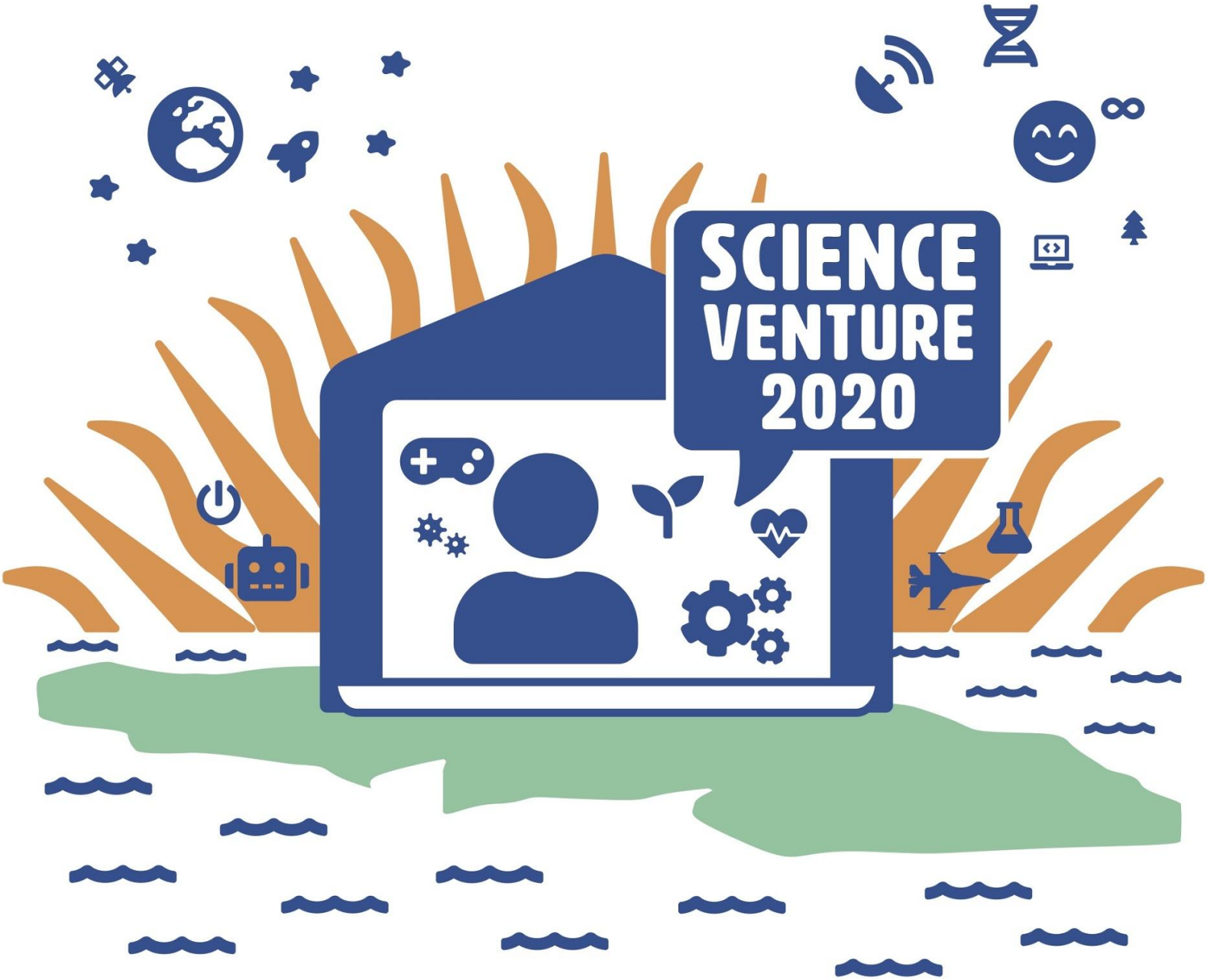


Arduinos

Arduino Motion Detector

Introduce the ultrasonic distance detector. Practice monitoring component values in the console. Learn about setting threshold values to determine effective movement. Code a program to detect motion and respond to readings above the programmed threshold.

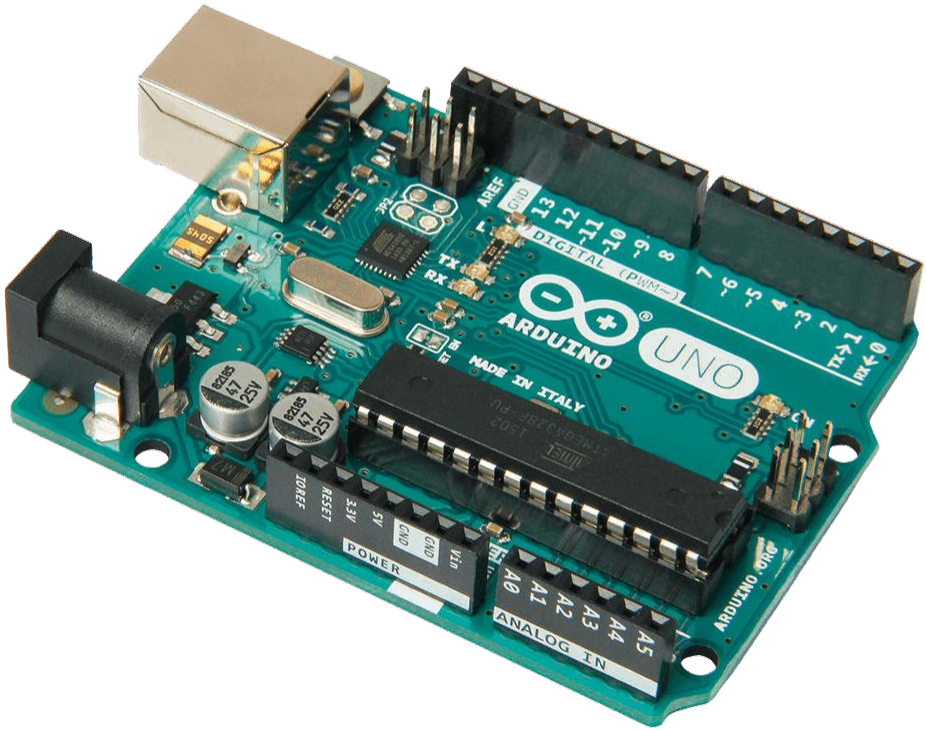
What is a Arduino?

Arduino is an open-source electronics platform based on easy-to-use hardware and software. Arduino boards are able to read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online. You can tell your board what to do by sending a set of instructions to the microcontroller on the board. To do so you use the Arduino programming language (based on Wiring), and the Arduino Software (IDE), based on Processing

Arduino coding environment: https://create.arduino.cc/editor/

Materials

- Arduino Uno Kit
  - USB Cable
  - Breadboard
  - Ultrasonic sensor
- Computer with Arduino Create

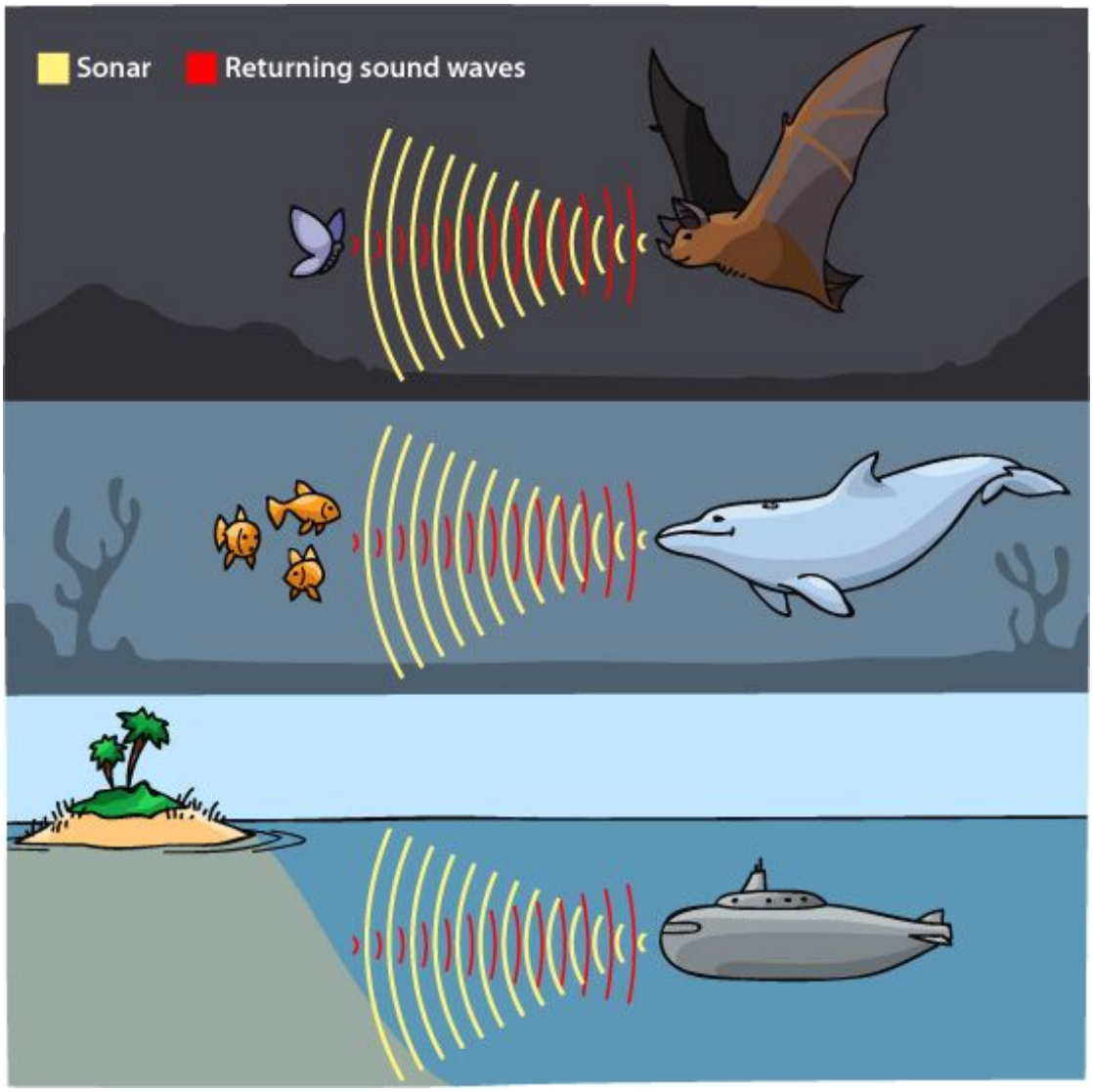


Arduino Motion Detector

Introduction to Ultrasonic Sensor

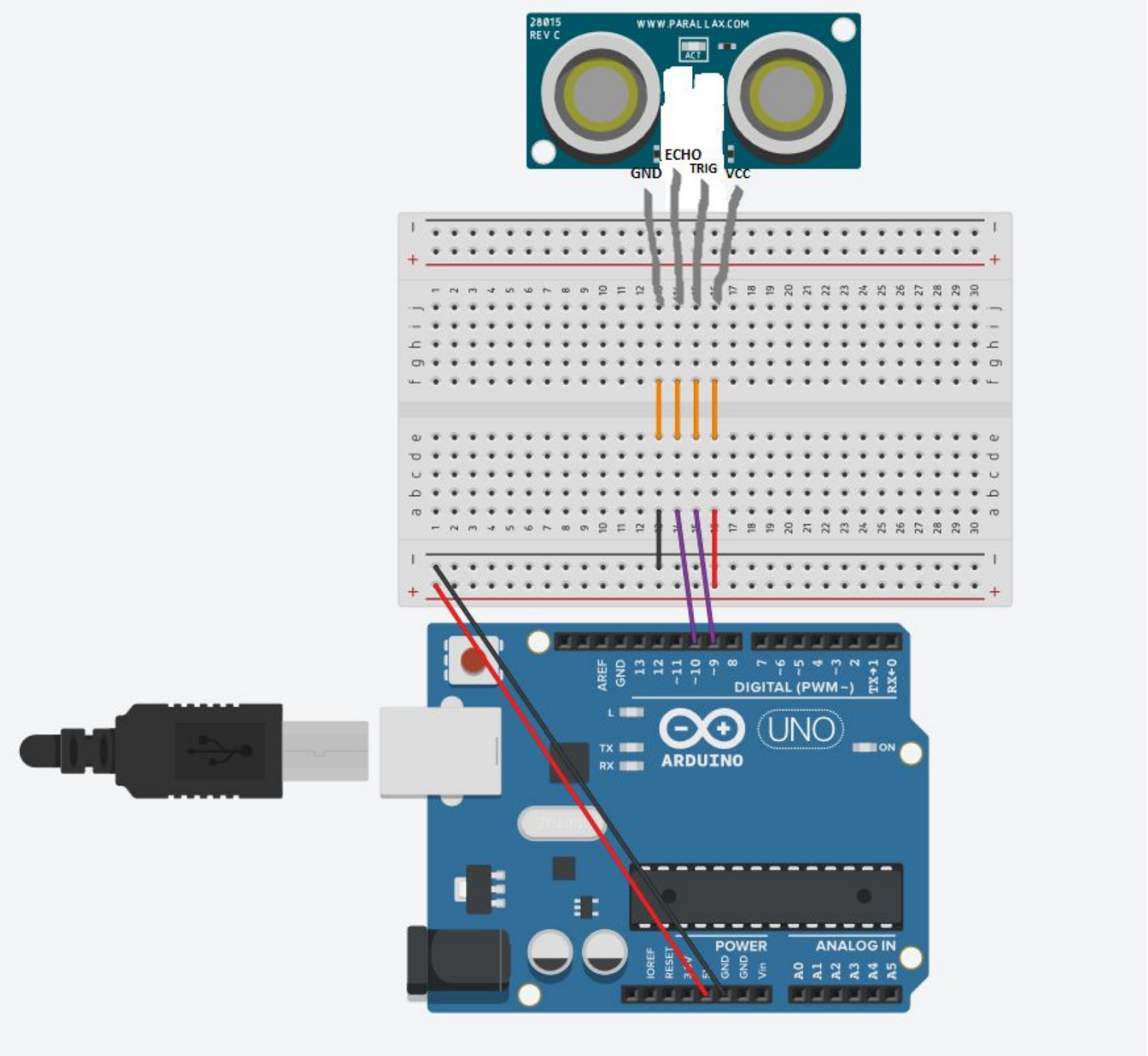
In this project we will introduce the HC-SR04 Ultrasonic sensor. It works by sending sound waves from the transmitter, which then bounce off of an object and then return to the receiver. You can determine how far away something is by the time it takes for the sound waves to get back to the sensor.

Ultrasonic waves are high frequency - mainly used for sensing distance over land. **Sonar** is the same concept, but can also use low frequency waves - like whale communication. Sonar is a natural technique for animals with poor sense of vision. Sonar is also used by military vehicles and submarines to detect enemies or obstacles surrounding their precious cargo.



Arduino Motion Detector

Wiring the Ultrasonic Distance Sensor



Arduino Motion Detector

Live coding and wiring of the motion detector

Make a motion sensor. Take a series of readings to calibrate our motion sensor - this is so we can tell what our readings will be when there is no motion. Then, we can set a minimum and maximum reading to detect when there is movement (by checking if values go outside of the min/max range).

When motion is detected print “Hello!” to the console.

Motion Detector Code: Step-by-step
Initializing variables

1) First set our constant variables - these will be our Sensor’s trigger and echo pins, and a constant number of readings to calibrate our sensor.
2) Declare some float variables for duration and distance (our sensor readings) and a Min and Max distance to set our range of values when we calibrate the sensor.

```
const int trigPin = 9;
const int echoPin = 10;
const int num_readings = 10;

float duration, distance;
float minDist, maxDist;
```

Void setup() { }

a) In our setup function, we can start by setting our pin modes. To use the ultrasonic distance detector, we need an output on our trigger pin and input for the echo pin.
b) Start our serial monitor communication with Serial.begin(9600);

```
void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  // calibrate reading data to get an idea of the min/max values it reads
  // try to be very still during the setup phase!
```

Arduino Motion Detector

3) Now we will use a For loop to calibrate the sensor: for (int i = 0; i < num_readings; i++) {} In the argument we use a variable int i to run through the loop multiple times, from 0 to 10 (our num_readings value).
4) In our loop, first set a LOW value on the trigPin. Use delays between each pin write.
5) Next set the pin to HIGH, wait 10ms, and reset the pin to low. This activates our sensor for 10ms so we can read how long the ultrasonic waves take to return.
6) Use duration = pulseIn(echoPin, HIGH); to set the duration variable with the echoPin's return value. This is how long the wave takes to return to the sensor.
7) Use the duration variable to calculate the distance. Print the value to the serial monitor.

```
for (int i = 0; i < num_readings; i++) {
    // set to LOW to make a well defined HIGH reading
    digitalWrite(trigPin, LOW);
    delayMicroseconds(2);

    // setting the trigger to HIGH for 10ms to tell the detector to send a wave
    digitalWrite(trigPin, HIGH);
    delayMicroseconds(10);
    digitalWrite(trigPin, LOW);   // set back to LOW

    // determines length of time the wave took to return
    duration = pulseIn(echoPin, HIGH);

    // calculates distance & prints to Serial
    distance = (duration*.0343)/2;
    Serial.print("Distance: ");
    Serial.println(distance);
    delay(1000);
```

8) Finally in our setup we will use an IF statement to set our Min and Max values.
9) On the first loop of our setup i==0, we set both min and max to the first distance reading.

Arduino Motion Detector

10) Else { We will use a pair of IF statements to compare the latest distance reading to the Max and Min. If the distance reading is greater than the Max or lesser than the Min, update the variable to the new reading.

```
if (i==0) {
        // the first time through the loop, initialize the min and max distance to
        // the first reading value.
        minDist = distance;
        maxDist = distance;
} else {
        // every other time through the loop, check to see if you have a
        // new min or max value.
if (distance < minDist) {
        minDist = distance;   // update minimum
}
if (distance > maxDist) {
        maxDist = distance;   // update maximum
}
}
 }
}
```

Arduino Motion Detector

Void loop() { }

1) Use the same code from the beginning of the calibration loop to take a reading from the sensor. Repeat the calculation code to set duration and distance values.
2) IF the distance is less than minDist OR greater than maxDist, do something to show we have triggered the motion detector with an unusual value. Here we print to the monitor.
3) Finish the loop function by adding a delay before the next motion sensor reading is taken

```
void loop() {
  // calculate distance in cm
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = (duration*.0343)/2;

  // if the reading is outside of the range in the initialization stage,
  // print something to the console.
  if (distance < minDist || distance > maxDist) {
        Serial.println("Hello World!");
  }

  delay(1000);
}
```

Arduino Motion Detector

motionDetector.ino - Complete Code

```
/*
 * Motion detector
 * See the following for tips on getting started, and for the math behind the distance calculation
 *
https://create.arduino.cc/projecthub/Isaac100/getting-started-with-the-hc-sr04-ultrasonic-sensor-036380
 *
 *
 * adapted for Science Venture, 2020
 *
 */

const int trigPin = 9;
const int echoPin = 10;
const int num_readings = 10;

float duration, distance;

float minDist, maxDist;

void setup() {
  pinMode(trigPin, OUTPUT);
  pinMode(echoPin, INPUT);
  Serial.begin(9600);
  // calibrate reading data to get an idea of the min/max values it reads
  // try to be very still during the setup phase!

  for (int i = 0; i < num_readings; i++) {
        // set to LOW to make a well defined HIGH reading
        digitalWrite(trigPin, LOW);
        delayMicroseconds(2);
```

Arduino Motion Detector

```
// setting the trigger to HIGH for 10ms to tell the detector to send a wave
        digitalWrite(trigPin, HIGH);
        delayMicroseconds(10);
        digitalWrite(trigPin, LOW);   // set back to LOW

        // determines length of time the wave took to return
        duration = pulseIn(echoPin, HIGH);

        // calculates distance & prints to Serial
        distance = (duration*.0343)/2;
        Serial.print("Distance: ");
        Serial.println(distance);
        delay(1000);

        if (i==0) {
        // the first time through the loop, initialize the min and max distance to
        // the first reading value.
        minDist = distance;
        maxDist = distance;
        } else {
        // every other time through the loop, check to see if you have a
        // new min or max value.
        if (distance < minDist) {
        minDist = distance;   // update minimum
        }
        if (distance > maxDist) {
        maxDist = distance;   // update maximum
        }
        }
  }
}
```

Arduino Motion Detector

```
void loop() {
  // calculate distance in cm
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = (duration*.0343)/2;

  // if the reading is outside of the range in the initialization stage,
  // print something to the console.
  if (distance < minDist || distance > maxDist) {
        Serial.println("Hello World!");
  }

  delay(1000);
}
```

Arduino Motion Detector

Add-On

Instead of printing Hello World to console, add another component to respond.
For example,

- Turn on LEDs
- Start a motor movement
- Code your own unique reaction!

Try adding a DC motor to our motion detector program.

DC Motors

A **DC motor** is one that has 2 wire connections - 1 for power and 1 for ground. This allows the motor to be powered either clockwise or counterclockwise by swapping the leads. The motor has one speed, when it is powered, and will run at that speed until stopped.

Arduino Motion Detector

MotionDetectorWavesBack.ino - LED and DC motor version

```
/*
 * Moton detector that waves back
 * See the following for tips on getting started, and for the math behind the distance calculation
 * https://create.arduino.cc/projecthub/Isaac100/getting-started-with-the-hc-sr04-ultrasonic-sensor-036380
 *
 * adapted for Science Venture, 2020
 *
 */

// declare the constants for the motor pins
const int dcPin = 6;
const int dcPin2 = 7;
const int LED = 13;

// motion sensor pins
const int trigPin = 9;
const int echoPin = 10;

// for calibration
const int num_readings = 10;
float minDist, maxDist;

// for calculating distance
float duration, distance;

void setup() {

  pinMode(trigPin, OUTPUT); //set pin Modes for Distance Detector
  pinMode(echoPin, INPUT);
  pinMode(dcPin, OUTPUT); //set pin Modes for DC motor
  pinMode(dcPin2, OUTPUT);
  pinMode(LED,OUTPUT); // set LED pin output

  Serial.begin(9600);
  // calibrate reading data to get an idea of the min/max values it reads
  // try to be very still during the setup phase!
```

Arduino Motion Detector

```
 for (int i = 0; i < num_readings; i++) {
        // set to LOW to make a well defined HIGH reading
        digitalWrite(trigPin, LOW);
        delayMicroseconds(2);

        // setting the trigger to HIGH for 10ms to tell the detector to send a wave
        digitalWrite(trigPin, HIGH);
        delayMicroseconds(10);

        digitalWrite(trigPin, LOW);   // set back to LOW

        // determines length of time the wave took to return
        duration = pulseIn(echoPin, HIGH);

        // calculates distance & prints to Serial
        distance = (duration*.0343)/2;
        Serial.print("Distance: ");
        Serial.println(distance);
        delay(1000);

        if (i==0) {
        // the first time through the loop, initialize the min and max distance to
        // the first reading value.
        minDist = distance;
        maxDist = distance;
        } else {
        // every other time through the loop, check to see if you have a
        // new min or max value.
        if (distance < minDist) {
        minDist = distance;   // update minimum
        }
        if (distance > maxDist) {
        maxDist = distance;   // update maximum
        }
        }
 }
} // ends setup() function
```

Arduino Motion Detector

```
void loop() {
  // calculate distance
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);

  duration = pulseIn(echoPin, HIGH);
  distance = (duration*.0343)/2;

  // if the reading is outside of the range in the initialization stage,
  // Light the LED and move the motor
  if (distance < minDist || distance > maxDist) {

     digitalWrite(LED,HIGH);

    // DC motor control
      digitalWrite(dcPin, LOW); //activate the DC motor, wait, then deactivate
      digitalWrite(dcPin2,HIGH);
      delay(1000);
      digitalWrite(dcPin2, LOW);

  }
  delay(1000);  // delay, then turn off LED
  digitalWrite(LED,LOW);
}
```

#SVatHome

Want to share your project or results with us?

Email or tag us @ScienceVenture

Have a question?

Reach us at svcamp@engr.uvic.ca