# Table of Contents

**Introduction**
- Parts
- Hardware

**3D Printing the Mailbox**
- Gmailbox Parts
- What if I don't have a 3D Printer?
- Slicer Settings for Cura
- CAD Assembly
- Design Source Files
- Tap Mounting Holes
- Making Threads
- Tap Holes in Case
- Tap Plate
- Install Plate
- Mounting Screws
- Mailbox Parts Assembly

**Putting it Together**
- Wiring
- Assembly
- Install HUZZAH to Mount
- Installed HUZZAH
- Install Servo
- Secure Servo
- Flag Servo Horn
- Install Flag
- Secure Flag
- Building The Mailbox
- Back Cover
- Bottom Cover
- Front Cover

**Adafruit IO Setup**

**IFTTT Setup**
- What happened with IFTTT?

**Zapier Setup**

**Arduino Setup**
- Installing Libraries

**Arduino Network Config**
- WiFi Config
- FONA Config
- Ethernet Config

**Code**
- Testing out your Gmailbox
- Help - my Gmailbox isn't working!
• Code
Introduction

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.

Ahhh, the charms of suburbia, mowing the lawn, white picket fences, running barefoot to the mailbox to see if the postal worker already dropped by. Sometimes we wax nostalgic for our childhood. Maybe we’ve become desensitized to the constant work email notification pop-ups on our smartphones.

Wouldn’t it be nice to get a friendlier heads up? Do you ever miss the old-school “You’ve got mail!” sound from your old AOL email client?

We’re going to make a small mailbox to sit on your desk and notify you about incoming emails. When a new email is received, it raises a flag and gently lowers it back down.

It's a fun, easy, introductory project to the Internet-of-Things by making things move using Adafruit IO with IFTTT.

Parts
You’ll need the following parts to complete this guide.
Supplies

Wire and handy supplies for soldering.

1 x Silicone Cover Stranded-Core Wire
This wire is soft and flexible enough to go into the enclosure for your Gmailbox

1 x Solder Wire
Solder Spool - 1/4 lb SAC305 RoHS lead-free / 0.031" rosin-core - 0.25 lb / 100 g

Tools

To build your Gmailbox, you'll need access to the following tools. Don't have something listed? Pick it up from the Adafruit shop.

1 x Wire Strippers
Hakko Professional Quality 20-30 AWG Wire Strippers
<table>
<thead>
<tr>
<th>Item</th>
<th>Quantity</th>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>Panavise</td>
<td>1</td>
<td>Panavise Jr.</td>
<td><a href="https://www.adafruit.com/product/151">https://www.adafruit.com/product/151</a></td>
</tr>
<tr>
<td>3D Printer</td>
<td>1</td>
<td>Ultimaker 2+ 3D Printer</td>
<td><a href="https://www.adafruit.com/product/2673">https://www.adafruit.com/product/2673</a></td>
</tr>
</tbody>
</table>

**Hardware**

You'll need some screws and nuts to secure the parts together. We have them linked here but you can source them from your local hardware supplier.

<table>
<thead>
<tr>
<th>Item</th>
<th>Quantity</th>
<th>Description</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>M2 x 12mm Machine Screws</td>
<td>2</td>
<td>Micro Servo – M2x 12mm Flat Head Phillips</td>
<td><a href="https://www.albanycountyfasteners.com/2MM-x-4-Phillips-Flat-Head-Machine-Screw-p/1011-1000.htm">https://www.albanycountyfasteners.com/2MM-x-4-Phillips-Flat-Head-Machine-Screw-p/1011-1000.htm</a></td>
</tr>
<tr>
<td>M2.5 x 6mm Machine Screws</td>
<td>4</td>
<td>Feather HUZZAH – M2.5 x 6mm Flat Head Phillips</td>
<td><a href="https://www.albanycountyfasteners.com/2-5-MM-x-45-Phillips-Flat-Head-Machine-Screw-p/1011-1002.htm">https://www.albanycountyfasteners.com/2-5-MM-x-45-Phillips-Flat-Head-Machine-Screw-p/1011-1002.htm</a></td>
</tr>
<tr>
<td>M3 x 8mm Machine Screws</td>
<td>2</td>
<td>Pole Bracket – M3 x 8mm Metric Pan Head Phillips</td>
<td><a href="https://www.albanycountyfasteners.com/Phillips-Pan-Head-Machine-Screw-M3-x-5-p/1066-1008.htm">https://www.albanycountyfasteners.com/Phillips-Pan-Head-Machine-Screw-M3-x-5-p/1066-1008.htm</a></td>
</tr>
</tbody>
</table>
3D Printing the Mailbox

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.

Gmailbox Parts

There are six 3D-printed parts which make up the Gmailbox:

- Mailbox Bottom
- Mailbox Top
- Servo Holder
• Mailbox Cover (Front)
• Mailbox Cover (Back, Micro-USB cutout)

What if I don't have a 3D Printer?

Not to worry! You can use a 3D printing service such as 3DHubs () or MakeXYZ () to have a local 3D printer operator 3D print and ship you parts to you. This is a great way to get your parts 3D printed by local makers. You could also try checking out your local Library or search for a Maker Space.

Slicer Settings for Cura

[Image]

Layer height: 0.2mm  
Line Width: 0.38mm  
Wall Thickness: 2 line count  
Infill: 20% triangle pattern  
Print Speed: 60mm/s  
Extruder/Bed Temps: 220c / 60c  
Supports: Nope!

CAD Assembly

This animation demonstrates how the parts fit together. The assembly was designed for 3D printing and optimized to print without any supports. The front cover hinges onto the case while the back cover snap fits shut. The bottom cover is clamped to the case. The STL files are oriented to print "As-is".

Design Source Files

The enclosure assembly was designed in Fusion 360. This can be downloaded in different formats like STEP, SAT and more. Electronic components like the board,
displays, connectors and more can be downloaded from our Fusion 360 CAD parts github repo.

If you need another format for your printer, use the following link:

- Download Files on GitHub
- Fusion 360 Source
- Download STLs on Thingiverse

**Tap Mounting Holes**
The mounting holes in the bottom cover and case will need to be tapped using an M2 and M2.5 tap tools. This will create threads inside the mounting holes to allow precise fastening.

**Making Threads**
A tapping tool has coarse and sharp edges that are designed to scope out excess material. A tap handle is used instead of a drill to minimize friction caused by rotational speed (PLA plastic melts!). Slowly twisting the handle produces better results. Try to be as straight as possible when tapping holes.
Tap Holes in Case
The two mounting holes on the side of the case feature geometry optimized for 3D printing. The part is printed vertically and produces holes that have slightly tighter tolerances. Use an M2 tapping tool to create threads in the mounting holes.

Tap Plate
The two mounting holes in the flat plate are also M2 sized and slightly tighter. These were also tapped. The M2 size tapping tool has a very small shank that did not fit my tap handle – So I designed and 3D printed one!

Install Plate
Place the plate over the side of the case with the ovular opening. Line up the mounting holes. Insert and fasten two M2 x 12mm flat head machine screws.
Mounting Screws
Continue to fasten the screws until the shank is about halfway through the case. Hold the parts together while fastening.

Mailbox Parts Assembly
Once all of the parts are printed, a dry fit test will ensure everything fits properly or may need a bit of sanding. Start with the case and back cover parts first. Press the back cover onto the end of the case with the flat side. Fit the lip over the case on one side and press fit the other side to snap the edges together. Insert the bottom cover in between the rail of the case and slide into the back cover. Fit the nubs from the front cover over the dimples in the lower corners of the case.

Putting it Together
Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.
Wiring

Make the following connections between the servo and the Feather HUZZAH:

- Servo Yellow to Feather Huzzah Digital Pin 14
- Servo Brown to Feather Huzzah GND Pin
- Servo Red to Feather Huzzah 3V Pin

Not comfortable with soldering? You can order a Feather HUZZAH ESP8266 with stacking header ()s pre-soldered, and use male/male jumper cables to connect the servo to your Feather HUZZAH.
Premium Male/Male Jumper Wires - 40 x 3" (75mm)
Handy for making wire harnesses or jumpering between headers on PCB's. These premium jumper wires are 3" (75mm) long and come in a 'strip' of 40 (4 pieces of each of...
https://www.adafruit.com/product/759

Assembly

Install HUZZAH to Mount
Use two M2.5 x 4mm flat head machine screws to secure the Adafruit Feather HUZZAH to the bottom cover part. Place the PCB over the standoffs and line up the mounting holes. Fastening screws into standoffs work best when tapped and threaded. USB port should be facing the edge of the mount.

Installed HUZZAH
Flat head machine screws had a tapered screw head. This is nice for low clearance mount holes. The mounting holes on near the antenna have a slightly smaller diameter and should be tapped when intending to secure all four mount holes. Be cautious not to over tighten or that could damage the antenna.
Install Servo
Grab the servo and orient to match the mounting holes on the 3d printed case. Insert the servo, shaft first, through the opening in between the mounting holes. Press firmly to fit through the wall. If it's too tight to push through, loosen the opening by filing edge with filing tool or sand paper.

Secure Servo
Insert and fasten two M2 x 12mm machine screws on the outside of the case through the mounting tabs in the micro servo. Fasten until fully tighten. The flag mount plate has chamfered holes to allow flat head screws to be flush with the surface.

Flag Servo Horn
Use the single arm servo horn that came with your submicro servo. Superglue will bond the two surfaces together. Place the servo horn over the flag and line up the mounting holes. Glue and set dry. Use one of the small screws to secure the flag to the servo.
Install Flag
Line up the servo horn with the splines on the sub-micro servo. The position of the flag will need to be tweaked and adjusted to properly line up with the expected angles of degree.

Secure Flag
Once the position is figured out, insert the small screw through the flag. Fasten the screw through the flag until the tip pokes through the servo horn. Then, press fit the servo horn onto the splines. Continue to fasten the screw and hold onto the flag until fully tightened.

Building The Mailbox
Now that we have our parts secured and mounted, we can put everything together!
Back Cover
Start with the back cover. Line up the curve with the case and fit one side of the lip onto the indentation on the case. While holding the case, firmly press the other side of the lip on the back cover. The cover should snap fit and lock into place.

Bottom Cover
Line up the bottom cover so the USB port on the HUZZAH is facing the back cover. Insert one side of the railing on the case to the lip on the bottom case. Firmly press down on the other side to snap it onto the rail of the case. Slide the bottom cover up and firmly press to wedge it into the back cover.

Front Cover
Fit one side of the tab over the dimple on the case. Pinch the ends of the case together to allow clearance for fitting the other tab. Clip the tab over the case and onto the dimple to secure the hinge.

Adafruit IO Setup

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.
If you do not already have an Adafruit IO account set up, head over to io.adafruit.com to link your Adafruit.com account to Adafruit IO.

We need to create a new feed to hold data for the incoming emails. Navigate to the feeds page on Adafruit IO. Then click actions -> create new feed, and name this feed gmail.

- If you do not already know how to create a feed, head over to Adafruit IO Basics: Feeds.

We're also going to need our Adafruit IO username and active key. Navigate to your profile and click the View AIO Key button to retrieve them. Write them down in a safe place, we'll need them for later.

Now that Adafruit IO is all set up, let's link IFTTT to your Gmail account.
IFTTT Setup

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.

What happened with IFTTT?

Google implemented changes to their API ecosystem that will impact the Gmail service on IFTTT. Unfortunately, this change means that all Gmail triggers were removed from IFTTT.

Please use the Zapier Setup page for this guide, instead of this page.

If This Then That is an internet service which can listen to services on the internet (such as a new tweet) and trigger physical device actions.

Once you're signed up and logged into IFTTT, navigate to the Create page to create your applet and click the +this button:

When prompted to Choose a Service, use the search-bar to look for Gmail.
Next, we'll need to choose a trigger. This is an event which occurs in Gmail that's used to cause our action ("then that") to occur. We'll need to retrieve any email which comes into our inbox.

Select Any new email in inbox

Once the trigger is selected, you'll be taken back to the If Then That page. This time, click on +that

On Step 4, search for Adafruit in the search bar and select Send data to Adafruit IO on the Choose Action page
On Step 5, select gmail as the Feed Name. On the Data to Save field, click Add ingredient and then select ReceivedAt from the dropdown.

Note: While we could select any data element from the gmail feed, we don't want to leak important information such as the sender or subject. So, we'll send date/time data.
Review the applet's settings to make sure they're correct. Then, click finish.
Next, we're going to set up our Arduino for programming.

Zapier Setup

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.
Zapier is a service for connecting web services to your Adafruit IO project. It offers support for Gmail, Twitter, and a few hundred other services. It's similar to IFTTT, but offers some additional flexibility.

First, you'll need an account at Zapier and an invite to the Adafruit IO app on Zapier. Visit zapier.com to sign up, and then use this link for the Adafruit IO app invite.

Launch the Zapier Editor to make a Zap (similar to an IFTTT action). Then, use the searchbar to search for Gmail.

Select New Email as a Gmail Trigger
1. Connect your Gmail account to Zapier
2. Then, click test to ensure it's correctly integrated with your Zapier account.

If you see Success, your Gmail was successfully connected.

Zapier will ask you to select a label or mailbox. Select Inbox.
Note: This step is optional, if no Label/Mailbox is selected, this Zap will trigger on all e-mails, including Drafts.

Next, Zapier will proceed to test your Gmail account. If it integrates, it'll show a success message. Click Next.

You'll also need an Action App. Search and select Adafruit IO (it's still in beta but that's o.k).

On the next step, select Send Value to Feed as an Adafruit IO action.

On "Select Adafruit IO Account", click Connect an Account and enter your Adafruit IO Active Key.

If you’re not sure where to find this, navigate to your Adafruit IO Profile and click View AIO Key.
On the Set up Adafruit IO Value feed, select Date from the dropdown for Value. Then, enter Gmail as the name of the Feed.
Now, Zapier will test your Zap integrated with Gmail! Click the Test button and it'll send a test value to the gmail feed.

Navigate to your gmail Adafruit IO Feed. You should see a new value created 'a few seconds ago'. If you don't, check over the previous steps.

Click Finish and give your Zap a name. Then, turn it on and you're ready to retrieve emails.

Note: Zapier retrieves emails every 5 minutes.

Arduino Setup

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.
Installing Libraries

This guide assumes you've completed the setup required to get your ESP8266 up and running with Arduino IDE and Adafruit IO.

- If you haven't yet set up your ESP8266 for use with Adafruit IO and the Arduino IDE, follow along with this guide. The setup only needs to be performed once.

The code for this example is contained within the Adafruit IO Arduino Library. Open the adafruitio_23_ifttt example from within the Arduino IDE: File->Examples->Adafruit IO Arduino->adafruitio_23_ifttt

Next, we'll perform the network configuration required for this sketch.

Arduino Network Config

To configure the network settings, click on the config.h tab in the sketch. You will need to set your Adafruit IO username in the IO_USERNAME define, and your Adafruit IO key in the IO_KEY define.
WiFi Config

WiFi is enabled by default in config.h so if you are using one of the supported WiFi boards, you will only need to modify the WIFI_SSID and WIFI_PASS options in the config.h tab.

```c
/******************** WIFI ********************/
// the AdafruitIO_WiFi client will work with the following boards...
// - Feather Huzzah ESP8266 -> https://www.adafruit.com
// - Feather M0 WiFi -> https://www.adafruit.com/products/3056
#define WIFI_SSID   "Test Wifi"
#define WIFI_PASS   "my wifi password"

// comment out the following two lines if you are using fona or ethernet
#include "AdafruitIO_WiFi.h"
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
```

FONA Config

If you wish to use the FONA 32u4 Feather to connect to Adafruit IO, you will need to first comment out the WiFi support in config.h

```c
/******************** WIFI ********************/
// the AdafruitIO_WiFi client will work with the following boards...
// - Feather Huzzah ESP8266 -> https://www.adafruit.com
// - Feather M0 WiFi -> https://www.adafruit.com/products/3056
#define WIFI_SSID   "Test Wifi"
#define WIFI_PASS   "my wifi password"

// comment out the following two lines if you are using fona or ethernet
#include "AdafruitIO_WiFi.h"
AdafruitIO_WiFi io(IO_USERNAME, IO_KEY, WIFI_SSID, WIFI_PASS);
```

Next, remove the comments from both of the FONA config lines in the FONA section of config.h to enable FONA support.
Ethernet Config

If you wish to use the Ethernet Wing to connect to Adafruit IO, you will need to first comment out the WiFi support in config.h

```c
#include "AdafruitIO_FONA.h"
AdafruitIO_FONA io(IO_USERNAME, IO_KEY);
```

Next, remove the comments from both of the Ethernet config lines in the Ethernet section of config.h to enable Ethernet Wing support.

```c
#include "AdafruitIO_Ethernet.h"
AdafruitIO_Ethernet io(IO_USERNAME, IO_KEY);
```

Next, we will look at how the example sketch works.

---

**Code**

Google implemented changes to their API ecosystem that impacted the Gmail applet on both IFTTT and Zapier. This guide is no longer possible as a result of these changes and should not be followed.
Let's take a dive into the code that powers our Gmailbox. It's not too hard to understand and this code can serve as an introduction to Arduino and Adafruit IO.

We'll start by setting up the servo holding the flag. We wired the servo to the Feather HUZZAH's pin 14. If you need to change that, you can modify `SERVO_PIN`. We also define the flag's up and down positions (what angle to set the servo's maximum to) as `FLAG_UP` and `FLAG_DOWN`.

Want the servo's flag to be held up longer? You can do that by modifying the `FLAG_DELAY` variable.

```cpp
// pin used to control the servo
#define SERVO_PIN 14

// Flag's up position, in degrees
#define FLAG_UP 0

// Flag's down position, in degrees
#define FLAG_DOWN 180

// How long to hold the flag up, in seconds
#define FLAG_DELAY 2
```

Next, we're going to create an instance of the servo class and set up the gmail feed from Adafruit IO.

```cpp
Servo servo;
AdafruitIO_Feed *gmail_feed = io.feed("gmail");
```

The `setup()` function starts a serial connection, attaches the servo to `SERVO_PIN`, and attempts to connect to Adafruit IO. Then, it creates a message handler to listen to the gmail feed. We'll also write the servo to its' resting state (the `FLAG_DOWN` position).

```cpp
void setup() {
  // start the serial connection
  Serial.begin(115200);

  // wait for serial monitor to open
  while(! Serial);

  // tell the servo class which pin we are using
  servo.attach(SERVO_PIN);

  // connect to io.adafruit.com
  Serial.print("Connecting to Adafruit IO");
  io.connect();

  // set up a message handler for the 'servo' feed.
  // the handleMessage function (defined below)
  // will be called whenever a message is
  // received from adafruit io.
  gmail_feed-&gt;onMessage(handleMessage);
}
The `loop()` is empty aside from a call to `io.run()`, a function which keeps the client connected to `io.adafruit.com`.

```cpp
void loop() {
    io.run();
}
```

The `handleMessage` function is called whenever a new data value is received from the gmail feed on Adafruit IO. Inside this function, we'll print to the serial monitor that a new email has been received.

Then, we'll set the flag to the `FLAG_UP` position to indicate a new email has been received. We'll wait `FLAG_DELAY` seconds and then move the flag back down by writing the servo to its' `FLAG_DOWN` position.

The `handleMessage` function is called when the gmail feed is updated by Zapier or IFTTT.

When it receives new data from the gmail feed, it will write the servo to the `flagUp` position, waits a second so that you can see it, and moves it back to the `flagDown` position.

```cpp
void handleMessage(AdafruitIO_Data *data) {
    Serial.println("You've got mail!");
    servo.write(FLAG_UP);
    // wait FLAG_DELAY seconds
    delay(FLAG_DELAY * 1000);
    servo.write(FLAG_DOWN);
}
```

**Testing out your Gmailbox**

Upload the code to your Feather and open the Arduino Serial Monitor (Tools -> Serial Monitor).
IFTTT Gmailbox
Connecting to Adafruit IO
Connected to Adafruit IO!

If you see the above, we're all set up to receive emails! Compose an email to yourself and send it! When the gmail feed receives new data, the serial monitor will print You've got mail!, put the flag up, and gently lower it.

Help - my Gmailbox isn't working!

I want to trigger my feed to show my friend/coworker the gmailbox, but don't want to clog my inbox

On the gmail feed page, you can click on the actions dropdown -> add data. You can enter any value, it doesn't matter for this project, and click create. You'll see a new data-point dropped onto the IO feed.

The flag on my mailbox isn't moving up and down

First, check that the feed is set up correctly. Visit your feed page () for gmail and ensure your feed was updated when an email arrived.

If your feed did update, but the servo flag didn't move, there's a chance it was wired incorrectly. Re-read the wiring diagram on the Wiring and Assembly page. Make sure SERVO_PIN in the sketch is connected to the correct pin on your Feather HUZZAH.
My feed isn't updating when I get a new email

Check over your Zap or IFTTT Applet configuration. Make sure that the email is connected correctly.

Code

```
// Adafruit IO IFTTT Example - Gmailbox
// Tutorial Link: https://learn.adafruit.com/gmailbox
//
// Adafruit invests time and resources providing this open source code.
// Please support Adafruit and open source hardware by purchasing
// products from Adafruit!
//
// Written by Brent Rubell for Adafruit Industries
// Copyright (c) 2018 Adafruit Industries
// Licensed under the MIT license.
//
// All text above must be included in any redistribution.

/**************************************************************************** Configuration */

// edit the config.h tab and enter your Adafruit IO credentials
// and any additional configuration needed for WiFi, cellular,
// or ethernet clients.
#include "config.h"

/**************************************************************************** Example Starts Here */

// pin used to control the servo
#define SERVO_PIN 14

// Flag's up position, in degrees
#define FLAG_UP 0

// Flag's down position, in degrees
#define FLAG_DOWN 180

// How long to hold the flag up, in seconds
#define FLAG_DELAY 2

// create an instance of the servo class
Servo servo;

// set up the 'servo' feed
AdafruitIO_Feed *gmail_feed = io.feed("gmail");

void setup() {
  Serial.begin(115200);

  while(! Serial);
```

©Adafruit Industries
Serial.print("IFTTT Gmailbox");

// tell the servo class which pin we are using
servo.attach(SERVO_PIN);

// connect to io.adafruit.com
Serial.print("Connecting to Adafruit IO");
io.connect();

// set up a message handler for the 'servo' feed.
// the handleMessage function (defined below)
// will be called whenever a message is
// received from adafruit io.
gmail_feed->onMessage(handleMessage);

// wait for a connection
while(io.status() < AIO_CONNECTED) {
  Serial.print(".");
delay(500);
}

// we are connected
Serial.println();
Serial.println(io.statusText());
gmail_feed->get();

// write flag to down position
servo.write(FLAG_DOWN);

}

void loop() {

// io.run(); is required for all sketches.
// it should always be present at the top of your loop
// function. it keeps the client connected to
// io.adafruit.com, and processes any incoming data.
io.run();

}

// this function is called whenever a 'gmail' message
// is received from Adafruit IO. it was attached to
// the gmail feed in the setup() function above.
void handleMessage(AdafruitIO_Data *data) {

  Serial.println("You've got mail!");
servo.write(FLAG_UP);
// wait FLAG_DELAY seconds
delay(FLAG_DELAY * 1000);
servo.write(FLAG_DOWN);

}