CircuitPython BLE Advertising Beacons

Created by John Park

https://learn.adafruit.com/circuitpython-ble-advertising-beacons

Last updated on 2023-08-29 04:22:21 PM EDT
Table of Contents

Overview 3
- Parts

Understanding BLE Advertising Beacons 6
- Advertising
- Other Uses

Simple Advertising Beacons 7
- CircuitPython Setup

CircuitPython for ItsyBitsy nRF52840 Express 8
- Set up CircuitPython Quick Start!
- Further Information

Simple Beacon Code in CircuitPython 10
- Text Editor
- CircuitPython Libraries & Code.py
- Code Explainer
- Advertising
- Usage

CLUE Beacon 14
- Libraries
- Text Editor
- Image Files and Code.py
- How It Works
- Setup
- Main Loop
- Buttons
- Beacon Advertising
- CLUE Beacon Usage
- Pick Beacon Advertisement
- QR Code
Overview

Mmmmm, bacon.....

What?

Oh. Beacons? Yes, beacons!

Beacons are low power, low cost Bluetooth LE devices that can be used for a very simple job: to advertise a URL to users running an app on their mobile device within range of the device!

Technically, beacons can also be used for a few different scenarios where a localized, wireless data broadcast is desirable, such as indoor location tracking, point of sales/ marketing, and experience personalization.

We'll create two version -- first, a simple beacon that can run on any CircuitPython capable nRF52840 board. Second, a custom version for the CLUE board that also shows QR codes and allows you to scroll among multiple beacon URL choices to advertise!

Parts

All you need to advertise your own BLE beacons is any Adafruit Bluefruit nRF52840 board:
Adafruit ItsyBitsy nRF52840 Express - Bluetooth LE
What's smaller than a Feather but larger than a Trinket? It's an Adafruit ItsyBitsy nRF52840 Express featuring the Nordic nRF52840 Bluetooth LE...
https://www.adafruit.com/product/4481

Adafruit CLUE - nRF52840 Express with Bluetooth LE
Do you feel like you just don't have a CLUE? Well, we can help with that - get a CLUE here at Adafruit by picking up this sensor-packed development board. We wanted to build some...
https://www.adafruit.com/product/4500

Adafruit Feather nRF52840 Express
The Adafruit Feather nRF52840 Express is the new Feather family member with Bluetooth Low Energy and native USB support featuring the nRF52840! It's...
https://www.adafruit.com/product/4062

Circuit Playground Bluefruit - Bluetooth Low Energy
Circuit Playground Bluefruit is our third board in the Circuit Playground series, another step towards a perfect introduction to electronics and programming. We've...
https://www.adafruit.com/product/4333
You'll also want a USB cable for programming it and power, or you can use a LiPoly battery for remote powering.

**USB cable - USB A to Micro-B**
This here is your standard A to micro-B USB cable, for USB 1.1 or 2.0. Perfect for connecting a PC to your Metro, Feather, Raspberry Pi or other dev-board or...
https://www.adafruit.com/product/592

**Lithium Ion Polymer Battery - 3.7v 500mAh**
Lithium-ion polymer (also known as 'lipo' or 'lipoly') batteries are thin, light, and powerful. The output ranges from 4.2V when completely charged to 3.7V. This...
https://www.adafruit.com/product/1578

**Adafruit Micro Lipo - USB Lilon/LiPoly charger**
Oh so adorable, this is the tiniest little lipo charger, so handy you can keep it any project box! Its also easy to use. Simply plug in the gold plated contacts into any USB port and a...
https://www.adafruit.com/product/1304
Adafruit LiIon/LiPoly Backpack Add-On for Pro Trinket/ItsyBitsy

If you have an ItsyBitsy or Pro Trinket you probably know it's the perfect little size for a portable project. This LiPoly backpack makes it really easy to do! Instead of wiring 2...

https://www.adafruit.com/product/2124

Additionally, you'll need a mobile device running an Eddystone beacon discovery app, such as Physical Web:

- Physical Web for iOS ()
- Physical Web for Android ()

Understanding BLE Advertising Beacons

Bluetooth LC beacons are low power devices that are used to broadcast a tiny amount of data to any device that's listening. This often takes the form of a mobile device running an app, such as Physical Web, that can read the beacon's advertising broadcast and display a URL the user can then visit.
There are two primary BLE beacon standards: Apple's iBeacon, and Google's Eddystone. In this guide, we'll use the Eddystone standard to create a beacon that advertises a website URL (which you can customize to any URL you like).

**Advertising**

When an Eddystone beacon is running, it transmits a packet of information that includes a Unique Identifier (UID), and another packet of information that includes a website URL.

Note, this data is transmitted with no need for connecting, pairing, or bonding with a mobile device. The beacon is pretty much just always sitting there yelling, "Hey! I'm a beacon! This is my name, and this is a URL I think you should visit!"

**Other Uses**

This guide focuses solely on using beacons to advertise a URL, however there are a few other common uses for beacons. These include indoor navigation, tracking (such as lost key finders like Tile), and interaction notifications, such as interactive museum tours.

---

**Simple Advertising Beacons**
CircuitPython Setup

For our simple beacon example, we'll use the ItsyBitsy nRF52840. We'll show you how to set it up with the proper libraries and code to use on the Itsy, but you can apply these instructions to the Feather nRF52840, Circuit Playground Bluefruit, or CLUE boards as well by downloading the appropriate version of CircuitPython for those boards.

CircuitPython for ItsyBitsy nRF52840 Express

CircuitPython () is a derivative of MicroPython () designed to simplify experimentation and education on low-cost microcontrollers. It makes it easier than ever to get prototyping by requiring no upfront desktop software downloads. Simply copy and edit files on the CIRCUITPY drive to iterate.

Set up CircuitPython Quick Start!

Follow this quick step-by-step for super-fast Python power :) 

Download the latest version of CircuitPython for this board via CircuitPython.org

Further Information

For more detailed info on installing CircuitPython, check out Installing CircuitPython ().

Click the link above and download the latest UF2 file.

Download and save it to your desktop (or wherever is handy).
Plug your Itsy nRF52840 into your computer using a known-good USB cable.

A lot of people end up using charge-only USB cables and it is very frustrating! So make sure you have a USB cable you know is good for data sync.

In the image, the Reset button is indicated by the magenta arrow, and the BTLE status LED is indicated by the green arrow.

Double-click the Reset button on your board (magenta arrow), and you will see the BTLE LED (green arrow) will pulse quickly then slowly blue. If the DotStar LED turns red, check the USB cable, try another USB port, etc.

If double-clicking doesn't work the first time, try again. Sometimes it can take a few tries to get the rhythm right!
You will see a new disk drive appear called ITSY840BOOT.

Drag the adafruit_circuitpython_etc.uf2 file to ITSY840BOOT.

The LED will flash. Then, the ITSY840BOOT drive will disappear and a new disk drive called CIRCUITPY will appear.

That's it, you're done! :)

---

**Simple Beacon Code in CircuitPython**

**Text Editor**

Adafruit recommends using the Mu editor for using your CircuitPython code with Adafruit boards. You can get more info in [this guide](https://docs.adafruit.com/).
Alternatively, you can use any text editor that saves files.

## CircuitPython Libraries & Code.py

To use with CircuitPython, you need to first install a few libraries, into the lib folder on your CIRCUITPY drive. Then you need to update code.py with the example script.

Thankfully, we can do this in one go. In the example below, click the Download Project Bundle button below to download the necessary libraries and the code.py file in a zip file. Extract the contents of the zip file, open the directory examples/ and then click on the directory that matches the version of CircuitPython you're using and copy the contents of that directory to your CIRCUITPY drive.

Your CIRCUITPY drive should now look similar to the following image:

```python
import time
import adafruit_ble
from adafruit_ble_eddystone import uid, url
radio = adafruit_ble.BLERadio()

eddystone_uid = uid.EddystoneUID(radio.address_bytes)
eddystone_url = url.EddystoneURL("https://adafruit/discord")

while True:
    radio.start_advertising(eddystone_uid)
    time.sleep(0.5)
    radio.stop_advertising()
    radio.start_advertising(eddystone_url)
    time.sleep(0.5)
    radio.stop_advertising()

    time.sleep(4)
```

---

© Adafruit Industries
Code Explainer

Libraries

Here's how this code works. First we import the `time` and `adafruit_ble` libraries.

Then, we import `uid` and `url` from the `adafruit_ble_eddystone` library. These two give us the capability to advertise a unique identifier (UID) and a website URL.

```python
import time
import adafruit_ble
from adafruit_ble_eddystone import uid, url
```

Radio & Eddystone Setup

Next we'll create an instance of the `BLERadio()` and then create a variable for the `eddystone_uid`. This is the unique ID that is used to differentiate beacons when multiple of them are present. We can re-use the BLE `radio.address_bytes` as our UID.

We'll also create a variable for the website URL we want to be advertised from our beacon, `eddystone_url`.

In this case, we are using `https://adafruit.discord` as the beacon URL, but you can swap in a URL of your choosing. Just be careful it isn't too long -- anything under 18 characters will work, otherwise you can use a URL shortener service, such as bit.ly.

```python
radio = adafruit_ble.BLERadio()
# Reuse the BLE address as our Eddystone instance id.
eddystone_uid = uid.EddystoneUID(radio.address_bytes)
eddystone_url = url.EddystoneURL("https://adafruit/discord")
```

You may get away with a longer URL than you think you should -- this is because for certain common URL prefixes (`https://www.` for example) and domains (`.com`, `.org`, etc.) we substitute them for a single non-printing byte.

Advertising

With setup complete, the main, ever-repeating loop of the code now takes over. Here, we alternate between advertising the beacon's UID and the beacon's URL, pause for four seconds, and repeat.
while True:
    # Alternate between advertising our ID and our URL.
    radio.start_advertising(eddystone_uid)
    time.sleep(0.5)
    radio.stop_advertising()

    radio.start_advertising(eddystone_url)
    time.sleep(0.5)
    radio.stop_advertising()

    time.sleep(4)

Usage

To use it, simply power up the beacon and then launch your Physical Web app (or other beacon-aware app) on your mobile device. You’ll see the app scan for beacons.

When the beacon is found, you will see some info about it, including a clickable link to take you to the advertised URL.

Any icons and descriptive text are scraped from the website itself, and not part of the data delivered by the beacon.

That's all there is to it! Next we'll create a more sophisticated version on the CLUE.
First, set up CircuitPython on the CLUE following the instructions on this page.

Libraries
Next, install the libraries needed. This guide page will show you where to download them.

You'll need the following libraries for this project:

- adafruit_bitmap_font
- adafruit_ble
- adafruit_ble_eddystone
- adafruit_bus_device
- adafruit_display_shapes
- adafruit_display_text
- adafruit_pybadger
- adafruit_register
- adafruit_lis3mdl.mpy
- adafruit_lsm6ds.mpy
- adafruit_miniqr.mpy
- neopixel.mpy
Adafruit recommends using the Mu editor for using your CircuitPython code with the Feather boards. You can get more info in [this guide](https://example.com).

Alternatively, you can use any text editor that saves files.

### Image Files and Code.py

This project uses a couple of .bmp image files -- you can get them by downloading the Project Zip link in the embed below. Unzip the downloaded zip archive and then drag the two image files to your CLUE board's CIRCUITPY drive.

Copy the code shown below, paste it into Mu. Save the code from Mu to the CLUE's CIRCUITPY drive as code.py

```python
# SPDX-FileCopyrigh...: MIT

### Eddystone Beacon for CLUE
This example broadcasts our Mac Address as our Eddystone ID and a link to a URL of your choice.
Hold the A button to display QR code, use B button to pick URL from the list.

```python
import time
from adafruit_pybadger import pybadger
import adafruit_ble
from adafruit_ble_eddystone import uid, url

radio = adafruit_ble.BLERadio()

# Reuse the BLE address as our Eddystone instance id.
eddystone_uid = uid.EddystoneUID(radio.address_bytes)

# List of URLs to broadcast here:
ad_url = [("https://circuitpython.org", "CirPy"),
         ("https://adafruit.it/discord","DISCORD"),
         ("https://forums.adafruit.com", "Forums"),
         ("https://learn.adafruit.com", "Learn")
]
pick = 0  # use to increment url choices

pybadger.play_tone(1600, 0.25)
pybadger.show_business_card(image_name="cluebeacon.bmp")

while True:
    pybadger.auto_dim_display(delay=3, movement_threshold=4)
    eddystone_url = url.EddystoneURL(ad_url[pick][0])

    if pybadger.button.a and not pybadger.button.b:  # Press button A to show QR code
        pybadger.play_tone(1200, 0.1)
        pybadger.brightness = 1
        pybadger.show_qr_code(data=ad_url[pick][0])  # Tests QR code
```

©Adafruit Industries
```python
time.sleep(0.1)  # Debounce

elif pybadger.button.b and not pybadger.button.a:  # iterate through urls to broadcast
    pick = (pick + 1) % len(ad_url)
    pybadger.play_tone(1600, 0.2)
    pybadger.brightness = 1
    pybadger.show_business_card(image_name="bg.bmp", name_string=ad_url[pick][1], name_scale=5,
                              email_string_one="", email_string_two=ad_url[pick][0])
    time.sleep(0.1)

elif pybadger.button.a and pybadger.button.b:
    pybadger.play_tone(1000, 0.2)
    pybadger.brightness = 1
    pybadger.show_business_card(image_name="cluebeacon.bmp")
    time.sleep(0.1)

# Alternate between advertising our ID and our URL.
radio.start_advertising(eddystone_uid)
radio.sleep(0.5)
radio.stop_advertising()
radio.start_advertising(eddystone_url)
radio.sleep(0.5)
radio.stop_advertising()

time.sleep(1)
```

---

**How It Works**

Like the simple example before, we will import libraries for time, ble, and eddystone support. We will also import the `adafruit_pybadger` library which makes it convenient to use the buttons, speaker, and screen on the CLUE.

```python
import time
from adafruit_pybadger import pybadger
```
import adafruit_ble
from adafruit_ble_eddystone import uid, url

Setup

We'll set up the radio and eddystone_uid as before, but this time instead of a single URL, we'll create a list of few to pick through.

This is a list of tuples, which each entry containing both a URL and a "nice name".

The pick variable will be used to select a URL from the list when the B button is pressed.

```python
radio = adafruit_ble.BLERadio()
# Reuse the BLE address as our Eddystone instance id.
eddystone_uid = uid.EddystoneUID(radio.address_bytes)
# List of URLs to broadcast here:
ad_url = ["https://circuitpython.org", "CirPy"],
       ["https://adafruit/discord", "DISCORD"],
       ["https://forums.adafruit.com", "Forums"],
       ["https://learn.adafruit.com", "Learn"
    ]
pick = 0  # use to increment url choices
```

We'll use the pybadger.play_tone() command to beep the on-board speaker, and the pybadger.show_business_card() command to display a .bmp image.

```python
pybadger.play_tone(1600, 0.25)
pybadger.show_business_card(image_name="cluebeacon.bmp")
```

Main Loop

In the main loop of the program we will set the display to auto dim after three seconds, and set the movement threshold so you can shake it a little to wake up the CLUE and undim it.

The eddystone_url is set to the first one in the list.

```python
pybadger.auto_dim_display(delay=3, movement_threshold=4)
eddystone_url = url.EddystoneURL(ad_url[pick][0])
```
Buttons

When the A button is pressed by itself, the `pybadger.show_qr_code()` command is run, displaying the automatically generated QR code image for the currently selected URL.

```python
if pybadger.button.a and not pybadger.button.b:  # Press button A to show QR code
    pybadger.play_tone(1200, 0.1)
    pybadger.brightness = 1
    pybadger.show_qr_code(data=ad_url[pick][0])  # Tests QR code
    time.sleep(0.1)  # Debounce
```

When the B button is pressed, the next URL in the list is selected, and this info is displayed on the screen using the `pybadger.show_business_card()` command.

```python
elif pybadger.button.b and not pybadger.button.a:  # iterate through urls to broadcast
    pick = (pick + 1) % len(ad_url)
    pybadger.brightness = 1
    pybadger.show_business_card(image_name="bg.bmp", name_string=ad_url[pick][1],
    email_string_one="", email_string_two=ad_url[pick][0])
    time.sleep(0.1)
```

When both A and B buttons are pressed at the same time, the display returns to the initial screen image.

```python
elif pybadger.button.a and pybadger.button.b:
    pybadger.play_tone(1000, 0.2)
    pybadger.brightness = 1
    pybadger.show_business_card(image_name="cluebeacon.bmp")
    time.sleep(0.1)
```

Beacon Advertising

Finally, we have the beacon advertising code, just as with the simple example. The beacon UID is advertised, then the URL is, and this repeats every second.

```python
# Alternate between advertising our ID and our URL.
radio.start_advertising(eddystone_uid)
    time.sleep(0.5)
radio.stop_advertising()

radio.start_advertising(eddystone_url)
    time.sleep(0.5)
radio.stop_advertising()

    time.sleep(1)
```
CLUE Beacon Usage
Power up the CLUE and you'll see the welcome screen.

Pick Beacon Advertisement
Press the B button to pick a URL from your list. There is a small beep when the button press is recognized -- you may need to hold it for a second first.

The current beacon name and URL will be displayed.
QR Code

Press the A button to go to QR code mode. This is great for any users without a BLE beacon app who want to use their camera to scan the URL instead.

Here is the auto-generated QR code. All made in CircuitPython!

You can press the B button to switch to the other URLs which will change the beacon advertisements almost instantly.

NOTE: Depending on the beacon app you’re running, you may need to refresh or delete the beacon cache in order to see the changed advertisement URL show up.