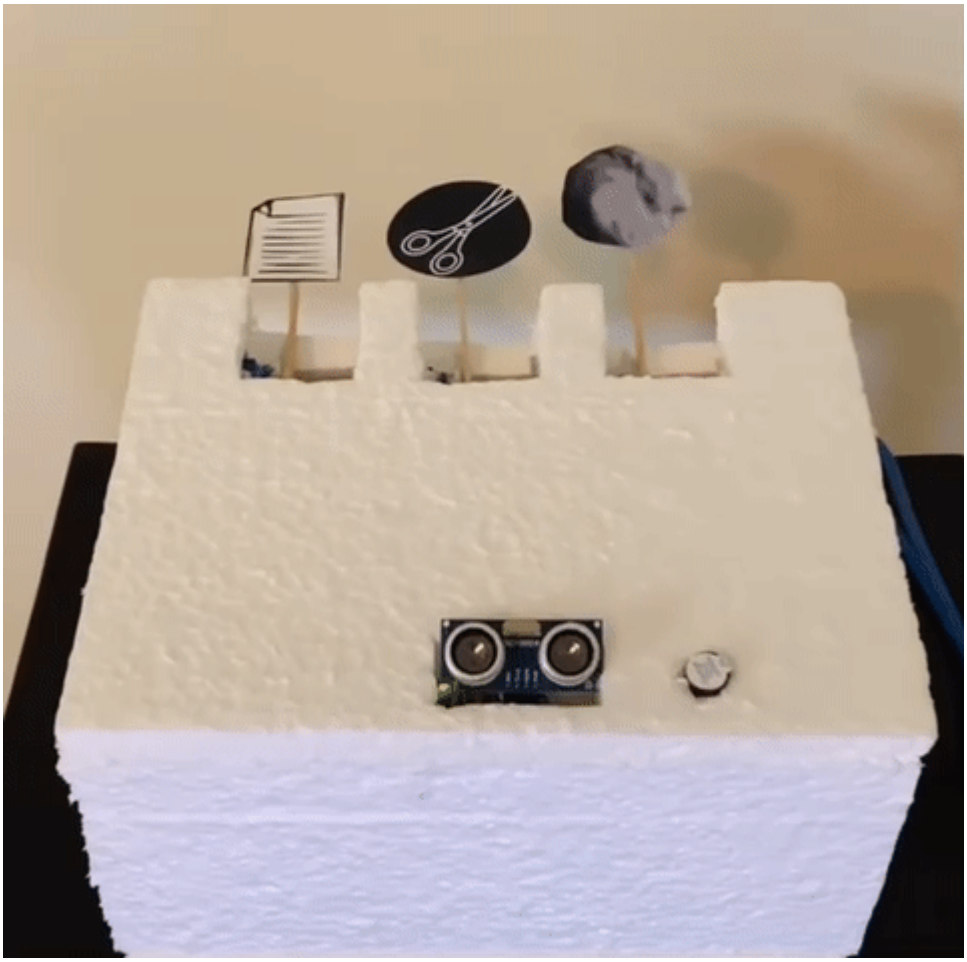
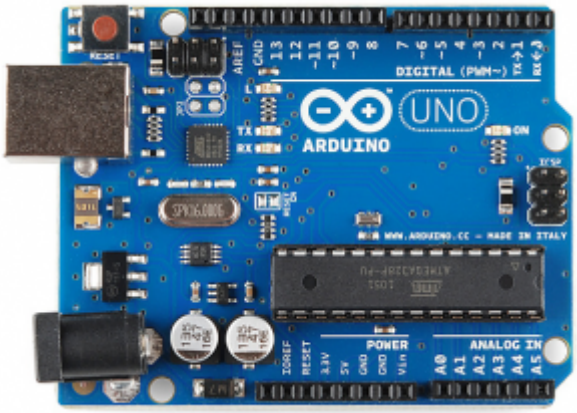
العب حجرة ورقة مقص مع الاردوينو

مقدمة

في هذا الدرس ستتعلم كيف تصنع روبوت يلعب لعبة حجرة ورقة مقص باستخدام الاردوينو وحساس المسافة ومحرك السيرفو.

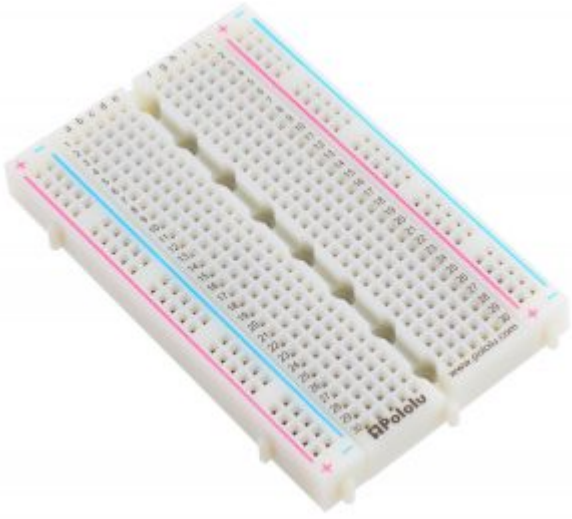


المواد والأدوات



1× اردوينو اونو

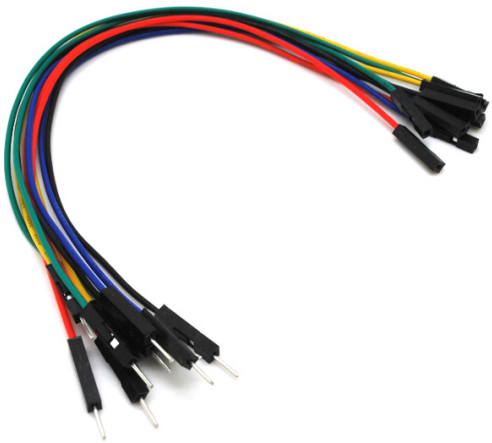
1× سلك الاردوينو



1× لوحة تجارب – حجم وسط

1× مصدر صوت

حزمة أسلاك توصيل (ذكر- ذكر)



حزمة أسلاك توصيل (ذكر – أنثى)

3× محرك سيرفو (TowerPro SG90 micro servo)

1× حساس المسافة (HC-SR04)

1× هيكل اللعبة

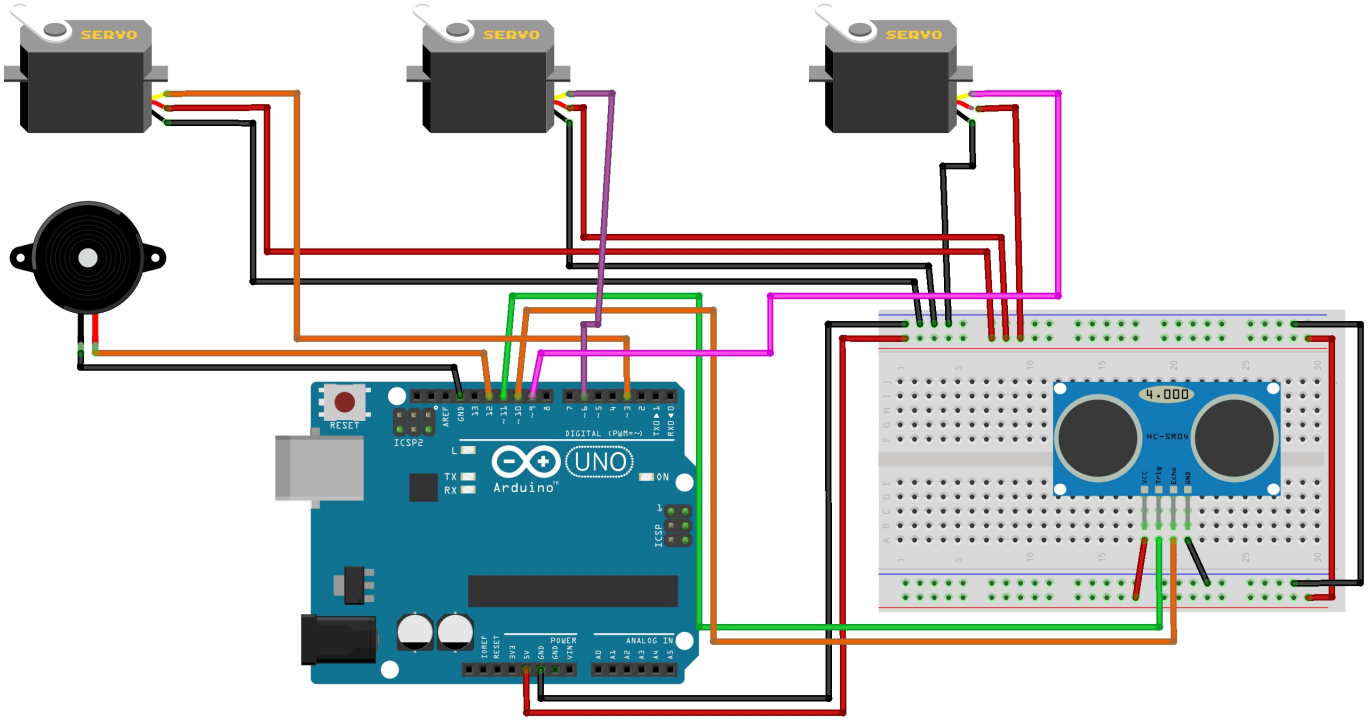
3× أعواد خشبية

1× غراء مسدس

توصيل الدائرة

للمزيد حول محرك السيرفو يمكنك الرجوع للدرس التالي محرك السيرفو.

وللمزيد حول حساس الموجات الفوق صوتية يمكنك الرجوع للدرس التالي حساس الموجات فوق الصوتية.



الهيكل

في هذا الدرس تم استخدام قالب فليني لمحاكاة نموذج لعبة حجرة ورقة مقص.

يمكنك اختيار شكل الهيكل المناسب لك سواء قالب فيليني أو كرتوني أو نموذج بالطابعة ثلاثية الأبعاد أو نموذج خشبي بجهاز الليزر.

بعد ذلك ثبت الدائرة الكهربائية على الهيكل كما في الشكل.

وثبت القصاصات المطبوعة (مقص وورقة وحجرة)على محركات السيرفو يمكنك استخدام أعواد خشبية لتثبيتها.

الكود البرمجي

ارفع الكود البرمجي على لوحة الاردوينو باستخدام برنامج اردوينو (IDE).

```
#include <Servo.h>
volatile long A;
float checkdistance_11_10()
{
digitalWrite(11, LOW);
delayMicroseconds(2);
digitalWrite(11, HIGH);
delayMicroseconds(10);
digitalWrite(11, LOW);
float distance = pulseIn(10, HIGH) / 58.00;
delay(10);
return distance;
}


Servo servo_3;
Servo servo_6;
Servo servo_9;


void setup()
```

```
{
A = 0;
pinMode(11, OUTPUT);
pinMode(10, INPUT);
pinMode(12, OUTPUT);
servo_3.attach(3);
servo_6.attach(6);
servo_9.attach(9);
}

void loop()
{
if (checkdistance_11_10() < 20) {
A = random(0, 4);
switch (A)
{
case 1:
tone(12,131);
delay(100);
noTone(12);
servo_3.write(179);
delay(1000);

servo_3.write(90);
delay(500);
break;


case 2:
tone(12,131);
delay(100);
noTone(12);
servo_6.write(179);
delay(1000);
servo_6.write(90);
delay(500);
break;


case 3:
tone(12,131);
delay(100);
noTone(12);
servo_9.write(179);
delay(1000);
servo_9.write(90);
delay(500);
break;
}
}
}
```

شرح الكود البرمجي

في هذا السطر تم استدعاء مكتبة محرك السيرفو: <Servo.h>.

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

هنا عرفنا المتغير A.

```
volatile long A;
```

في الدالة ()checkdistance_11_10 ستتم قراءة المسافة من حساس المسافة بواسطة المدخل ECHO وتتم معالجتها بالقسمة على 58 بعد ذلك يتم إرسالها إلى المخرج TRIG.

```
float checkdistance_11_10()
{
digitalWrite(11, LOW);
delayMicroseconds(2);
digitalWrite(11, HIGH);
delayMicroseconds(10);
digitalWrite(11, LOW);
float distance = pulseIn(10, HIGH) / 58.00;
delay(10);
return distance;
}
```

في هذا الأسطر تم تعريف المتغيرات الخاصة بمحركات السيرفو.

```
Servo servo_3;
Servo servo_6;
Servo servo_9;
```

في الدالة ()setup تم تعيين القيمة الابتدائية للمتغير A وتساوي صفر.

المنفذ الرقمي 11 سيتم ربطه مع TRIG في حساس المسافة كمخرج.

والمنفذ الرقمي 10 سيتم ربطه مع المدخل ECHO في حساس المسافة كمدخل.

المنفذ الرقمي 12 سيتم ربطه مع مصدر الصوت كمخرج.

المنفذ الرقمي 3 يتم ربطه مع محرك السيرفو الأول servo_3.

المنفذ الرقمي 6 يتم ربطه مع محرك السيرفو الثاني servo_6.

المنفذ الرقمي 9 يتم ربطه مع محرك السيرفو الثالث servo_9.

```
void setup()
{
```

```
A = 0;
pinMode(11, OUTPUT);
pinMode(10, INPUT);
pinMode(12, OUTPUT);
servo_3.attach(3);
servo_6.attach(6);
servo_9.attach(9);
}
```

في الدالة ()loop سيتم قراء المسافة إذا كانت أصغر من 20 فسيتم تحريك محركات السيرفو بشكل عشوائي.

المتغير A سيحمل القيم العشوائية الثلاثة لمحركات السيرفو وسيتم الخلط بينها.

القيمة العشوائية الأولى: سيتم تشغيل مصدر الصوت بنغمة معينة ثم سيتوقف عن العمل وسيعمل محرك السيرفو الأول servo_3 بزاوية 179 ثم سيتوقف ويعمل بزاوية 90 وسيتوقف عن العمل استعدادًا للذهاب للقيمة العشوائية التالية.

القيمة العشوائية الثانية: سيتم تشغيل مصدر الصوت بنغمة معينة ثم سيتوقف عن العمل وسيعمل محرك السيرفو الثاني servo_6 بزاوية 179 ثم سيتوقف ويعمل بزاوية 90 وسيتوقف عن العمل استعدادًا للذهاب للقيمة العشوائية التالية.

القيمة العشوائية الثالثة: سيتم تشغيل مصدر الصوت بنغمة معينة ثم سيتوقف عن العمل وسيعمل محرك السيرفو الثالث servo_9 بزاوية 179 ثم سيتوقف ويعمل بزاوية 90 وسيتوقف عن العمل استعدادًا للذهاب للقيمة العشوائية التالية.

```
void loop()
{
if (checkdistance_11_10() < 20) {
A = random(0, 4);
switch (A)
{
case 1:
tone(12,131);
delay(100);
noTone(12);
servo_3.write(179);
delay(1000);

servo_3.write(90);
delay(500);
break;


case 2:
tone(12,131);
delay(100);
noTone(12);
servo_6.write(179);
delay(1000);
servo_6.write(90);
delay(500);
break;


case 3:
```

```
tone(12,131);
delay(100);
noTone(12);
servo_9.write(179);
delay(1000);
servo_9.write(90);
delay(500);
break;
}
}
}
```

يمكنك اختبار اللعبة بعد رفع الكود البرمجي.

لا تنسَ فصل مصدر الطاقة بعد الانتهاء من استخدام النظام.