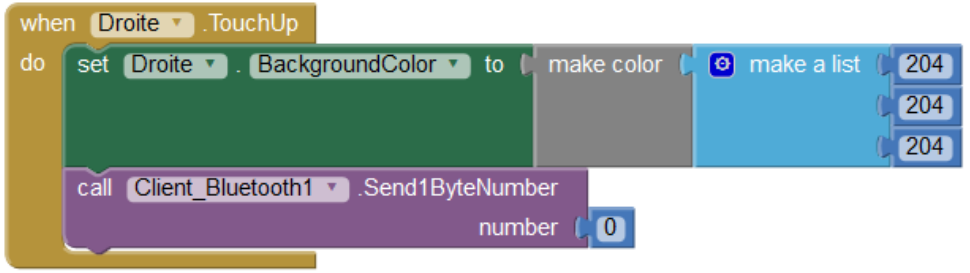


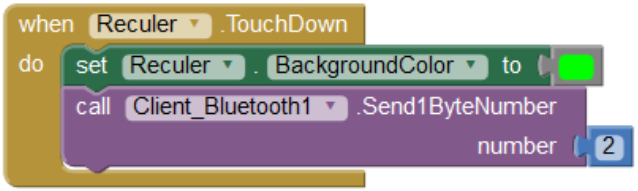
AppInventor 2

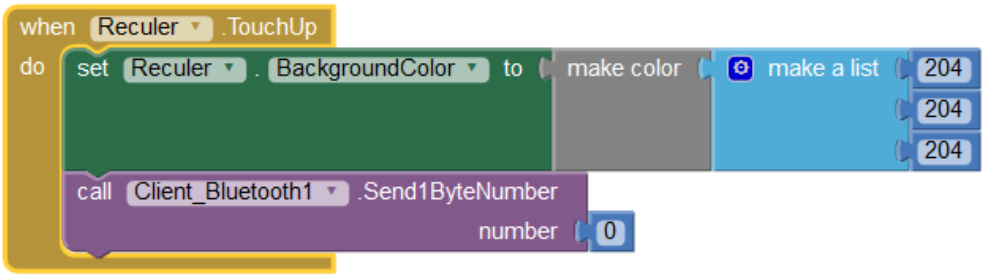




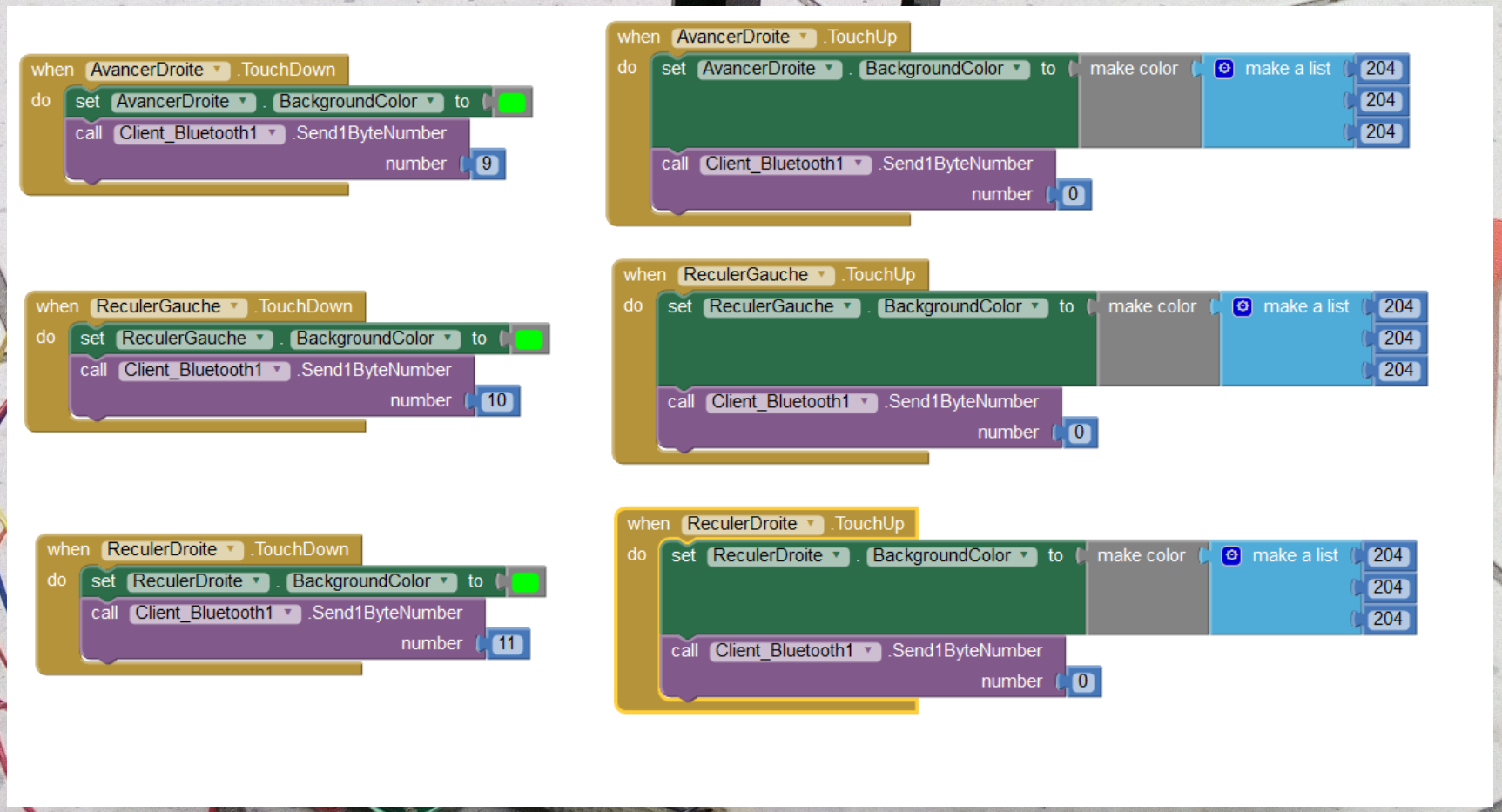




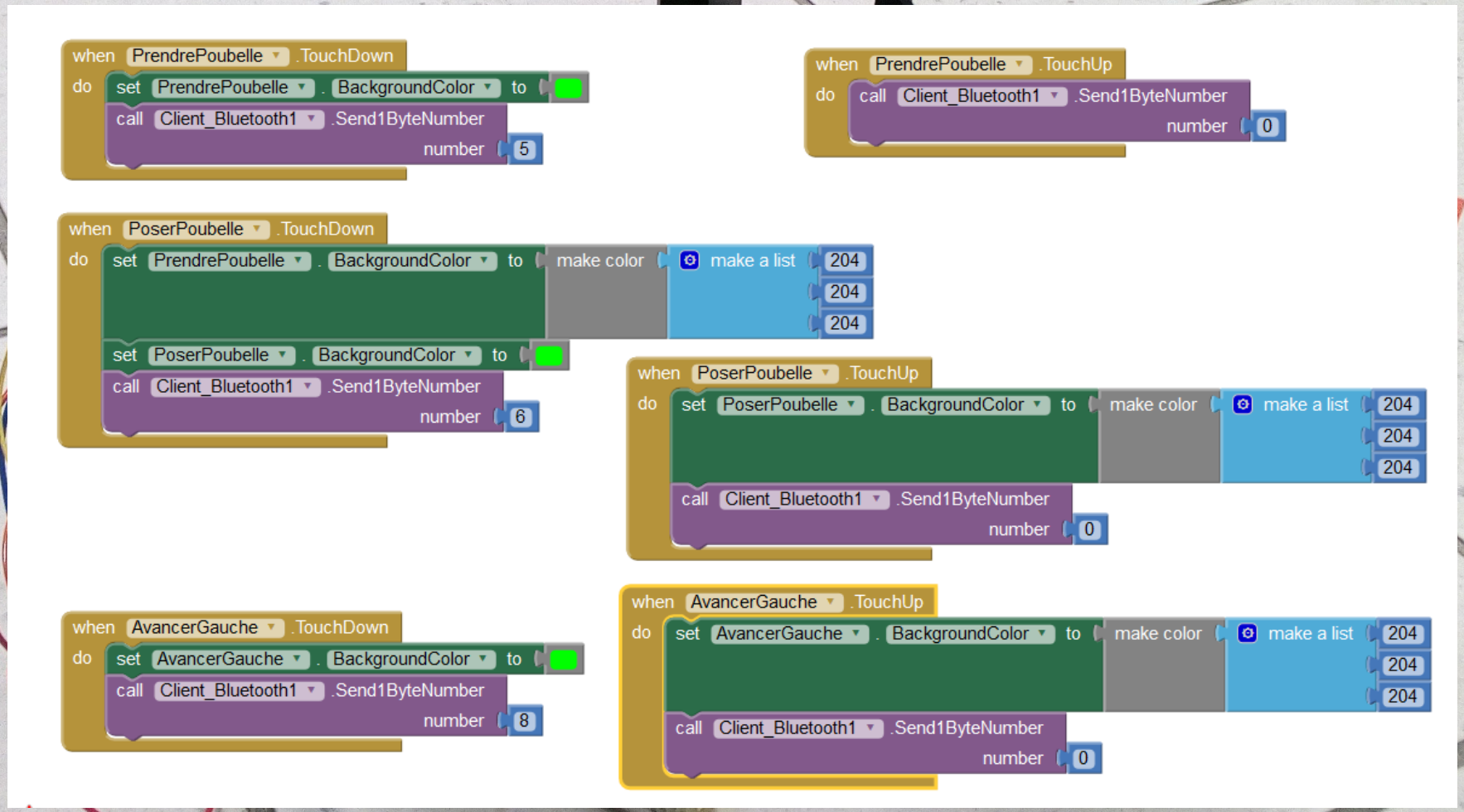




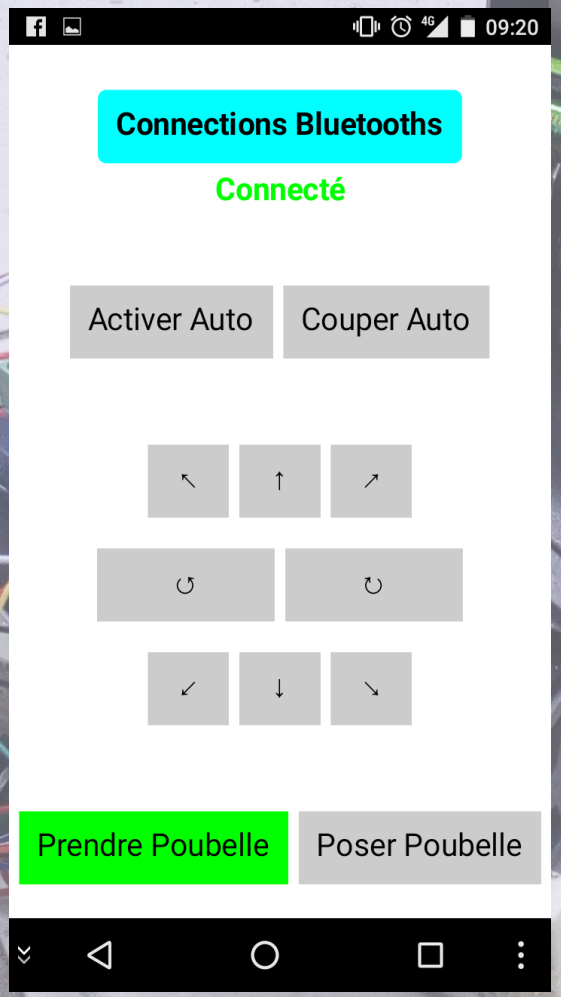
AppInventor 2

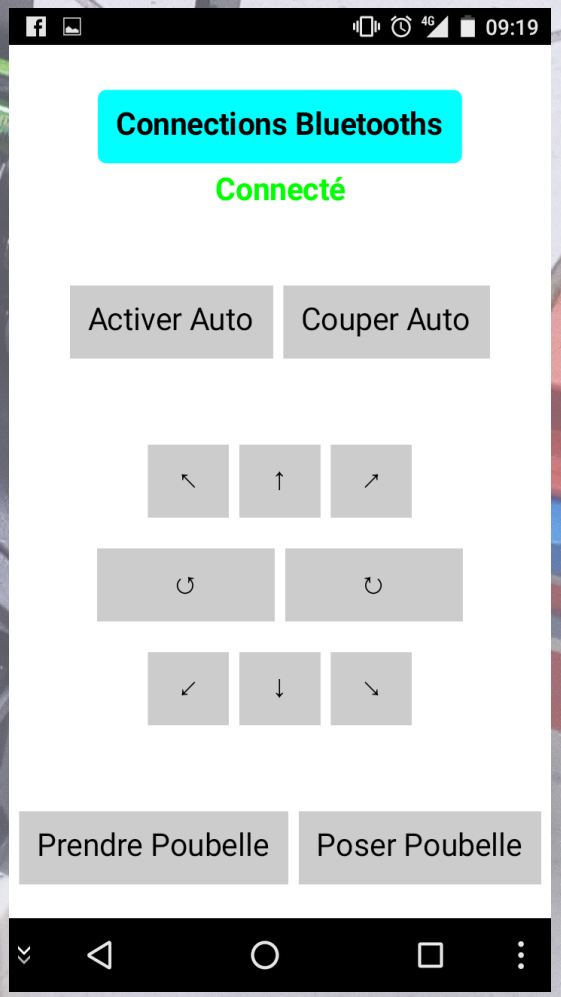


AppInventor 2

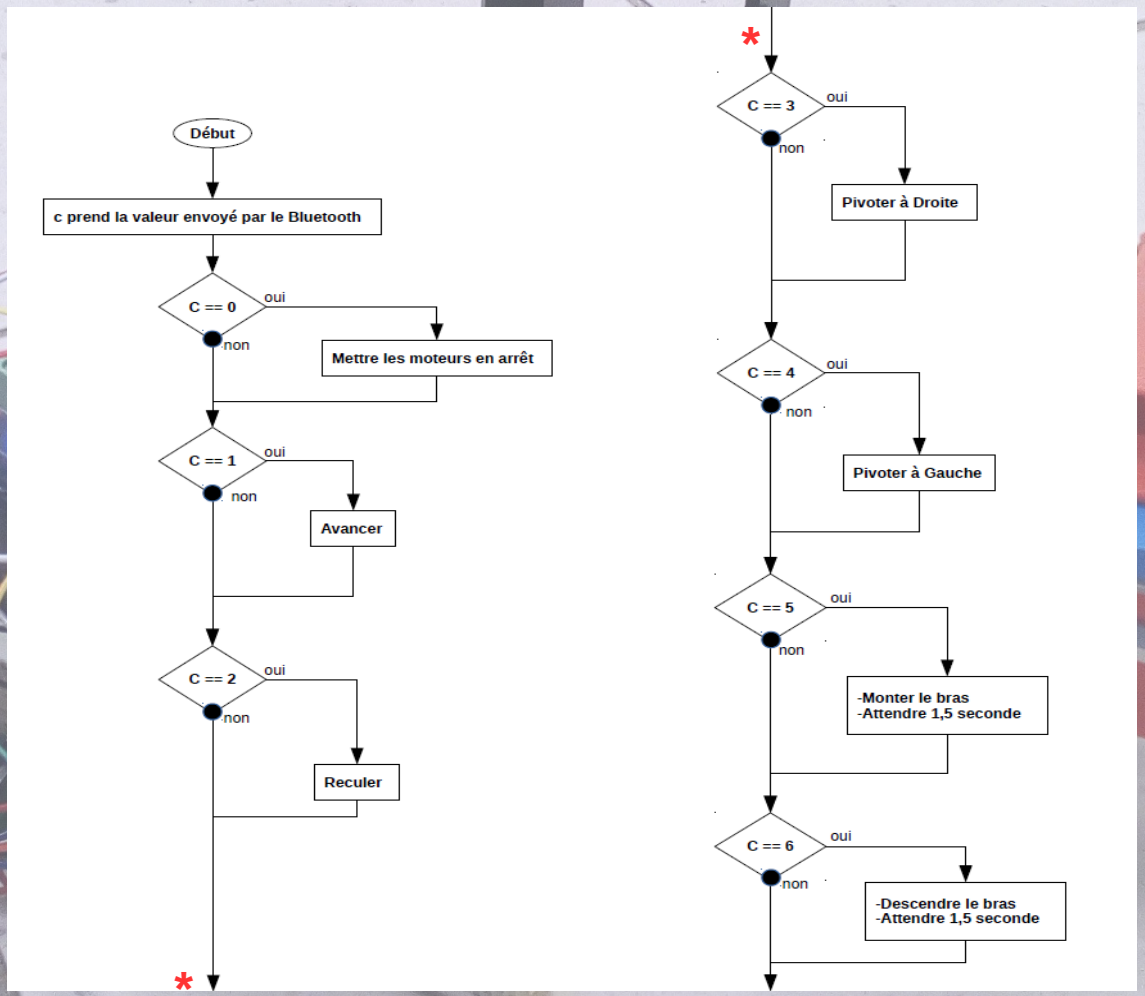


AppInventor 2

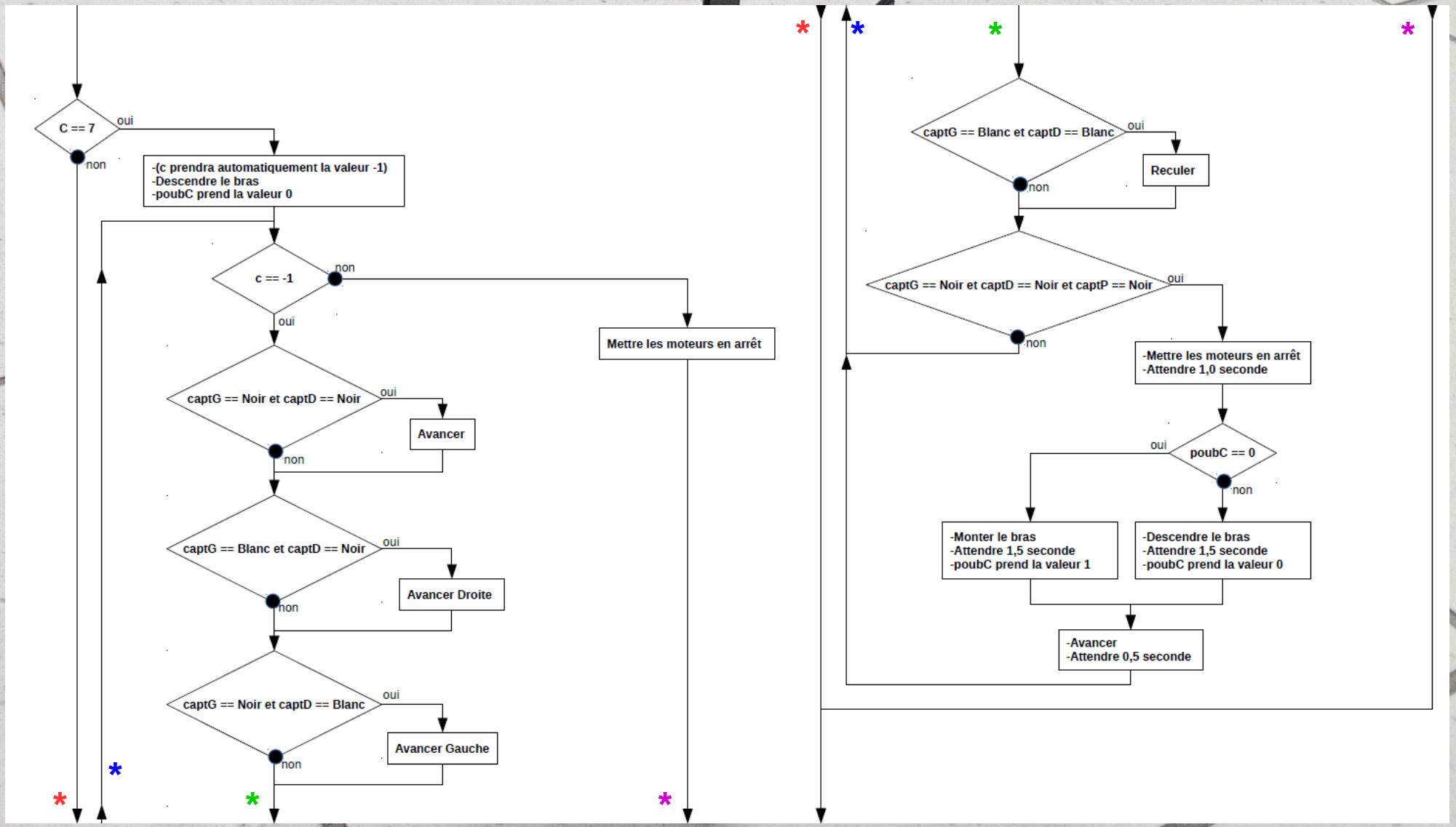




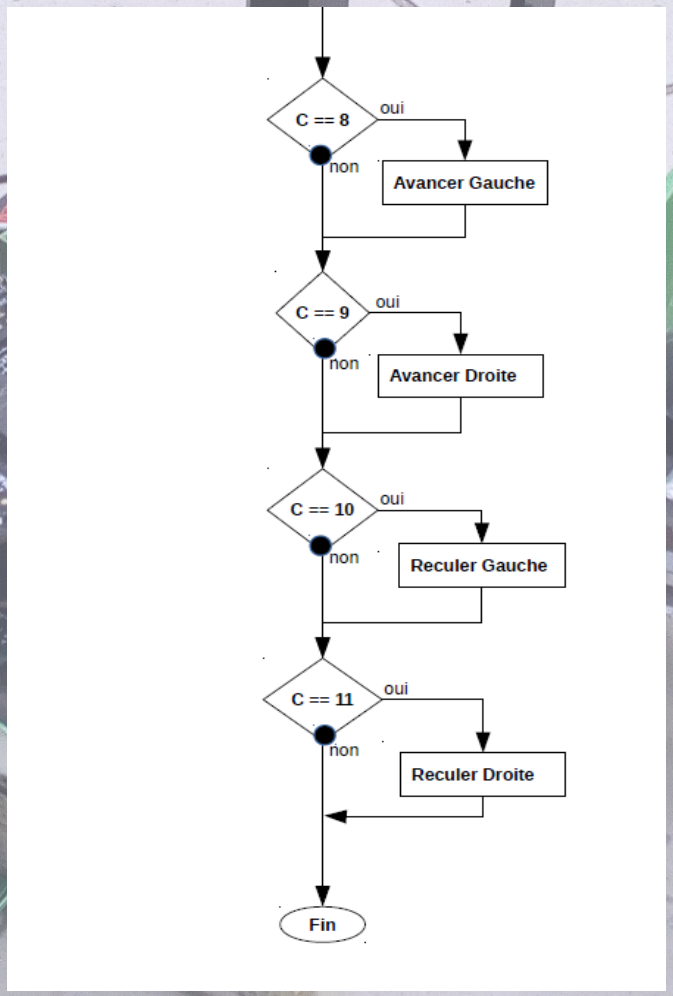
Logigramme ARDUINO



Logigramme ARDUINO

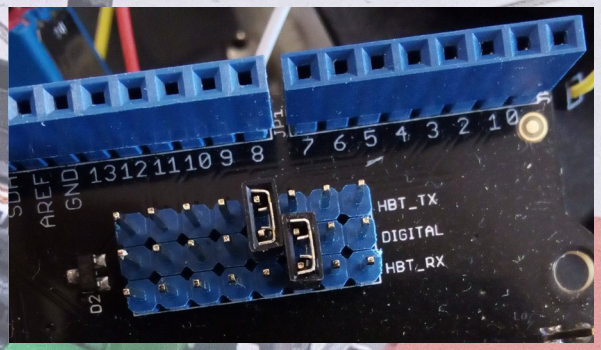


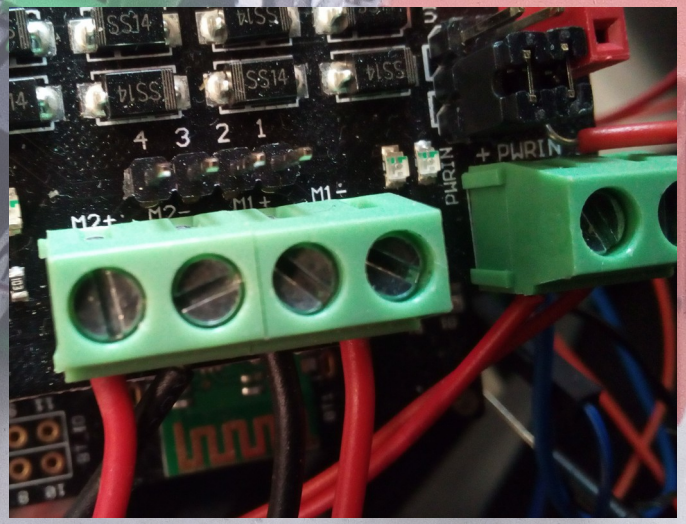
Logigramme ARDUINO



Programme ARDUINO

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

#define RX 3  //pour la carte bluetooth
#define TX 2

int motD = 5;  //branchements des moteurs sur la carte moteur
int motG = 6;
int dirD = 4;
int dirG = 7;

int pinServo=8;  //branchement du servo moteur sur le pin 8

Servo monServo;
SoftwareSerial mySerial(RX, TX);

int c = 0;
int poubC = 0;  //Pour PoubelleCharge, 0 quand la poubelle est au sol

//Creation des fonctions de deplacement
void Avancer();
void Reculer();
void PivoterGauche();
void PivoterDroite();
void AvancerGauche();
void AvancerDroite();
void ReculerGauche();
void ReculerDroite();
void Arret();
```





Programme ARDUINO

```
void setup()
{
  pinMode(motD, OUTPUT);
  pinMode(motG, OUTPUT);
  pinMode(dirD, OUTPUT);
  pinMode(dirG, OUTPUT);

  monServo.attach(pinServo);

  analogWrite(motD, 0);
  analogWrite(motG, 0);  //Met les moteurs en arret

  Serial.begin(9600);
  Serial.println("Debut ");
  mySerial.begin(9600);
  mySerial.print("AT+NAME?");
  delay(1000);

  while (mySerial.available())
  {
    Serial.print(char(mySerial.read()));
  }

  Serial.println();
  mySerial.print("AT+PIN?");
  delay(1000);
  while (mySerial.available())
  {
    Serial.print(char(mySerial.read()));
  }

  Serial.println();
  mySerial.print("AT+UART?");
  delay(1000);
  while (mySerial.available())
  {
    Serial.print(char(mySerial.read()));
  }
  *
```

```
*Serial.println();
  mySerial.print("AT+ROLE?");
  delay(1000);

  while (mySerial.available())
  {
  Serial.print(char(mySerial.read()));
  }
  Serial.println();
  delay(1000);
}
```

Programme ARDUINO

```
void loop()
{
  while (mySerial.available())
  {
    c = int(mySerial.read());
    Serial.println(c);  //A utiliser pour avoir les valeurs de "c"
    switch (c)
    {
      case 0:
        Arret();
        break;

      case 1:
        Avancer();
        break;

      case 2:
        Reculer();
        break;

      case 3:
        PivoterDroite();
        break;

      case 4:
        PivoterGauche();
        break;

      case 5:
        monServo.write(60);
        Serial.println("Monter bras");
        delay(1500);
        break;

      case 6:
        monServo.write(170);
        Serial.println("Descendre bras");
        delay(1500);
        break;
```

```
      case 7:  //Debut du mode automatique
        Serial.println("Debut du mode AUTO");
        c = int(mySerial.read());
        monServo.write(170);
        poubC = 0;
        while (c==-1)  //c a pour valeur -1 ici
        {

          if (analogRead(3)>=400 && analogRead(5)>=400)
          {
            Serial.println("Ligne Droite");
            digitalWrite(dirD, HIGH);
            digitalWrite(dirG, HIGH);
            analogWrite(motD, 150);
            analogWrite(motG, 150);
          }

          if (analogRead(3)<=400 && analogRead(5)>=400)
          {
          Serial.println("Ligne Tourne Droite");
          digitalWrite(dirD, HIGH);
          digitalWrite(dirG, LOW); analogWrite(motD, 150);
          analogWrite(motG, 50);
          }

          if (analogRead(3)>=400 && analogRead(5)<=400)
          {
            Serial.println("Ligne Tourne Gauche");
            digitalWrite(dirD, LOW);
            digitalWrite(dirG, HIGH);
            analogWrite(motD, 50);
            analogWrite(motG, 150 );
          }

          if (analogRead(3)<400 && analogRead(5)<400)
          {
            Serial.println("Pas de Ligne ");
            digitalWrite(dirD, LOW);
            digitalWrite(dirG, LOW);
            analogWrite(motD, 100);
            analogWrite(motG, 100);
          }
```

Programme ARDUINO

```
if (analogRead(1)>=400 && analogRead(3)>=400 && analogRead(5)>=400)
{
  Arret();
  delay(1000);
  if (poubC==0)
  {
    delay(5000);
    monServo.write(60);
    delay(1500);
    poubC=1;
  }

  else if (poubC==1)
  {
    monServo.write(170);
    delay(1500);
    poubC=0;
  }

  digitalWrite(dirD, HIGH);
  digitalWrite(dirG, HIGH);
  analogWrite(motD, 150);
  analogWrite(motG, 150);
  delay(500);
 }

}

Arret();
break;
```

*

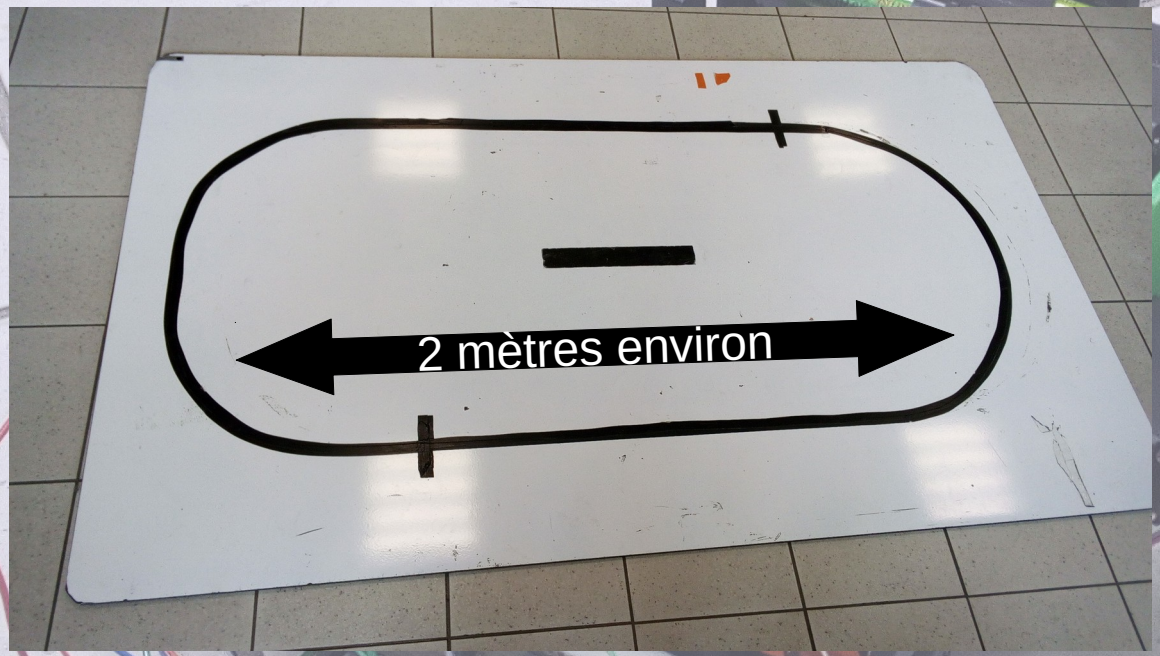
```
*  case 8:
     AvancerGauche();
     break;

   case 9:
     AvancerDroite();
     break;

   case 10:
     ReculerGauche();
     break;

   case 11:
     ReculerDroite();
     break;


  }
 }
}
```

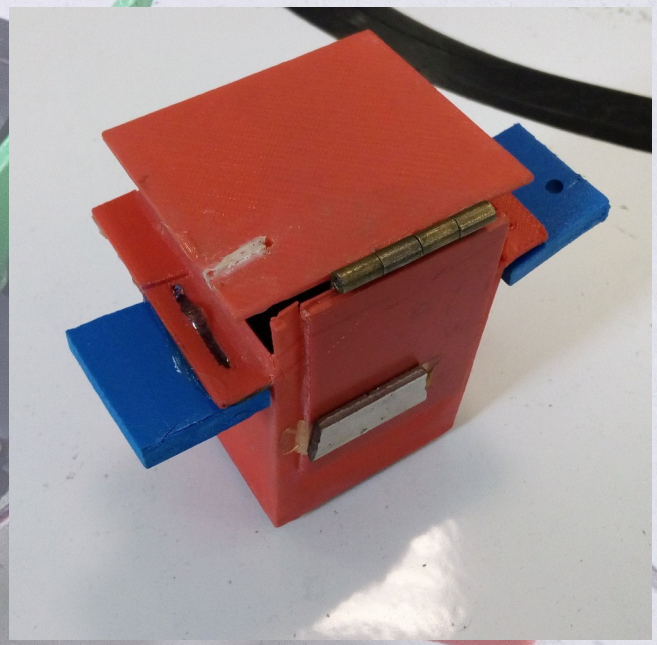
Programme ARDUINO

```
void Avancer()
  {
    Serial.println("Avancer");
    digitalWrite(dirD, HIGH);
    digitalWrite(dirG, HIGH);
    analogWrite(motD, 255);
    analogWrite(motG, 241);
  }

void Reculer()
  {
    Serial.println("Reculer");
    digitalWrite(dirD, LOW);
    digitalWrite(dirG, LOW);
    analogWrite(motD, 255);
    analogWrite(motG, 232);
  }

void PivoterGauche()
  {
    Serial.println("Pivoter Gauche");
    digitalWrite(dirD, HIGH);
    digitalWrite(dirG, LOW);
    analogWrite(motD, 210);
    analogWrite(motG, 200);
  }

void PivoterDroite()
  {
    Serial.println("Pivoter Droite");
    digitalWrite(dirD, LOW);
    digitalWrite(dirG, HIGH);
    analogWrite(motD, 210);
    analogWrite(motG, 200);
  }
```

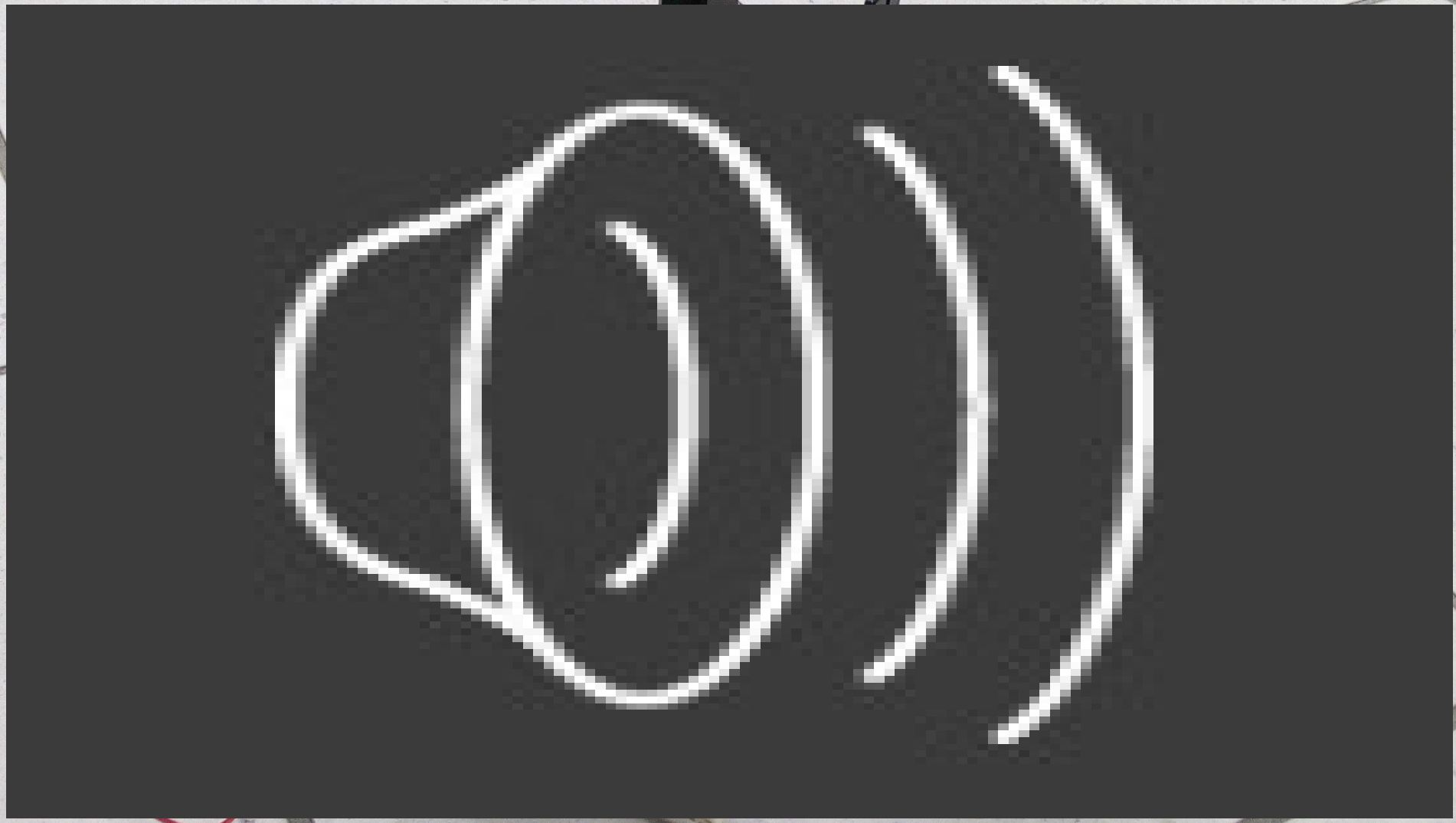
```
void AvancerGauche()
  {
    Serial.println("Avancer Gauche");
    digitalWrite(dirD, HIGH);
    digitalWrite(dirG, HIGH);
    analogWrite(motD, 255);
    analogWrite(motG, 97);
  }

void AvancerDroite()
  {
    Serial.println("Avancer Droite");
    digitalWrite(dirD, HIGH);
    digitalWrite(dirG, HIGH);
    analogWrite(motD, 113);
    analogWrite(motG, 250);
  }

void ReculerGauche()
  {
    Serial.println("Reculer Gauche");
    digitalWrite(dirD, LOW);
    digitalWrite(dirG, LOW);
    analogWrite(motD, 250);
    analogWrite(motG, 90);
  }

void ReculerDroite()
  {
    Serial.println("Reculer Droite");
    digitalWrite(dirD, LOW);
    digitalWrite(dirG, LOW);
    analogWrite(motD, 110);
    analogWrite(motG, 250);
  }

void Arret()
  {
    Serial.println("Arret");
    digitalWrite(dirD, LOW);
    digitalWrite(dirG, LOW);
    analogWrite(motD, 0);
    analogWrite(motG, 0);
  }
```

Matériel pour le test





Robot en mode automatique



Robot en mode manuel