





## レッスン4 障害物回避車

---




### セクションのポイント

学習の喜びは、あなたの車を制御する方法だけでなく、あなたの車を守る方法も知っています。  
だから、あなたの車を衝突から遠ざけてください。

学習パーツ:

-  超音波モジュールを組み立てる方法を学ぶ
-  ステアリングの使用に慣れる
-  車の回避の原則について学ぶ
-  プログラムで障害物回避車を実現する

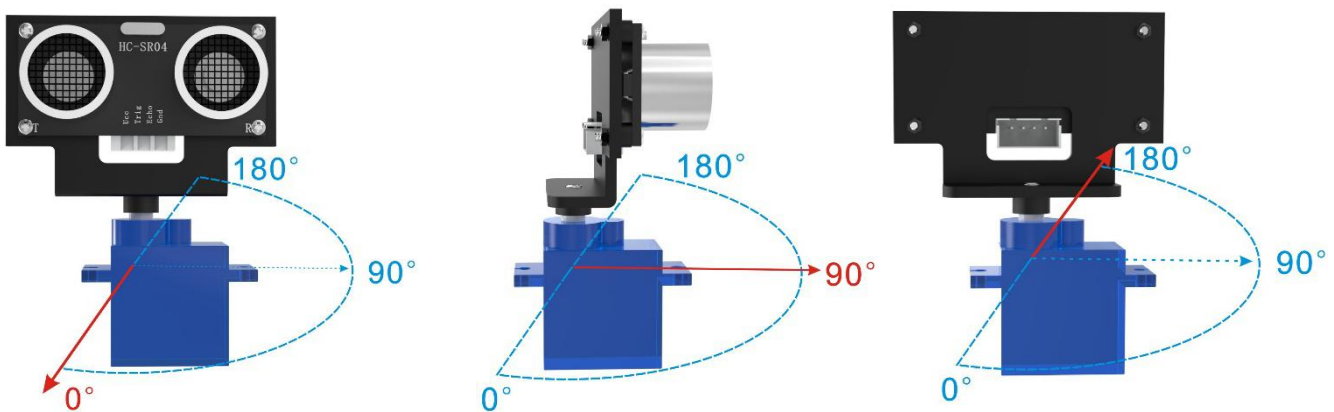
準備:

-  カー（バッテリー付き）
-  USB ケーブル
-  超音波クレードルヘッドのスーツ

## I. コネクション

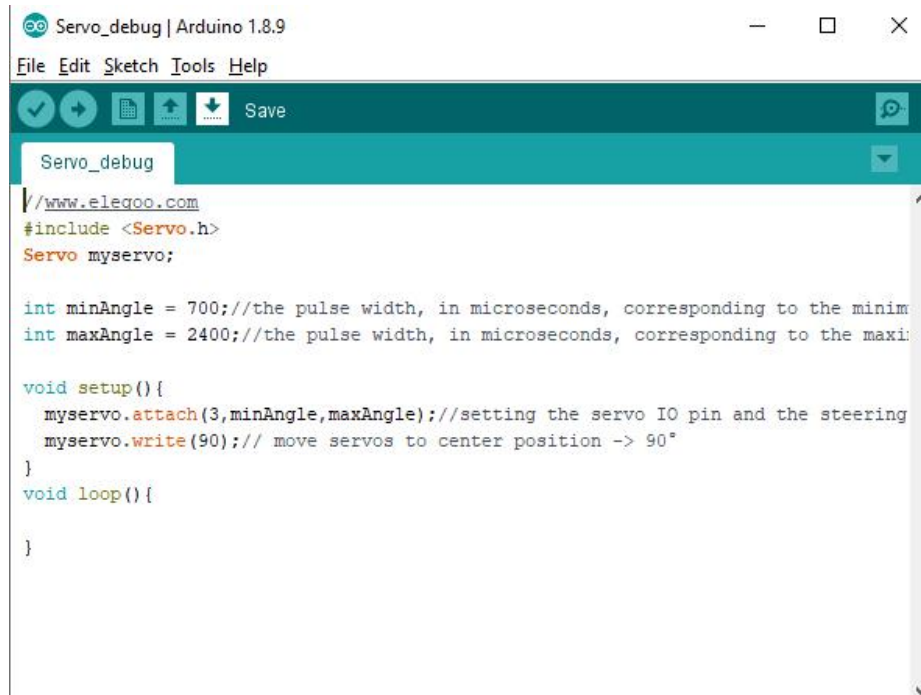
**ヒント：** 弊社の製品は製造時に正確に修正されているため、サーボと超音波モジュールを取り外さない場合は、次の「接続」手順を無視できます。

超音波センサーモジュールホルダーを組み立てるときは、サーボも 180 度回転できるようにサーボをデバッグする必要があります。



**ステップ 1：** UNO をコンピューターに接続し、パス “¥Lesson 4 Obstacle Avoidance Car¥Servo\_debug¥ Servo\_debug. ino” で Servo\_debug コードファイルを開きます。





```

Servo_debug | Arduino 1.8.9
File Edit Sketch Tools Help

Servo_debug

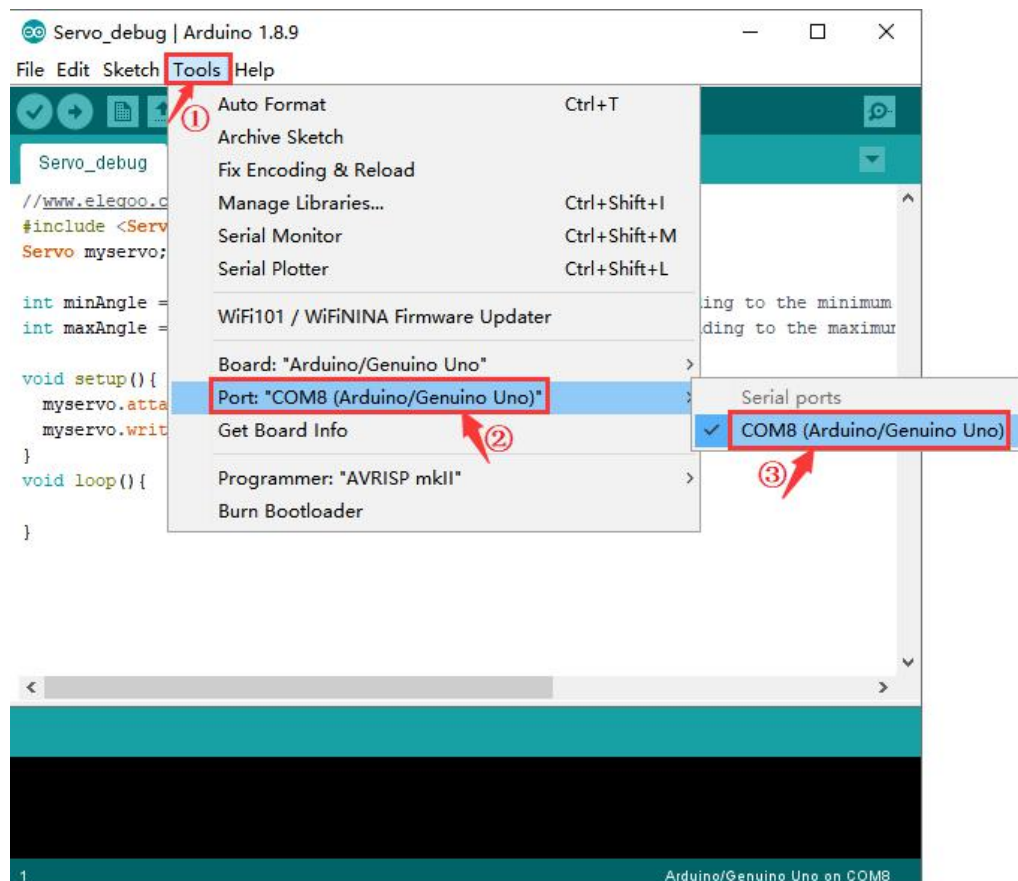
//www.elegoo.com
#include <Servo.h>
Servo myservo;

int minAngle = 700;//the pulse width, in microseconds, corresponding to the minimum
int maxAngle = 2400;//the pulse width, in microseconds, corresponding to the maximum

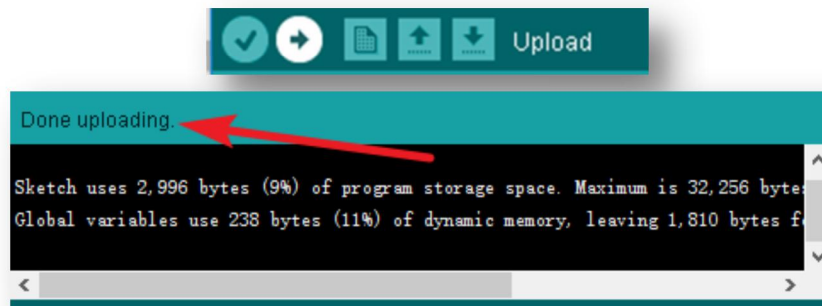
void setup() {
  myservo.attach(3,minAngle,maxAngle);//setting the servo IO pin and the steering
  myservo.write(90);// move servos to center position -> 90°
}
void loop() {

}
  
```

ステップ2: Arduino IDE で[Tool]-> [Port]と[Board]を選択します。



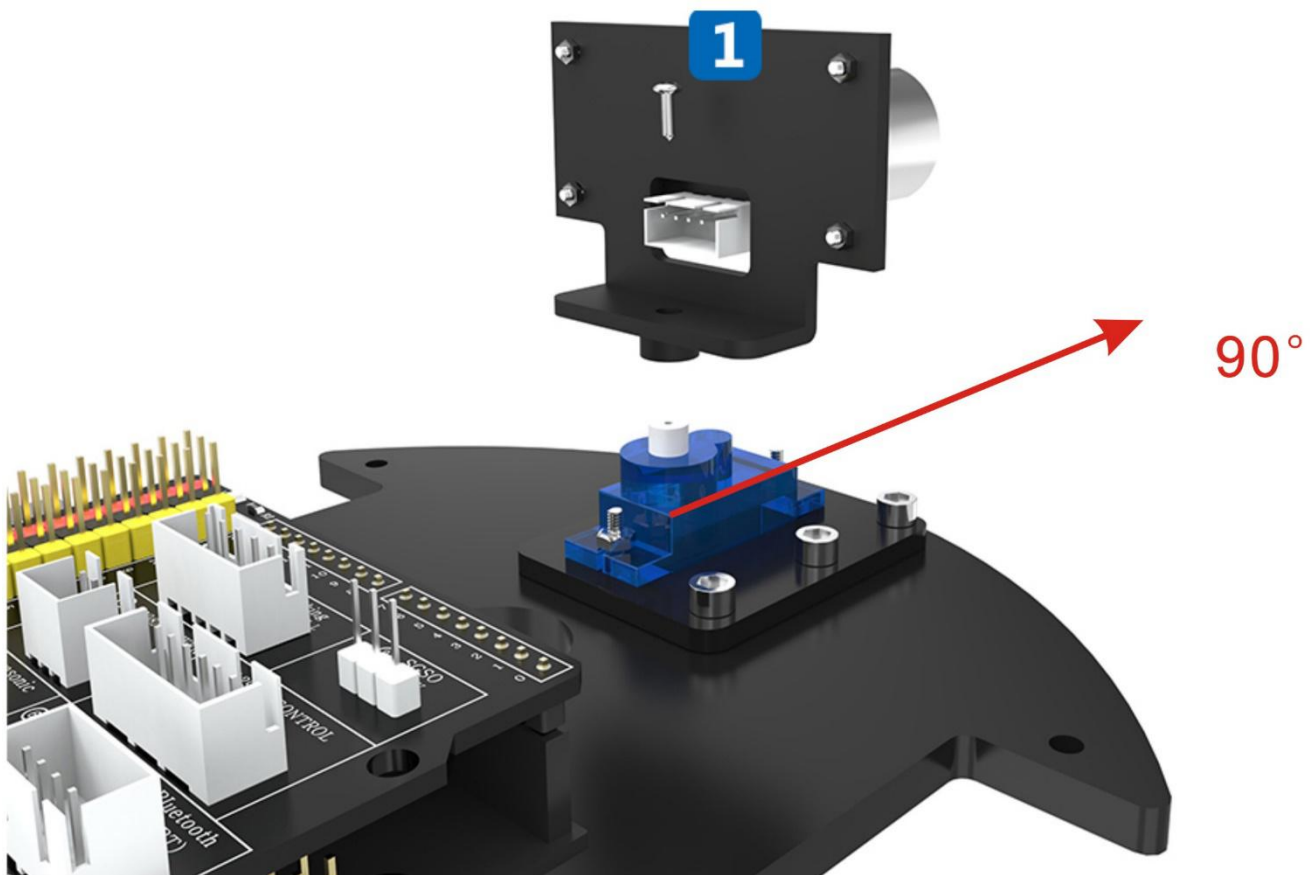
ステップ 3: 矢印ボタンをクリックして、コードを UNO コントローラボードにアップロードします。



アップロードが完了すると、サーボが 90 度回転して静止します。

ステップ 4 : 超音波センサーモジュールを 90 度で組み立てます。

マイクロサーボの全ての歯の角度は 15 度で、90 度の方向の真ん中に取り付けると、左右に 15 度に回転します。つまり、マイクロサーボの実際の取り付け角度は 85 度または 105 度です。



## ご注意：サーボモーターに関するよくある質問

1 電源を入れるたびにマイクロサーボが反時計回りに 15 度回転するのはなぜですか？

これは SG90 マイクロサーボの正常な動作であり、プログラムでの通常の使用には影響しません。

プログラムで制御しなかった場合は、電源を入れる前に、手で回転させるまたはマイクロサーボに接続されている配線を外してください。

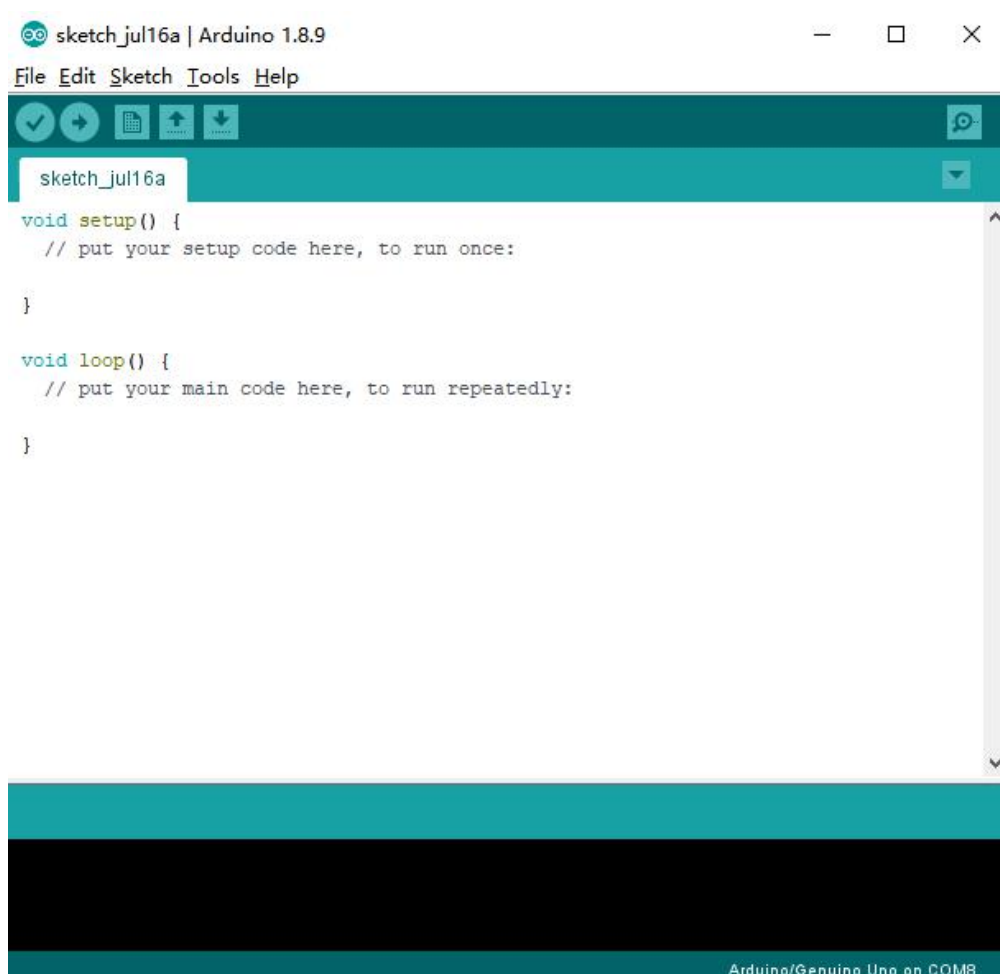
2 なぜマイクロサーボは制御不能で回転し続けるのですか？

「myservo.attach (3, 700, 2400)」を使用して、マイクロサーボに 0 から 180 の範囲の角度を指定します。範囲を超えると、マイクロサーボはこの角度を認識せず、回転を続けます。

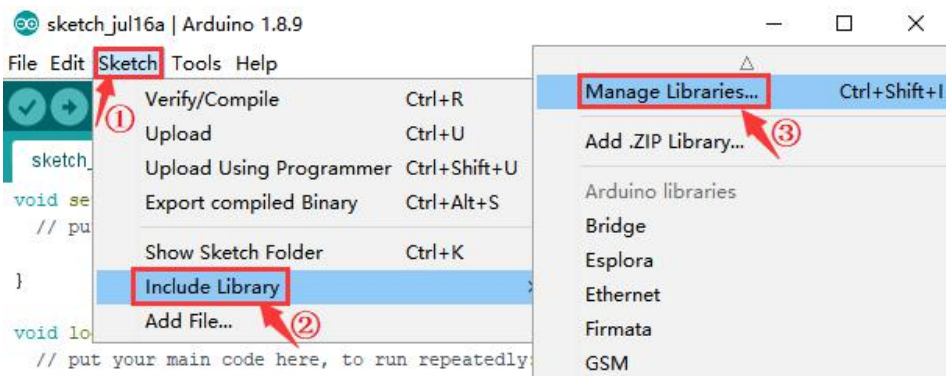
## Ⅱ. プログラムをアップロード

プログラムはライブラリ<servo.h>を使用するため、まずはライブラリをインストールする必要があります。

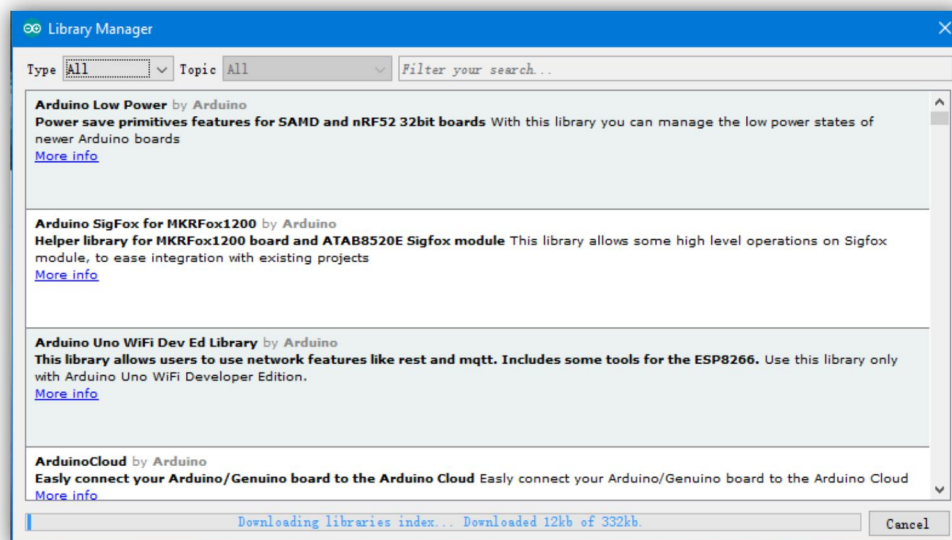
Arduino ソフトウェアを開く



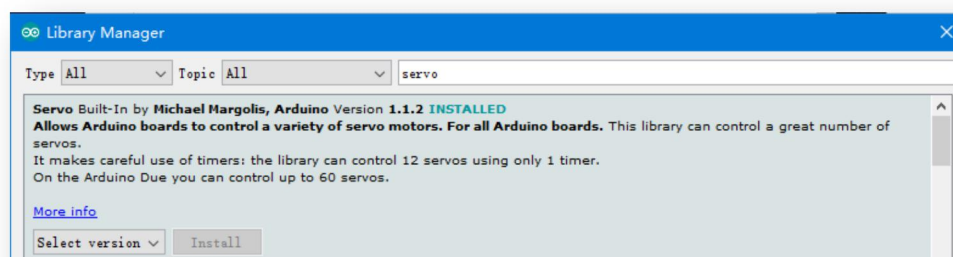
[Sketch]-> [Include Library]-> [Manage Libraries]を選択します



「Downloading libraries index」が完了するまで待機してください。

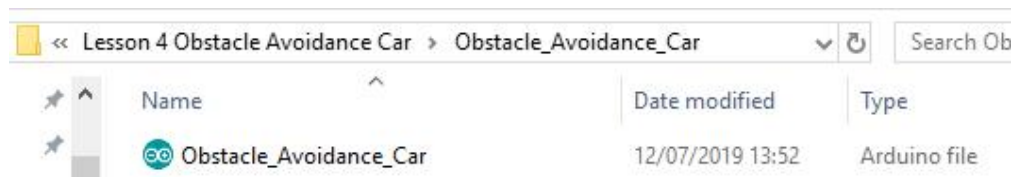


最新バージョンのサーボを検索してインストールしてください。 次の画像は、Servo ライブラリがすでにインストールされていることを示しています。





UNO コントローラボードをコンピュータに接続し、パス “¥Lesson 4 Obstacle Avoidance Car¥Obstacle\_Avoidance\_Car¥Obstacle\_Avoidance\_Car. ino” のコードファイルを開きます。プログラムを UNO ボードにアップロードします。



プログラムを UNO コントロールボードにアップロードした後、ケーブルを外し、車を地面に置き、電源をオンにします。

車が前進し、クラウドプラットフォームが回転を続け、距離測定センサーが継続的に動作することが観測できます。前方に障害物がある場合、クラウドプラットフォームは停止し、車両は障害物を迂回するように方向を変えます。障害物を迂回した後、クラウドプラットフォームは再び回転し続け、車も移動します。

### Ⅲ. 原理の紹介

まず、SG90 サーボについて学びましょう：

#### SG90 Servo



**180 angle steering gear**

**Rotation angle is from 0 to 180**

**Brown line —GND**

**Red line —SV**

**Orange line —signal(PWM)**

分類：180 ステアリングギア

サーボには通常に 3 つの制御ラインがある：電源、グランド、信号

サーボピンの定義：brown line—GND, red line—5V, orange—signal.

### サーボの仕組み：

サーボの信号変調チップがコントローラーボードから信号を受信すると、サーボは基本的な DC 電圧を取得します。サーボ内部には基準電圧を生成するリファレンス回路もあります。これらの 2 つの電圧は互いに比較され、その差を出力されます。次に、モーターチップが差を受け取り、回転速度、方向、角度を決定します。2 つの電圧に差がない場合、サーボは停止します。

### サーボの制御方法：

サーボの回転を制御するには、時間パルスを約 20 ミリ秒、高レベルのパルス幅を約 0.5 ミリ秒～2.5 ミリ秒にする必要があります。これは、サーボの制限角度と一致しています。

たとえば 180° のサーボを例にとると、対応する制御関係は以下のとおりです：

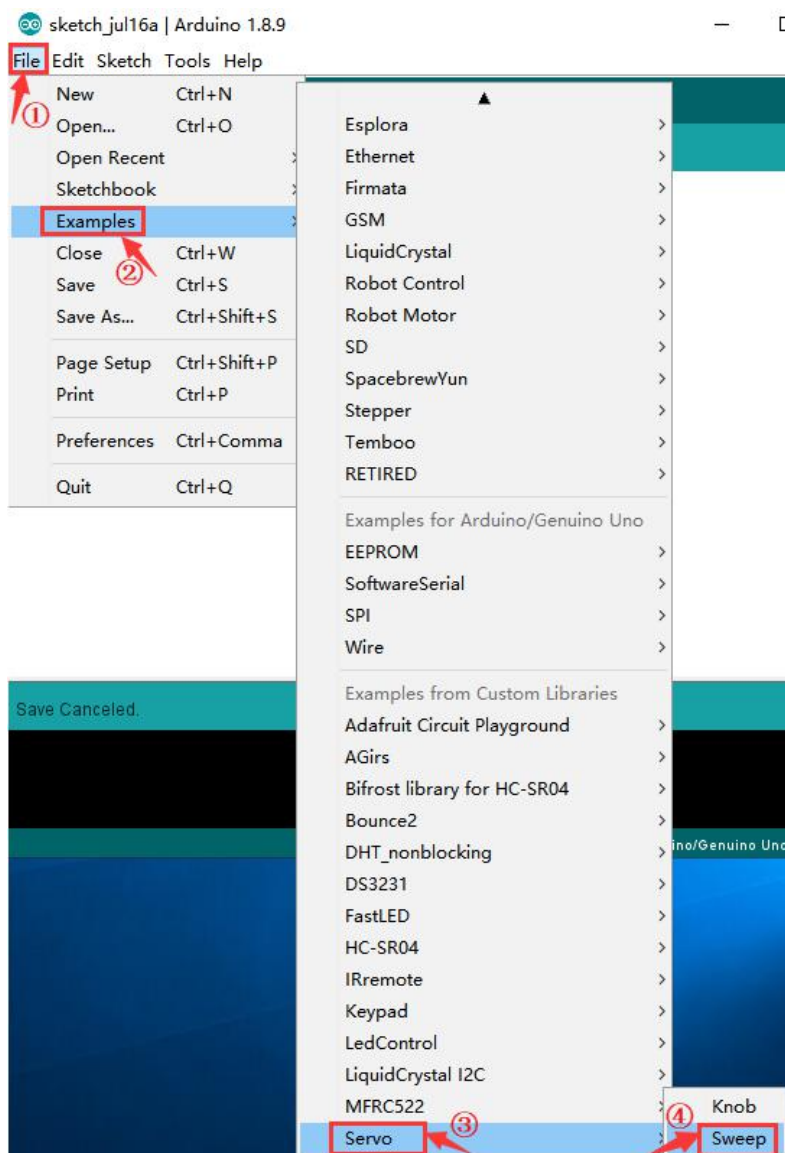
0.5ms	0 degree
1.0ms	45 degree
1.5ms	90 degree
2.0ms	135 degree
2.5ms	180 degree



## サンプルプログラム:

Arduino IDE を開いて次の手順で選択します。

**“File->Examples->Servo->Sweep”**



次に、超音波センサーモジュールを見てみましょう。



## モジュールの特徴：テスト距離、高精度モジュール

**製品の応用：** ロボットの障害物回避、物体の距離テスト、液体試験テスト、公共安全の監督、駐車場試験

## 主な技術パラメータ

- (1) : 使用電圧: DC---5V
- (2) : 静電流: 2mA 未満
- (3) : レベル出力: 5V 以上
- (4) : レベル出力: 0 以下
- (5) : 検出角度: 15 度以下
- (6) : 検出距離: 2cm-450cm
- (7) : 高精度: 最大 0.2cm

ラインの接続方法 : VCC、trig（制御終了）、echo（受信終了）、GND

## モジュールの仕組み:

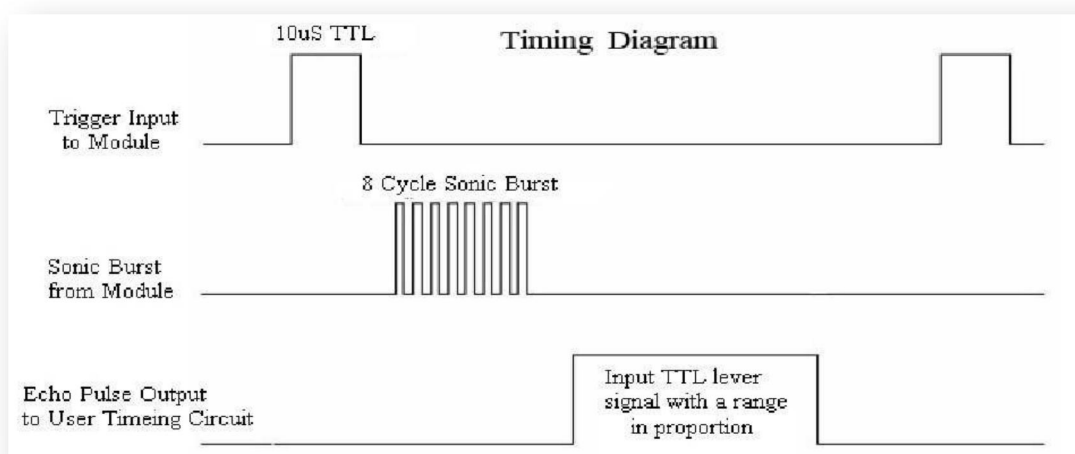
- (1) TRIG の I/O ポートをトリガー・レンジに適用し、一度は少なくとも  $10\mu s$  の高レベルの信号を与えます;
- (2) モジュールは 40kHz の 8 つの方形波を自動的に送信し、信号が自動的に返されるかどうかをテストします;
- (3) 受信した信号がある場合、モジュールは ECHO の I/O ポートを介して高レベルパルスを出します。高レベルパルスの持続時間は、波の送信と受信の間の時間です。したがって、モジュールは時間に応じて距離を知ることができます。

$$\text{テスト距離} = (\text{高レベル時間} \times \text{音速 (340M / S)}) / 2$$

$$\text{Testing distance} = (\text{high level time} \times \text{velocity of sound (340M/S)}) / 2;$$

## 実際の運用:

タイミング図は以下に示します。トリガー入力に short10uS パルスを供給するだけでレンジングが開始します。その後、モジュールは 40 kHz で 8 サイクルの超音波バーストを送信し、エコーを高めます。エコーは、パルス幅と範囲が比例する距離オブジェクトです。トリガー信号を送信してからエコー信号を受信するまでの時間間隔を介して範囲を計算できます。式： $\mu\text{S} / 58 = \text{センチメートル}$  または  $\mu\text{S} / 148 = \text{インチ}$ ; または：範囲=高レベル時間\*速度 (340M / S) / 2; エコー信号へのトリガー信号を防ぐために、60ms 以上の測定サイクルを使用することをお勧めします。



/\*Ultrasonic distance measurement Sub function\*/

int Distance\_test()

{ digitalWrite(Trig, LOW);

    delayMicroseconds(2);

    digitalWrite(Trig, HIGH);

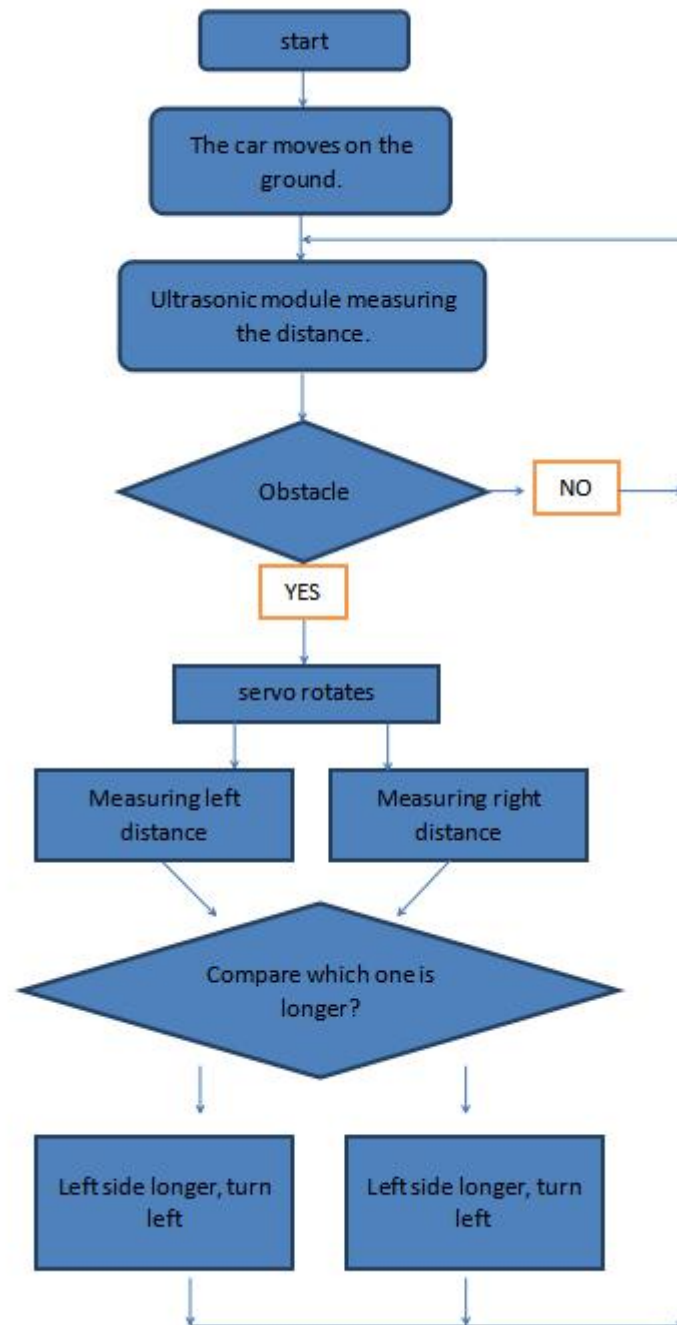
    delayMicroseconds(20);

    digitalWrite(Trig, LOW);

    float Fdistance = pulseIn(Echo, HIGH);

    Fdistance= Fdistance/58;

```
return (int)Fdistance;  
}
```



上の画像より、障害物回避車の原理は非常に簡単であることを理解できます。超音波センサーモジュールは、車と障害物間の距離を何度も検出し、データをコントローラーボードに送信します。その後、車が停止し、サーボを回転させて、左側と右側を検出します。別の側からの距離を比較した後、車はより長い距離の側に曲がり、前進します。その後、超音波センサーモジュールが再び距離を検出します。

## コードレビュー:

```
if(rightDistance > leftDistance) {
    right();
    delay(360);
}
else if(rightDistance < leftDistance) {
    left();
    delay(360);
}
else if((rightDistance <= 40) || (leftDistance <= 40)) {
    back();
    delay(180);
}
else {
    forward();
}
```