# Overview

A sound sensor is a component that receives sound waves and converts them into electrical signal. It detects the sound intensity in ambient environment like a microphone. The microphone on the sensor module can convert audio signals into electrical signals (analog quantity). We must then convert the analog signal into a digital signal by using a separate chip before we can access it in the Raspberry Pi.

# Step 1: Connect It

Use the following wiring diagram to connect the sensor to your Raspberry Pi



**Important Notes:**

* The wire colors do not matter. Just use whatever is available at your station.
* **Make sure that you are plugging the wires into the correct location on both the Pi, the Sensor, and the analog to digital converter (especially the power and ground)**
* The wires move around a lot, so make sure they are firmly in place when you are testing

# Step 2: Code It

We have provided you with a Python code template that can run this sensor/device. The code is located at:

**/home/pi/Documents/rpi-iot-demos/sound.py**

Use the Geany editor to open this file (NOTE: this will be slow, so be patient). Once it is open, take some time to read the code to see how it works.

Once you are ready, do the following:

1. Find the function called loop().
   1. This function starts when the script runs, and runs forever
2. Modify this function so that it gets a reading (using the read() function), and prints the value out to the screen.

def loop():

count = 0

while True:

voiceValue = ADC.read(0)

if voiceValue:

print ("Value:", voiceValue)

if voiceValue < 90:

print ("I’m hearing something!!!")

time.sleep(0.2)

# Step 3: Run / Test It

1. To run the script, open a Linux console and navigate to the folder with your code
   1. Helpful Linux Commands:  
      **ls** lists the contents of the directory  
      **cd <folder\_name>** opens a folder (don’t type the < > characters)  
      **cd ..** exits the folder you are currently in  
      **cd ~** takes you back to your home folder
2. Run the python program by typing the following:

**python3 sound.py**

1. Try making noises and see what happens!
   1. If it doesn’t work, try getting closer to the microphone
   2. You will have to tune the code to reliably differentiate between sounds and background noise. Try changing the 90 in the code to other values to see if you can find a good threshold value.
2. **To exit the program, press Ctrl-C in the terminal**

# Step 4: Make It Control Something

To make your program send MQTT messages, modify your main program to look like the following:

# --------------------------------------------

# Main Program Starts Here

# --------------------------------------------

if \_\_name\_\_ == '\_\_main\_\_':

try:

comm.connect()

setup()

print("Program Running. Waiting for a Sound")

loop()

except KeyboardInterrupt:

pass

Now, go to the loop() function you modified earlier. You can use the comm.send function to send a message when the sound is heard, etc.

comm.send("CHANNEL NAME GOES HERE", "MESSAGE GOES HERE")