End-to-End IoT

IFTTT + MQTT; Internet-WiFi-BLE

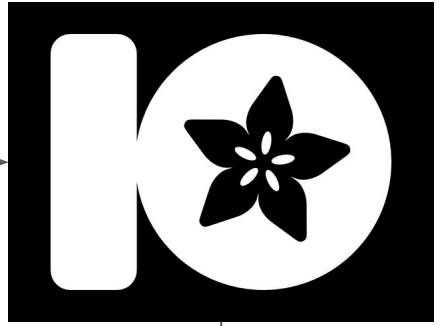
Steven Knudsen, PhD, PEng
Mentor, The Pod
knud@ualberta.ca

the POD
invent & innovate

pod—innovation.ca

IFTTT

ifthisthenthat



Objective: Control a BLE-enabled device from your phone

Overview of Main Steps

1. Set up MQTT (Adafruit IO)
2. Set up IFTTT
3. Confirm RPi can poll MQTT server, get command from IFTTT
4. Set up BLE connection with Raspberry Pi (RPi)
    a. Control a PWM signal
5. Bridge MQTTT polling and BLE commands
6. Done!

Information and Caveats

Sources

- github.com/knud
  - HackED2020Workshop
  - RB_Nano_v1 : WIP for Nordic nRF51822

Bluetooth Low Energy

- Many options for platforms that are easy to work with
  - Adafruit and mbed are two good ones
  - Chose Nordic SDK only because using it a lot these days

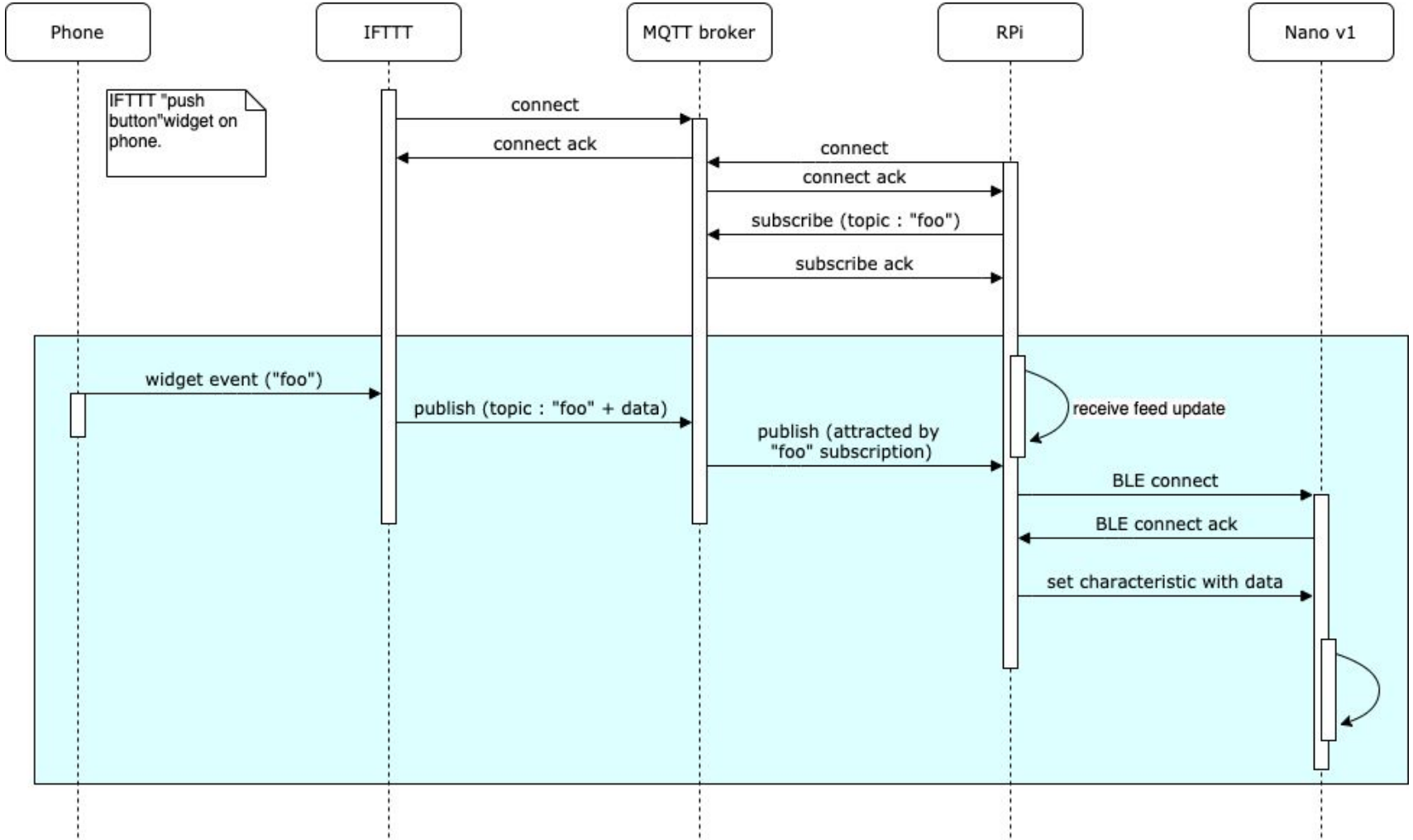
Information and Cavets

- github.com/IanHarvey/bluepy
  - Has issues with the nRF51822 when hosted on RPi
  - Solid on laptop and with nRF52xxx

MQTT — mqtt.org

An extremely lightweight publish/subscribe messaging transport to support machine-to-machine (M2M), Internet of Things (IoT) interactions.

- Small; Windows executable is 1.4 MB


Phone
IFTTT
MQTT broker
RPi
Nano v1
IFTTT "push button"widget on phone.
connect
connect ack
connect
connect ack
subscribe (topic : "foo")
subscribe ack
widget event ("foo")
publish (topic : "foo" + data)
publish (attracted by "foo" subscription)
receive feed update
BLE connect
BLE connect ack
set characteristic with data

Set up MQTT

- Sign up with io.adafruit.com (aka AIO)
- Create new dashboard and open
- Create a new feed and block
- Set up / customize the block
- Record AIO key
  - Needed for IFTTT and RPi client

Profile Feeds Dashboards Triggers Services AIO Key

eeknud / Dashboards

Actions

Name	Key	Created At
HackED2020	hacked2020	January 13, 2020
Welcome Dashboard	welcome-dashboard	October 31, 2017

Loaded in 0.13 seconds.

Help
Get Help
Quick Guides
API Documentation
FAQ
Terms of Service
Privacy Policy
Send Feedback

Explore
Learn
IO Plus
News

"Don't fight forces, use them" - R. Buckminster Fuller

Profile Feeds

AIO Key

eeknud / Dashboards / H

HackED2020

Motor

ON

Help

Get Help
Quick Guides
API Documentation
FAQ
Terms of Service
Privacy Policy

Send Feedback

ckminster Fuller

Create a new block

Click on the block you would like to add to your dashboard. You can always come back and switch the block type later if you change your mind.

ON

RESET

HELLO WORLD!

#00ACEC

Hello world! This is a different type of text block that works for larger quantities of data

Streaming Data
Text Blocks
Welcome Dashboard

Profile Feeds

AIO Key

eeknud / Dashboards / H

HackED2020

Motor

ON

Choose feed

Momentary Button: A momentary button works similarly to a hardware push button.

If you have lot of feeds, you may want to use the search field. You can also create a feed quickly below.

MomButton Create

Group / Feed	Last value	Recorded
☑ MomButton		8 minutes
☐ Motor Control	ON	about 2 ho...
☐ Welcome Feed		about 1 year

Previous step Next step

Help

Get Help
Quick Guides
API Documentation
FAQ
Terms of Service
Privacy Policy

Send Feedback

Explore

Learn
IO Plus
News

"Don't fight forces, use them" - R. Buckminster Fuller

Profile Feeds

AIO Key

eeknud / Dashboards / H

HackED2020

Motor

ON

Help

Get Help
Quick Guides
API Documentation
FAQ
Terms of Service
Privacy Policy

Send Feedback

ckminster Fuller

Block settings

In this final step, you can give your block a title and see a preview of how it will look. Customize the look and feel of your block with the remaining settings. When you are ready, click the "Create Block" button to send it to your dashboard.

Block Title (optional)

My Block

Button Text

On

Press Value

1

Release Value

0

Leave this field blank to not send anything when the button is released.

Color

#1b9af7

Block Preview

My Block

On

Momentary Button A momentary button works similarly to a hardware push button.

Test Value

45

Published Value

0 bytes

Previous step

Create block

Profile Feeds Dashbo

AIO Key

eeknud / Dashboards / HackED

HackED2020

Motor

ON

My Block

On

Help

Get Help
Quick Guides
API Documentation
FAQ
Terms of Service
Privacy Policy

Learn
IO Plus
News

R. Buckminster Fuller

YOUR AIO KEY

Your Adafruit IO key should be kept in a safe place and treated with the same care as your Adafruit username and password. People who have access to your AIO key can view all of your data, create new feeds for your account, and manipulate your active feeds.

If you need to regenerate a new AIO key, all of your existing programs and scripts will need to be manually changed to the new key.

Username: eeknud

Active Key: b25c2c664f0545a799b273029bfee3ce

REGENERATE AIO KEY

Hide Code Samples

Arduino

```
#define IO_USERNAME  "eeknud"
#define IO_KEY       "b25c2c664f0545a799b273029bfee3ce"
```

Linux Shell

```
export IO_USERNAME="eeknud"
export IO_KEY="b25c2c664f0545a799b273029bfee3ce"
```

Scripting

```
ADAFRUIT_IO_USERNAME = "eeknud"
ADAFRUIT_IO_KEY = "b25c2c664f0545a799b273029bfee3ce"
```

Set up IFTTT

- Create a new widget
- Select an action (“This”)
  - Button
- Select a service (“That”)
  - Choose Adafruit service

Create your own

If This Then That

Build your own service on the IFTTT Platform

Choose a service

Step 1 of 6

button

Choose trigger

Step 2 of 6

Button press

This trigger fires every time you press the button.

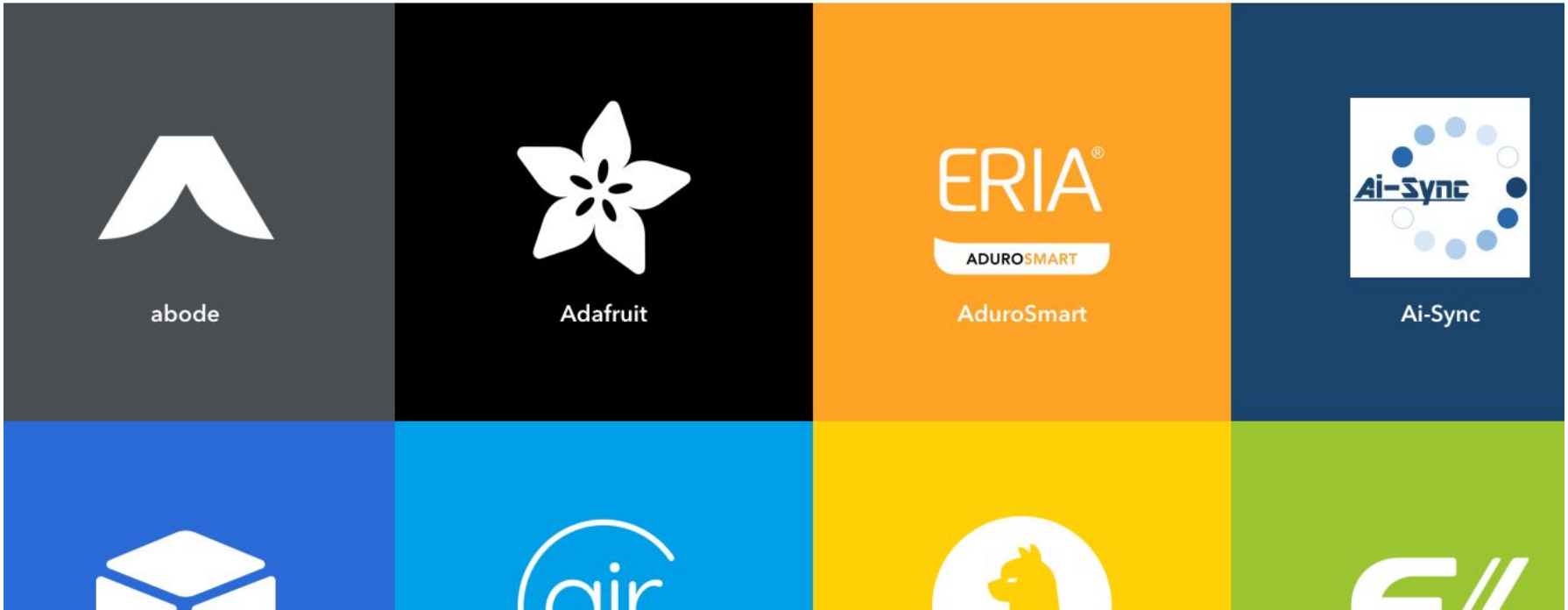
Don't see what you're looking for? **Suggest a new trigger**

If Then That

Choose action service

Step 3 of 6

Search services



Choose action

Step 4 of 6

Send data to Adafruit IO

This Action will send data to a feed in your Adafruit IO account.

Don't see what you're looking for? **Suggest a new action**

Complete action fields

Step 5 of 6

Feed name

MomButton

The name of the feed to save data to.

Data to save

ON

The data to be saved to your feed.

Add ingredient

Create action

Review and finish

Step 6 of 6

Send data to MomButton feed

27/140

by eeknud

Receive notifications when this Applet runs

Finish

So far...

- At this point the IFTTT widget should activate and be available on your phone
  - BTW, you need an IFTTT account and their app
- Activating the widget sends the data you specified to the Adafruit feed
- The Adafruit MQTT broker makes note of it for any subscribers

But we need a subscriber...

Raspberry Pi

- Need to be sure that `bluepy`, `requests`, and `adafruit-io` are installed.
  - `sudo pip3 install …`
- Need `MQTTRelay.py` from `github.com/knud/HackED2020Workshop`

```python
#!/usr/bin/env python
# -*- coding: utf-8 -*-

from __future__ import print_function, unicode_literals
from pprint import pprint
from bluepy import btle
from bluepy.btle import Scanner, DefaultDelegate, Peripheral, UUID, BTLEException
from time import sleep
from Adafruit_IO import Client, RequestError
import re
import os
from os import system

ADAFRUIT_IO_KEY = 'b25c2c664f0545a799b273029bfee3ce'
ADAFRUIT_IO_USERNAME = 'eeknud'

```

```python
#-----------------------------------------------------------------------
# BLE stuff
#-----------------------------------------------------------------------

# create a delegate class to receive the BLE broadcast packets
class ScanDelegate(DefaultDelegate):
    def __init__(self):
        DefaultDelegate.__init__(self)

    # when this python script discovers a BLE broadcast packet, print a message with the device's MAC address
    def handleDiscovery(self, dev, isNewDev, isNewData):
        if isNewDev:
            print ( "Discovered device %s" % dev.addr )
        elif isNewData:
            print ( "Received new data from %s" % dev.addr )


```

```
76  def commandToPeripheral(blePeripheral, characteristic, commandData):
77      try:
78          blePeripheral.writeCharacteristic(characteristic.getHandle(), commandData, withResponse=False)
79      except BTLEException as e:
80          print("exception : "+str(e))
81
82  def updateNano(peripherial, newState):
83      numAttempts = 5
84      attempt = 1
85      commandDelivered = False;
86      while attempt <= numAttempts and not commandDelivered:
87          try:
88              peripherial.connect(rb_nanov1Device.addr, btle.ADDR_TYPE_RANDOM)
89              if peripherial.getState() == "conn":
90  #                print('connected')
91                  if newState == "ON":
92                      commandToPeripheral(peripherial, commandCharacteristic, commandStringON)
93                  else:
94                      commandToPeripheral(peripherial, commandCharacteristic, commandStringOFF)
95                  commandDelivered = True
96              else:
97                  print('failed to connect')
98          except BTLEException as e:
99  #            print("exception : "+str(e))
100             print("Error: Unable to connect to Nano")
101             sleep(0.05)
```

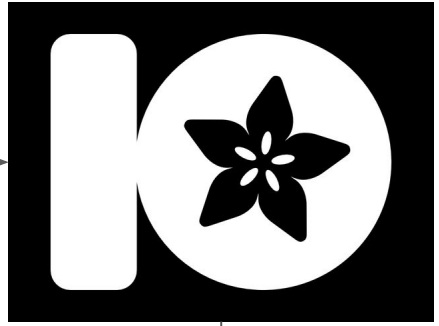
```
print('-----------------------------')
print('Scanning 5 s for Nano v1')
print('-----------------------------')

# create a scanner object that sends BLE broadcast packets to the ScanDelegate
scanner = Scanner().withDelegate(ScanDelegate())

# create a list of unique devices that the scanner discovered during a 10-second scan
devices = scanner.scan(5.0)

# for each device  in the list of devices

for dev in devices:
#    print ( "Device %s (%s), RSSI=%d dB" % (dev.addr, dev.addrType, dev.rssi) )

    # For each of the device's advertising data items, print a description of the data type and value of the data itself
    # getScanData returns a list of tupples: adtype, desc, value
    # where AD Type means "advertising data type," as defined by Bluetooth convention:
    # https://www.bluetooth.com/specifications/assigned-numbers/generic-access-profile
    # desc is a human-readable description of the data type and value is the data itself\
    found = False
    for (adtype, desc, value) in dev.getScanData():
        if value == "HackED_PWM":
            found = True
            print ( "  %s = %s" % (desc, value) )
    if found == True:
        rb_nanov1Device = dev
        break;
```

```
153 if found == True:
154     # print  the device's MAC address, its address type,
155     # and Received Signal Strength Indication that shows how strong the signal was when the script received the broadcast.
156     print('----------------------------------------------------------------------------')
157     print("Found RedBear Nano v1 with address %s (%s), RSSI=%d dB" % (rb_nanov1Device.addr, rb_nanov1Device.addrType, rb_nanov1Device.rssi)
158     print('----------------------------------------------------------------------------')
159
160     # connect to the reader
161     if rb_nanov1Device.connectable:
162         try:
163             rb_nanov1 = Peripheral(rb_nanov1Device.addr, btle.ADDR_TYPE_RANDOM)
164 #            rb_nanov1.setDelegate(ReceptionDelegate())
165             services = rb_nanov1.getServices()
166             for s in services:
167                 if s.uuid == "00001523-1212-efde-1523-785feabcd123":
168                     print(s.uuid)
169                     characteristics = s.getCharacteristics()
170                     for c in characteristics:
171                         print("%s: %s" % (c.uuid, c.propertiesToString()))
172                         if c.uuid == "00001524-1212-efde-1523-785feabcd123":
173                             dataNotifyCharacteristic = c
174 #                             enable_notify(rb_nanov1, dataNotifyCharacteristic)
175                         if c.uuid == "00001525-1212-efde-1523-785feabcd123":
176                             commandCharacteristic = c
177
178         except BTLEException as e:
179             print("exception : "+str(e))
180             print("Error: Unable to connect to Nano and find its services")
```

```
189
190         # define the commands to control the remote device
191         commandStringOFF = b"\x00"
192         commandStringON = b"\xFF"
193
194         aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)
195
196         try:
197             motorControlFeed = aio.feeds('motor-control')
198         except RequestError:
199             print("feed error")
200
201         # get the current motor state
202         motorControlState = aio.receive(motorControlFeed.key)
203         motorState = motorControlState.value
204         print("Initial Motor State : %s" % (motorState) )
205         sleep(0.1)
206         updateNano(rb_nanov1, motorState)
207
208         # Look for any changes initiated by IFTTT and relayed by Adafruit IO
209         while True:
210             motorControlState = aio.receive(motorControlFeed.key)
211 #             print("Motor State : %s" % (motorControlState.value) )
212             if motorControlState.value != motorState:
213                 print("Motor State changed to: %s" % (motorControlState.value) )
214                 motorState = motorControlState.value
215                 updateNano(rb_nanov1, motorState)
216
217             sleep(3)
218     else:
219         print ("not connectable")
```

IFTTT

ifthisthenthat



Objective: Control a BLE-enabled device from your phone