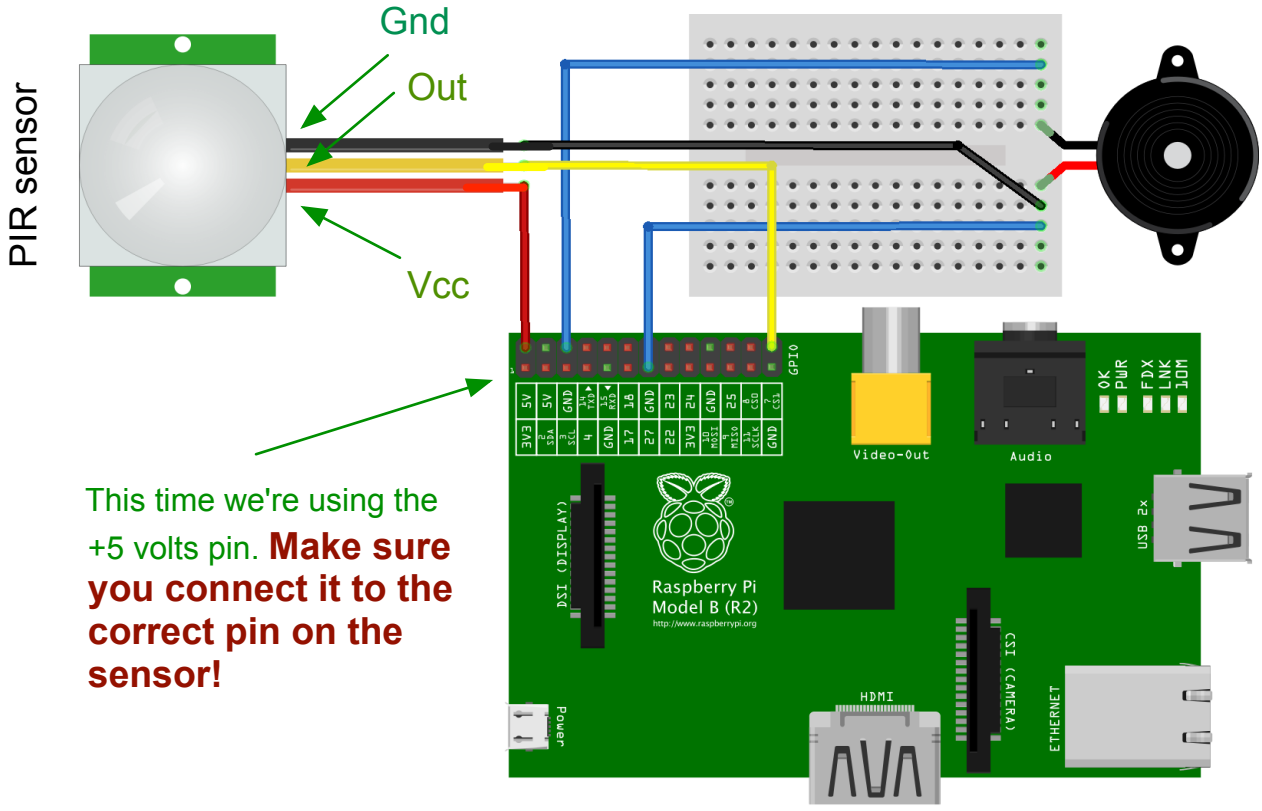


This time we're using the +5 volts pin. **Make sure you connect it to the correct pin on the sensor!**

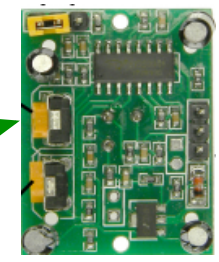
The PIR sensor uses Infra-red light to detect movement. Can you see it?

The sensor's output is 0 when no motion is detected, then changes to 1 when something moves.

Use the Python code over the page (note that it uses the buzz function we wrote before).

Too sensitive? You can adjust its range:

Sensitivity



Can you add a visual warning? Add an led to the circuit and then make it flash when the alarm is triggered?

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BOARD)

# Setup the pins we'll use
GPIO_PIR = 26
GPIO.setup(13,GPIO.OUT)
GPIO.setup(GPIO_PIR,GPIO.IN)

#define a function to make the buzzer buzz
def buzz(freq,dur):
    repeat = dur/(2*freq)
    for count in (range(int(repeat))):
        GPIO.output(13,GPIO.HIGH)
        time.sleep(freq)
        GPIO.output(13,GPIO.LOW)
        time.sleep(freq)

print "PIR Alarm active (CTRL-C to exit)"

Current_State  = 0
Previous_State = 0

try:

    print "Waiting for PIR to settle ..."

  # Loop until PIR output is 0
    while GPIO.input(GPIO_PIR)==1:
        Current_State  = 0

    print "  Ready"

  # Loop until users quits with CTRL-C
    while True :

    # Read PIR state
        Current_State = GPIO.input(GPIO_PIR)

        if Current_State==1 and Previous_State==0:

            # PIR is triggered
            print "  ALARM!"
            buzz(0.005, 2)
            # Record previous state
            Previous_State=1

        elif Current_State==0 and Previous_State==1:

        # PIR has returned to ready state
            print "  Ready"
            Previous_State=0

        # Wait for 10 milliseconds
        time.sleep(0.01)

except KeyboardInterrupt:
    print "  Quit"
    # Reset GPIO settings
    GPIO.cleanup()
```

The indentation is really important in this code. Try to using the tab key rather than lots of spaces.

Use the dotted lines to help you get each block of code aligned correctly.