ESP8266 Motion Detector to IFTTT

Version 2016-06-29 – R.Grokett

Overview

We just installed a Cat Door in our garage and I wanted to see how many times per day (actually night) our cat went in and out the door. We could tell the cat was using the door as we would find it outside sometimes and inside sometimes.

So, breaking out my parts box, I found a PIR motion sensor, an ESP8266 and a 5V battery.

I used the Adafruit HUZZAH ESP8266 as it has 5V regulator for powering the 3.3v ESP as well as good tutorials for initially getting it set up. I also used the Arduino IDE with the ESP8266 library, since I was already quite familiar with using it with the Huzzah ESP8266.

I decided to interface this to IFTTT (www.ifttt.com) to trigger any number of types of events. Initially, just an email each time motion was detected.

Note that IFTTT requires HTTPS SSL encryption. Thus this project includes code for that.

Parts List

- Adafruit HUZZAH ESP8266 https://www.adafruit.com/product/2471
- PIR Motion Detector such as https://www.adafruit.com/products/189
- FTDI or USB console cable https://www.adafruit.com/products/954 or equiv
- 5V power supply or 5V USB battery (for portable use)
- Breadboard, wire, box to put everything in
- Arduino IDE with ESP8266 extension package installed *(see Initial Setup below)*
- Download the ESP8266_PIR software from GitHub

https://github.com/rgrokett/ESP8266_PIR

Important Initial Setup of ESP8266

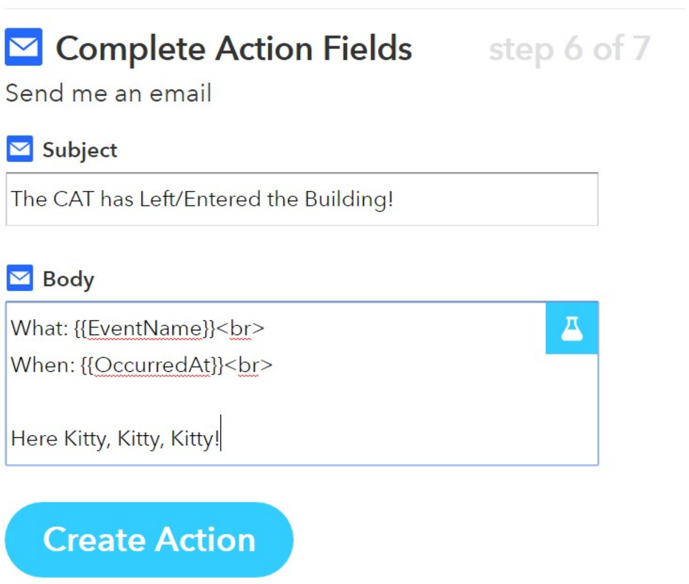
Before beginning the project, you should become familiar with the Adafruit HUZZAH board and programming it using the Arduino IDE. The best way is to use the excellent Adafruit tutorial:

https://learn.adafruit.com/adafruit-huzzah-esp8266-breakout/using-arduino-ide

You must be able to program your ESP8266 and connect wirelessly to it via browser as described in their tutorial.   Once completed, THEN continue below.

IFTTT Setup

1. Go to www.ifttt.com
2. Log in. If you don't have an account, you can sign up. It's free.
3. Once logged in, click on **Channels**
4. In the Search Channels box, enter "Maker"
5. Scroll down and click on the Maker icon
6. On the Maker Channel screen, you need to copy the "Your Key is:" key field. This is needed for the .ino program later on.
7. Click on **Create a New Recipe**
8. Click the "this" of **ifthisthenthat**…
9. Again type "Maker" into the Search Channels box
10. For Choose a Trigger, there is only one big gray box with "Receive a web Request". Click it
11. For Complete Trigger Fields, enter "**pirtrigger**" and click Create. This is the event name used in the ESP8266 .ino software.
12. Click the "that" of ifthisthenthat…
13. Type "email" into the Search Channels box. You could change this to do other things, like send SMS messages, etc. But let's stick to email. You can always edit later.
14. For Choose an Action, click the big gray box with "Send me an email".
15. You can edit the text or just leave it as is. Some of the fields aren't used, but they just show up blank in your email.


Complete Action Fields step 6 of 7

Send me an email

Subject

The CAT has Left/Entered the Building!

Body

What: {{EventName}}
When: {{OccurredAt}}

Here Kitty, Kitty, Kitty!

16. Click Create Action. You will see a screen that just describes what this recipe will do. You can edit the Recipe Title, or just leave as is.
17. NOTE that it uses the email address you entered when you signed up for IFTTT.
18. You now have an IFTTT Recipe.

Any IFTTT Recipe that uses Maker channel can be used, as long as it is called “pirtrigger”. (You can change the trigger name in the ESP8266_PIR.ino program below, if desired.)

Software

You should program and test the ESP8266 before adding its hardware wiring.

1. **STOP**! Be sure you have already completed the Adafruit tutorial software setup of the Arduino IDE and tested the ESP8266 with your WiFi network as described in the Initial Setup section above!

2. Ok, download the ESP8266_PIR software from GitHub (https://github.com/rgrokett/ESP8266_PIR/)

3. Copy the ESP8266_PIR subdirectory into your Arduino IDE development directory. This folder has the 3 software files needed.
```
ESP8266_PIR.ino
HTTPSRedirect.h
HTTPSRedirect.cpp
```

4. Double-click the ESP8266_PIR.ino program to load it into your Arduino IDE.

5. Using Arduino IDE, edit the ESP8266_PIR.ino and insert your WiFi SSID and PASSWORD into the appropriate places.

6. Update the API_KEY with your IFTTT API KEY copied previously. You can look on Channels -> Search Channels -> Maker in IFTTT for it, if needed.

7. You can also change a few variables, described here:

```
const char* ssid     = "{YOUR_WIFI_SSID}"; // Your WiFi SSID
const char* password = "{YOUR_WIFI_PWD}";  // Your WiFi Password

const char* api_key  = "aBc1fakekey2ab3cBA";   // Your API KEY from
https://ifttt.com/maker
const char* event    = "pirtrigger";       // Your IFTTT Event Name

bool verifyCert = false; // Select true if you want SSL certificate
validation

int PIRpin = 14;      // GPIO 14 (PIR Sensor)
int MOTION_DELAY = 15; // Delay in seconds between events to keep from
flooding IFTTT & emails
```

IFTTT requires HTTPS SSL and HTTPS 302 Redirection. The ESP8266 Library (installed in the Adafruit Tutorial) contains the HTTPS SSL functions and an extension of that library was developed by https://github.com/electronicsguy/ESP8266/tree/master/HTTPSRedirect to handle the HTTPS 302 Redirection. Since this code wasn't in the ESP8266 Library, I have included a copy or you can get the latest version from the URL above and add the .cpp and .h files to the ESP8266_PIR folder.

The "WiFiClientSecure" provides SSL encryption, so the messages are always sent encrypted, but by default, the verification of the IFTTT's SSL Certificate is turned off. You can turn it on by changing `verifyCert = true;`
This requires the SHA1 Fingerprint of the IFTTT server to be used to verify the certificate.

```
const char* SHA1Fingerprint="A9 81 E1 35 B3 7F 81 B9 87 9D 11 DD 48 55
43 2C 8F C3 EC 87";
```

This fingerprint is initially retrieved from the IFTTT server using the Linux command:

```
$ openssl s_client -servername maker.ifttt.com -connect
maker.ifttt.com:443 | openssl x509 -fingerprint -noout
```

Replace colons with spaces in result and update the ESP8266_PIR.ino as needed.
You should NOT have to change this unless IFTTT changes their SSL Certificate.
Again, you can bypass this check by setting `verifyCert = false;`

The IFTTT server initially returns a 302 Redirect message back, so the “HTTPSRedirect.cpp” software invisibly handles resending the request to the new host.

8. Compile and Upload the program using the FTDI or USB console cable just like shown in the Adafruit tutorial. Remember, you have to press the tiny GPIO0 and RESET buttons on the HUZZAH ESP8266 (aka *bootload* mode) to allow the upload to occur.

9. When the program finishes loading, open a Serial Monitor, set to 115,200 baud, and press the ESP8266 RESET button to restart the program running.

10. It should display the IP address in the Serial Monitor once connected to your Wifi. Also, the onboard Red LED should blink 4 times signifying it’s successfully connected.

11. Time for wiring the PIR and power…

Hardware - PIR

1. Temporarily unplug the FTDI/USB cable from the PC to power off the ESP8266.

2. Wire the PIR sensor as follows. Note that the PIR is powered by 5V, but its I/O line is 3.3v which makes it directly compatible with the ESP8266’s 3.3v GPIO pins.

ESP8266 HUZZAH	PIR SENSOR
GND	GND
V+	VCC
GPIO 14	OUT

3. With the PIR now wired in, reconnect the FTDI/USB cable to the PC.
4. Again, start the Serial Monitor from the Arduino IDE.
5. Reset the ESP8266 and you should see the LED blink 4 times and the IP address displayed again.
6. If you move in front of the PIR, the Serial Monitor should register the event and send off to IFTTT. If IFTT trigger is successful, you should see a 200 OK HTTP response message and text and receive an email.

```
< HTTP/1.1 200 OK
< Server: Cowboy
< Connection: keep-alive
< X-Powered-By: Sad Unicorns
< X-Top-Secrettt: VG9vIGVhc3k/IElmIHlvdSBFK3.../NlY3JldEBlIHdnQgTWFrZXJzLg==
< Content-Type: text/html; charset=utf-8
< Content-Length: 50
< Etag: W/"32-44d0098f"
< Date: Wed, 29 Jun 2016 21:25:32 GMT
< Via: 1.1 vegur
<
```

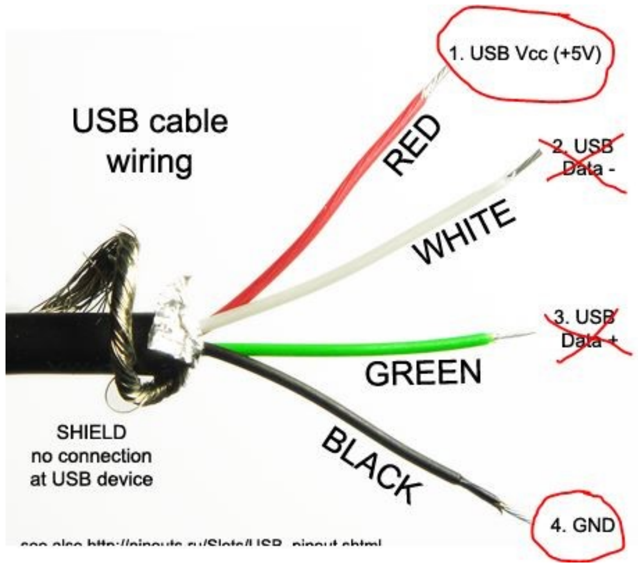
```
* Connection #0 to host maker.ifttt.com left intact
* Closing connection #0
* SSLv3, TLS alert, Client hello (1):
Congratulations! You've fired the pirtrigger event
```

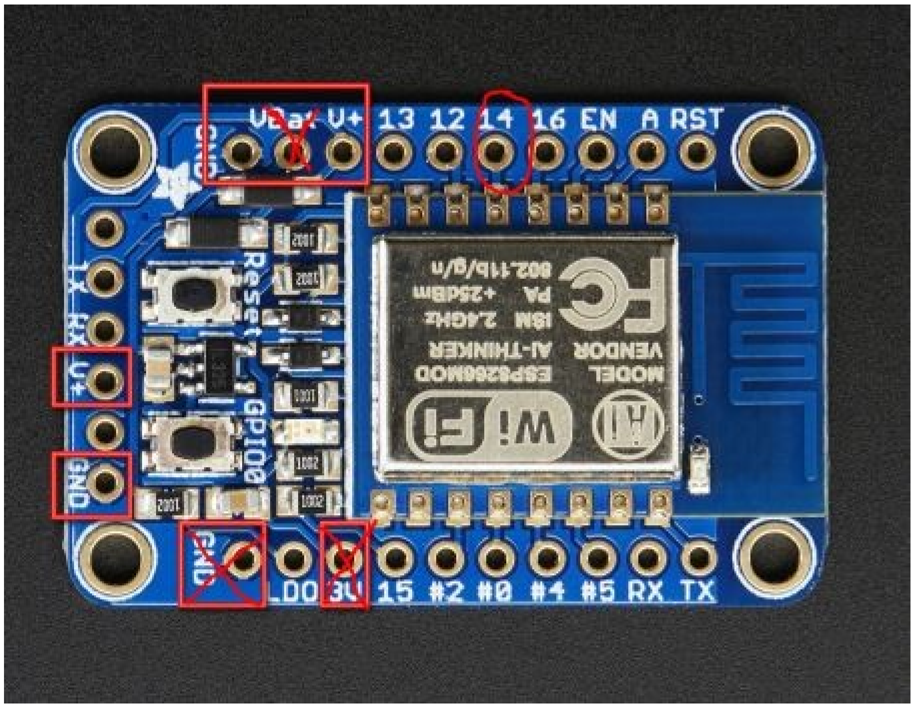
Hardware - Power

1. Since the Adafruit HUZZAH ESP8266 supports operating on 5V, a 5V USB battery or USB power supply can be used. Since we tied the PIR sensor directly to one of the V+ pins, the battery or power supply is going to have to connect to the V+ and GND that the FTDI/USB cable connected to. So if you need to reprogram the ESP8266, you will need to disconnect these.

ESP8266 HUZZAH SIDE CONNECTOR	5V POWER
TX	n/a
RX	n/a
V+	5V +
GND	GND

You may need to build a USB-to-ESP cable by taking an unused USB cable and cutting it:





2. Once ready, plug in the USB connector to your power supply or battery.
3. After a few seconds, you should see the red LED blink four times to indicate it connected to your WiFi.
4. Do some motion and after a minute or so, an email should arrive!

   Have Fun!