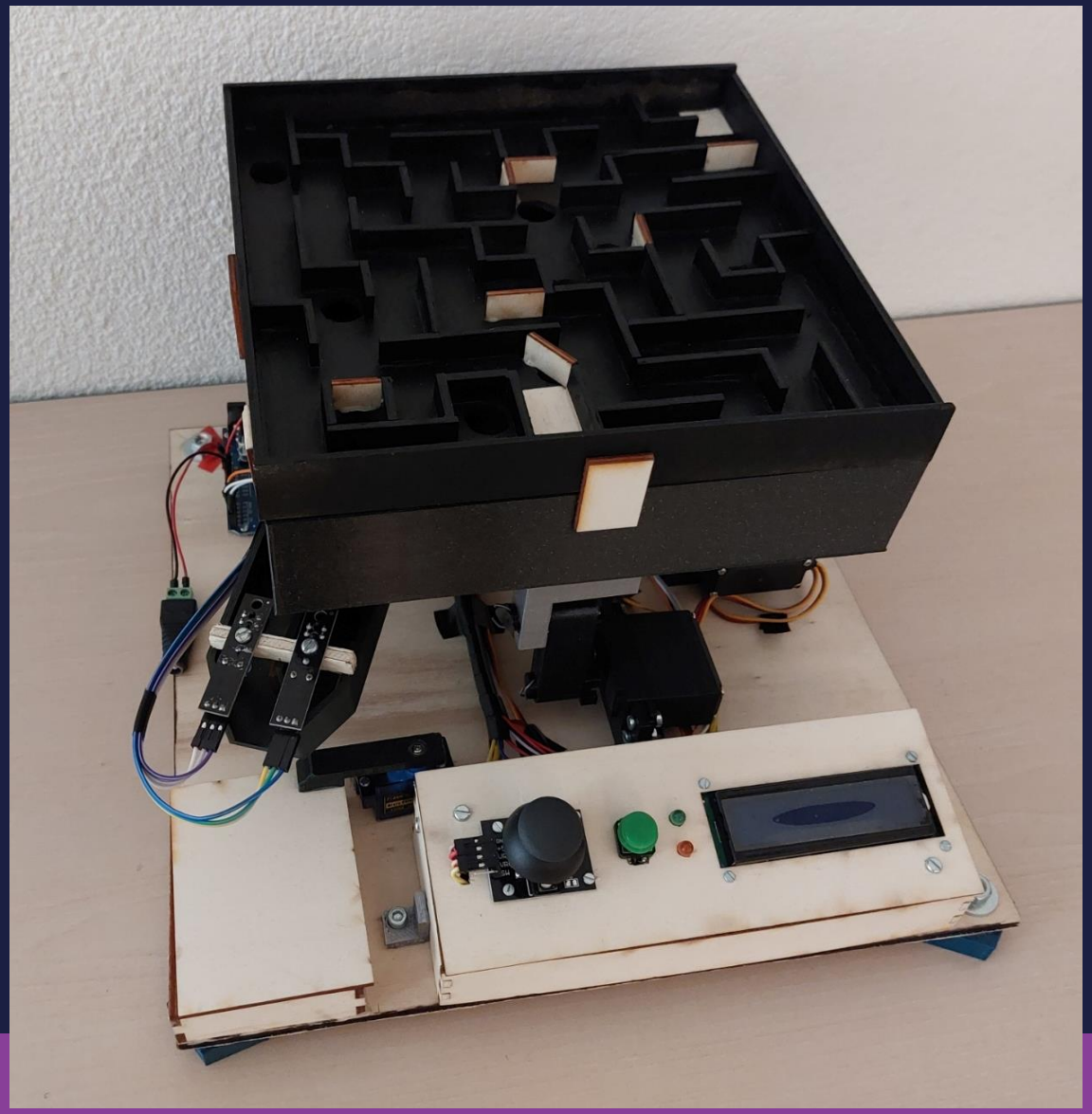
Maze Game

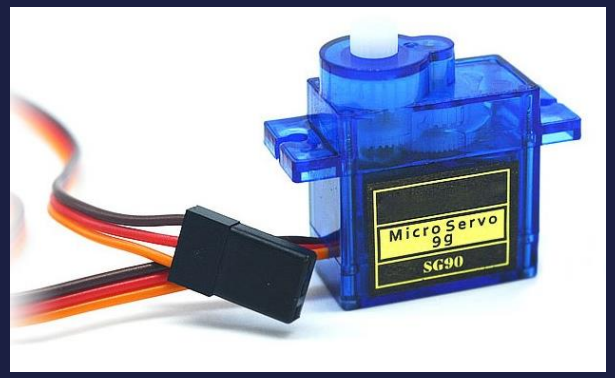
Ortstadt Julius - G4 Peip2

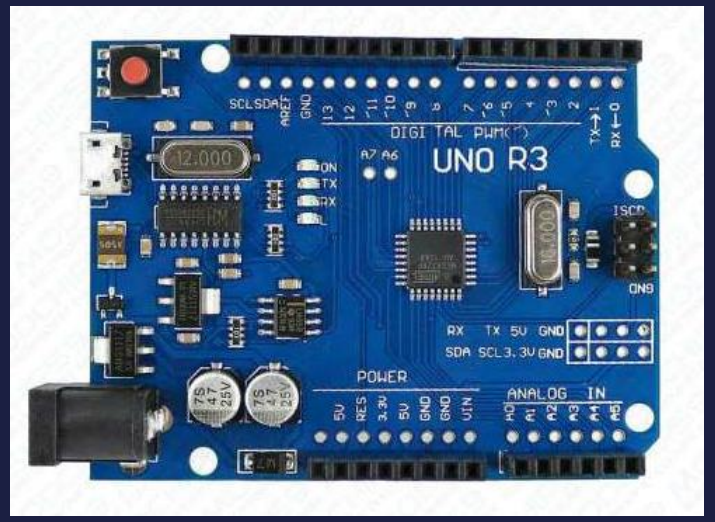
https://github.com/JuliusOrtstadt/Maze_Game

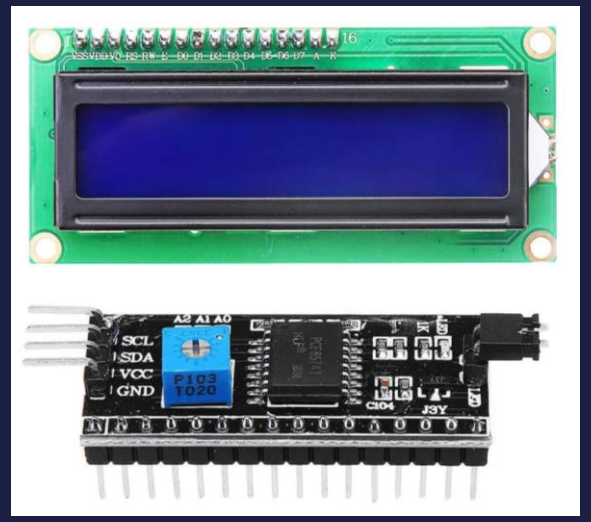
Présentation

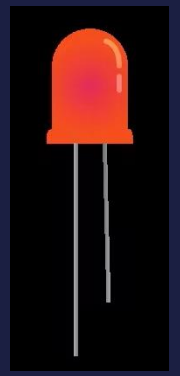


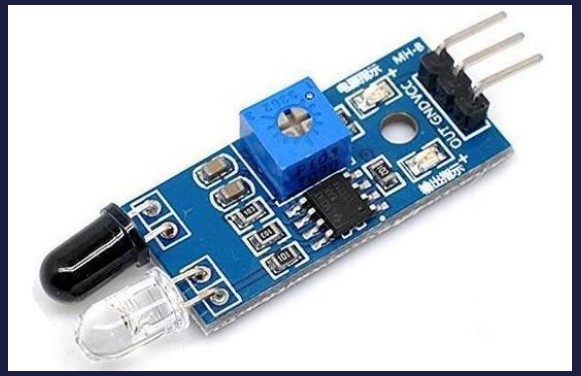
Les composants



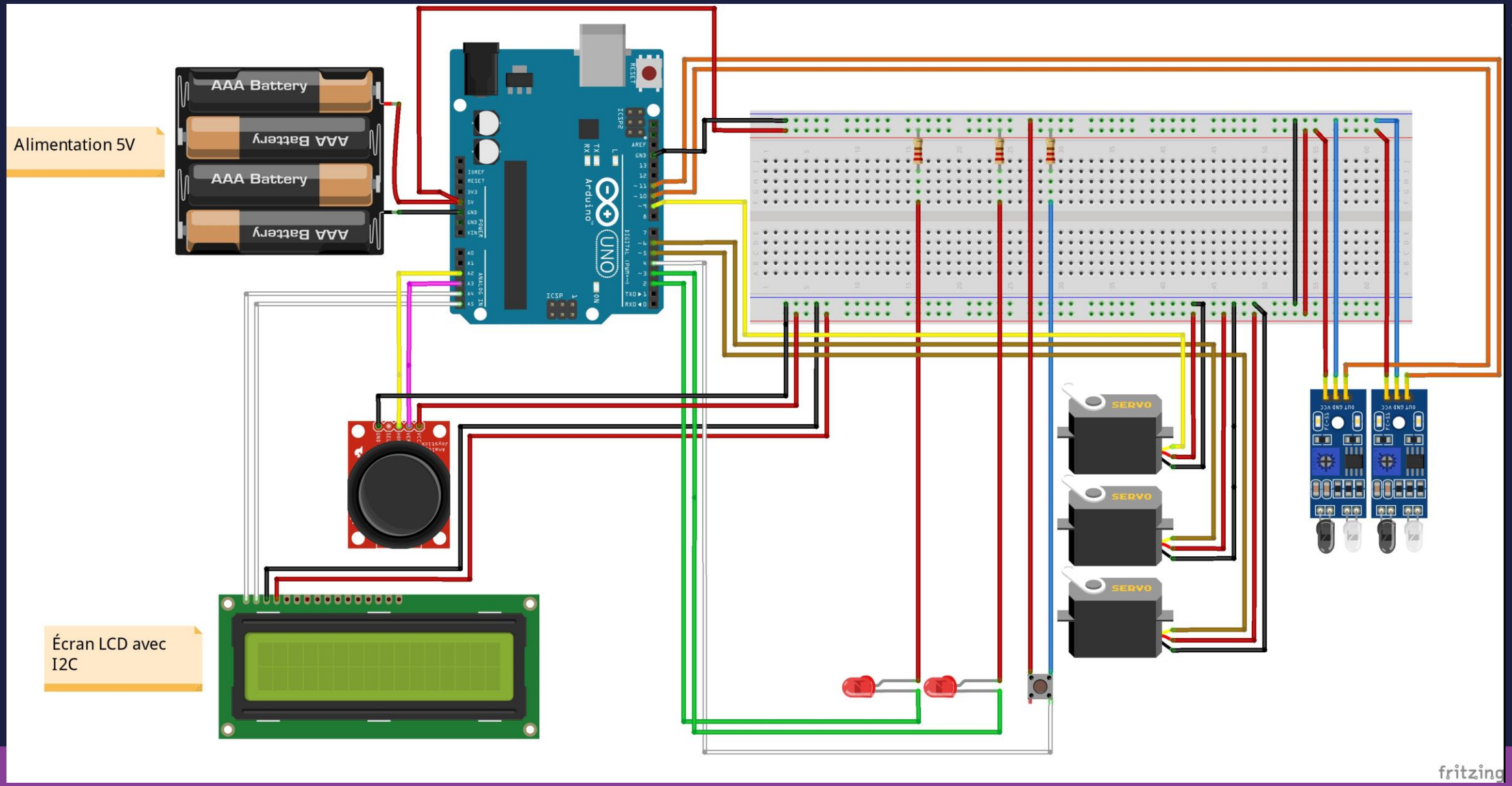








Le branchement



Le code

```
/*--- Fonction pour afficher les messages sur l'écran ---*/
void lcd_msg (String c, int i, int j){
  lcd.setCursor(i,j);
  lcd.print(c);
}

void loop() {
  // put your main code here, to run repeatedly:
  int score = 0;           // Score du joueur -> temps mis pour gagner
  int countdown = 4;       // Compteur pour démarrer la partie

  lcd.clear();
  lcd_msg("Valider pour",0,0);
  lcd_msg("jouer",0,1);

  delay(20);

  if (digitalRead(bouton) == HIGH){
    bouton_app = true;

    lcd.clear();
    lcd_msg("Liberation de ",0,0);
    lcd_msg("la bille",0,1);
    delay(2000);

    ServoPorte.write(70); // Ouverture de la porte pour la bille
```

Le code

```
while(bouton_app == true){
  lcd.clear();
  lcd_msg("Placer bille sur",0,0);    // Étapes à réalisés par le joueur
  lcd_msg("start et valider",0,1);

  delay(20);

  if (digitalRead(bouton) == HIGH){
    lcd.clear();

    /*--- Coutdown pour le début de la partie ---*/
    while (countdown != 0){
      unsigned long currentMillis = millis();       // État de millis actuel
      if (currentMillis - previousMillis > 1000){
        previousMillis = currentMillis;
        countdown--;
        lcd_msg("La partie debute",0,0);
        lcd_msg("dans " + String(countdown) + "s",0,1);
      }
    }
    digitalWrite(LED_V, LOW);
    digitalWrite(LED_O, HIGH);
    game = true;                  // Début de la partie
    bouton_app = false;
  }
}
```

Le code

```
while (game == true){
  unsigned long currentMillis = millis();      // État de millis actuel
  /*---------- Timer ----------*/
  if (currentMillis - previousMillis > 1000){
    previousMillis = currentMillis;
    score++;
  }

  lcd.clear();
  lcd_msg("Score: " + String(score) + "s",0,0);          // Affichage du temps écoulé = score

  /*---------- Contrôle de la partie ----------*/
  int valX = analogRead(joyX);     // Valeur en X du joystick
  int valY = analogRead(joyY);     // Valeur en X du joystick

  // Conversion des valeurs en des valeurs entre 10 et 170
  valX = map(valX, 0, 1023, 170, 10);
  valY = map(valY, 0, 1023, 170, 10);

  // Application de ces valeurs sur les servos
  ServoX.write(valX);
  ServoY.write(valY);
```

Le code

```
// Détection si le joueur perd
if (digitalRead(detecPerd) == LOW){ // Bille détectée du côté gagnant
  digitalWrite(LED_V, HIGH);        // LED verte éteinte
  digitalWrite(LED_O, LOW);         // LED orange allumée

  lcd.clear();
  lcd_msg("Vous avez perdu!",0,0);

  delay(1500);
  ServoPorte.write(posIniPorte);  // Fermeture de la porte pour la bille
  game = false;                   // Arrêt de la partie
}

// Détection si le joueur gagne
if (digitalRead(detecGagn) == LOW){ // Bille détectée du côté gagnant
  digitalWrite(LED_V, HIGH);        // LED verte éteinte
  digitalWrite(LED_O, LOW);         // LED orange allumée

  score = score - 1;                // Tolérance pour que la bille soit acheminée vers le capteur

  lcd.clear();
  lcd_msg("Vous avez gagne",0,0);
  lcd_msg("Score: " + String(score) + "s",0,1);

  delay(1500);
  ServoPorte.write(posIniPorte);  // Fermeture de la porte pour la bille
  delay(2000);
  game = false;                   // Arrêt de la partie
}
```

Problèmes rencontrés et solutions

Démonstration