Sarah Wood

Embedded Programming

Chapter 11

Assignment 1

IMPORTANT NOTE if using the Grove shield AND the Ethernet shield!!

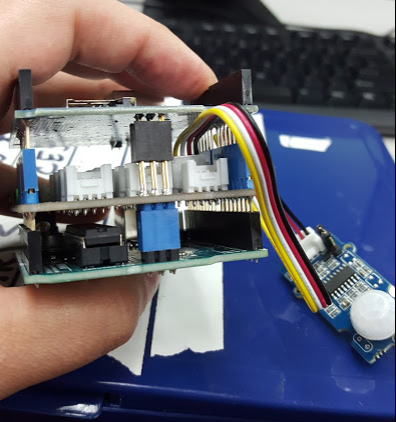
1. Install the Grove shield onto the Arduino UNO.
2. Plug the PIR module into the Grove shield and gently flex the cable away from the USB port to be parallel to the PCB.
3. Holding the Grove cable in place, install the Ethernet shield on top of the Arduino/Grove assembly. Release Grove cable, and place PIR sensor as desired (I used non-conductive tape).

Your layers should now be as follows:

Ethernet (top)

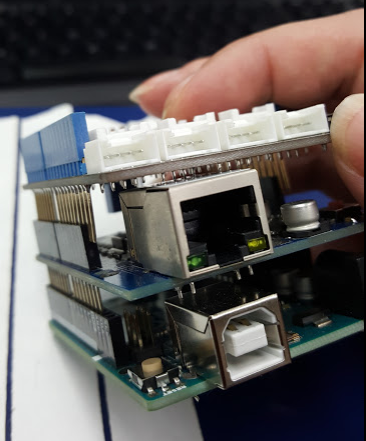
Grove (PIR module poking out) (middle)

Arduino(bottom, as pictured)



This is correct. Note the air-space between boards.

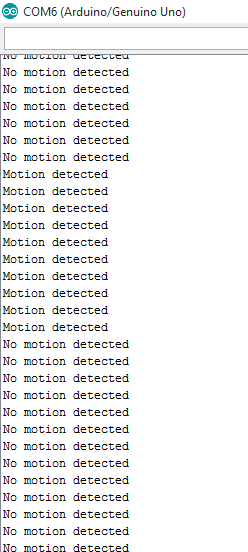
If your “sandwich” layers are Arduino, ETH, Grove; YOU WILL HAVE PROBLEMS! The PCB contacts will short out on the metal jack casings!



(Glad I caught this before I put power through anything!)

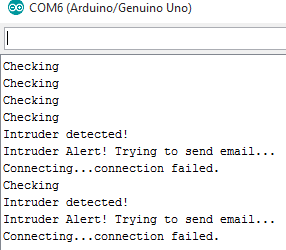
If you really need your sandwich layers to leave the Grove module on top, FOR THE LOVE OF PETE, install header extensions!

Screen Caps



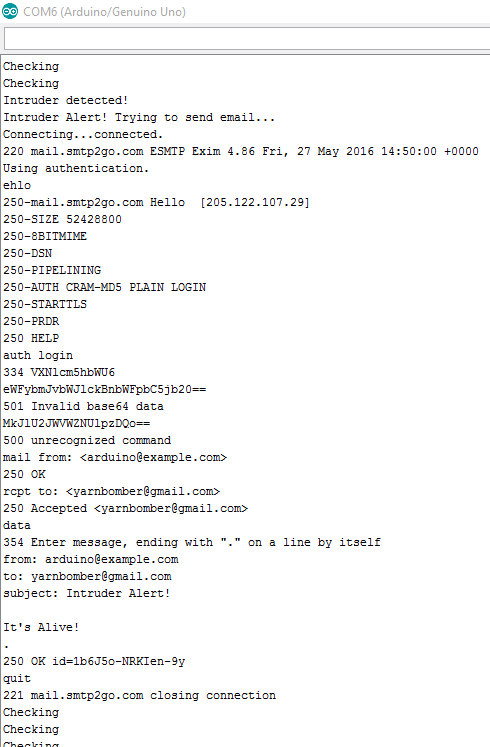
Basic hardware functionality is OK. Very wide “field of vision”.

Burglar Alarm code compiled and uploaded cleanly. I added some serial outputs, since I don’t have an Ethernet cable with me right now.



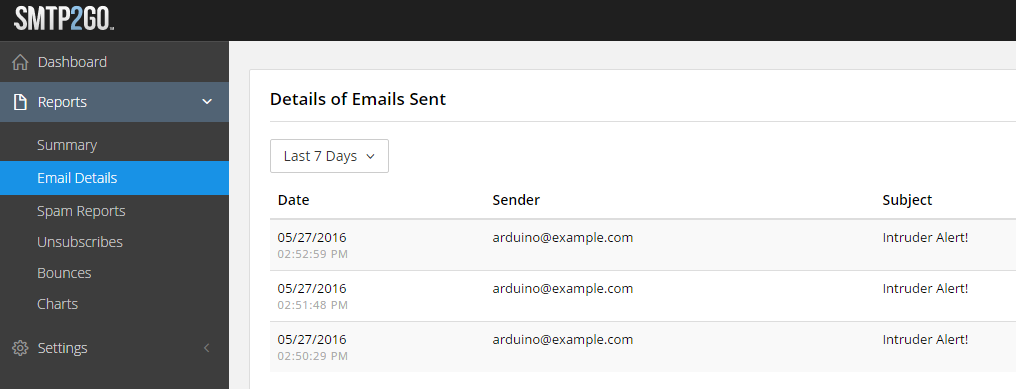
I expected the failed connection, but the program is doing its thing.

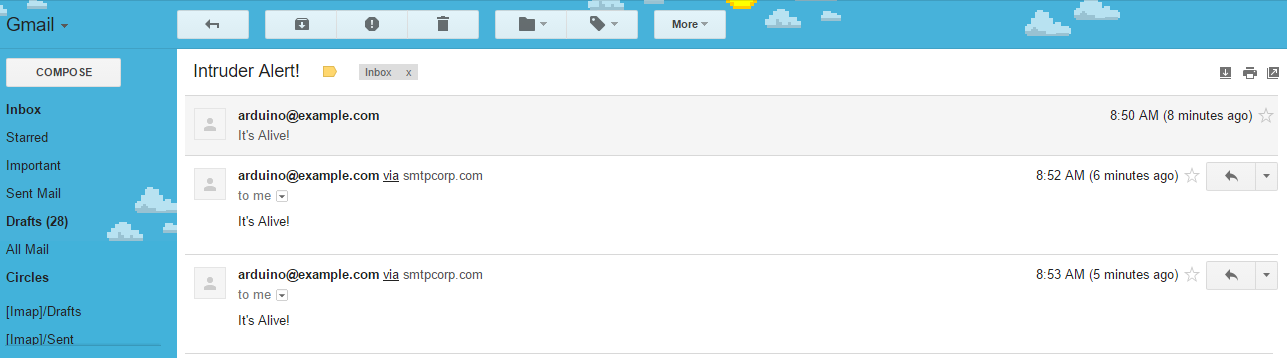
Ok, we’re plugged in. I plugged into a random Ethernet cable in the classroom, but I never did find a valid IP address for it. (The following may not be entirely Kosher, but it worked.) I retrieved the IP address for the Windows machine I use in class, and entered it into my sketch. I then unplugged it from the back of the desktop machine, and plugged it into the Arduino. I kept my USB connected to Arduino as well, so I could monitor serial output. Here’s what I got.



It’s Alive!

I let it cycle three times before restoring Ethernet to my desktop machine. Here are some shots of the emails I sent/received. (Sometimes my watch will “clump” together similar notifications. That’s why there are only two showing. Please ignore my power bill.)







Source Code

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

EmailProximityAlarm.cpp

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

#include <SPI.h>

#include <Ethernet.h>

#include "burglar\_alarm.h"

const unsigned int PIR\_INPUT\_PIN = 2;

const unsigned int SMTP\_PORT = 2525; // as per SMTP2go

const unsigned int BAUD\_RATE = 9600;

const String USERNAME = "eWFybmJvbWJlckBnbWFpbC5jb20=="; // Encoded in Base64.

const String PASSWORD = "MkJlU2JWVWZNUlpzDQo=="; // Encoded in Base64.

byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

IPAddress my\_ip(192, 168, 2, 120);

// Insert IP address of your SMTP server below! // inserted SMTP2go static IP

IPAddress smtp\_server(216, 22, 15, 250);

PassiveInfraredSensor pir\_sensor(PIR\_INPUT\_PIN);

SmtpService smtp\_service(smtp\_server, SMTP\_PORT, USERNAME, PASSWORD);

BurglarAlarm burglar\_alarm(pir\_sensor, smtp\_service);

void setup() {

Ethernet.begin(mac, my\_ip);

Serial.begin(BAUD\_RATE);

delay(20 \* 1000);

}

void loop() {

burglar\_alarm.check();

delay(3000);

}

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

burglar\_alarm.h

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/\*\*\*

\* Excerpted from "Arduino: A Quick-Start Guide, Second Edition",

\* published by The Pragmatic Bookshelf.

\* Copyrights apply to this code. It may not be used to create training material,

\* courses, books, articles, and the like. Contact us if you are in doubt.

\* We make no guarantees that this code is fit for any purpose.

\* Visit http://www.pragmaticprogrammer.com/titles/msard2 for more book information.

\*\*\*/

#ifndef \_\_BURGLAR\_ALARM\_H\_\_

#define \_\_BURGLAR\_ALARM\_H\_\_

#include "pir\_sensor.h"

#include "smtp\_service.h"

class BurglarAlarm {

PassiveInfraredSensor \_pir\_sensor;

SmtpService \_smtp\_service;

void send\_alarm() {

Email email(

"arduino@example.com",

"yarnbomber@gmail.com",

"Intruder Alert!",

"It's Alive!"

);

Serial.println("Intruder Alert! Trying to send email...");

\_smtp\_service.send\_email(email);

}

public:

BurglarAlarm(

const PassiveInfraredSensor& pir\_sensor,

const SmtpService& smtp\_service) :

\_pir\_sensor(pir\_sensor),

\_smtp\_service(smtp\_service)

{

}

void check() {

Serial.println("Checking");

if (\_pir\_sensor.motion\_detected()) {

Serial.println("Intruder detected!");

send\_alarm();

}

}

};

#endif

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

email.h

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/\*\*\*

\* Excerpted from "Arduino: A Quick-Start Guide, Second Edition",

\* published by The Pragmatic Bookshelf.

\* Copyrights apply to this code. It may not be used to create training material,

\* courses, books, articles, and the like. Contact us if you are in doubt.

\* We make no guarantees that this code is fit for any purpose.

\* Visit http://www.pragmaticprogrammer.com/titles/msard2 for more book information.

\*\*\*/

#ifndef \_\_EMAIL\_\_H\_

#define \_\_EMAIL\_\_H\_

class Email {

String \_from, \_to, \_subject, \_body;

public:

Email(

const String& from,

const String& to,

const String& subject,

const String& body

) : \_from(from), \_to(to), \_subject(subject), \_body(body) {}

const String& getFrom() const { return \_from; }

const String& getTo() const { return \_to; }

const String& getSubject() const { return \_subject; }

const String& getBody() const { return \_body; }

};

#endif

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

pir\_sensor.h

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

#ifndef \_pir\_sensor\_h

#define \_pir\_sensor\_h

class PassiveInfraredSensor {

int \_input\_pin;

public:

PassiveInfraredSensor(const int input\_pin) {

\_input\_pin = input\_pin;

pinMode(\_input\_pin, INPUT);

}

const bool motion\_detected() const {

return digitalRead(\_input\_pin) == HIGH;

}

};

#endif

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

smtp\_service.h

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

/\*\*\*

\* Excerpted from "Arduino: A Quick-Start Guide, Second Edition",

\* published by The Pragmatic Bookshelf.

\* Copyrights apply to this code. It may not be used to create training material,

\* courses, books, articles, and the like. Contact us if you are in doubt.

\* We make no guarantees that this code is fit for any purpose.

\* Visit http://www.pragmaticprogrammer.com/titles/msard2 for more book information.

\*\*\*/

#ifndef \_\_SMTP\_SERVICE\_\_H\_

#define \_\_SMTP\_SERVICE\_\_H\_

#include "email.h"

class SmtpService {

boolean \_use\_auth;

IPAddress \_smtp\_server;

unsigned int \_port;

String \_username;

String \_password;

void read\_response(EthernetClient& client) {

delay(4000);

while (client.available()) {

const char c = client.read();

Serial.print(c);

}

}

void send\_line(EthernetClient& client, String line) {

const unsigned int MAX\_LINE = 256;

char buffer[MAX\_LINE];

line.toCharArray(buffer, MAX\_LINE);

Serial.println(buffer);

client.println(buffer);

read\_response(client);

}

public:

SmtpService(

const IPAddress& smtp\_server,

const unsigned int port) : \_use\_auth(false),

\_smtp\_server(smtp\_server),

\_port(port) { }

SmtpService(

const IPAddress& smtp\_server,

const unsigned int port,

const String& username,

const String& password) : \_use\_auth(true),

\_smtp\_server(smtp\_server),

\_port(port),

\_username(username),

\_password(password) { }

void send\_email(const Email& email) {

EthernetClient client;

Serial.print("Connecting...");

if (client.connect(\_smtp\_server, \_port) <= 0) {

Serial.println("connection failed.");

} else {

Serial.println("connected.");

read\_response(client);

if (!\_use\_auth) {

Serial.println("Using no authentication.");

send\_line(client, "helo");

}

else {

Serial.println("Using authentication.");

send\_line(client, "ehlo");

send\_line(client, "auth login");

send\_line(client, \_username);

send\_line(client, \_password);

}

send\_line(

client,

"mail from: <" + email.getFrom() + ">"

);

send\_line(

client,

"rcpt to: <" + email.getTo() + ">"

);

send\_line(client, "data");

send\_line(client, "from: " + email.getFrom());

send\_line(client, "to: " + email.getTo());

send\_line(client, "subject: " + email.getSubject());

send\_line(client, "");

send\_line(client, email.getBody());

send\_line(client, ".");

send\_line(client, "quit");

client.println("Disconnecting.");

client.stop();

}

}

};

#endif