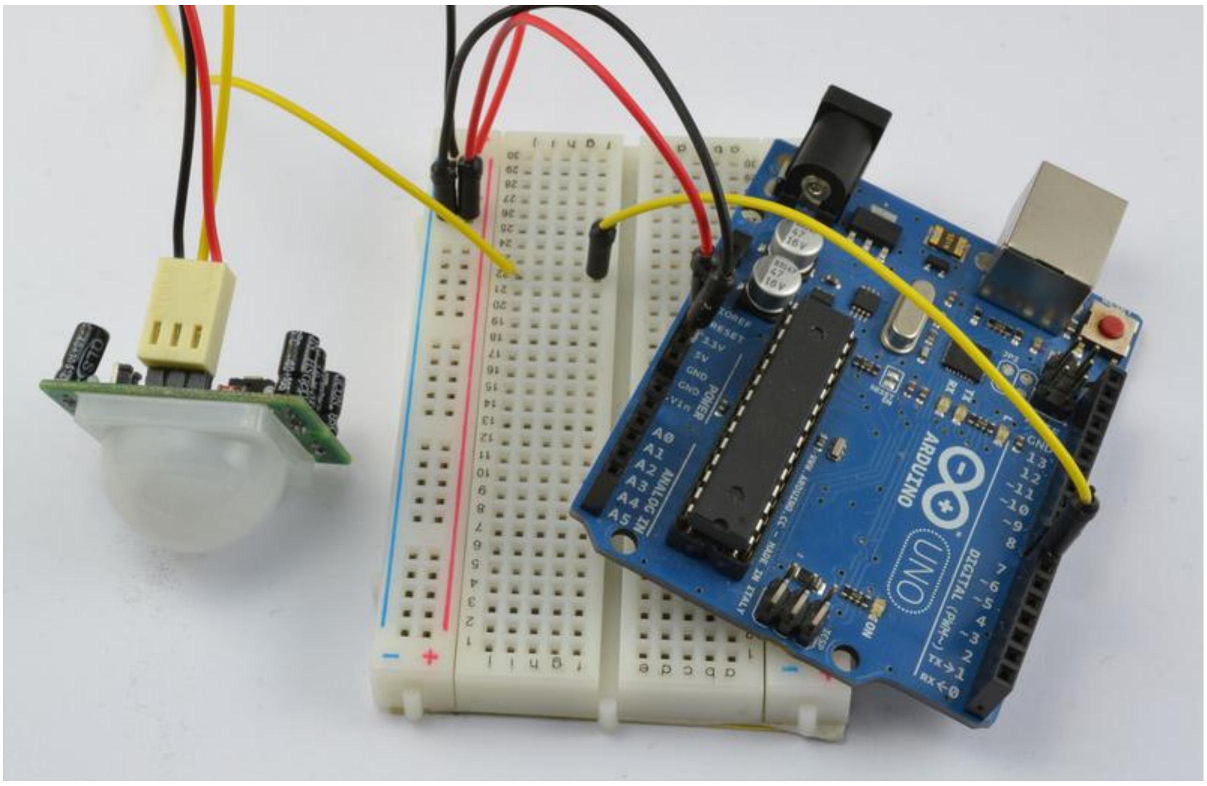
Arduino Lesson 17. Email Sending Movement Detector

Created by Simon Monk



https://learn.adafruit.com/arduino-lesson-17-email-sending-movement-detector

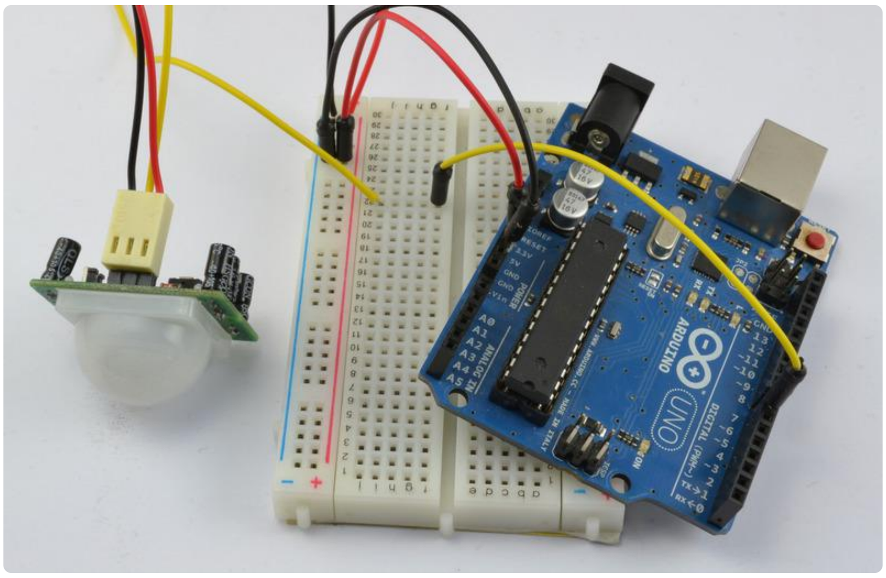
Last updated on 2023-08-29 02:16:52 PM EDT

Table of Contents

Overview

In this lesson you will learn how to use a PIR movement detector with an Arduino and to have the Arduino communicate with a Python program running on your computer to send an email whenever movement is detected by the sensor.

The Arduino is the heart of this project. It 'listens' to the PIR sensor and when motion is detect, instructs the computer via the USB port to send an email.



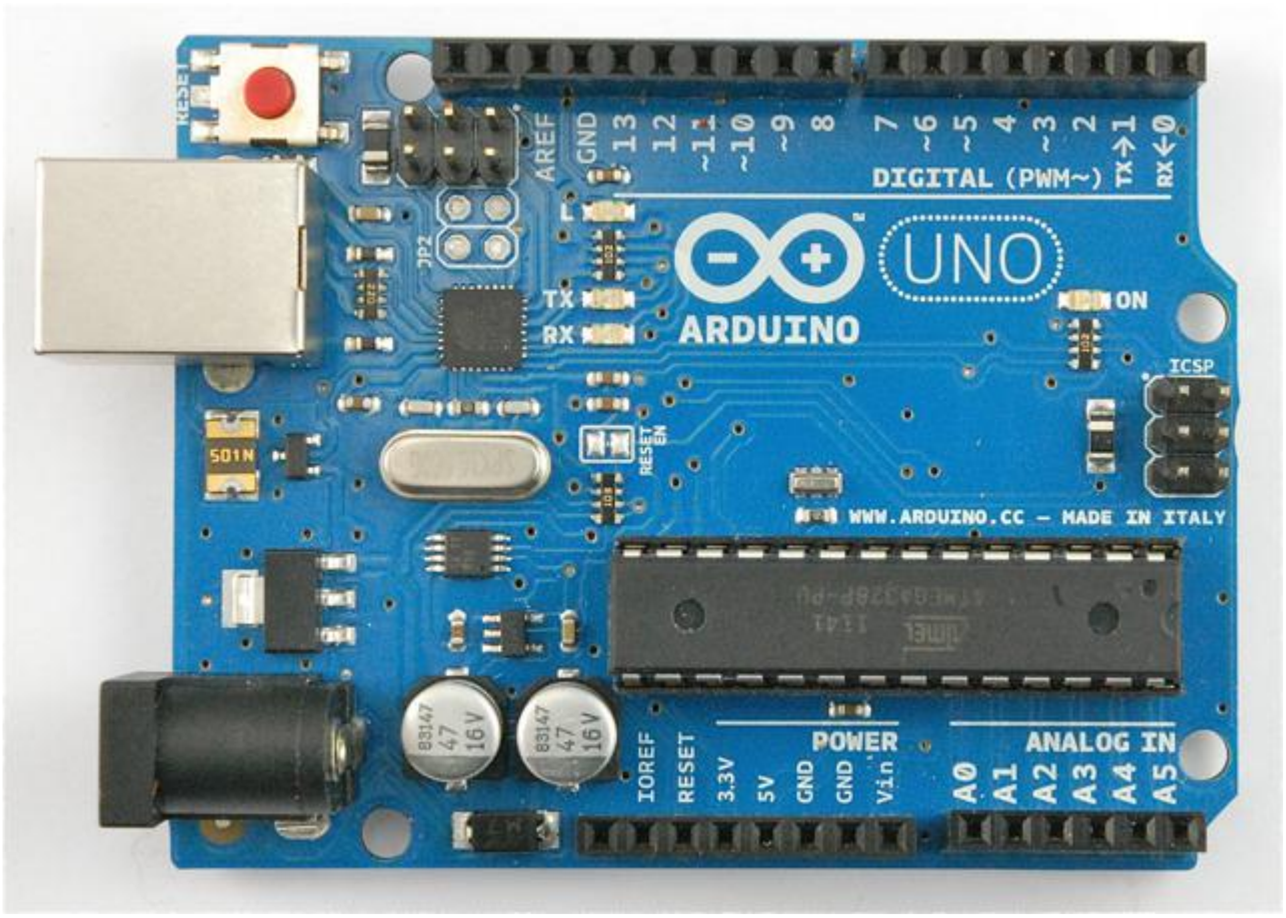
Parts

To build the project described in this lesson, you will need the following parts.

You will also need a computer with an Internet connection (so you can send email thru it)!

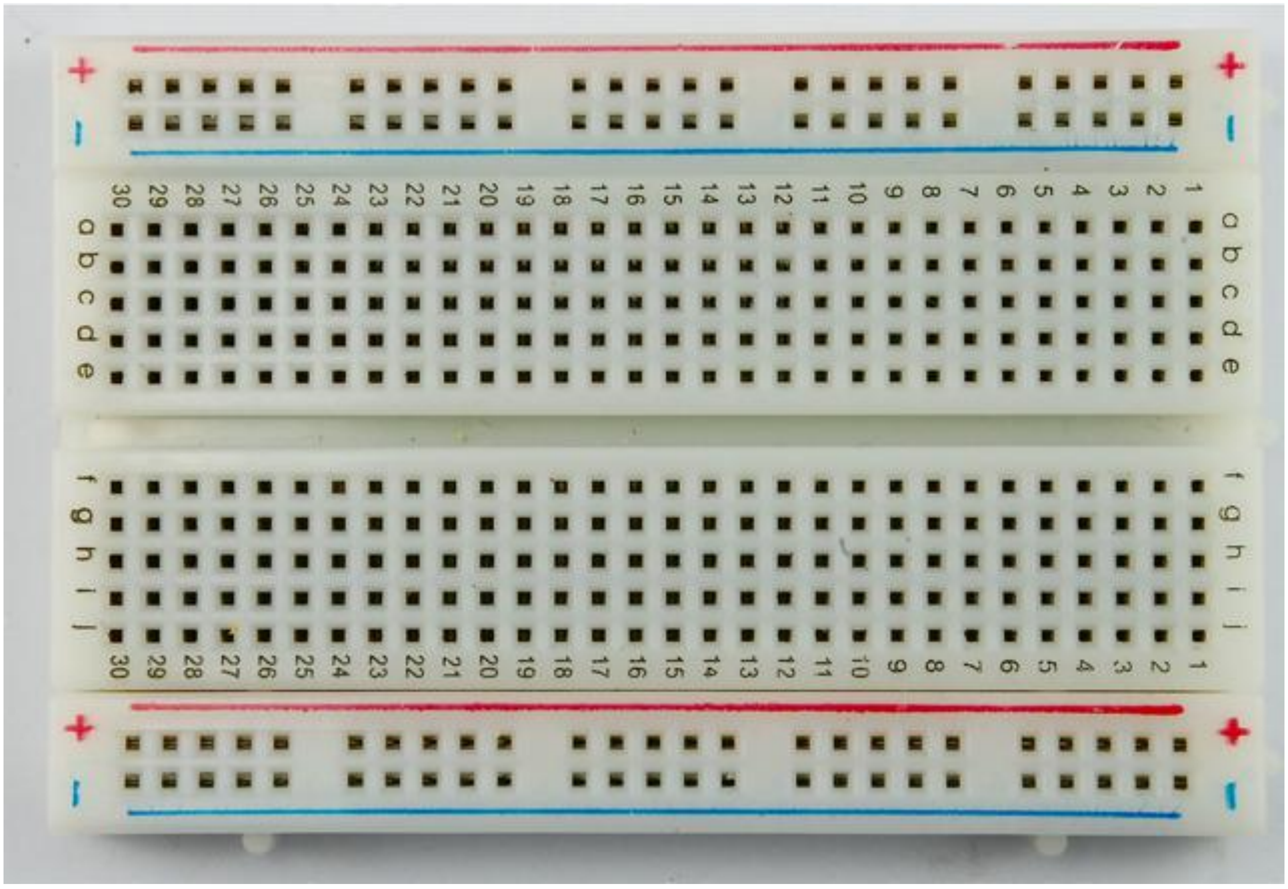
Part
Qty

PIR Sensor 1

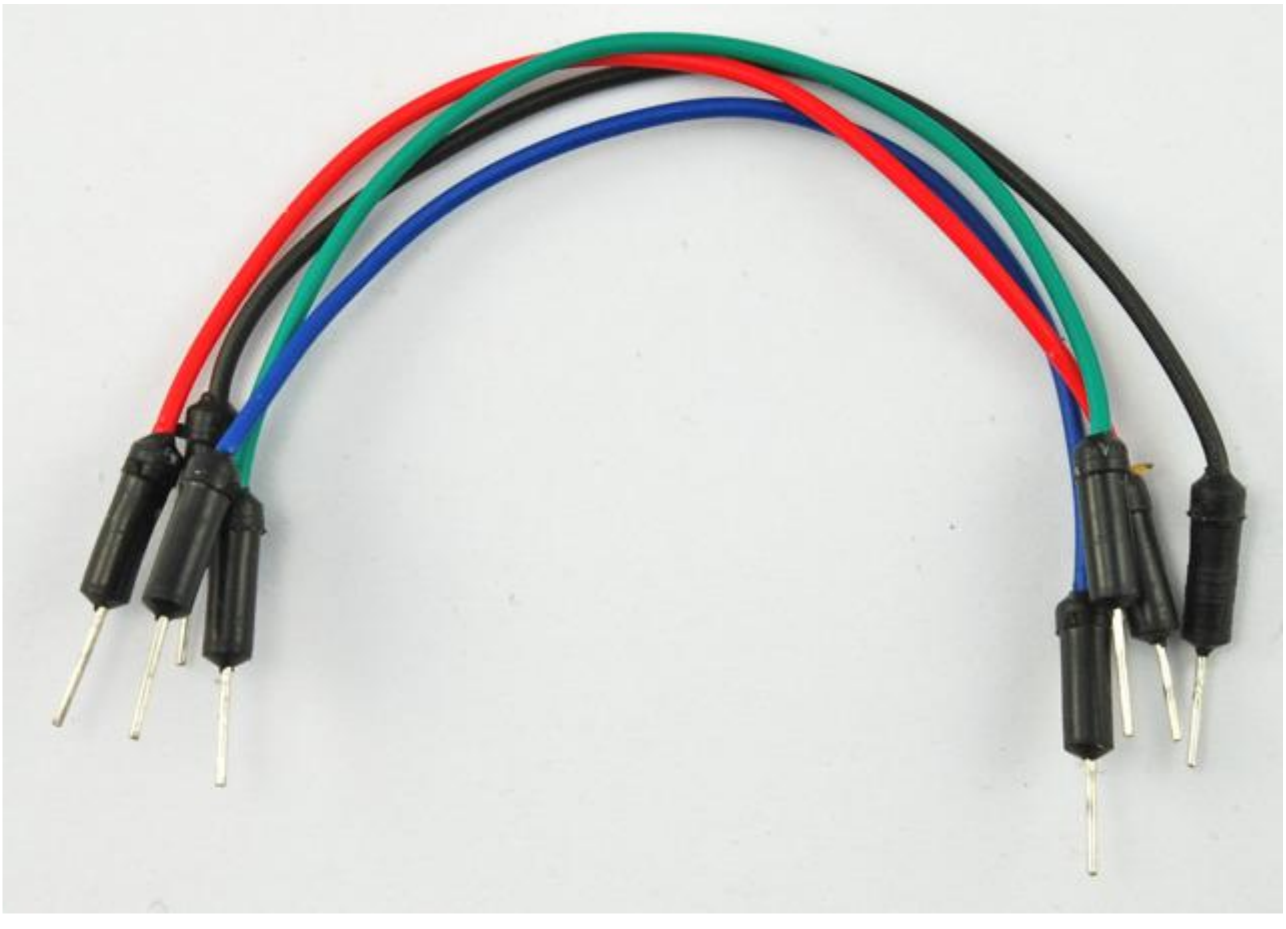


Arduino Uno R3

1



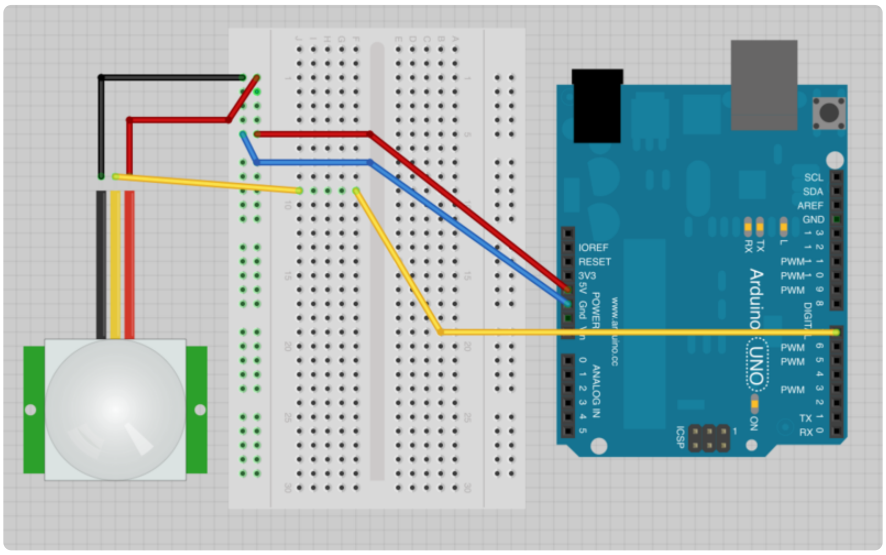
Half-sized Breadboard 1



Jumper wire pack

Breadboard Layout

The only thing that you are connecting to the Arduino is the PIR sensor, so you could if you prefer simply push the wires attached to the PIR sensor directly into the Arduino board. However, the wires from the sensor, are a bit loose in the Arduino sockets so it is probably better to use the breadboard layout below.



Arduino Code

The Arduino will send a message over USB Serial connection whenever movement is detected. However, this could have the potential to generate a lot of emails. For this reason the Arduino sends a different message if its too soon to send another email.

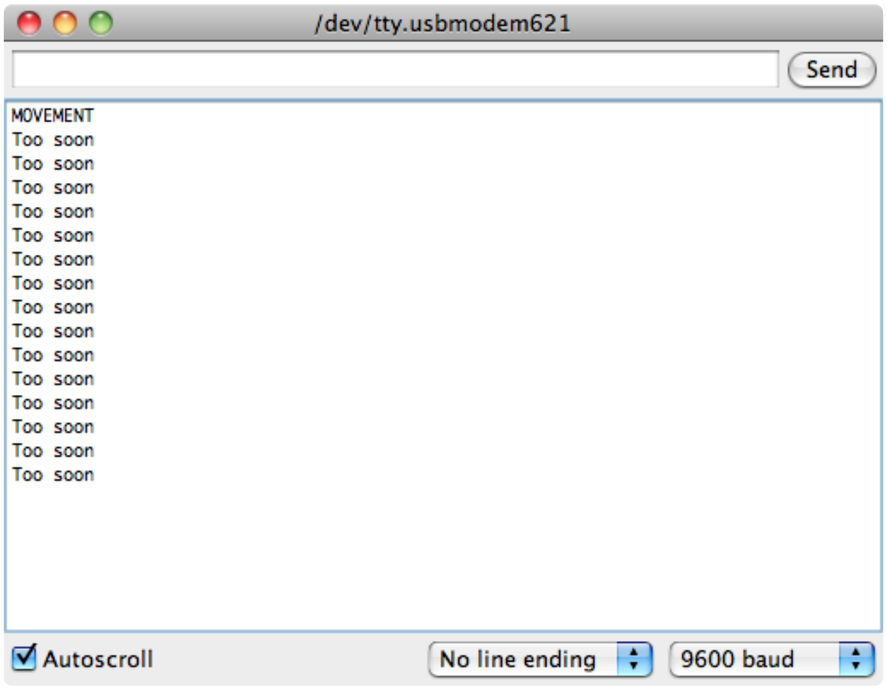
```
int pirPin = 7;

int minSecsBetweenEmails = 60; // 1 min

long lastSend = -minSecsBetweenEmails * 1000l;

void setup()
{
  pinMode(pirPin, INPUT);
  Serial.begin(9600);
}

void loop()
{
  long now = millis();
  if (digitalRead(pirPin) == HIGH)
  {
    if (now &gt; (lastSend + minSecsBetweenEmails * 1000l))
    {
      Serial.println("MOVEMENT");
      lastSend = now;
    }
    else
    {
      Serial.println("Too soon");
    }
  }
  delay(500);
}
```

The variable “minSecsBetweenEmails” can be changed to whatever you feel is a reasonable value. Here it is set to 60 seconds, so emails will not be sent at a rate of more than one a minute.

To keep track of when the last request to send an email was sent, a variable “lastSend” is used. This is initialized to a negative number, equal to the negative of the number of milliseconds specified in the “minSecsBetweenEmails” variable. This ensures that the PIR can be triggered immediately that the Arduino sketch starts.

Within the loop, the function “millis()” is used to get the number of milliseconds since the Arduino started and compare it with that last time the alarm was triggered and only if it is more than the specified number of seconds since last time does it send the message “MOVEMENT”. Otherwise even though movement has been detected, it just sends the message “Too soon”.

Before you link things up to your Python program, you can test the Arduino setup by just opening the Serial Monitor on the Arduino IDE.



```
MOVEMENT
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
Too soon
```

Installing Python and PySerial

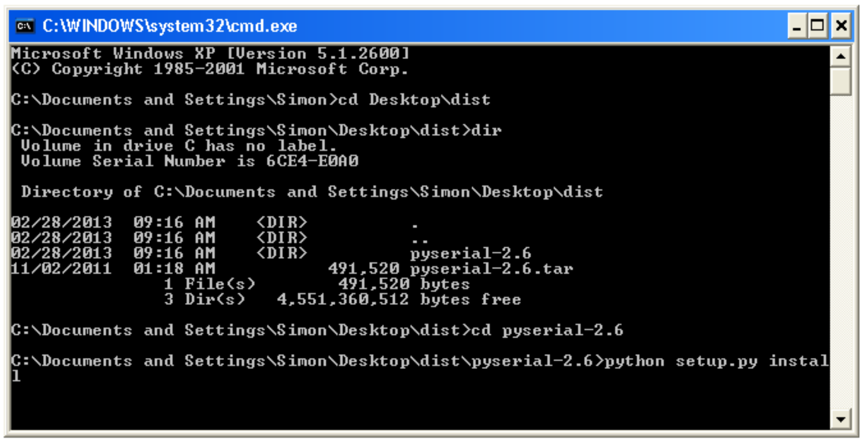
If you are using a Mac or Linux computer, the Python is already installed. If you are using Windows, then you will need to install it. In either case, you will also need to install the PySerial library to allow communication with the Arduino.

Install Python on Windows

To install Python on Windows, download the installer from http://www.python.org/getit/ ().

This project was built using Python 2.7.3

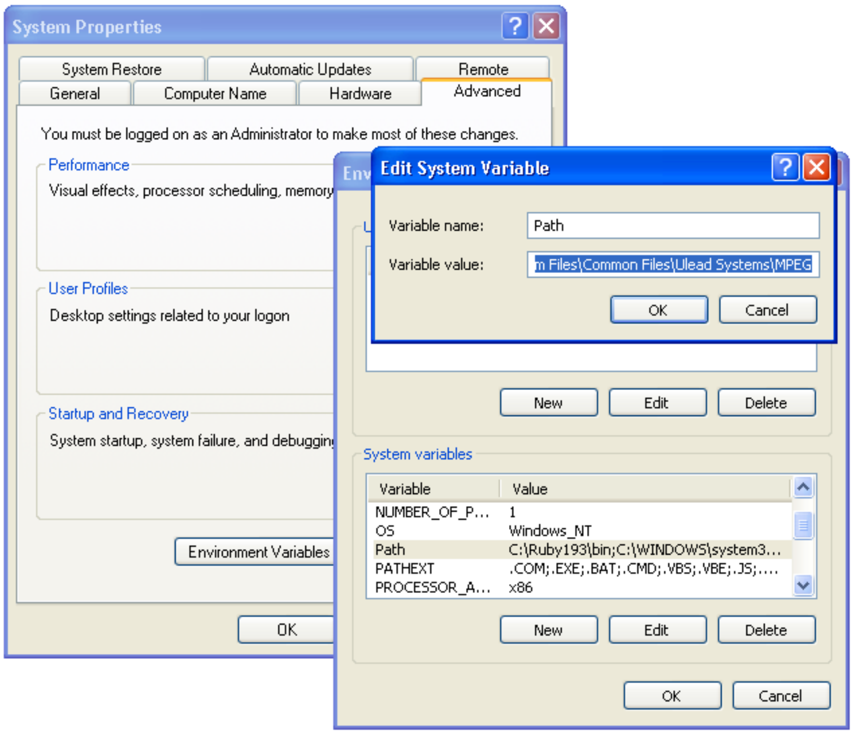
There are some reported problems with PySerial on Windows, using Python 3, so stick to Python 2.



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Simon>cd Desktop\dist

C:\Documents and Settings\Simon\Desktop\dist>dir
 Volume in drive C has no label.
 Volume Serial Number is 6CE4-E0A0

 Directory of C:\Documents and Settings\Simon\Desktop\dist

02/28/2013  09:16 AM    <DIR>          .
02/28/2013  09:16 AM    <DIR>          ..
02/28/2013  09:16 AM    <DIR>          pyserial-2.6
11/02/2011  01:18 AM           491,520 pyserial-2.6.tar
               1 File(s)        491,520 bytes
               3 Dir(s)   4,551,360,512 bytes free

C:\Documents and Settings\Simon\Desktop\dist>cd pyserial-2.6

C:\Documents and Settings\Simon\Desktop\dist\pyserial-2.6>python setup.py instal
l
```

Once Python is installed, you will find a new Program Group on your Start menu. However, we are going to make a change to Windows to allow you to use Python from the Command Prompt. You will need this to be able to install PySerial.

We are going to add something to the PATH environment variable.



You must be logged on as an Administrator to make most of these changes.

Visual effects, processor scheduling, memory

Desktop settings related to your logon

System startup, system failure, and debuggin

Variable name: Path

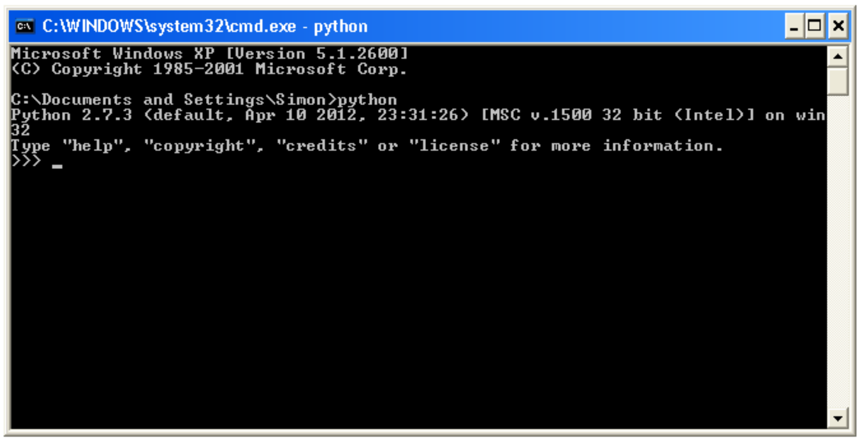
Variable value: m Files\Common Files\Ulead Systems\MPEG

Variable	Value
NUMBER_OF_P...	1
OS	Windows_NT
Path	C:\Ruby193\bin;C:\WINDOWS\system3...
PATHEXT	.COM;.EXE;.BAT;.CMD;.VBS;.VBE;.JS;....
PROCESSOR_A...	x86

To do this, you need to go to the Windows Control panel and find the System Properties control. Then click on the button labelled “Environment Variables” and in the window that pops-up select “Path” in the bottom section (System Variables). Click “Edit” and then at the end of the “Variable Value” without deleting any of the text already there, add the text: ;C:\Python27

Don't forget the ";" before the new bit!

To test that it worked okay, start a new Command Prompt (DOS Prompt) and enter the command “python”. You should see something like this:



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Simon>python
Python 2.7.3 (default, Apr 10 2012, 23:31:26) [MSC v.1500 32 bit (Intel)] on win
32
Type "help", "copyright", "credits" or "license" for more information.
>>> _
```

Install PySerial

Whatever your operating system, download the .tar.gz install package for PySerial 2.6 from https://pypi.python.org/pypi/pyserial ()

This will give you a file called: pyserial-2.6.tar.gz

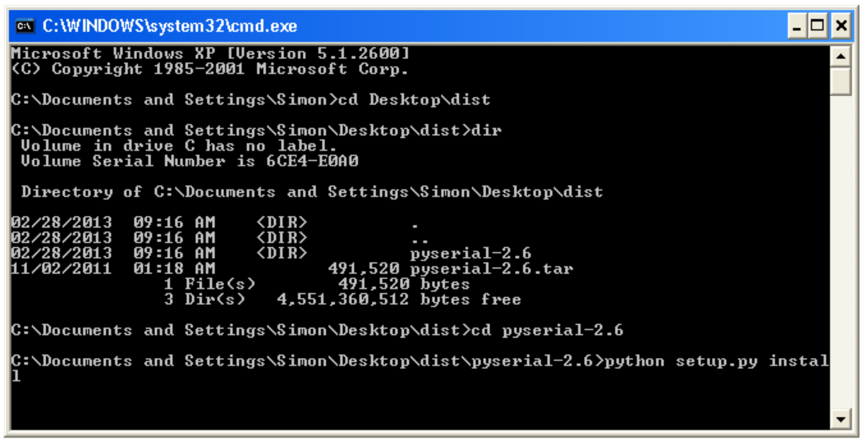
If you are using windows you need to uncompress this into a folder. Unfortunately, it is not a normal zip file, so you may need to download a tool such as 7-zip (http://www.7-zip.org/ ()).

If you are using a Mac or Linux computer, then open a Terminal session, 'cd' to wherever you downloaded pyserial-2.6.tar.gz and then issue the following command to unpack the installation folder.

```
$ tar -xzf pyserial-2.6.tar.gz
```

The rest of the procedure is the same whatever your operating system. Use you Comamnd Prompt / Terminal session and “cd” into the pyserial-2.6 folder, then run the command:

```
sudo python setup.py install
```



```
Microsoft Windows XP [Version 5.1.2600]
(C) Copyright 1985-2001 Microsoft Corp.

C:\Documents and Settings\Simon>cd Desktop\dist

C:\Documents and Settings\Simon\Desktop\dist>dir
 Volume in drive C has no label.
 Volume Serial Number is 6CE4-E0A0

 Directory of C:\Documents and Settings\Simon\Desktop\dist

02/28/2013  09:16 AM    <DIR>          .
02/28/2013  09:16 AM    <DIR>          ..
02/28/2013  09:16 AM    <DIR>          pyserial-2.6
11/02/2011  01:18 AM           491,520 pyserial-2.6.tar
               1 File(s)        491,520 bytes
               3 Dir(s)   4,551,360,512 bytes free

C:\Documents and Settings\Simon\Desktop\dist>cd pyserial-2.6

C:\Documents and Settings\Simon\Desktop\dist\pyserial-2.6>python setup.py instal
l
```

Python Code

Now, you need to create the Python program. To do this, copy the code below into a file called “movement.py”. On Mac / Linux you can use the “nano” editor, on Windows, it is probably easiest to make the file using the Python editor 'IDLE” which is available from the Python program group on your start menu.

```
import time
import serial
import smtplib

TO = 'putyour@email.here'
GMAIL_USER = 'putyour@email.here'
GMAIL_PASS = 'putyourpasswordhere'

SUBJECT = 'Intrusion!!'
TEXT = 'Your PIR sensor detected movement'

ser = serial.Serial('COM4', 9600)

def send_email():
    print("Sending Email")
    smtpserver = smtplib.SMTP("smtp.gmail.com",587)
    smtpserver.ehlo()
    smtpserver.starttls()
    smtpserver.ehlo
    smtpserver.login(GMAIL_USER, GMAIL_PASS)
    header = 'To:' + TO + '\n' + 'From: ' + GMAIL_USER
    header = header + '\n' + 'Subject:' + SUBJECT + '\n'
    print header
    msg = header + '\n' + TEXT + ' \n\n'
    smtpserver.sendmail(GMAIL_USER, TO, msg)
    smtpserver.close()

while True:
    message = ser.readline()
    print(message)
    if message[0] == 'M' :
        send_email()
    time.sleep(0.5)
```

Before you run the Python program, there are some configuration changes that you need to make. These are all up near the top of the file.

The program assumes that the emails are being set from a gmail account. So, if you don't have one, you might like to make yourself one, even if it is just for this project.

Change the email address next to “TO” to the email that you want to receive the notifications. This does not have to be your email address, but probably will be.

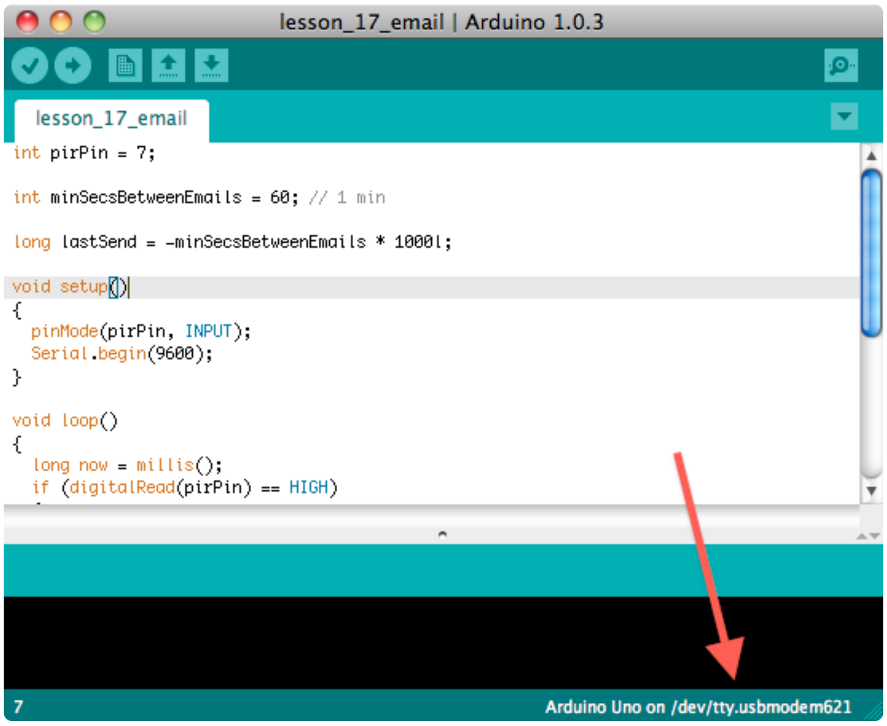
Change the email address next to “GMAIL_USER” to the email address of your gmail address and alter the password on the next line to the password you use to retrieve your emails.

If you want to, you can also change the subject line and text of the message to be sent, on the lines that follow.

You will also need to set the serial port of the Arduino by editing the line below:

```
ser = serial.Serial('COM4', 9600)
```

For Windows, this will be something like “COM4” for Mac and Linux, something like “/dev/tty.usbmodem621”. You can find this by openning the Arduino IDE and in the bottom right corner, it will show you the port that is connected to the Arduino.



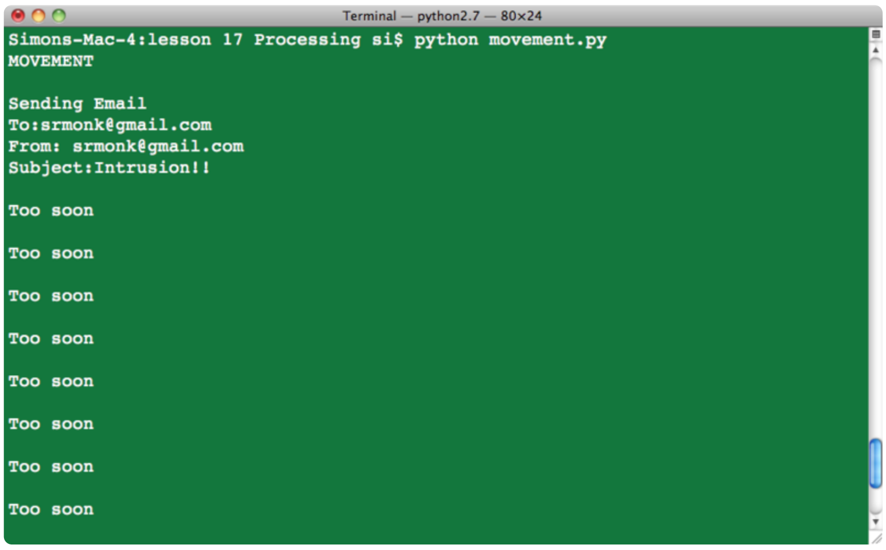
```
int pirPin = 7;

int minSecsBetweenEmails = 60; // 1 min

long lastSend = -minSecsBetweenEmails * 1000l;

void setup()
{
  pinMode(pirPin, INPUT);
  Serial.begin(9600);
}

void loop()
{
  long now = millis();
  if (digitalRead(pirPin) == HIGH)
```

```
7                                    Arduino Uno on /dev/tty.usbmodem621
```

When you have made these changes, you can run the program from Command Prompt / Terminal with the command:

```
python movement.py
```

When a movement is triggered you should get some trace like this and shortly after an email will arrive in your inbox.



```
Simons-Mac-4:lesson 17 Processing si$ python movement.py
MOVEMENT

Sending Email
To:srmonk@gmail.com
From: srmonk@gmail.com
Subject:Intrusion!!

Too soon

Too soon

Too soon

Too soon

Too soon

Too soon

Too soon

Too soon
```

Notice also the "Too soon" messages.

Other Things to Do

Now that you have a means of sending email from your Arduino, this opens up all sorts of possibilities You could add different types of sensor, and perhaps send yourself hourly temperature reports by email.

The PIR sensor could be used directly with the Arduino to play a warning tone or turn on LEDs.

About the Author.

Simon Monk is author of a number of books relating to Open Source Hardware. The following books written by Simon are available from Adafruit: Programming Arduino (http://adafru.it/1019), 30 Arduino Projects for the Evil Genius (http://adafru.it/868) and Programming the Raspberry Pi ().