FunHouse Motion Detecting Lights with LIFX Bulbs

Created by John Park

https://learn.adafruit.com/funhouse-motion-detecting-lighting-for-lifx-bulbs

Last updated on 2022-12-01 04:04:22 PM EST

Table of Contents

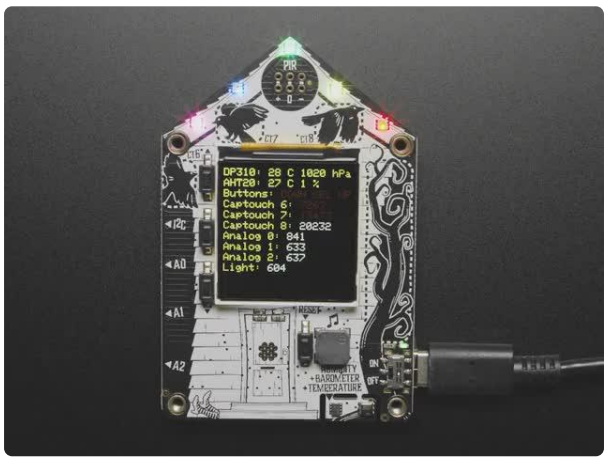
Overview

You can build your own motion detection system to control your room lighting, using the FunHouse, LIFX WiFi light bulbs, and CircuitPython.

Set up a high tech night light. Or a darkroom warning light. Or a psychedelic chill-out zone. All of these things are possible with the mini PIR sensor that detects the presence of a person (or other large IR heat source) and flips the bulb to the color and brightness you decide.

Also uses the FunHouse buttons and display to pause and arm the system as well as to adjust the active timer.

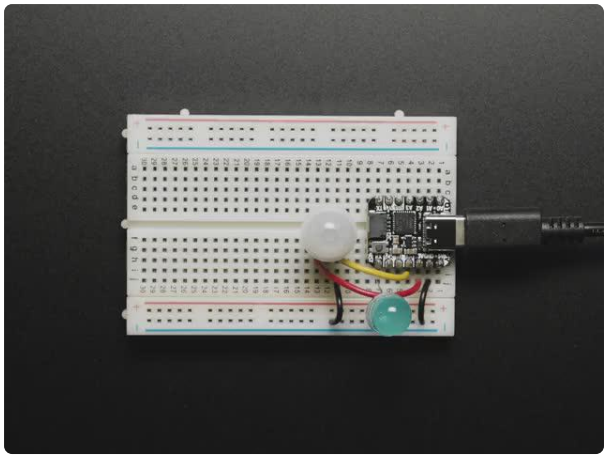
Parts



Adafruit FunHouse - WiFi Home Automation Development Board

Home is where the heart is...it's also where we keep all our electronic bits. So why not wire it up with sensors and actuators to turn our house into an electronic wonderland....

https://www.adafruit.com/product/4985

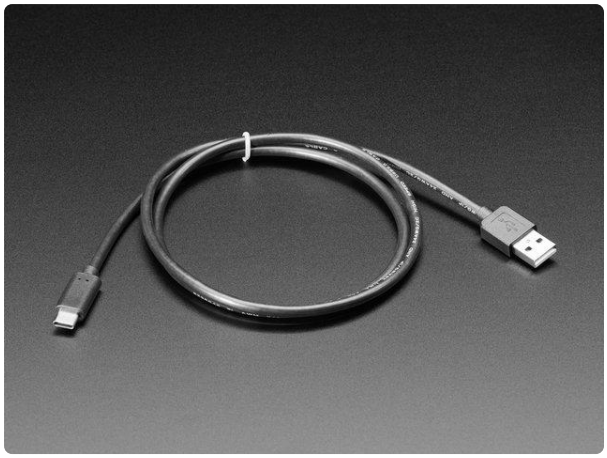


[Breadboard-friendly Mini PIR Motion Sensor with 3 Pin Header]
PIR sensors are used to detect motion from pets/humanoids from about 5 meters away (possibly works on zombies, not guaranteed). This sensor is much smaller than most PIR modules, which...
https://www.adafruit.com/product/4871

LIFX WiFi Light Bulb

WiFi enabled, hub-less, color/color-temperature changing smart bulb by LIFX () (there are other WiFi bulbs out there, but this project is specifically coded for LIFX brand bulbs).

Any of their bulbs will work, and you can modify the code to control multiple bulbs if you like.

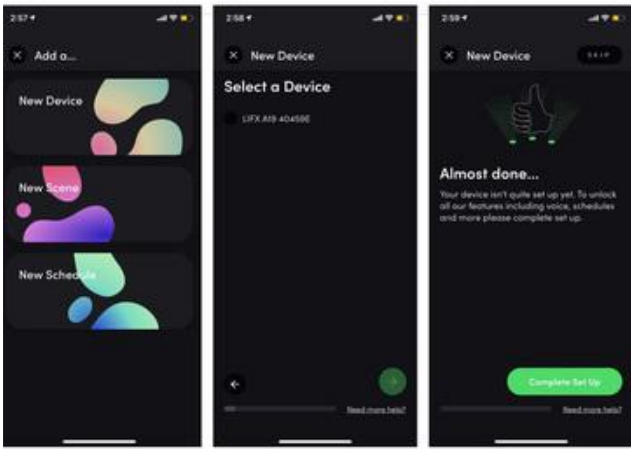


[USB Type A to Type C Cable - approx 1 meter / 3 ft long]
As technology changes and adapts, so does Adafruit. This USB Type A to Type C cable will help you with the transition to USB C, even if you're still...
https://www.adafruit.com/product/4474

[5V 2A Switching Power Supply w/ USB-A Connector]
Our 5V 2A USB power adapter is the perfect choice for powering single-board computers like Raspberry Pi, BeagleBone, or anything else that's power-hungry!This adapter was...
https://www.adafruit.com/product/1994

LIFX Bulb Setup

The LIFX bulbs have a robust API and developer support. In order to use them in your own projects, follow these steps.



Set up bulb with LIFX app

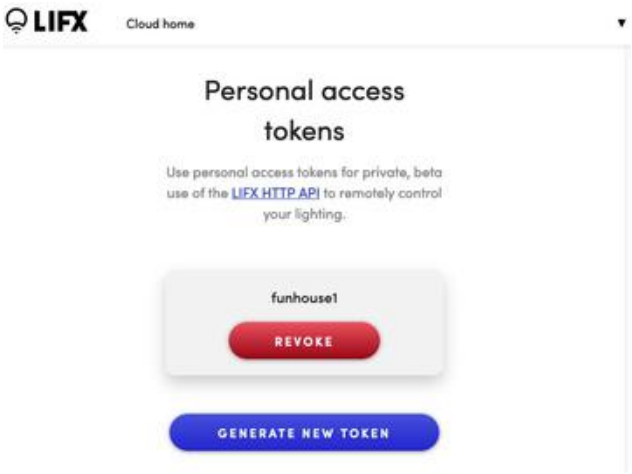
Screw the bulb into a light socket and power it on.

Follow the instructions that came with the bulb for using with the LIFX mobile app for iOS and Android. This process will get your bulb connected to your WiFi, and allow you to control the bulb directly from your mobile device.

During the setup, name your bulb Lamp so that it will be the same name used in our CircuitPython code.

LIFX Cloud API Access Token

Next, you'll sign in to the LIFX Cloud (). (You'll use the same account credentials you set up with the mobile app.)



In the account settings page, click on Generate New Token to create your HTTP API access token, which will be used in the secrets.py file on your FunHouse in order to communicate with the bulb. You can read about it here in the LIFX Developer Zone ().

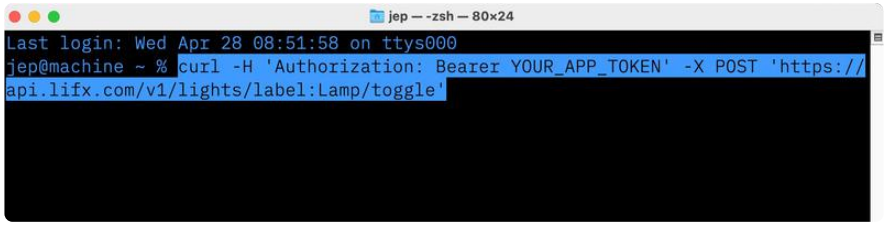
Write down this token in a secure place, you'll need it later.

> If you ever need to reset your bulb to factory settings, turn the switch on and off five times in a row!

Test the Connection

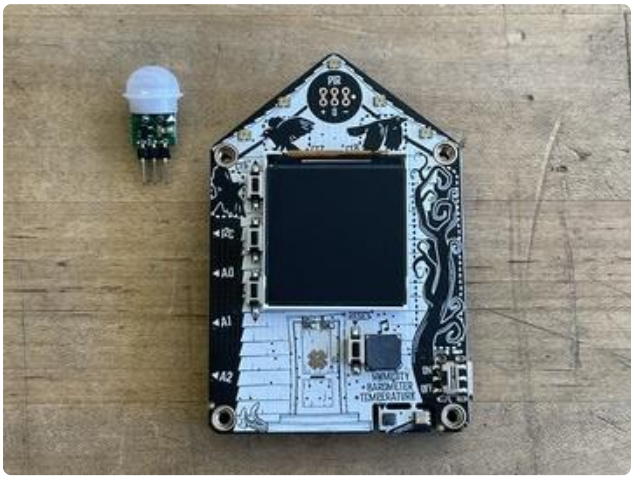
Now that you have your token, you can try out a quick test from a command line terminal. Copy and paste the following, substituting the `YOUR_APP_TOKEN` for your unique token instead:

```
curl -H 'Authorization: Bearer YOUR_APP_TOKEN' -X POST 'https://api.lifx.com/v1/lights/label:Lamp/toggle'
```



Press enter and your light will toggle off or on each time you repeat the command, and the light will report its status.

Add the Mini PIR Sensor



Align the Polarity

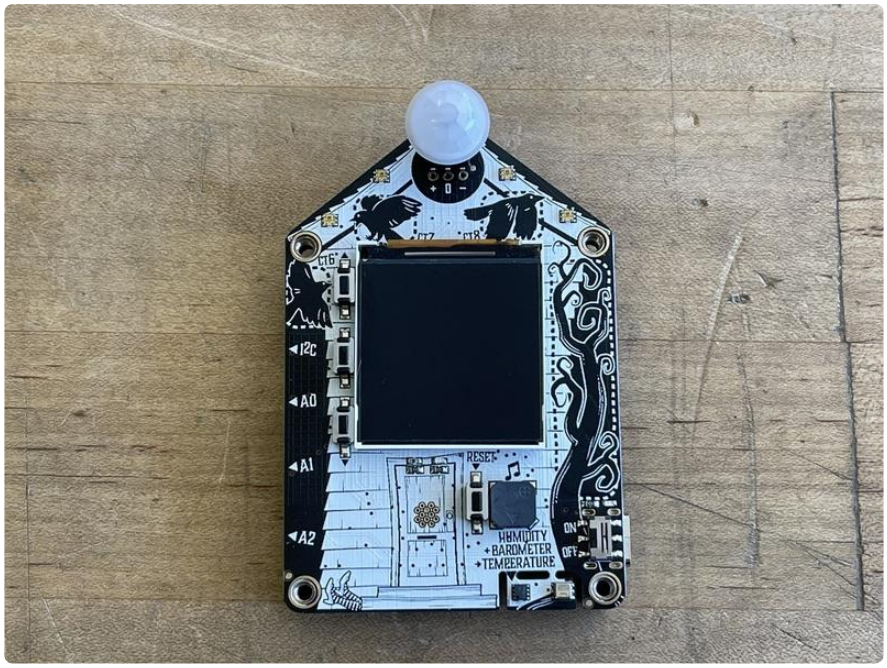
Adding the mini PIR sensor to the FunHouse is pretty simple, you just need to check the polarity.

Look for the + symbol on the PIR sensor's PCB and match it up with the + symbol on the front side of the FunHouse board's PIR row.

Insert Sensor

Align the pins with the socket holes on the front side of the board.

The first time you insert the sensor the fit is tight, but just apply firm down ward pressure while wiggling the board a little bit and it will go in just fine.



Code the FunHouse Lighting for LIFX

Install CircuitPython

The first think to do is install CircuitPython on your Funhouse. Follow this guide () to get it set up with the latest release version.

Shhhh... Secrets

In order for the Funhouse to connect to the internet, you'll need to include a secrets.py file on the board that contains your WiFi access point ssid and password, as well as your AIO key.

Follow this guide page () to get your secrets.py file set up.

Add LIFX Token to secrets.py

Add your LIFX token to the secrets.py file on it's own line. It should look something like this:

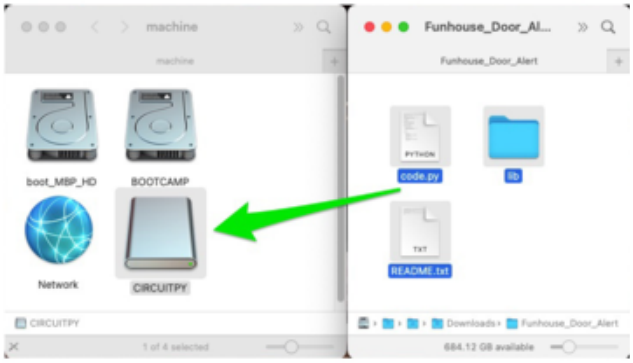
```
# This file is where you keep secret settings, passwords, and tokens!
    # If you put them in the code you risk committing that info or sharing it

secrets = {
  'ssid' : 'your_access_point',
  'password' : 'your_wifi_password',
  'lifx_token' : 'YOUR_APP_TOKEN'
}
```

Text Editor

Adafruit recommends using the Mu editor for using your CircuitPython code with the Funhouse. You can get more info in this guide ().

Download the Project Bundle

Your project will use a specific set of CircuitPython libraries and the code.py file. In order to get the libraries you need, click on the Download Project Bundle link below, and uncompress the .zip file.



Next, drag the contents of the uncompressed bundle directory onto you microcontroller board CIRCUITPY drive, replacing any existing files or directories with the same names, and adding any new ones that are necessary.

```
# SPDX-FileCopyrightText: 2021 John Park for Adafruit Industries
# SPDX-License-Identifier: MIT
# FunHouse PIR Motion Sensor for LIFX light bulbs
import time
import ssl
import socketpool
import wifi
import adafruit_requests
from adafruit_funhouse import FunHouse
import adafruit_lifx

# Get wifi details and more from a secrets.py file
try:
    from secrets import secrets
except ImportError:
    print("WiFi and API secrets are kept in secrets.py, please add them there!")
    raise

# choose colors here. Note formatting differences.
default_bulb_color = "#002010"
default_led_color = 0x002010
tripped_bulb_color = "#440044"
tripped_led_color = 0x440044

# Set up ESP32-S2 and adafruit_requests session
wifi.radio.connect(ssid=secrets["ssid"], password=secrets["password"])
pool = socketpool.SocketPool(wifi.radio)
http_session = adafruit_requests.Session(pool, ssl.create_default_context())

# Add your LIFX Personal Access token to secrets.py
# (to obtain a token, visit: https://cloud.lifx.com/settings)
lifx_token = secrets["lifx_token"]

# Set this to your LIFX light separator label
# https://api.developer.lifx.com/docs/selectors
lifx_light = "label:Lamp"

# Initialize the LIFX API Client
lifx = adafruit_lifx.LIFX(http_session, lifx_token)

# List all lights
lights = lifx.list_lights()
# print(lights)  # uncomment for lots of LIFX light info

funhouse = FunHouse(default_bg=0x000F20, scale=3)

pir_state = 0
running_state = False
trip_time = 30  # seconds to stay tripped, adjust this with buttons while running
```

```
funhouse.peripherals.dotstars.fill(default_led_color)


def set_label_color(conditional, index, on_color):
    if conditional:
        funhouse.set_text_color(on_color, index)
    else:
        funhouse.set_text_color(0x606060, index)


# Create the labels
funhouse.display.show(None)
up_label = funhouse.add_text(text="+", text_position=(3, 6), text_color=0x606060)
down_label = funhouse.add_text(text="-", text_position=(3, 40), text_color=0x606060)
running_label = funhouse.add_text(
    text="paused", text_position=(2, 68), text_color=0x606060
)
time_label = funhouse.add_text(
    text=trip_time, text_scale=2, text_position=(30, 25), text_color=0x606060
)

funhouse.display.show(funhouse.splash)

# Turn on the light
print("Turning on light...")
lifx.toggle_light(lifx_light)

# Set the light's brightness
light_brightness = 0.65
lifx.set_brightness(lifx_light, light_brightness)
lifx.set_color(
    lifx_light, power="on", color=default_bulb_color, brightness=light_brightness
)


while True:

    if funhouse.peripherals.button_up:
        trip_time = trip_time + 1
        funhouse.set_text(trip_time, time_label)
        funhouse.set_text_color(0xFFFFFF, up_label)
        time.sleep(0.2)
    else:
        funhouse.set_text_color(0x606060, up_label)

    if funhouse.peripherals.button_sel:
        trip_time = abs(trip_time - 1)
        funhouse.set_text(trip_time, time_label)
        funhouse.set_text_color(0xFFFFFF, down_label)
        time.sleep(0.2)
    else:
        funhouse.set_text_color(0x606060, down_label)

    if funhouse.peripherals.button_down:
        if running_state is False:  # it's currently paused, so unpause it
            running_state = True  # flip the state
            funhouse.set_text("..prepping..", running_label)
            time.sleep(6)  # pause to get out of range
            funhouse.set_text("sensing...", running_label)

        else:  # it's currently running, so pause it
            running_state = False
            funhouse.set_text("paused", running_label)
            time.sleep(0.5)

    # when sensor is tripped, set the color x amount of time
    if running_state is True and funhouse.peripherals.pir_sensor and pir_state is 0:
        funhouse.peripherals.dotstars.fill(tripped_led_color)
        funhouse.set_text("tripped", running_label)
```

```
        lifx.set_color(
            lifx_light,
            power="on",
            color=tripped_bulb_color,
            brightness=light_brightness,
        )
        prior_trip_time = trip_time  # store the state of the trip time value
        for _ in range(trip_time):
            time.sleep(1)
            trip_time = trip_time - 1
            funhouse.set_text(trip_time, time_label)
        pir_state = 1
        trip_time = prior_trip_time  # restore the trip time value

    # return to default color
    elif (
            running_state is True and not funhouse.peripherals.pir_sensor and
pir_state is 1
    ):
        funhouse.peripherals.dotstars.fill(default_led_color)
        funhouse.set_text("sensing...", running_label)
        lifx.set_color(
            lifx_light,
            power="on",
            color=default_bulb_color,
            brightness=light_brightness,
        )
        funhouse.set_text(trip_time, time_label)
        pir_state = 0
```

Code Walkthrough

How It Works

Here's how the code works.

Libraries

First, you'll import the libraries needed including `time`, `ssl`, `socketpool`, `wifi`, and `adafruit_requests` for creating the connection to the bulb, `adafruit_funhouse` for all of the convenient functions for using the sensors, display, NeoPixels, and buttons on the FunHouse easily, and the `adafruit_lifx` library for specific bulb functions.

```
import time
import ssl
import socketpool
import wifi
import adafruit_requests
from adafruit_funhouse import FunHouse
import adafruit_lifx
```

WiFi Setup

Next, the code checks the secrets.py file for the WiFi credentials.

```
try:
    from secrets import secrets
except ImportError:
    print("WiFi and API secrets are kept in secrets.py, please add them there!")
    raise
```

Color Variables

You can pick your standby and active (sensor has been "tripped") colors here. Note that these are hexidecimal color values formatted two ways for the bulb, which uses HTML style hex formatting, and the FunHouse DotStars, which use more standard hex code formatting.

```
default_bulb_color = "#002010"
default_led_color = 0x002010
tripped_bulb_color = "#440044"
tripped_led_color = 0x440044
```

Requests Session

Here the ESP32-S2 sets up a WiFi HTTP socket session.

```
wifi.radio.connect(ssid=secrets["ssid"], password=secrets["password"])
pool = socketpool.SocketPool(wifi.radio)
http_session = adafruit_requests.Session(pool, ssl.create_default_context())
```

LIFX Setup

Next you'll set up the LIFX bulb using your token from the secrets.py file, the bulb name, and the http session. The `lights` variable is set by querying the LIFX light list.

```
lifx_token = secrets["lifx_token"]
lifx_light = "label:Lamp"
lifx = adafruit_lifx.LIFX(http_session, lifx_token)
lights = lifx.list_lights()
```

FunHouse Setup

The FunHouse object is created, setting a background color for the screen and a text scale of 3 and the DotStars are turned on to their default color.

Three variables are created to store the PIR sensor state, the running/paused state, and the initial on timer value.

```
funhouse = FunHouse(default_bg=0x000F20, scale=3)

pir_state = 0
running_state = False
trip_time = 30  # seconds to stay tripped, adjust this with buttons while running

funhouse.peripherals.dotstars.fill(default_led_color)
```

Text Display

Next the screen labels are set up, including a function for toggling label colors when buttons are pressed.

```
def set_label_color(conditional, index, on_color):
    if conditional:
        funhouse.set_text_color(on_color, index)
    else:
        funhouse.set_text_color(0x606060, index)


# Create the labels
funhouse.display.show(None)
up_label = funhouse.add_text(text="+", text_position=(3, 6), text_color=0x606060)
down_label = funhouse.add_text(text="-", text_position=(3, 40), text_color=0x606060)
running_label = funhouse.add_text(
    text="paused", text_position=(2, 68), text_color=0x606060
)
time_label = funhouse.add_text(
    text=trip_time, text_scale=2, text_position=(30, 25), text_color=0x606060
)

funhouse.display.show(funhouse.splash)
```

Turn on the Light

The final step of setup is to turn on the LIFX bulb with a brightness of 65% and the default teal color.

```
lifx.toggle_light(lifx_light)
light_brightness = 0.65
lifx.set_brightness(lifx_light, light_brightness)
lifx.set_color(
```

```
    lifx_light, power="on", color=default_bulb_color, brightness=light_brightness
)
```

Main Loop

The main loop of the program does the following:

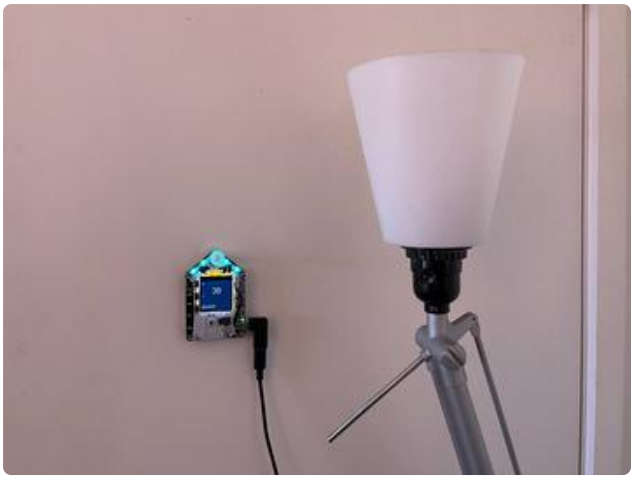
- When the top button is pressed, increase the timer variable and highlight the text
- When the middle button is pressed, decrease the timer variable and highlight the text
- When the bottom button is pressed toggle the `running_state` value and swap the text
- Watch for the PIR sensor value to change -- if the sensor is tripped, set the LIFX `set_color` and the DotStars to the "tripped" color, and then countdown the timer. When the timer runs down, return to the default color.

```
while True:

    if funhouse.peripherals.button_up:
        trip_time = trip_time + 1
        funhouse.set_text(trip_time, time_label)
        funhouse.set_text_color(0xFFFFFF, up_label)
        time.sleep(0.2)
    else:
        funhouse.set_text_color(0x606060, up_label)

    if funhouse.peripherals.button_sel:
        trip_time = abs(trip_time - 1)
        funhouse.set_text(trip_time, time_label)
        funhouse.set_text_color(0xFFFFFF, down_label)
        time.sleep(0.2)
    else:
        funhouse.set_text_color(0x606060, down_label)

    if funhouse.peripherals.button_down:
        if running_state is False:  # it's currently paused, so unpause it
            running_state = True  # flip the state
            funhouse.set_text("..prepping..", running_label)
            time.sleep(6)  # pause to get out of range
            funhouse.set_text("sensing...", running_label)

        else:  # it's currently running, so pause it
            running_state = False
            funhouse.set_text("paused", running_label)
            time.sleep(0.5)

    # when sensor is tripped, set the color x amount of time
    if running_state is True and funhouse.peripherals.pir_sensor and pir_state is 0:
        funhouse.peripherals.dotstars.fill(tripped_led_color)
        funhouse.set_text("tripped", running_label)
        lifx.set_color(
            lifx_light,
            power="on",
            color=tripped_bulb_color,
            brightness=light_brightness,
        )
```

```
        prior_trip_time = trip_time  # store the state of the trip time value
        for _ in range(trip_time):
            time.sleep(1)
            trip_time = trip_time - 1
            funhouse.set_text(trip_time, time_label)
        pir_state = 1
        trip_time = prior_trip_time  # restore the trip time value

    # return to default color
    elif (
            running_state is True and not funhouse.peripherals.pir_sensor and
pir_state is 1
    ):
        funhouse.peripherals.dotstars.fill(default_led_color)
        funhouse.set_text("sensing...", running_label)
        lifx.set_color(
            lifx_light,
            power="on",
            color=default_bulb_color,
            brightness=light_brightness,
        )
        funhouse.set_text(trip_time, time_label)
        pir_state = 0
```

Use the FunHouse Lighting for LIFX Sensor

Here's how to use the FunHouse Lighting for LIFX controller.



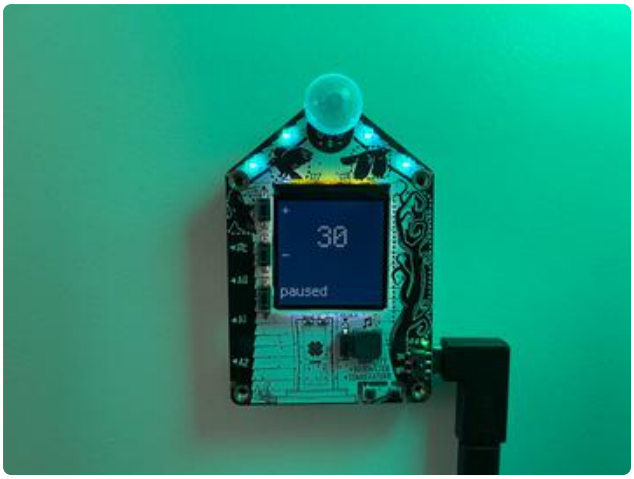
Mount the FunHouse

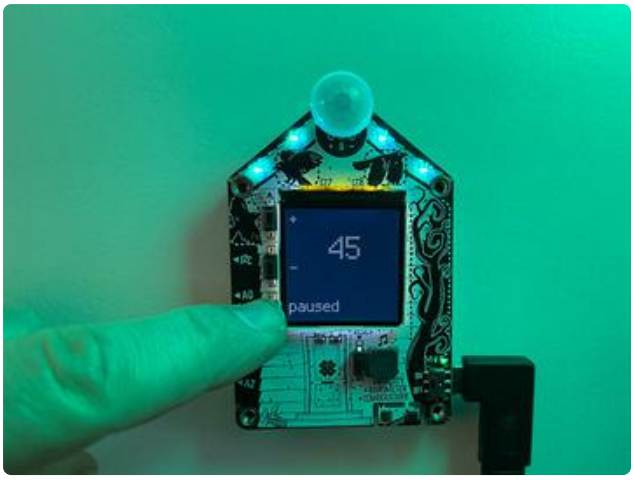
Mount the FunHouse to your wall using the mounting plate, or rest it on a desk or other surface. Make sure the mini PIR sensor has a clear view of the area you want to motion detect for people entering the space.

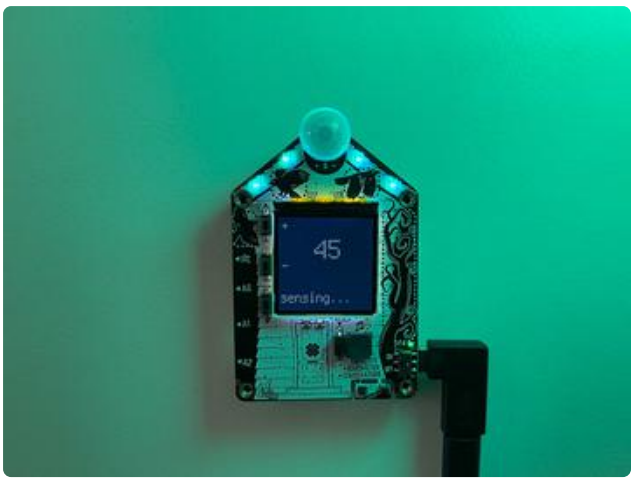
Turn on the LIFX bulb switch.

Startup

The FunHouse will start up and turn on the LIFX bulb to the default color set in code.







Adjust Timer and Arm Sensor

You can use the top and middle buttons on the FunHouse to adjust the timer.

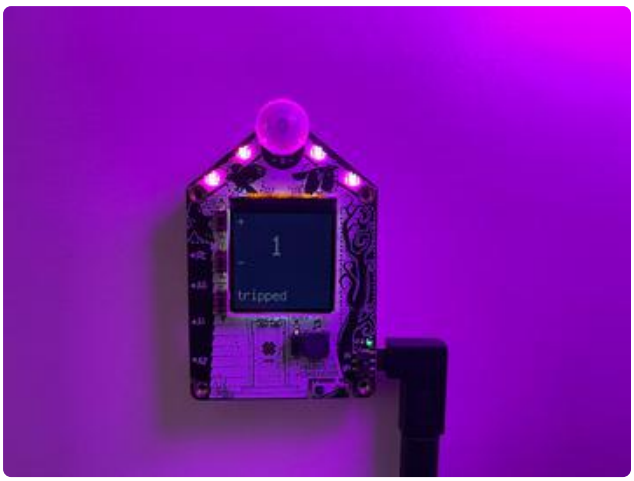
When ready to use the sensor, press the bottom button, which will change the display from paused to ..prepping.. and then to sensing...

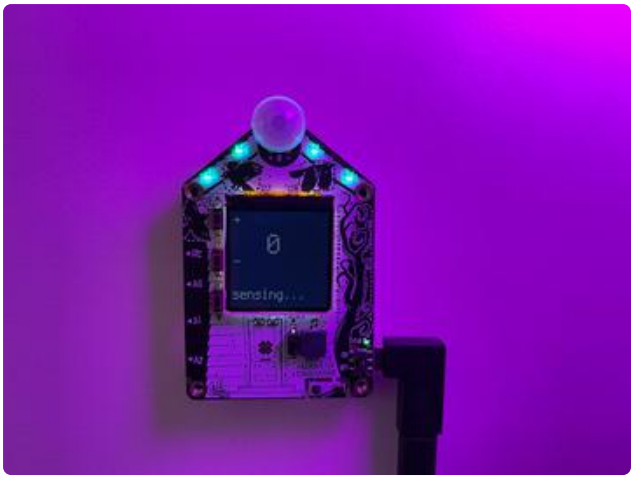
It is now sensing for big IR sources, such as, well, you!

Motion Detected

When a person is detected the Dotstars will change to the tripped color and the command will be sent over WiFi to the LIFX bulb (via the Internet).

The tripped color will remain while the timer counts down.





No Motion Detected

Once the timer has run down and no motion is detected, the bulb will return to the default color.