# Ultrasonic Sensor Radar Tutorial 1

## this tutorial calculates creates a radar of the object in front of the sensor using arduino for data creation and processing for displaying data

*\*Uses Arduino IDE*

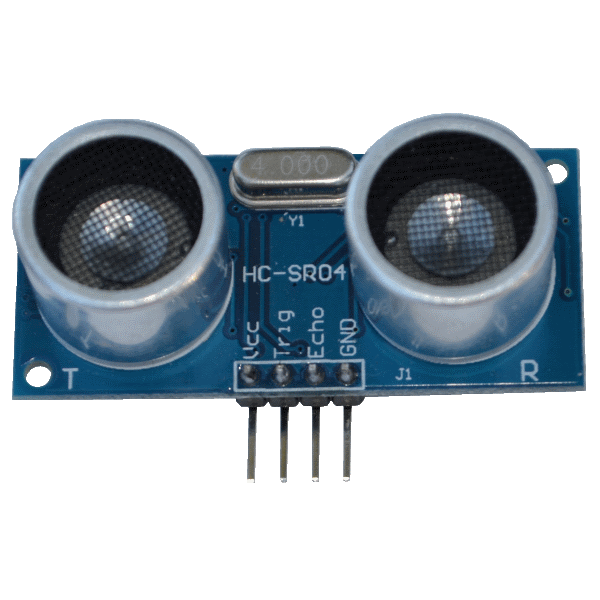
*\*Uses Processing IDE*

The following components will be used:

* 5- Male to Male 20cm Jumper Wires
* 5- Male to Female 20cm Jumper Wires
* 1- HC-SR04 Ultrasonic Sensor
* 1- SG-90 Servo Motor
* 1- 400 Point Breadboard
* PastelTech Uno Board Arduino Compatible
* USB B Cable
* Energy and willingness to learn

# Components Explained:

* What is an ultrasonic distance sensor?

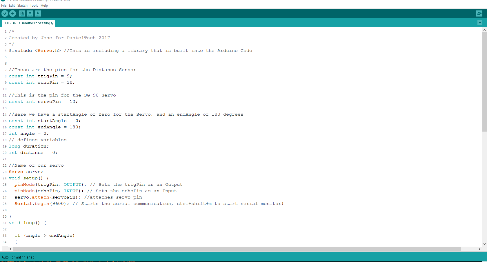


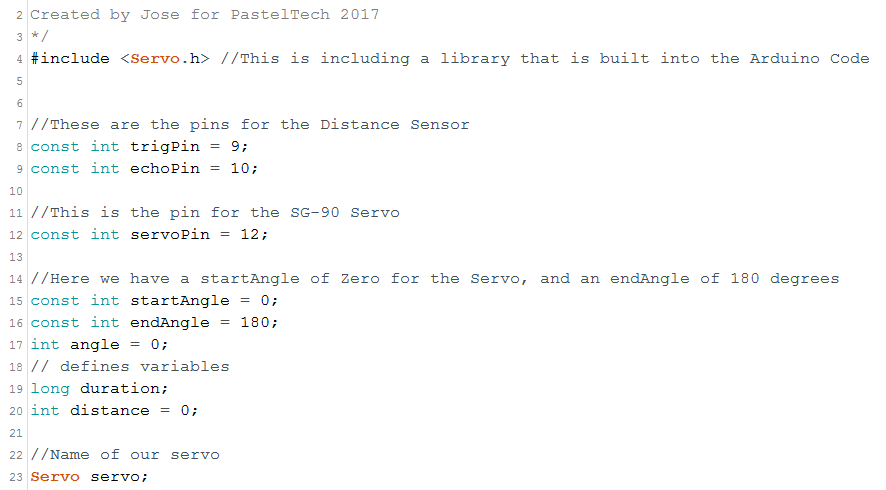
* + The left speaker of the ultrasonic distance sensor emits a 40Hz tone for 10 Microseconds (.00001s). The right receiver then waits for an echo to bounce back. It measures the time difference and with a little bit of knowledge of the speed of sound you can calculate the distance that the object is at. This also means that temperature affects the sensor. As in warmer weather sound travels faster. However, at this scale, no error correction is needed.
  + As said earlier, the sensor waits for the echo and if the echo takes too long, it can cause the system to hang up waiting for that audio echo to return, that’s why we added a limit of 7000 Microseconds in the pulseIn method call in the Arduino code.
* What is an SG-90 Servo Motor?
  + This servo can turn from 0 degrees all the way to 180 degrees in one-degree increments. It’s a starter component, as nearly all the gearing mechanism is plastic, so be careful! Try not to overload or stop the mechanism from turning. Engineering is about designing things that fit their use cases.
  + The SG-90 turns one degree after it receives a distance measurement from the Arduino Code.

# How does the Code Work?

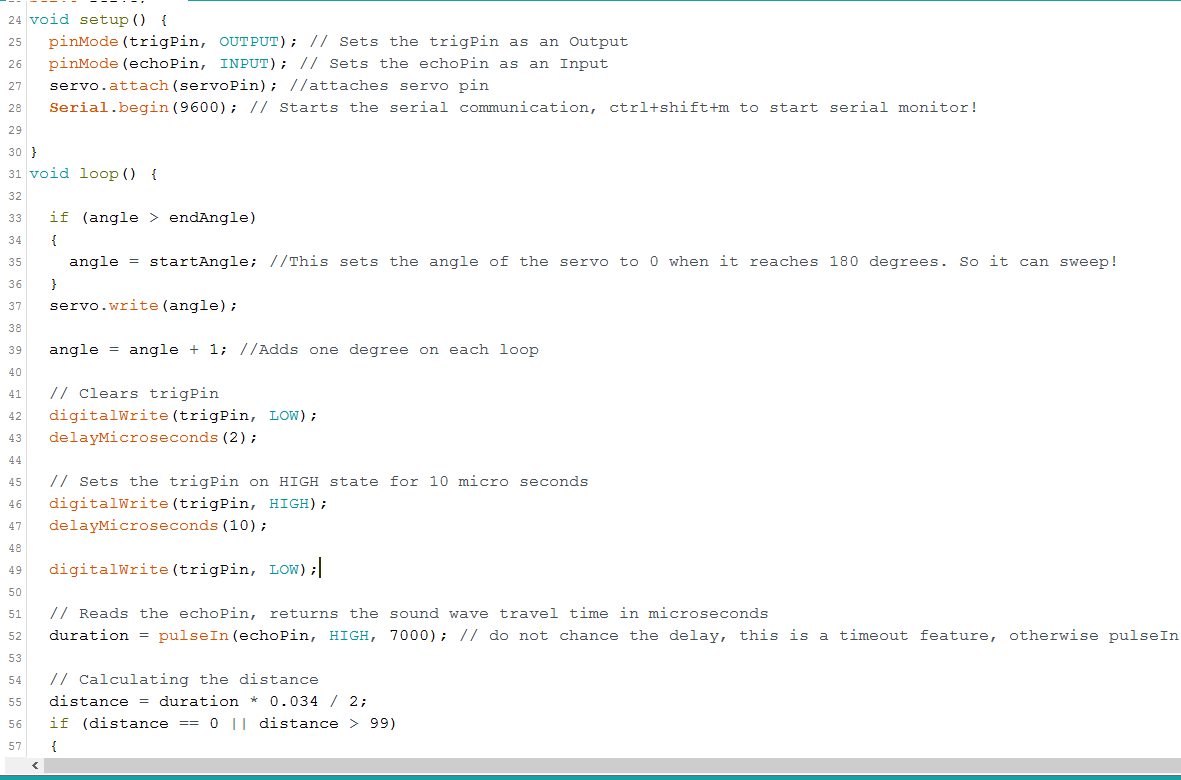
*try to understand why something works, not just make people think you do*

# Arduino Code Overview

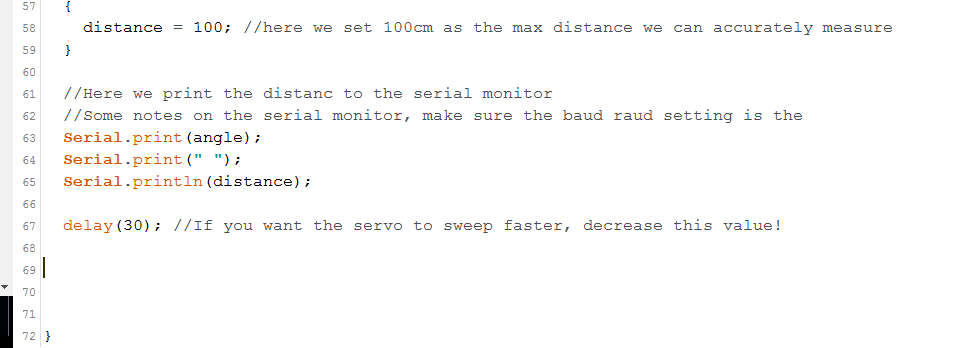
* The lowest level of Code is the Arduino Code, we connect the ultrasonic sensor and the sg-90 servo to the included PastelTech Uno. Both components use 5v. You can just connect straight to the 5v outputs on the board.
  + The Ultrasonic sensor uses a trigPin (pin #9) and an echoPin (pin #10).
  + The SG-90 Servo using a servoPin (pin #12).
* Here we see the Arduino we recommend opening the Arduino folder containing the Ultrasonic Sensor file 
* Here we need to declare the pins for both the Ultrasonic sensor and the SG-90 Servo



* On line 8 we set the const int pins for the Ultrasonic sensor
* We use const int because it cannot be changed, we cannot set the trigPin = 1000; later on in the program.
* We set a servo pin for the servo on line 12
* On line 15-20, we have the lines solving for the distance of the Arduino sensor, we need the current angle, in this case just angle, a start and end angle. The duration that it takes the signal to echo back, and finally the distance, that is calculated later.



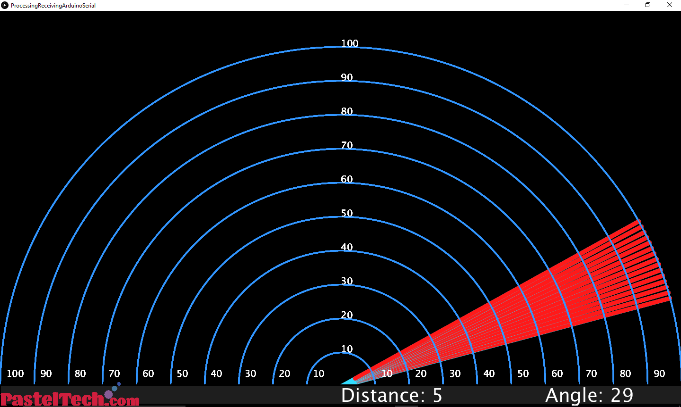
* From lines 24 to 28, we have the setup method, we set the trigPin as an OUTPUT, and the echoPin as an INPUT
* On line 27 we attach the servo
* On line 28 we begin our serial monitor, where our values are printed to, access it by pressing ctrl+shift+m
* Make sure the baud rate is 9600, when checking the Serial Monitor
* From line 31 we have the void loop() method
  + This will loop continuously, forever.
  + On line 33 there is an if statement checking if the servo is has spun 180**°** if so, it sets the angle to equal startAngle which equals 0.
  + On line 37, we set servo.write(angle)
  + On line 38, we add one degree to the current angle
  + From lines 42 to 49, we first clear the trigPin, then we set the pin to HIGH on line 46, this sends a sound signal out for 10 Microseconds
  + Then we set our pin back to LOW
  + On line 52, we calculate the duration from the echoPin, by checking for HIGH, and waiting 7000 Microseconds.
  + On line 55 we finally calculate the distance.

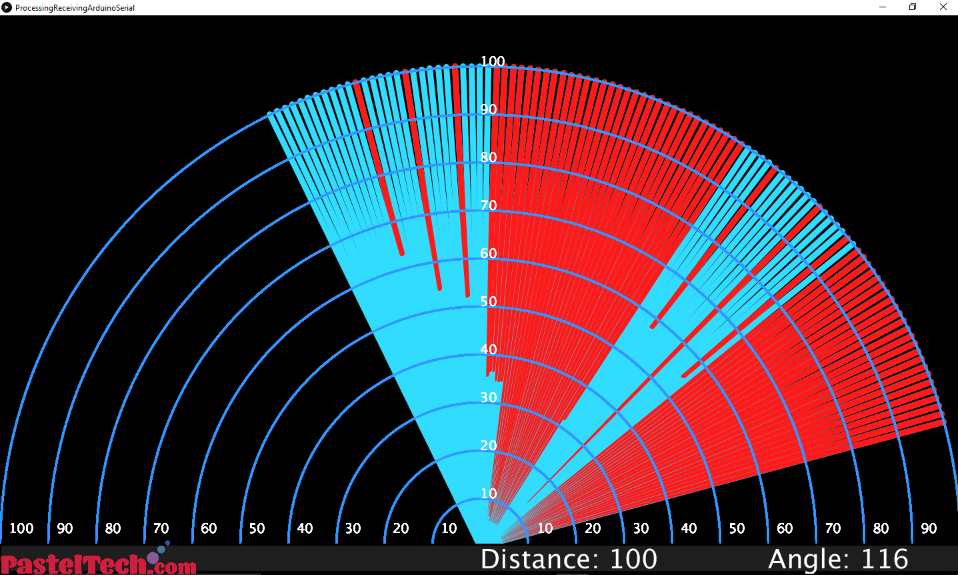


* On line 63-65, we use Serial.print(angle); to print the currentAngle of the SG-90, we print a “ “ space, then we print the distance we calculated earlier.
* On line 67 we delay(30) milliseconds, you can increase this to make the servo slower or faster, experiment!

# processing code overview

* The processing code takes the Serial Monitor data that the Arduino is sending and processes it into the following data display.





* The processing sketch is much more complicated than the Arduino sketch, the processing sketch itself has helpful comments explaining what the program is doing.
* Here everything that is marked in red shows that there is an object in the way.