ARDUINO TUTORIAL

**Arduino a general Introduction - (Getting started)**

 *This is a very basic tutorial of Arduino. It is a general introduction of Arduino. How*

*should one can get started with Arduino this tutorial science canvas is explaining.*

**Detailed Tutorial**

**1. Introduction:** Arduino is an Integrated Development Environment based upon Processing. It has made very easy several things namely these are embedded system, physical computing, robotics, automation and other electronics based things.

**2. Software Required:** The Arduino comes for the following operating systems. You may go for any of these.

1. Windows
2. Linux
3. Mac OS

[*You may download the software from here.*](http://www.arduino.cc/en/Main/Software)

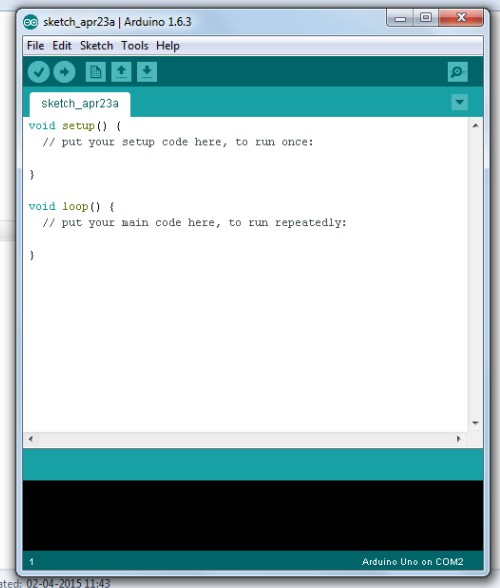
**3. Installation of FTDI:** Arduino Boards works on Virtual serial port supported by FTDI's FT232RL. You will need to install this to your computer. After installation note down the COM PORT No. This COM PORT No. will be required later in this tutorial.

[*Here is the detailed tutorial of Robo India to install FT232 RL.*](http://roboindia.com/tutorials/tutorial.php?tutorial_id=8)

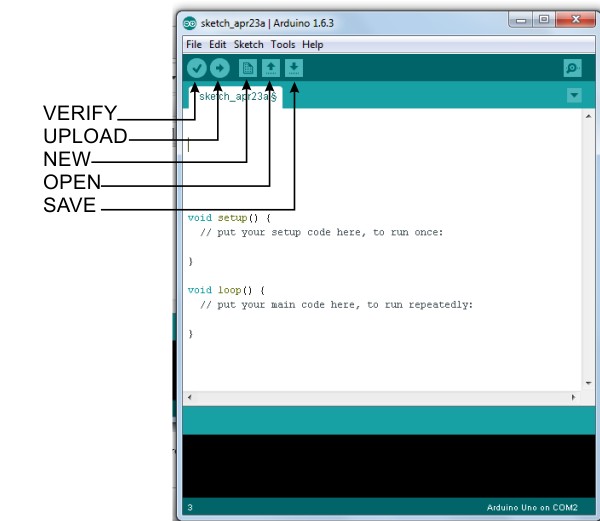
**4. Arduino Software Introduction:** Some general Understanding:

1. Code we write in Arduino is known as **SKETCH**.
2. Compilation of the code is known as **VERIFY**.
3. Transfer of the code from computer to Arduino board is known as **UPLOAD**.
4. If you directly hit **UPLOAD** button of Arduino software, the software will first VARIFY the code and then will transfer that code to Arduino Board.

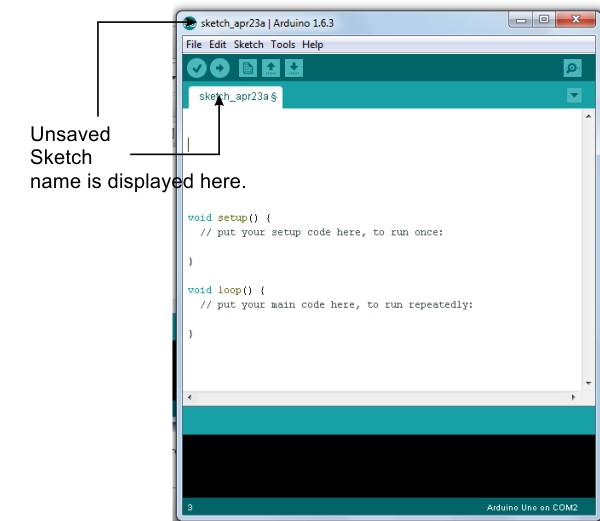
4.1 After installation of FTDI open Arduino Software. It looks like the following screenshot.



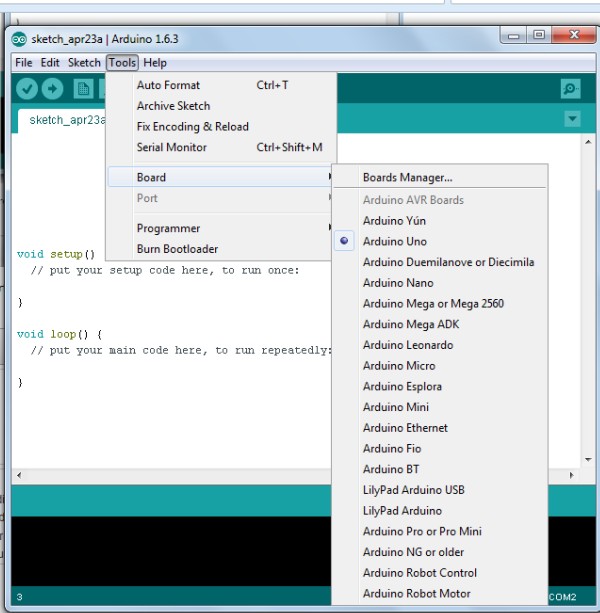
4.2 Buttons of Arduino software:



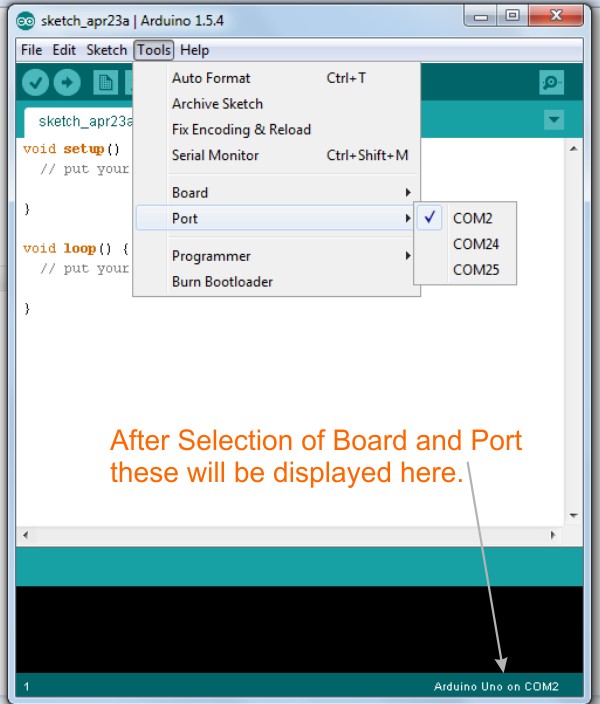
4.3 Untill you save your project, it displays its name as ***sketch\_date****.*Default folder to save Arduino code(sketch) is *My Documents/Arduino*or similar location for Mac OS and Linux. Arduino saves each code in a folder extension of code is***.ino***. By default it creates a folder and in that folder create a file with the extension we have seen. It is to be noted that name of the folder and containing file should be same ( If you want to rename the sketch then change name of both).



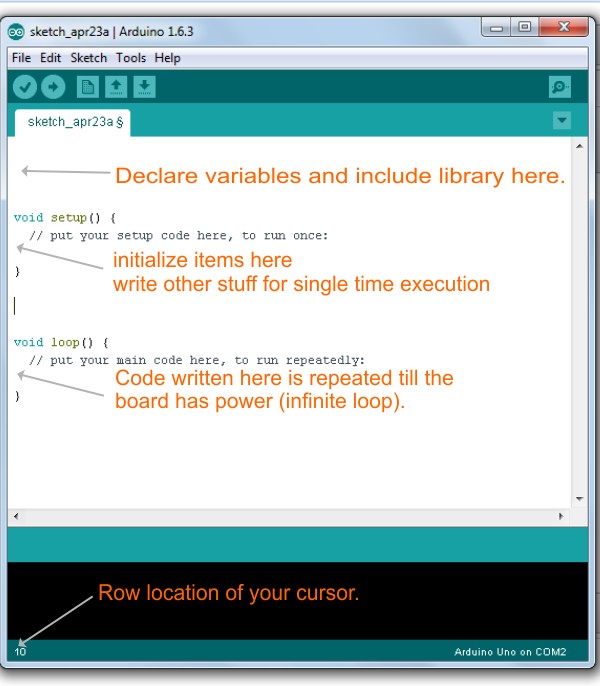
4.4 Selection of Board: Select the Arduino Board you are having from the available list.



4.5 Selection of Port: The port number you wrote is to be selected here(The Board should be attached to computer).



4.6 General understandings for programming:



Now you are ready to write you code. Once you have written your code save it and press UPLOAD button.

|  |  |
| --- | --- |
| **Arduino Digital Output - LED Blinking** | |
| http://roboindia.com/tutorials/admin/uploads/Icon-image-led-blink-tutori.jpg | ***This tutorial explains how to take digital output from Arduino. One of the basic tutorials for Arduino. The output is taken on a LED that blinks for an interval of 1 second.*** |

Detailed Tutorial

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **1. Introduction:** A step by step illustrated very basic tutorial for Arduino. Here we are taking digital output on a LED. This LED remains ON for one second and OFF for another, this loop runs for an infinite time.  1.1 LED Pins:  LED has two pin interface. Both of these pins are to be given input supply to the LED. Long legs is for positive supply the smaller one is for negative supply. following image depicts it clearly.  [LED-Pins-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/LED-blinking-arduino/LED-Pins-robo-india.jpg)  Digital means either 0/1, in other words HIGH/LOW or ON/OFF. So in the form of digital output we shall get either +5V or 0V on digital pin of Arduino. How does it happen that is illustrated ahead.  This tutorial is made on Original Arduino UNO .  **2. Required Hardware**  Following Hardware will be required to perform this LED blinking sketch.   |  |  |  |  | | --- | --- | --- | --- | | * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance |  |  |  | |  |  |  |  | |  |  |  |  | | **3. Building Circuit**  Make following circuit with the help of above mentioned components.  3.1 You may go with original Arduino UNO Board-  [Arduino-LED-BlinkUNO-B](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/LED-blinking-arduino/Arduino-LED-BlinkUNO-B.jpg) |  |  |  | | Here is the schematic of this circuit-  [Arduino-LED-Blink-UNO](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/LED-blinking-arduino/Arduino-LED-Blink-UNO.jpg)  **4. Programming:**  Once we are done with circuit part, here is our programme to this circuit.  [*You may download this code (Arduino Sketch) from here.*](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/LED-blinking-arduino/LED_Blink.zip)  // Digital output tutorial by Science Canvas  // Digital output is taken on a LED that remains ON for one second and  // OFF for another.  // Defining Pin 2 as LED.  const int LED = 2; // from the circuit we can see that we have connected LED on Pin 2  void setup() {      pinMode(LED, OUTPUT); // Defining LED pin as OUTPUT Pin.  }  // Below mentioned code runs for ever(infinite loop)  void loop() {    digitalWrite(LED, HIGH); // LED gets turned ON (1/HIGH/+5V)    delay(1000); // Waiting for one second.    digitalWrite(LED, LOW); // LED gets OFF (0/LOW/0V/GND)    delay(1000); // here and above Delay is in mili second (1000 = 1 second)  }  **Arduino Analog Output - LED fade in and fade out.**  http://roboindia.com/tutorials/admin/uploads/analog-output-icon-robo-ind.jpg*This tutorial explains how to take analog output from Arduino. One of the basic tutorials for Arduino. The output is taken on a LED that fades in and fades out. The video of output is also included.*  **Detailed Tutorial**  **1. Introduction:** A step by step illustrated very basic tutorial for Arduino. Here we are taking analog output on a LED. This LED gets fade in and then fade out. Arduino gives analog output in range of 0 to 255. Technically the output is digital but in the form of PWM, but it seems to be analog. Arduino Boards have 6 PWM(Analog Pins) these are PIN No. 3,5,6,9,10,11.  This tutorial is made on Original Arduino UNO .  **2. Required Hardware**  Following Hardware will be required to perform this LED fade in and fade out circuit.   * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance   **3. Building Circuit**  Make following circuit with the help of above mentioned components.  3.1 You may go with original Arduino UNO Board-  [Arduino-LED-BlinkUNO-B](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/LED-blinking-arduino/Arduino-LED-BlinkUNO-B.jpg)  Here is the schematic of this circuit-  [Arduino-LED-Blink-UNO](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/LED-blinking-arduino/Arduino-LED-Blink-UNO.jpg)  **4. Programming:**  Once we are done with circuit part, here is our programme to this circuit.  [You may download this code (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/analog-fade-in-out/Analog_Output_LED_Fade_In_Out.zip)  /\*  Tutorial for Analog Output.  Analog output is taken through PWM.  Prepared by Science Canvas   \*/  int LED\_ao = 3; // The LED attached to Pin 3 for analog output.  void **setup**() {     pinMode(LED\_ao, OUTPUT); // Declaring Pin as output.  }   void **loop**() {   // Range of PWM is 0 to 255. So we are running FOR LOOP for 1 to 255.    for (int brightness=1; brightness<=255; brightness++) // For loop for Fade In effect.      {        analogWrite(LED\_ao, brightness); // LED will glow for value of 1 to 255.        delay(20); // Small delay to see fade effect.      }    for (int brightness=255; brightness>0; brightness--) // Same FOR LOOP for Fade out effect.      {        analogWrite(LED\_ao, brightness);        delay(20);      }  }  **Arduino - Digital Input & Output | Pushbutton & LED**  http://roboindia.com/tutorials/admin/uploads/Digital-input-robo-india-ar.jpg  *This tutorial explains basic concepts of arduino. Digital input and digital output. Digital input is taken through push button and that is detected by Arduino. This input is processes by Arduino and it send digital command to attached LED.*  *When the button is pressed LED glows.*  **Detailed Tutorial**  **1. Introduction:** A step by step illustrated basic tutorial for Arduino. In this tutorial we are taking digital input from a push button switch. That input is read by Arduino board and decision is taken accordingly. When we press the button LED glows. Thus this tutorial is for both digital input and digital output.  **1.1 Digital Input:** Digital input means when we are supplying ***HIGH/1/+5V or LOW/0/GND*** to the Arduino board. On the contrary digital output means when we are taking ***HIGH/1/+5V or LOW/0/GND*** from the Arduino.  In this example Pin No. 2 of Arduino is connected to +5V through a switch and the same pin is also connected to GND via 10K resistance. This resistance keeps this pin at GND and when we press button it gets connected to +5V. So we can understand that if switch is released the Input Pin (Pin No.2) is *GND/LOW/0* and when switch is presses it is *+5V/HIGH/1.*  **2. Required Hardware**  Following Hardware will be required to perform this LED fade in and fade out circuit.   * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance/10kΩ resistor * Mini Push Button   **3. Building Circuit**  Make following circuit with the help of above mentioned components.  3.1 You may go with original Arduino UNO Board-  [Arduino-digital-input-push-button-UNO-Board-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-digital-input/Arduino-digital-input-push-button-UNO-Board-robo-india.jpg)  here is the schematic:  [Arduino-digital-input-push-button-UNO-Board-robo-india-sch](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-digital-input/Arduino-digital-input-push-button-UNO-Board-robo-india.jpg)  **4. Programming:**  Once we are done with circuit part, here is our programme to this circuit. All of the commands are explained in the comment section. Output video is attached at the last of this tutorial, This example will be clear after watching the video.  [You may download codes (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-digital-input/Digital_input_through_button.zip)  This code has a simple function to perform, if Button is pressed glow the LED.  /\*  Tutorial on Digital input through button.  \*/  const int BUTTON = 2; // Pushbutton Input to Pin No.2  const int LED = 3; // Output LED to Pin No. 3  int BUTTONState = 0; // To store input status  void **setup**() {    pinMode(LED, OUTPUT); // Define LED pin as output.    pinMode(BUTTON, INPUT); // Define BUTTON pin as Input.  }  void **loop**(){    BUTTONState = digitalRead(BUTTON); // Reading input from Button Pin.    if (BUTTONState == HIGH) // Checking if Input from button is HIGH (1/+5V)     {       digitalWrite(LED, HIGH); // If input is High make LED ON (HIGH/1/+5V)     }    else    {       digitalWrite(LED, LOW); // For every other condition make LED OFF (0/GND/LOW)    }  }  **Arduino - Peizo Buzzer**  http://roboindia.com/tutorials/admin/uploads/buzzer.jpg  *This tutorial of Robo India is on Peizo Buzzer. Here two examples are included one plays beep sound and other one plays a song.*  **Detailed Tutorial**  **1. Introduction:** This tutorial expalins how to use a Peizo Buzzer on Arduino. When voltages are supplied to its terminal it generates beep sound. This buzzer is used to give beep sound to various embedded systems.  **2. Required Hardware**  Following Hardware will be required to perform this sketch of shift register.   * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance/10kΩ resistor * Peizo buzzer   **3. Building Circuit**  Make following circuit with the help of above mentioned components.  3.1 You may go with original Arduino UNO Board-  [UNO-Board-buzzer-arduino-tutorial-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-buzzer/UNO-Board-buzzer-arduino-tutorial-robo-india.jpg)  Here is the schematic of this circuit-  [UNO-Board-buzzer-arduino-tutorial-robo-india-sch](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-buzzer/UNO-Board-buzzer-arduino-tutorial-robo-india-sch.jpg)  **4. Programming 1 - Playing a beep sound.** Once we are done with circuit part, here is our programme to this circuit. Every command of the following programme is explained in the comment section.  [You may download this code (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-buzzer/Buzzer_1_robo_india.zip)  /\*  Beep sound generation Tutorial  \*/  const int Buzzer = 9;  void **setup**()  {    pinMode(Buzzer, OUTPUT);  }  void **loop**()  {    digitalWrite(Buzzer, HIGH);    delay(400);    digitalWrite(Buzzer, LOW);    delay(2000);  }  **Programming 2 - Playing a song:**  Cicruit is same as above mentioned. Copy this code and upload to your Arduino board.  /\*  Tone generation Tutorial    note  frequency    c     262 Hz    d     294 Hz    e     330 Hz    f     349 Hz    g     392 Hz    a     440 Hz    b     494 Hz    C     523 Hz  \*/    const int Buzzer = 9;  const int songLength = 18;  char notes[] = "cdfda ag cdfdg gf "; // a space represents a rest  int beats[] = {1,1,1,1,1,1,4,4,2,1,1,1,1,1,1,4,4,2};  int tempo = 150; // Speed of tempo  void **setup**()  {    pinMode(Buzzer, OUTPUT);  }  void **loop**()  {    int i, duration;      for (i = 0; i < songLength; i++) // step through the song arrays    {      duration = beats[i] \* tempo;  // length of note/rest in ms        if (notes[i] == ' ') // is this a rest?      {        delay(duration); // then pause for a moment      }      else // otherwise, play the note      {        tone(Buzzer, frequency(notes[i]), duration);        delay(duration); // wait for tone to finish      }      delay(tempo/10); // brief pause between notes    }        while(true){} // Remove this line if you want to play this song for ever.  }  int frequency(char note)  {    int i;    const int numNotes = 8; // number of notes we're storing    char names[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };    int frequencies[] = {262, 294, 330, 349, 392, 440, 494, 523};    for (i = 0; i < numNotes; i++)    {      if (names[i] == note)      {        return(frequencies[i]);      }    }    return(0);  }  **Arduino - LDR (Light-Dependent Resistor)**  http://roboindia.com/tutorials/admin/uploads/LDR-ICON-Robo-India.jpg*This tutorial explains concept and how to use LDR (Light Dependent Resistor) with Arduino. An example is included in which a LED is controlled on the basis of LDR.*  **Detailed Tutorial**  **1. Introduction:**  A step by step illustrated tutorial to use LDR on Arduino.  **1.1 LDR:**  A LDR (Light Dependent Resistor) or a photo resistor is a photo conductive sensor. It is a variable resistor and changes its resistance in a proportion to the light exposed to it. It’s resistance decreases with the intensity of light.  **2. Required Hardware**  Following Hardware will be required to perform this LDR circuit.   * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance/10kΩ resistor * LDR   **3. Building Circuit**  Make following circuit with the help of above mentioned components. Some key points to understand about the circuit- LDR is connected to a 10 Resistance in series. +5 Voltage is applied to this arrangement. As the light intensity changes LDR value changes thus the voltage drop on LDR will change and we are going to measure that voltage change.  3.2 You may go with original Arduino UNO Board-  [UNO-Board-Arduino-LDR-tutorial-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-LDR/UNO-Board-Arduino-LDR-tutorial-robo-india.jpg)  Here is the schematic of this circuit-  [UNO-Board-Arduino-LDR-tutorial-robo-india-sch](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-LDR/UNO-Board-Arduino-LDR-tutorial-robo-india-sch.jpg)  **4. Programming:**  Once we are done with circuit part, here is our programme to this circuit. Every command of the following programme is explained in the comment section. A few point to consider for this sketch.  1. It reads LDR value and prints them on Serial monitor. Once you upload this programme to your Arduino board open serial monitor and observe how values are changing with the change of Light intensity.  2. The attached LED glows in analog mode according to the LDR Values.  3. There is a condition of threshold; The attached LED remains OFF for all the values below Threshold limit. You can set your own threshold limit. In this programme we have given 800 as threshold. You can set t threshold to any value between 0 and 1023.  [You may download this code (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-LDR/LDR.zip)  // Tutorial on LDR.  int LDR = A0; // LDR input at A0 pin.  int LED = 3; // LED is connected to PWM Pin 3.  int LDRReading = 0; // to store input value of LDR  int lEDBrightness = 0; // to store the value of LED Brightness  int threshold\_val = 800; // Check your threshold and modify it.  void **setup**(){  **Serial**.begin(9600); // initializing serail communication.    pinMode(LED, OUTPUT); // Defining LED pin as output.  }  void **loop**(){    LDRReading = analogRead(LDR); // Reading LDR Input.  **Serial**.println(LDRReading); // Printing LDR input value.      if (LDRReading >threshold\_val){ // Condition to make LED ON.    lEDBrightness = map(LDRReading, 0, 1023, 0, 255); // Converting LDR to LED Brightness.    analogWrite(LED, lEDBrightness); // Writing Brightness to LED.    }    else{    analogWrite(LED, 0); // If LDR is below threshold make LED OFF.    }      delay(300); // delay to make output readable on serial monitor.  }  **Arduino Servo Control**  http://roboindia.com/tutorials/admin/uploads/arduino-servo-control-icon-.jpg*This tutorial is about servo control on Arduino. Arduino has got a library for servo control. This tutorial explains how to control servo by using in-built library of Arduino. It has got two examples of servo control to give a better understanding of servo control.*  **Detailed Tutorial**  **1. Introduction:** A step by step illustrated basic tutorial for Arduino. This tutorial explains Servo motor control through Arduino. We have included two examples in this tutorial.  **1.1 Servo Motor:** The meaning of servo is feedback. So a servo is an actuator that takes feedback itself and moves precisely. Example of preciseness and feedback can be understood by a daily life example, suppose you pull up/down glass of your car window, You push up/down the power window button and keep on watching the actual position of glass as it reaches to the desired position you releases the button. So this is feedback, we are taking feedback here, if it would be an feedback based system we would have to tell open window by 10%-20%.  Lets come back to servo motor. Servo motor rotate from 0 degree to 180degree. We send the command to servo as it reaches to the commanded value it stops there. The below diagram exhibits how it rotates and its wire interface.  [servo-basics-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/servo-arduino/servo-basics-robo-india.jpg)  **1.2 Servo and Arduino:**  Arduino has got a library to control servo. It is – Servo.h.  This library can control above shown servo motors. This library supports up to 12 servos on most Arduino boards and 48 servos on Arduino Mega. It disables analogWrite() for Pin 9 & Pin 10 except Arduino Mega.  This tutorial is made on Original Arduino UNO .  **2. Required Hardware**  Following Hardware will be required to perform this LED fade in and fade out circuit.   * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance/10kΩ resistor * Servo Motor * 1k potentiometer   **3. Building Circuit - 1** This is first example of servo control. Servo alone is controlled through Arduino. In this examples servo takes 5 motions-  1. 0 degree to 45 degree  2. 46 degree to 90 degree  3. 91 degree to 135 degree  4. 136 degree to 180 degree  5. 180 degree to 0 degree (Back to zero)  After completing every movement it stops there for one second.  To run this practical example make following circuit with the help of above mentioned components.  3.1 You may go with original Arduino UNO Board-  [Arduino-UNO-board-servo-control-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/servo-arduino/Arduino-UNO-board-servo-control-robo-india.jpg)  **4. Programming - 1:**  Once we are done with circuit part, here is our programme to this circuit. All of the commands are explained in the comment section. Output video is attached at the last of this tutorial, This example will be clear after watching the video.  [You may download this code (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/servo-arduino/Servo1.zip)  // Servo Control Tutorial#1  // Arduino Servo Library can add up to 12 servo on most of Arduino boards.  #include <Servo.h> // Includes servo library.    Servo servo\_1; // Creating Servo object.    int servo\_pos = 0; // Storing servo position (0 degree to 180 degree)    void **setup**()  {    servo\_1.attach(3); // Attaching servo to Pin No.3  }      void **loop**()  {    for(servo\_pos = 0; servo\_pos <= 45; servo\_pos++) // loop to go to 45 degree from 0 degree.     {                                   // increment of 1 degree in each step      servo\_1.write(servo\_pos); // commanding servo to reach at Servo\_pos.      delay(15); // waiting a bit for the servo to reach commanded position.     }    delay(1000); //Delay of 1 second to observe stoppage of servo.    for(servo\_pos = 46; servo\_pos <= 90; servo\_pos++) // loop to go to 90 degree from 46 degree.     {      servo\_1.write(servo\_pos);  **5. Circuit -2:**  This is second example of Servo Control. In this example We are controlling servo through Analog input. Analog input is taken from a potentiometer. And command of motion is given to the attached servo as per the input we are getting from potentiometer.Make the following circuit-  5.1 Or You may go with original Arduino Board:  Arduino-UNO-board-servo-Potentiometer-robo-india  here is the schematic:  [Arduino-UNO-board-servo-Potentiometer-robo-india-sch](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/servo-arduino/Arduino-UNO-board-servo-Potentiometer-robo-india-sch.jpg)  **6. Programming - 2:**  Here is programming for example - 2 (above mentioned circuit).  Points to under stand.  1. Analog input is taken through Potentiometer on Pin A0. It will give us value from 0 to 1023.  2. The servo library need value 0 to 180 to give command to servo.  3. So we will convert our input values (0-1023) to the command value for servo (0-180).  4. This converted value is sent to the servo through Pin 3.  The following programme explains all of the command in comment sections.  [You may download the codes (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/servo-arduino/Servo2.zip)  // Servo Control Tutorial#2  // Arduino Servo Library can add up to 12 servo on most of Arduino boards.  #include <Servo.h> // Includes servo library.    Servo servo\_1; // Creating Servo object.    int input\_pin = A0; // Analog Input Pin (Potentiometer)  int input\_val; // to store analog Input value (0-1023)  int servo\_angle =0; // Servo angle value (0-180 degree)    void **setup**()  {    servo\_1.attach(3); // Atteching Arduino's Pin 3 to servo.  }    void **loop**()  {    input\_val = analogRead(input\_pin); // To read analog input value (0-1023)    servo\_angle = map(input\_val, 0, 1023, 0, 179); // Converting input value (0-1023) to servo angle (0-180)    servo\_1.write(servo\_angle); // Commanding servo to reach servo angle    delay(15); // waiting for servo to reach commanded angle.  }  **Arduino - Ultrasonic Range sensor HC-SR04**  http://roboindia.com/tutorials/admin/uploads/Range-finder-hc-sr04.jpg*This explain the working concept of****Ultrasonic range sensor. This tutorial is based on easy to use Library. Various built in function of the library gives ease of use to HC-SR04 Ultra sonic range sensor.***  **Detailed Tutorial**  **1. Introduction:** A step by step illustrated tutorial to use Ultrasonic Range Sensor HC-SR04 on Arduino. lets see the working of the sensor in an easy way. This sensors transmits ultrasonic waves and receives its echo. We use the time difference between sending and receiving echo. That time interval is used to measure range or distance.This sensor is very useful to construct autonomous robots.  **1.1 Ultrasonic Library:** Here is the library to use this sensor. [*You may download it from here*](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-hc-sr04/Ultrasonic.zip)*.*  Once you have downloaded this library. Extract this and copy the folder to *My Documents/Arduino/Libraries*or other similar locations.   **2. Required Hardware**  Following Hardware will be required to perform this sketch of shift register.   * Arduino Board * Breadboard * Male to male jumper wire * LEDs * 100ohm resistance/10kΩ resistor * Ultrasonic Range Sensor HC-SR04   **3. Building Circuit**  Make following circuit with the help of above mentioned components.  3.1 You may go with original Arduino UNO Board-  [UNO-utra-sonic-sensor-hc-sr04-arduino-robo-india](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-hc-sr04/UNO-utra-sonic-sensor-hc-sr04-arduino-robo-india.jpg)    **4. Programming:**  Once we are done with circuit part, here is our programme to this circuit. Every command of the following programme is explained in the comment section.  Some salient features of the library.  1. It measures echo time and range.  2. Range it measures in Inch and Centimeter.  3. Maximum range it measure 51 cm.  4. Multiple sensor may be attached.  It has got four functions- All these functions are illustrated in the following sketch.  1. Initializing  2. Find range in cm  3. find range in Inch  4. Find echo time in mili seconds  [You may download this code (Arduino Sketch) from here.](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-hc-sr04/Ultrasonic.zip)  // HC-SR04 Utra sonic ranging module on Arduino  #include  Ultrasonic ultrasonic(6,7); // (Trig PIN,Echo PIN)  int range\_inch = 0; // Variable to store range in Inch  int range\_cm = 0; // Variable to store range in Centi meter  int time\_ms = 0; // Varibale to store echo time in ms  void **setup**() {  **Serial**.begin(9600); // Initializing serial communication.  }  void **loop**()  {    range\_inch = ultrasonic.Ranging(INC); // function returns range in Inch.    range\_cm = ultrasonic.Ranging(CM);    // function returns range in cm.    time\_ms = ultrasonic.Timing();        // function returns echo time in ms  **Serial**.print(range\_inch);  **Serial**.println(" Inch" ); // Printing range in Inch  **Serial**.print(range\_cm); // Printing range in CM  **Serial**.println(" cm" );  **Serial**.print(time\_ms); // Printing echo time in ms  **Serial**.println(" ms" );    delay(200); // waiting for a while  }  **5. Output:**After uploading this sketch open serial monitor and you will see the following like result.  **[ultra-sonic-sensor-robo-ind](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-hc-sr04/ultra-sonic-sensor-robo-ind.jpg)**  **Note:** This sensor works on the concept of trig and echo. So if you cover the sensor with some object, in this situation it will not receive any echo. Thus this situation is considered as the maximum range.  **Obstacle Avoiding Robot**  http://roboindia.com/tutorials/admin/uploads/dp-obstacle-avoiding.jpg *Presents obstacle avoiding robot based on SR-04 Ultrasonic Range finder and Arduino platform.*  **Detailed Tutorial**  **1. Introduction:**  A step by step guide for to make Obstacle avoiding Robot using Robo India's Motor Shield and Phantom Chassis.  [*You may need this tutorial to assemble Phantom Chassis. While you assemble chassis. Mark Name on wires so that you can identify them after assembly.*](http://roboindia.com/tutorial.php?tutorial_id=34)  **1.2 The motor shield**   Robo India offers Arduino compatible Motor shield that can run 2 servo and 2 stepper or 4 DC motor. This shield uses shift register IC. Data is given in serial mode that is converted to parallel by this shift register IC. Connection are as follows.  *Board to Shift Register-*   * Digital Pin 8 - Data Pin * Digital Pin 12 - Latch Pin * Digital Pin 4 - Clock Pin * Digital Pin 7 - Enable Pin   *Shift Register to Motors-*   * Q0 - M3 A * Q1 - M2 A * Q2 - M1 A * Q3 - M1 B * Q4 - M2 B * Q5 - M4 A * Q6 - M3 B * Q7 - M4 B   [Shield-motor-robo-india](https://www.adafruit.com/products/81)  **2.3 Ultrasonic Library:**   Here is the library to use this sensor. [*You may download it from here*](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-hc-sr04/Ultrasonic.zip)*.*  Once you have downloaded this library. Extract this and copy the folder to *My Documents/Arduino/Libraries*or other similar locations.  **2. Assembly and Connections**  After assembling the robot make the connection as given in the following diagram. The Power Jumper on the motor shield works as Switch to motor it will be useful while debugging.  [obstacle-avoiding-robot-cir](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/obstacle-avoiding-robot-cir.jpg)  **2.1 Assembly**  **Servo Direction-**  **[servo_direction_obstacle_av](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/servo_direction_obstacle_av.jpg)**  After assembling the robot will look like -   [01-obstacle-avoiding-robot](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/01-obstacle-avoiding-robot.jpg)  [02-obstacle-avoiding-robot](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/02-obstacle-avoiding-robot.jpg)  [03-obstacle-avoiding-robot](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/03-obstacle-avoiding-robot.jpg)  **2.2 Testing motor Connection**  It is difficult to find GND and positive supply terminal of motor. Transfer the following code to the Arduino Board. The robot should move in forward direction, if it not doing so, interchange the of the wire of motor. e.g. suppose left motor is rotating in back ward direction and right motor is rotating in forward direction then you have to interchange the wires of M3 terminal.  [*You may download this sketch from here.*](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-dtmf-robot/Forward_robot_motion.zip)  //Robotic Motion Tutorial  // This codes runs robot in the forward direction.  // Shield Pins Declaration  int dataPin = 8;  int latchPin = 12;  int clockPin = 4;  int en = 7;  void **setup**()  {      pinMode(dataPin, OUTPUT); // Setting up the motor shield.      pinMode(latchPin, OUTPUT);      pinMode(clockPin, OUTPUT);      pinMode(en, OUTPUT);      digitalWrite(en, LOW);     forward(); // This funtion for forward robot motion       }  void **loop**()  {  }  void forward(void){          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 3);          digitalWrite(latchPin, HIGH);  }  void backward(void){          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 164);          digitalWrite(latchPin, HIGH);  }  void turn\_left(void){          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 161);          digitalWrite(latchPin, HIGH);  }  void turn\_right(void){          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 38);          digitalWrite(latchPin, HIGH);  }  void halt(void){          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 32);          digitalWrite(latchPin, HIGH);  }  **2.2 Programming Obstacle Avoiding Robot**  Till the above step you have made all of the connection and your motors are connected to the correct terminals. The following programme is based upon following algorithm.  [obstalce2](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/obstalce2%20.png)  This is a very basic algorithm for obstacle avoiding robot. We kept this simple so that beginners can understand the concept. You may improve the programme by enhancing algorithm.  It has following predefined function for robot motion-  1. forward() : forward movement of robot.  2. backward() : backward movement of robot.  3. turn\_left() : for turning left.  4. turn\_right(): for turning right.  5. halt() : for stopping robot.  [*You may download this sketch from here.*](http://roboindia.com/tutorials/admin/source32145898/tutorials%20Images/arduino-obstacle-avoider/obstacle_avoiding_robot.zip)  // Tutorial Obstacle avoiding robot    #include <Servo.h> // Includes servo library.  #include  // Includes SR-04 Sensor Library.  Ultrasonic ultrasonic(A0,A1); // (Trig PIN,Echo PIN)  Servo servo\_1; // Creating Servo object.  // declaring Motor Shield  int dataPin = 8;  int latchPin = 12;  int clockPin = 4;  int en = 7;  // Variable to store distance  int left\_d = 0;  int right\_d = 0;  int front\_d = 0;  int max\_d = 50; // Max distance to obastacle  void **setup**()  {      // setting up shield.      pinMode(dataPin, OUTPUT);      pinMode(latchPin, OUTPUT);      pinMode(clockPin, OUTPUT);      pinMode(en, OUTPUT);      digitalWrite(en, LOW);        servo\_1.attach(10); // Attaching servo to Pin No.10      servo\_1.write(90); // Initial position      delay(350);    }      void **loop**()  {     front\_d = ultrasonic.Ranging(CM); // measuring fornt distance     if (front\_d < max\_d)     {       halt();       get\_d();       if(right\_d > max\_d)       {         turn\_right();         delay(400);         forward();       }       else if ( left\_d > max\_d)       {         turn\_left();         delay(400);         forward();       }       else {         backward();         delay (500);         halt();        }     }     else{      forward();     }  }  void forward(void){ // function for forward movement.          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 3);          digitalWrite(latchPin, HIGH);  }  void backward(void){ // function for forward movement.          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 164);          digitalWrite(latchPin, HIGH);  }  void turn\_left(void){ // function for left turn.          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 161);          digitalWrite(latchPin, HIGH);  }  void turn\_right(void){ // function for Right turn.          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 38);          digitalWrite(latchPin, HIGH);  }  void halt(void){ // function for stopping robot.          digitalWrite(latchPin, LOW);          shiftOut(dataPin, clockPin, LSBFIRST, 32);          digitalWrite(latchPin, HIGH);  }  void get\_d(void) // Fuction to get distances.  {      servo\_1.write(0); // Right Position      delay(500);      right\_d = ultrasonic.Ranging(CM);      servo\_1.write(90); // Front Positon      delay(500);      front\_d = ultrasonic.Ranging(CM);      servo\_1.write(180); // Left position of servo      delay(500);      left\_d = ultrasonic.Ranging(CM);      servo\_1.write(90); // back to front      delay(250);  }  **3. Resources:**  [*Robo India's SR-04 Ultrasonic Sensor Tutorial.*](http://roboindia.com/tutorial.php?tutorial_id=28)  [*Robo India's Arduino Servo control Tutorial*](http://roboindia.com/tutorial.php?tutorial_id=15)  [*This motor shield is based on Adafruit Motor Shield. Resources can be found here.*](https://www.adafruit.com/products/81) |  |  |  | |  |  |  |  | |

Using Sensors :

**ARDUINO MOTION SENSOR ALARM TUTORial**

[](http://301o583r8shhildde3s0vcnh.wpengine.netdna-cdn.com/wp-content/uploads/2014/06/pirPinout.jpg)Have you ever wanted to build a project that had could detect the presence of a person in a room? If so, then you can do this very easily using the PIR (Passive Infra Red) Motion sensor. This arduino motion sensor can detect the presence of a person in a room. Therefore, you can build projects such as burglar alarms, automated appliances, etc. Attach this motion sensor along with an arduino and place it in your room to build an intruder detection system.

This tutorial will show you to interface a motion sensor with an arduino and to use it to build a burglar alarm. This system detects the presence of an intruder in your room and sends a signal to the arduino. The arduino then creates an alarm sound using a buzzer to scare the intruder away.

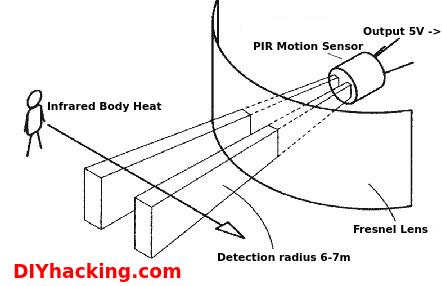
**What are the stuff required to do this project?**

Hardware:

1. [Arduino](http://robokits.co.in/shop/index.php?main_page=product_info&cPath=6_72&products_id=388)
2. PIR [Motion Sensor](http://www.digibay.in/pir-motion-sensor-passive-infrared-sensor).
3. A [piezo](http://probots.co.in/index.php?main_page=product_info&cPath=15_30&products_id=53&zenid=e34d797d5e63611814ef9d0d4bf0571a" \t "_blank) buzzer or an 8 ohm [speaker](http://www.ebay.in/itm/2pcs-Speaker-8Ohm-for-Electronic-Projects-DIYs-/131354513851?pt=IN_Home_Decor&hash=item1e9556d1bb).
4. 9V Battery and connector.
5. Connecting wires.

**How does it work?**

Here, we are using a PIR motion sensor. PIR stands for Passive InfraRed. This motion sensor consists of a fresnel lens, a infrared detector and supporting detection circuitry. The lens on the sensor focuses any infrared radiation present around it towards the infrared detector. Our bodies generate infrared heat and as a result this gets picked up by the motion sensor. The sensor outputs a 5V signal for a period of one minute as soon as it detects the presence of a person. It offers a tentative range of detection of about 6-7 m and is highly sensitive.

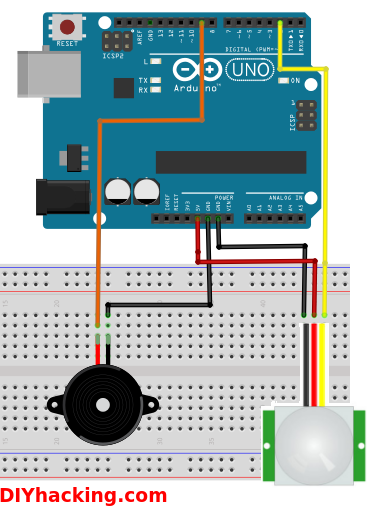
[](http://301o583r8shhildde3s0vcnh.wpengine.netdna-cdn.com/wp-content/uploads/2014/06/PIR.jpg)

When the PIR motion sensor detects a person, it outputs a 5V signal to arduino. Thus, an interrupt on the arduino is triggered. And we define what the arduino should do as it detects an intruder. Here, we are creating an alarm sound through a piezo buzzer. Therefore, when the sensor detects an intruder, an alarm sound will get triggered through the buzzer.

Working of a PIR motion sensor

[](http://301o583r8shhildde3s0vcnh.wpengine.netdna-cdn.com/wp-content/uploads/2014/10/BurglarAlarmWorking-min.jpg)The piezo buzzer is activated through the arduino using PWM signals. The source code for this project at the end of this tutorial will show you how to do this.

#### Step 1: Connecting the PIR arduino motion sensor

[](http://301o583r8shhildde3s0vcnh.wpengine.netdna-cdn.com/wp-content/uploads/2014/11/connection.png)The connections required for interfacing the arduino motion sensor and the piezo buzzer is really simple. Connect the motion sensor to arduino as per the following connection diagram. Connect the Vcc and GND of the sensor to arduino’s 5V and GND pins. Next, connect the Output signal pin of the motion sensor to arduino’s digital pin no 2 (interrupt pin 0).

After hooking up the arduino motion sensor, we have to connect the piezo buzzer to this system. To do this, connect the negative terminal of the buzzer (black wire) to arduino’s GND pin. And the positive terminal of the buzzer (red wire) to arduino’s digital pin no 9.

#### Step 2: Uploading the code for the arduino motion sensor and piezo buzzer

Now, we have to test the arduino motion sensor along with the piezo buzzer to see if it works. To do this, download the code for the arduino motion sensor project from [here](http://diyhacking.com/projects/MotionSensorAlarm.ino). Next, open this code in the arduino IDE. Upload the code to your arduino. Check if it works by opening the serial monitor (at a baud rate setting of 115200bps).

Note that the motion sensor will take a minute to get calibrated with the surroundings after you power it up. The sensor gets activated whenever you are within 6-7 m of the radius of the arduino PIR motion sensor. After which, the piezo buzzer will begin to make an alarm sound. And the string – “Intruder detected” gets printed on your serial monitor. The sensor outputs a LOW signal (0V) after a minute when you are no longer near the radius of the PIR motion sensor. Depending on the PIR motion sensor you posses, it’s sensitivity and delay time could be adjusted. So, adjust it correspondingly till it reaches the sweet spot.

/\*

Arduino Motion Sensor Burglar Alarm Project

\*/

int speakerOut = 9;//Piezo buzzer's positive terminal is connected to digital pin 9

byte names[] = {'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C'};

int tones[] = {1915, 1700, 1519, 1432, 1275, 1136, 1014, 956};

byte melody[] = "2d2a1f2c2d2a2d2c2f2d2a2c2d2a1f2c2d2a2a2g2p8p8p8p";

// count length: 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0

// 10 20 30

int count = 0;

int count2 = 0;

int count3 = 0;

int MAX\_COUNT = 24;

int statePin = LOW;

void siren();

volatile byte intruder;

void setup()

{

Serial.begin(115200);

attachInterrupt(0, intruder\_detect, RISING);//Initialize the intterrupt pin for the motion sensor (Arduino digital pin 2)

intruder = 0;

}

void loop()

{

}

void intruder\_detect()//This function is called whenever an intruder is detected by the arduino

{

intruder++;

Serial.println("Intruder detected");

for(int i=0; i<3; i++)//Play the alarm three times

siren();

}

void siren()//This function will make the alarm sound using the piezo buzzer

{

for (count = 0; count < MAX\_COUNT; count++) {

for (count3 = 0; count3 <= (melody[count\*2] - 48) \* 30; count3++) {

for (count2=0;count2<8;count2++) {

if (names[count2] == melody[count\*2 + 1]) {

analogWrite(speakerOut,1023);

delayMicroseconds(tones[count2]);

analogWrite(speakerOut, 0);

delayMicroseconds(tones[count2]);

}

if (melody[count\*2 + 1] == 'p') {

// make a pause of a certain size

analogWrite(speakerOut, 0);

delayMicroseconds(100);

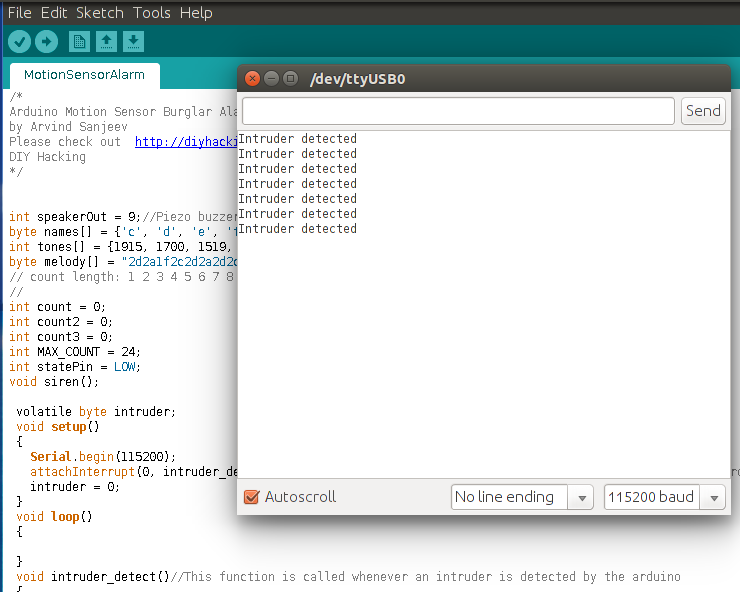
}

}

}

}

}

[](http://301o583r8shhildde3s0vcnh.wpengine.netdna-cdn.com/wp-content/uploads/2014/11/monitor.png)

Arduino Motion Sensor serial monitor

Here, whenever the PIR motion sensor detects a person, it outputs a digital HIGH signal to arduino’s digital pin 2. This immediately triggers an interrupt function called: intruder\_detect(). You can program the arduino to do something when it detects an intruder by editing the function: intruder\_detect(). Currently, it first prints the string: “Intruder detected”. Next, it runs a function called: siren() three times. The function: siren(), consists of the code required for creating the alarm sound on the piezo buzzer or speaker. It does this by writing PWM signals of varying period and frequencies to the piezo buzzer.You will also notice that there is a delay associated with the motion sensor after each detection. Depending on the sensor, you may be able to adjust this delay. So, use this arduino motion sensor to build burglar alarm systems, home automation systems or any simple gadget which prevents people from getting into your room ;)

# Arduino Sound Detection Sensor Tutorial

# Basic Description

[](http://www.shareasale.com/r.cfm?u=1129382&b=497343&m=48335&afftrack=&urllink=www%2Eicstation%2Ecom%2Fsound%2Ddetection%2Dsensor%2Dmodule%2Dsmart%2Dvehicle%2Darduino%2Dp%2D1690%2Ehtml)This module allows you to detect when sound has exceeded a set point you select.  Sound is detected via a microphone and fed into an LM393 op amp.

The sound level set point is adjusted via an on board potentiometer.    When the sound level exceeds the set point, an LED on the module is illuminated and the output is sent low.

# Uses for the Arduino Sound Detector

Given that this device measures whether or not sound has exceeded a threshold,  you’re basically left with determining what it is you want to do.   What I mean by this is that you can do something when it is quiet and/or you can do something when it is loud.  For example:

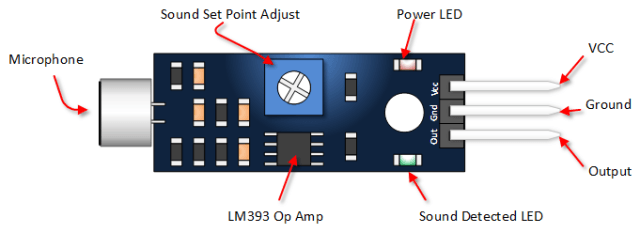
* You could detect whether or not a motor is running.
* You could set a threshold on pump sound so that you know whether or not there is cavitation.
* In the presence of no sound,  you might want to create an ambiance by turning on music.
* In the presence of no sound and no motion, you may go into an energy savings mode and turn off the lights.

# Arduino Sound Detection Sensor Pin Outs

The image and table below detail the controls, pin outs, and other key components.

When referring sensititivity, I mean this:

* When less sensitive,  it takes more sound to trigger the device
* When more sensitive, it takes less sound to trigger the device

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/07/Arduino-Sound-Detection-Sensor-Pin-Outs.png)

| **Parameter** | **Value** |
| --- | --- |
| VCC | 5 Vdc from your Arduino |
| Ground | GND from your Arduino |
| Out | Connect to Digital Input Pin |
| Power LED | Illuminates when power is applied |
| Sound Detection LED | Illuminates when sound is detected |
| Sound Set Point Adjust | CW = More Sensitive CCW = Less Sensitive |

# Arduino Sound Detection Sensor Tutorial

## Connect the Sound Sensor Module to your Arduino

This is your typical three pin hook up.

## [Arduino Sound Detection Sensor Tutorial](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/07/Arduino-Sound-Detection-Sensor-Tutorial.png)

## Copy This Tutorial Sketch and Upload It

//Arduino Sound Detection Sensor Module

int soundDetectedPin = 10; // Use Pin 10 as our Input

int soundDetectedVal = HIGH; // This is where we record our Sound Measurement

boolean bAlarm = false;

unsigned long lastSoundDetectTime; // Record the time that we measured a sound

int soundAlarmTime = 500; // Number of milli seconds to keep the sound alarm high

void setup () {

Serial.begin(9600);

pinMode (soundDetectedPin, INPUT) ; // input from the Sound Detection Module

}

void loop () {

soundDetectedVal = digitalRead (soundDetectedPin) ; // read the sound alarm time

if (soundDetectedVal == LOW) // If we hear a sound

{

lastSoundDetectTime = millis(); // record the time of the sound alarm

// The following is so you don't scroll on the output screen

if (!bAlarm){

Serial.println("LOUD, LOUD");

bAlarm = true;

}

}

else

{

if( (millis()-lastSoundDetectTime) > soundAlarmTime && bAlarm){

Serial.println("quiet");

bAlarm = false;

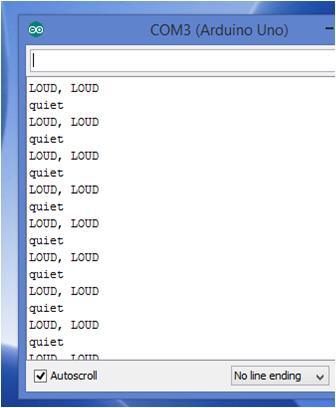
}

}

}

Run The Sketch and Verify Output

Once your sketch is running,  you will want to open your serial monitor.    Make some loud noises and view the result.   Your output should look something like the picture below:

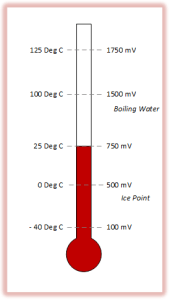
[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/07/Arduino-Sound-Detection-Moduel-Tutorial-Output.jpg)

# Simple TMP 36 Arduino Thermometer User Manual

# Easy to Use Temperature Sensor

[](http://www.amazon.com/gp/product/B007STHA22/ref=as_li_qf_sp_asin_il_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B007STHA22&linkCode=as2&tag=leaacicarbatf-20)It really doesn’t get much easier to measure temperature with an Arduino than with Analog Device’s TMP36 temperature sensor.  Supply it 5 volts and ground from your Arduino, connect it’s output to an analog pin, add a little code and you’re measuring temperature.

# Power Requirements

The TMP 36 needs power that is between 2.7 and 5.5 volts.  On your Arduino, you have two pins that supply voltages in this range.  The first is the 3.3V output and the other is the 5V output.  In my test, I used the 5 volt output.

**Accuracy**

The manufacturer’s data sheet indicates that the typical accuracy is plus or minus one degree C.   The worse case accuracy is stated as +/- 2 degrees C.

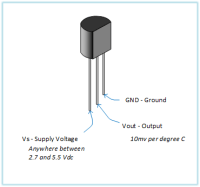
**The Range**

This sensor is suppose to have a range from -40 degrees C to +125 degree C.   For my American brothers (who are going metric inch by inch), that’s from -40 to 257 degrees F.

# The Output (or Scale Factor)

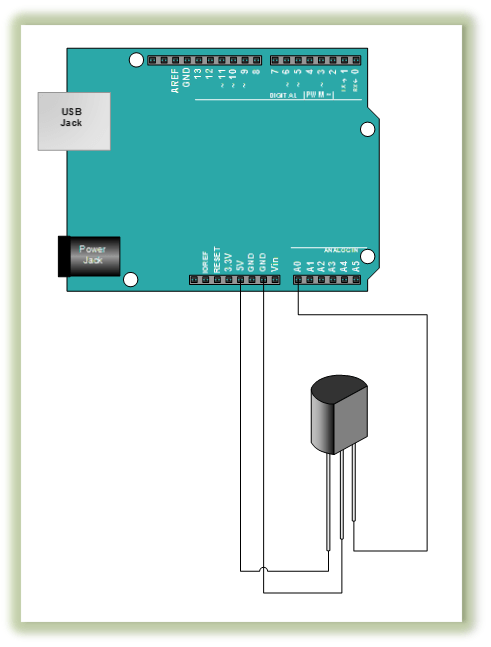
Zero degrees Celsius (ice point) is 500 mV.  This is often referred to as the offset.Every degree of change from that point will either increase or decrease this value by 10 mV (the scale factor).   So at five degrees Celsius, the device would output 550 mV.  At a negative 10 degrees Celsius, the output will be 400 mV.

# The TMP36 Pin outs

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2014/08/Pin-Out.png)The TMP36 pinouts  are identified in the diagram on the left.  You can click on it to get a larger view.

# TMP 36 Connections

As I mentioned before, you need just three connections and a USB cable to your computer.  See below.

[](http://i2.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2014/08/TMP-36-Hook-Up.png)

# Comments About My Code

* I wanted to see the measurement in millivolts, so I mutliplied my input value by 5000.
* I wanted to see my measurement in degrees F, so I converted the degrees C to F.  Its really simple.  All you have to do is multiply the measurement in C by 1.8 and then add 32.

# The Code

/\*The simplest TMP 36 Thermometer\*/

const int analogIn = A0;

int RawValue= 0;

double Voltage = 0;

double tempC = 0;

double tempF = 0;

void setup(){

  Serial.begin(9600);

}

void loop(){

RawValue = analogRead(analogIn);

  Voltage = (RawValue / 1023.0) \* 5000; // 5000 to get millivots.

  tempC = (Voltage-500) \* 0.1; // 500 is the offset

  tempF = (tempC \* 1.8) + 32; // conver to F

  Serial.print("Raw Value = " );  // shows pre-scaled value

  Serial.print(RawValue);

  Serial.print("\t milli volts = "); // shows the voltage measured

  Serial.print(Voltage,0); //

  Serial.print("\t Temperature in C = ");

  Serial.print(tempC,1);

  Serial.print("\t Temperature in F = ");

  Serial.println(tempF,1);

  delay(500);

}

### 

# Arduino IR Obstacle Sensor Tutorial

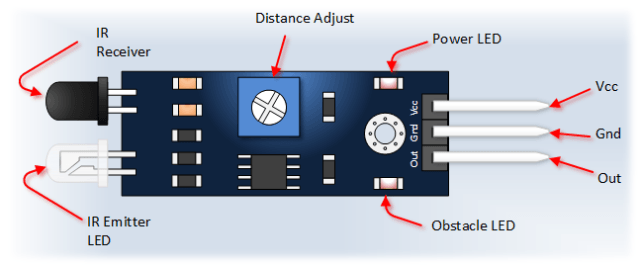
# Arduino Infrared Collision Avoidance

[](http://www.shareasale.com/r.cfm?u=1129382&b=497343&m=48335&afftrack=&urllink=www%2Eicstation%2Ecom%2Fobstacle%2Davoidance%2Dsensor%2Dmodule%2Dinfrared%2Dmodule%2Dreflection%2Dphot%2Dp%2D2432%2Ehtml)This is yet another one of those modules with cool possibilities.   You could for example, sound an alarm when something got too close or you could change the direction of  a robot or vehicle.

The device consists of an Infrared Transmitter, an Infrared Detector, and support circuitry.  It only requires three connections. When it detects an obstacle within range it will send an output low.

# IR Obstacle Detection Module Pin Outs

The drawing and table below identify the function of module pin outs, controls and indicators.

[](http://i1.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/Arduino-IR-Collision-Detection-Module-Pin-Outs.png)

**Pin, Control Indicator Description**

Vcc 3.3 to 5 Vdc Supply Input

Gnd Ground Input

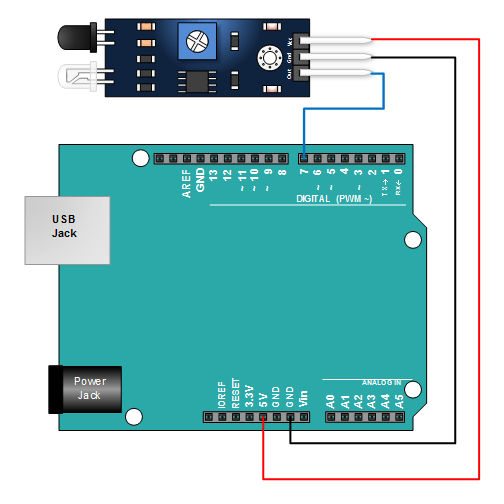
Out Output that goes low when obstacle is in range

Power LED Illuminates when power is applied

Obstacle LED Illuminates when obstacle is detected

Distance Adjust Adjust detection distance. CCW decreases distance.  
 CW increases distance.

IR Emitter Infrared emitter LED

[](http://i1.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/IR-Collision-Detection-Module-Arduino-Tutorial-Hook-Up.png)IR Receiver Infrared receiver that receives signal transmitted by Infrared emitter.

## Connect the Arduino to the Detection Module

Use the picture .  It only requires three wires.

## Copy, Paste and Upload the Sample Sketch

// IR Obstacle Collision Detection Module

int LED = 13; // Use the onboard Uno LED

int isObstaclePin = 7; // This is our input pin

int isObstacle = HIGH; // HIGH MEANS NO OBSTACLE

void setup() {

pinMode(LED, OUTPUT);

pinMode(isObstaclePin, INPUT);

Serial.begin(9600);

}

void loop() {

isObstacle = digitalRead(isObstaclePin);

if (isObstacle == LOW)

{ Serial.println("OBSTACLE!!, OBSTACLE!!");

digitalWrite(LED, HIGH);

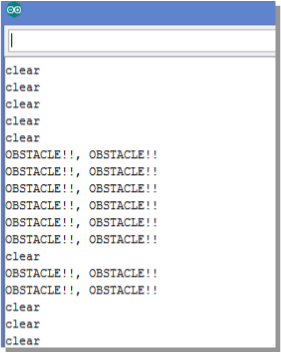
} else {

Serial.println("clear");

digitalWrite(LED, LOW);

} delay(200);

}

[](http://i1.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/IR-Obstacle-Detection-Module-Arduino-Tutorial-Output.png)Test the Tutorial Sketch

Move your hand towards the IR LEDs.  As you near them, the Output LED on the module and the LED for pin 13 on your Arduino will illuminate.  Open your serial monitor and vary the distance of your hand while viewing the serial monitor.   The output should look like the picture :

# Arduino Capacitive Touch Sensor Tutorial

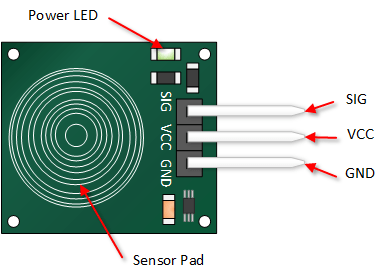
# A Low Cost Reliable Input Device

[](http://www.shareasale.com/r.cfm?u=1129382&b=302497&m=32431&afftrack=&urllink=www%2Edx%2Ecom%2Fp%2Fcatalex%2Ddigital%2Dcapacitive%2Dtouch%2Dsensor%2Dswitch%2Dmodule%2Dfor%2Darduino%2Dblue%2Dblack%2D230867%23%2EVawjmvnIwR4)This device uses your body as part of the circuit.  When you touch the sensor pad, the capacitance of the circuit is changed and is detected.  That detected change in capacitance results in the output changing states.

When I first got this,  I expected a glitchy device,  that while functional,  would occasionally have unpredictable output results.I may have been wrong.   After playing for a few hours, I can’t seem to get it to do anything other than what I expected it to do.  If you’re looking for robust user input,  this might do the trick.

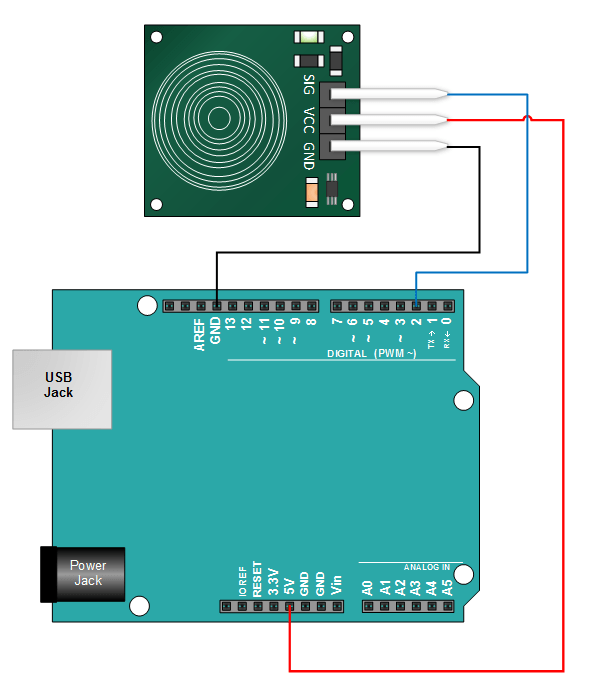
# Capacitive Touch Sensor Pin Outs

Like  a lot of the sensors out there, this is three pin sensor.  You provide, power, ground and monitor the output.

[](http://i2.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/07/Capacitive-Touch-Sensor-Pin-Outs.png)

Connect the Touch Sensor to Your Arduino

This is a real simple set up.  You will know that you have power properly applied when the green LED is on.

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/07/Arduino-Capacitive-Touch-Sensor-Tutorial.png)

**Copy, Paste and Upload the Arduino Sketch**

The sketch below provides an output to your serial monitor indicating whether or not the sensor is pressed.

// Capacitive Touch Sensor Tutorial

// When Sig Output is high, touch sensor is being pressed

#define ctsPin 2 // Pin for capactitive touch sensor

int ledPin = 13; // pin for the LED

void setup() {

Serial.begin(9600);

pinMode(ledPin, OUTPUT);

pinMode(ctsPin, INPUT);

}

void loop() {

int ctsValue = digitalRead(ctsPin);

if (ctsValue == HIGH){

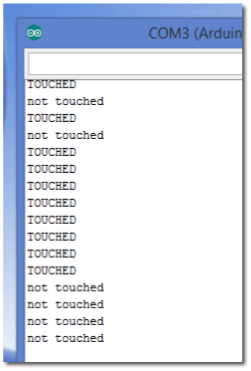
digitalWrite(ledPin, HIGH);

Serial.println("TOUCHED");

} else {

digitalWrite(ledPin,LOW);

Serial.println("not touched");

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/07/Arduino-Capacitive-Touch-Sensor-Tutorial-Output.png) }

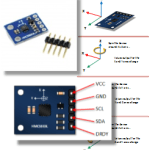
delay(500);

}

Test Your Arduino Sketch

Once you’ve uploaded the sketch, open your serial monitor.   Touch the sensor pad while looking at the monitor.   You should see an output that looks something like the picture below.

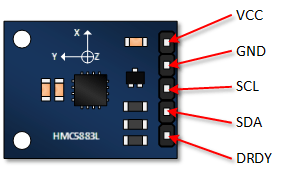
# Arduino Magnetometer Compass (GY-273 HMC5883L) Tutorial

[](http://i2.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/09/GY-273-Arduino-Tutorial.png)Based on the Honeywell HMC5883L,  the GY-273 sensor module allows you add an electronic compass to your projects.  While coding for the device can be a bit of a chore,  there available libraries that make putting it to practical us a snap.

The device communicates with your Arduino via I2c.

This tutorial will show you how to connect the sensor and run sample sketch.  It will also point you to  a fantastic Adafruit library that will make using the device easy.

# GY-273 Sensor Module Pin-outs

The device has five pins.  In most applications  (and this tutorial),  you will only be concerned with four of them.**[](http://i2.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/09/GY-273-Magnetometer-Pin-Outs.png)** This is a pretty complicated device, that by itself could take pages to describe.   This tutorial simply highlights the basics.

The Honeywell HMC5883L sensor used in the module is sensitive to the earths magnetic fields in three axes.  These axes are labeled as X, Y, and Z.     An output is provided for each of these axes that describes the position of these axes relative to the earth’s magnetic field.

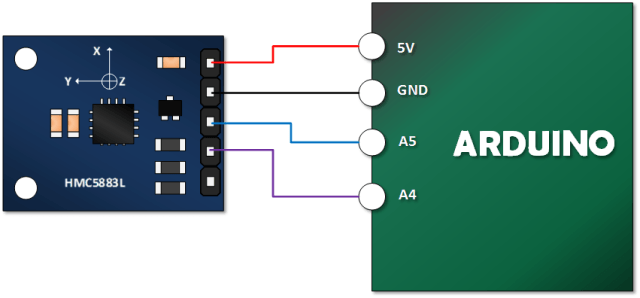
As illustrated, twisting or turning the device  will provide the corresponding outputs.

# [GY-273 Magnetometer Rotation vs Output](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/09/GY-273-Magnetometer-Rotation-vs-Output.png)

In this tutorial, you will connect the GY-273, load a basic sketch and see an output to your serial monitor.  The purpose of this tutorial is to verify that you can read and output and to

## Connect the Arduino to the GY-273

Just four connections are required.  Refer to the picture below.

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/09/GY-273-Magnetometer-Tutorial-hook-up.png)

## Copy, Paste and Upload the GY-273 Example Sketch

#include <Wire.h> //I2C Arduino Library

#define addr 0x1E //I2C Address for The HMC5883

void setup(){

Serial.begin(9600);

Wire.begin();

Wire.beginTransmission(addr); //start talking

Wire.write(0x02); // Set the Register

Wire.write(0x00); // Tell the HMC5883 to Continuously Measure

Wire.endTransmission();

}

void loop(){

int x,y,z; //triple axis data

Wire.beginTransmission(addr); //Tell the HMC what regist to begin writing data into

Wire.write(0x03); //start with register 3.

Wire.endTransmission();

Wire.requestFrom(addr, 6); //Read the data.. 2 bytes for each axis.. 6 total bytes

if(6<=Wire.available()){

x = Wire.read()<<8; //MSB x

x |= Wire.read(); //LSB x

z = Wire.read()<<8; //MSB z

z |= Wire.read(); //LSB z

y = Wire.read()<<8; //MSB y

y |= Wire.read(); //LSB y

} // Show Values

Serial.print("X Value: ");

Serial.println(x);

Serial.print("Y Value: ");

Serial.println(y);

Serial.print("Z Value: ");

Serial.println(z);

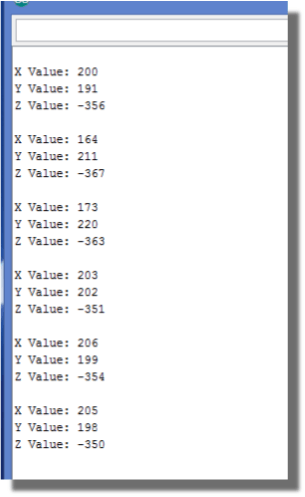
Serial.println();

delay(500);

}

Open the Serial Monitor and Verify Your Output.

Your output should look something like this. Once you’ve established that you have an output, try turning your device along it’s various axes and view the results.

[](http://i1.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/09/GY-273-Arduino-Tutorial-Output.png)

# Arduino Flame Detector Sensor Tutorial

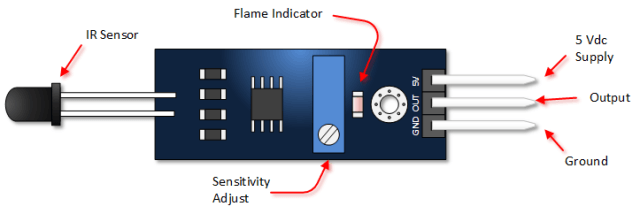
# Overview

[](http://i1.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/Flame-Sensor-Module.jpg)The Logo Version 1.3 Flame Detector module is representative of the many similar devices that are designed to interface with micro-controllers.  This particular device consists of an IR detector, op amp circuitry, sensitivity adjustment and an LED indicator.

The IR detector is sensitive to light wavelengths typical of flames.  When a flame is present, the module will turn on it’s red LED and it will set it’s output low.

# Flame Sensor Module Pin Outs

Only three connections are required to allow this device to work with your Arduino.  The following picture and table describe these pins.

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/flame-sensor-Module.png)

**Pin or Component Function**

5V 5 Vdc supply Input

Out Goes low when flame is detected

GND Ground input

Flame Indicator Illuminates when flame is sensed

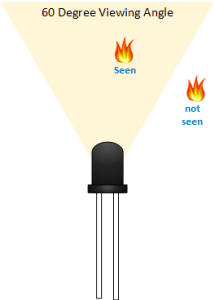
Sensitivity Adjust CW is more sensitive  
 CCW is less sensitive

IR Detector Has a 60 degree view angle and is sensitive to the wavelengths typical of flames

# Application Considerations

## Viewing Angle

As previously mentioned, the viewing angle is at sixty degrees.   Thus the sensor view is incredibly important as you design your projects.

[](http://i0.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/flame-sensor-Module-viewing-angle.png)

## [flame sensor Module arduino hookup](http://i1.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/flame-sensor-Module-arduino-hookup.png)Flame Sensor Module Wavelength

The flame sensor module detects wavelengths from 760nm-1100nm.   The are other sources of heat that will also detect this wavelength.  It is therefore important that you ensure that the only source of this particular range will be the flame that you want to detect.  Otherwise, your project may be riddled with false measurements.

## Spark Dectection?

During the course of testing this device, I used a defective lighter.  It took several attempts to see a flame.   However, each time I tried the sparks from lighter caused the LED to flash.

Connect The Arduino to the Flame Sensor Module

Use the schematic pic to connect your micro-controller to the flame detector.

Copy, Paste and Upload the Arduino Flame Detector Module Sketch

The sketch below is pretty straight forward.   It uses the LED connected to Pin 13 of serial monitor of your Arduino IDE to see the output.

// Flame Sensor Module

int LED = 13; // Use the onboard Uno LED

int isFlamePin = 7; // This is our input pin

int isFlame = HIGH; // HIGH MEANS NO FLAME

void setup() {

pinMode(LED, OUTPUT);

pinMode(isFlamePin, INPUT);

Serial.begin(9600);

} void loop() {

isFlame = digitalRead(isFlamePin);

if (isFlame== LOW)

{ Serial.println("FLAME, FLAME, FLAME");

digitalWrite(LED, HIGH);

} else {

Serial.println("no flame");

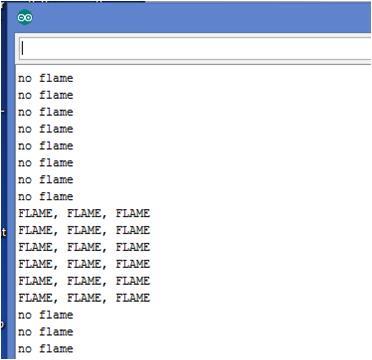
digitalWrite(LED, LOW);

}

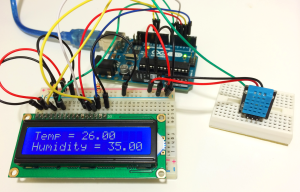
}

Verify Operation of the Flame Detector Module and Adjust Sensitivity

Open the Serial Monitor on your Arduino program.  Move a flame in and out of the viewing angle of the sensor.  You should see an output that looks something like the picture below.   You should also see the red LED illuminate on your module and you should see also see the module LED connected to pin 13 of your Arduino light up.

[](http://i2.wp.com/henrysbench.capnfatz.com/wp-content/uploads/2015/06/Flame-Sensor-Module-Tutorial-Output.jpg)

# Arduino Humidity Sensor (DHT11) Tutorial

[](http://i0.wp.com/www.circuitbasics.com/wp-content/uploads/2015/09/Arduino-DHT11-Humidity-and-Temperature-Sensor-With-LCD-Output.png)

Because of their low cost and small size, [DHT11 humidity and temperature](http://www.amazon.com/gp/product/B00K8PR16I/ref=as_li_qf_sp_asin_il_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B00K8PR16I&linkCode=as2&tag=circbasi-20&linkId=O2PUVUZRKYXH4RTS) sensors are perfect for lots of different DIY electronics projects. Some projects where the DHT11 would be useful include remote weather stations, home environment control systems, and agricultural/garden monitoring systems. The DHT11 is a digital sensor that lets you easily get relative humidity and temperature readings in your projects. In this post, I’ll first go into a little background on what humidity is, then I’ll explain how the DHT11 measures humidity. After that, I’ll show you how to connect the DHT11 to the [Arduino](http://www.amazon.com/gp/product/B008GRTSV6/ref=as_li_qf_sp_asin_il_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B008GRTSV6&linkCode=as2&tag=circbasi-20&linkId=SO3YILYNYLXWVBKI) and give you some example code so you can use the DHT11 in your own projects.

DHT11 Technical Specifications:

* Humidity Range: 20-90% RH
* Humidity Accuracy: ±5% RH
* Temperature Range: 0-50 °C
* Temperature Accuracy: ±2% °C
* Operating Voltage: 3V to 5.5V

## What is Relative Humidity?

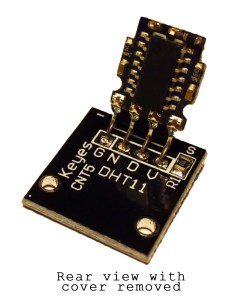
The DHT11 humidity and temperature sensor measures relative humidity (RH) and temperature. Relative humidity is the ratio of water vapor in air vs. the saturation point of water vapor in air. The saturation point of water vapor in air changes with temperature. Cold air can hold less water vapor before it is saturated, and hot air can hold more water vapor before it is saturated. The formula for relative humidity is as follows:

Relative Humidity = (density of water vapor / density of water vapor at saturation) x 100%

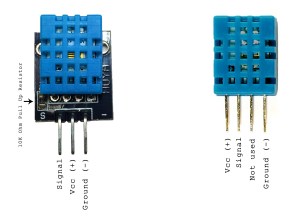
Basically, relative humidity is the amount of water in the air compared to the amount of water that air can hold before condensation occurs. It’s expressed as a percentage.  For example, at 100% RH condensation (or rain) occurs, and at 0% RH, the air is completely dry.

## How the DHT11 Measures Humidity and Temperature

[](http://i2.wp.com/www.circuitbasics.com/wp-content/uploads/2015/09/DHT11-Temperature-and-Humidity-Sensor-Inside-Front-with-Cover-Removed.jpg)The DHT11 calculates relative humidity by measuring the electrical resistance between two electrodes. The humidity sensing component of the DHT11 is a moisture holding substrate (usually a salt or conductive plastic polymer) with the electrodes applied to the surface. When water vapor is absorbed by the substrate, ions are released by the substrate which increases the conductivity between the electrodes. The change in resistance between the two electrodes is proportional to the relative humidity. Higher relative humidity decreases the resistance between the electrodes while lower relative humidity increases the resistance between the electrodes. Inside the DHT11 you can see electrodes applied to a substrate on the front of the chip:

[](http://i0.wp.com/www.circuitbasics.com/wp-content/uploads/2015/09/DHT11-Temperature-and-Humidity-Sensor-Inside-Back-with-Cover-Removed.jpg)The DHT11 converts the resistance measurement to relative humidity on an IC mounted to the back of the unit and transmits the humidity and temperature readings directly to the Arduino. This IC also stores the calibration coefficients and controls the data signal transmission between the DHT11 and the Arduino:

The temperature readings from the DHT11 come from a surface mounted [NTC temperature sensor](http://www.amazon.com/gp/product/B00GD471PO/ref=as_li_qf_sp_asin_il_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B00GD471PO&linkCode=as2&tag=circbasi-20&linkId=UJTHZ5Z3JDMKGOCK) (thermistor) built into the unit. To learn more about thermistors and how to use them on the Arduino, check out our [Arduino Thermistor Temperature Sensor Tutorial](http://www.circuitbasics.com/arduino-thermistor-temperature-sensor-tutorial).

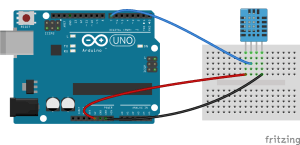
[](http://i2.wp.com/www.circuitbasics.com/wp-content/uploads/2015/10/DHT11-Pinout-for-three-pin-and-four-pin-types-2.jpg)

The DHT11 uses one signal wire to transmit sensor readings to the Arduino digitally. The power comes from separate 5V and ground wires. A 5K – 10K Ohm pull-up resistor is connected from the signal line to 5V to make sure the signal level stays high by default (see the datasheet for specifics on how the signal is sent).

There are two different variations of the DHT11 sensor you might come across. One type has four pins, and the other type is mounted to a small PCB that has three pins. The PCB mounted version with three pins is nice since it includes a surface mounted 10K Ohm pull up resistor for the signal line:

## How to Set Up the DHT11 on an Arduino

The DHT11 is really easy to connect to the Arduino:

[](http://i2.wp.com/www.circuitbasics.com/wp-content/uploads/2015/10/DHT11-Humidity-and-Temperature-Sensor-2.png)

## Code for Serial Monitor Output of Readings

Before we can use the DHT11 on the Arduino, we need to install the [DHTLib library](http://playground.arduino.cc/Main/DHTLib" \o "DHTLib Library" \t "_blank), which contains all of the functions we will need to get the humidity and temperature readings from the sensor. It’s easy to install. Just download the DHTLib.zip file below, and open up the Arduino IDE. Then go to Sketch>Include Library>Add Library and select the DHTLib.zip file.

http://i1.wp.com/www.circuitbasics.com/wp-content/plugins/download-manager/assets/file-type-icons/zip.png?w=750

**DHTLib.zip**2.41 KB

[**Download**](http://www.circuitbasics.com/how-to-set-up-the-dht11-humidity-sensor-on-an-arduino/)

After the library is installed, upload this example program to the Arduino, and open the serial monitor. You should see the humidity and temperature readings displayed at one second intervals:

### #include <dht.h>

### dht DHT;

### #define DHT11\_PIN 7

### void setup() {

### Serial.begin(9600);

### } void loop() {

### int chk = DHT.read11(DHT11\_PIN);

### Serial.print("Temperature = ");

### Serial.println(DHT.temperature);

### Serial.print("Humidity = ");

### Serial.println(DHT.humidity);

### delay(1000);

### }

### This simple program will take humidity and temperature data from the sensor and print it to the serial monitor. What if you don’t need to actually see the humidity and temperature readings, but need them to calculate or control other things? Just use DHT.humidity and DHT.temperature as variables in any function and include the following lines of code in your program to initialize the sensor. The DHT11 signal pin (in #define DHT11\_PIN 7) can be changed to any other available digital pin:

### #include <dht.h>

### dht DHT;

### #define DHT11\_PIN 7

### void setup(){

### }

### void loop() {

### int chk = DHT.read11(DHT11\_PIN);

### delay(1000);

### }

### Output Humidity and Temperature Readings to an LCD Display

### If you do want the actual humidity and temperature readings from the DHT11, a nice way to display them is on a[16X2 LCD](http://www.amazon.com/gp/product/B00E5YJCBA/ref=as_li_qf_sp_asin_il_tl?ie=UTF8&camp=1789&creative=9325&creativeASIN=B00E5YJCBA&linkCode=as2&tag=circbasi-20&linkId=B4272KMLIJ4QDXHU). To do this, first follow our tutorial on [How to Set Up an LCD Display on an Arduino](http://www.circuitbasics.com/how-to-set-up-an-lcd-display-on-an-arduino/), then upload this code to the Arduino:

### #include <dht.h>

### #include <LiquidCrystal.h>

### LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

### dht DHT;

### #define DHT11\_PIN 7

### void setup() {

### lcd.begin(16, 2);

### }

### void loop() {

### int chk = DHT.read11(DHT11\_PIN);

### lcd.setCursor(0,0);

### lcd.print("Temp: ");

### lcd.print(DHT.temperature);

### lcd.print((char)223);

### lcd.print("C");

### lcd.setCursor(0,1);

### lcd.print("Humidity: ");

### lcd.print(DHT.humidity);

### lcd.print("%");

### delay(1000);

### }