Make your own Button!

Diego Echeverri - @diegoeche - diego@geeny.io

Who am I?

1. Backend Engineer
2. Maker Wannabe

This Presentation

1. What is IoT?
2. What Geeny does?

The Workshop!

1. Connect to the Rpi Zero
2. GPIO Programming
3. Use Twilio SMS service
4. Make it Geeny Enabled

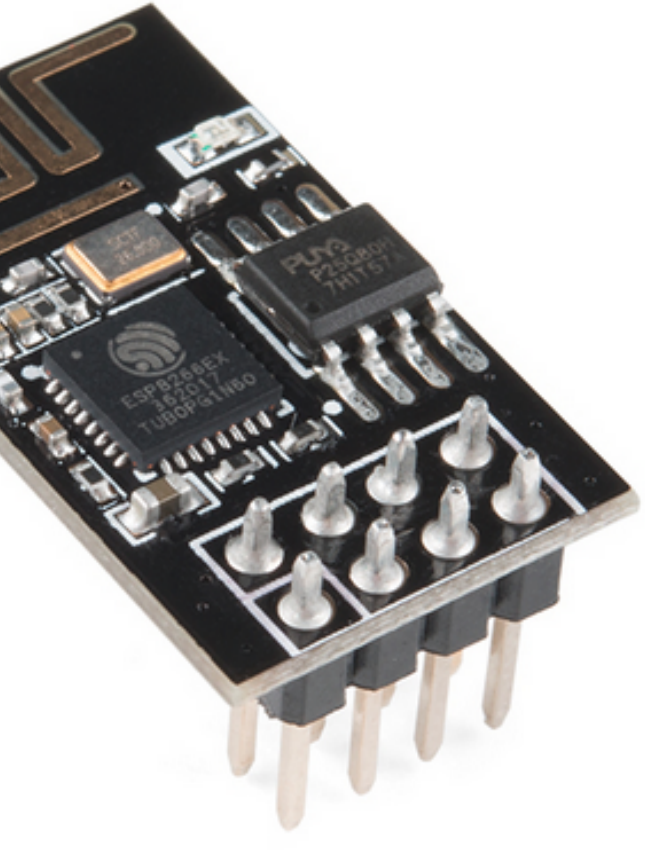
What is IoT?

Cheap Silicon + Low Power + Internet

= IoT

Raspberry PI Zero W

- Wifi
- 512MB
- GPIO
- $10



WiFi Module - ESP8266

WRL-13678 ROHS

28

$6.95

Volume sales pricing

Shipping outside of the US
Click here for info

- 1 +

ADD TO CART

Quantity discounts available

DESCRIPTION FEATURES DOCUMENTS

The ESP8266 WiFi Module is a self contained SOC with integrated TCP/IP protoc
give any microcontroller access to your WiFi network. The ESP8266 is capable of
an application or offloading all Wi-Fi networking functions from another applicatio
Each ESP8266 module comes pre-programmed with an AT command set firmwa
can simply hook this up to your Arduino device and get about as much WiFi-abilit
Shield offers (and that's just out of the box)! The ESP8266 module is an extremel
board with a huge, and ever growing, community.

Mouse over to zoom in

Orange Pi Zero H2+ Quad Core Open-source 256MB developme beyond Raspberry Pi

4.9 (1360 votes) | 1365 orders

Price: **US $6.99** / piece

Shipping: **US $4.12 to Germany via China Post Registered Air M**

Estimated Delivery Time: 26-49 days

Quantity: – 1 + piece (1575 pieces available)

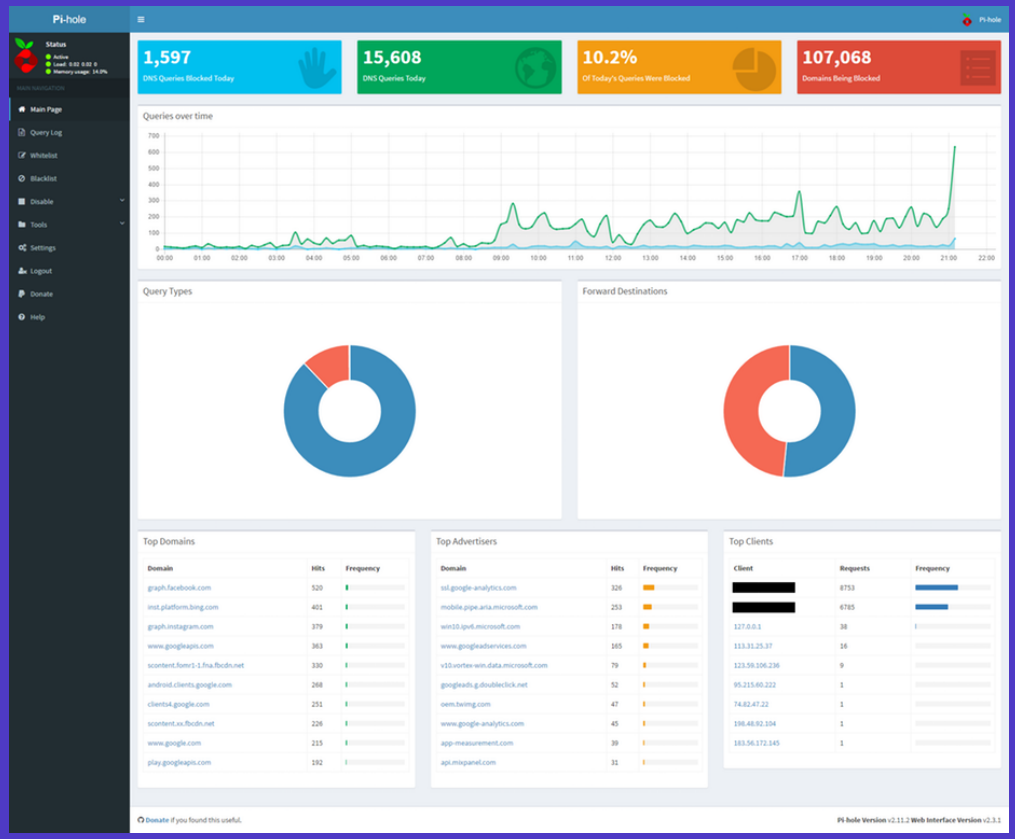
Total Price: **US $11.11**

Buy Now | Add to Cart

Add to Wish List (6947 Adds)

Return Policy: Returns accepted if product not as described, buyer pays retu or keep the product & agree refund with seller. View details

Seller Guarantees: On-time Delivery **60 days**


Pi-hole
Status
Active
Load: 0.02 0.02 0
Memory usage: 14.0%
MAIN NAVIGATION
Main Page
Query Log
Whitelist
Blacklist
Disable
Tools
Settings
Logout
Donate
Help
Pi-hole
1,597
DNS Queries Blocked Today
15,608
DNS Queries Today
10.2%
Of Today's Queries Were Blocked
107,068
Domains Being Blocked
Queries over time
Query Types
Forward Destinations
Top Domains
Domain Hits Frequency
graph.facebook.com 520
inst.platform.bing.com 401
graph.instagram.com 379
www.googleapis.com 363
scontent.fomr1-1.fna.fbcdn.net 330
android.clients.google.com 268
clients4.google.com 251
scontent.xx.fbcdn.net 226
www.google.com 215
play.googleapis.com 192
Top Advertisers
Domain Hits Frequency
ssl.google-analytics.com 326
mobile.pipe.aria.microsoft.com 253
win10.ipv6.microsoft.com 178
www.googleadservices.com 165
v10.vortex-win.data.microsoft.com 79
googleads.g.doubleclick.net 52
oem.bwimg.com 47
www.google-analytics.com 45
app-measurement.com 39
api.mixpanel.com 31
Top Clients
Client Requests Frequency
[redacted] 8753
[redacted] 6785
127.0.0.1 38
113.31.25.37 16
123.59.106.236 9
95.215.60.222 1
74.82.47.22 1
198.48.92.104 1
183.56.172.145 1
Donate if you found this useful.
Pi-hole Version v2.11.2 Web Interface Version v2.3.1

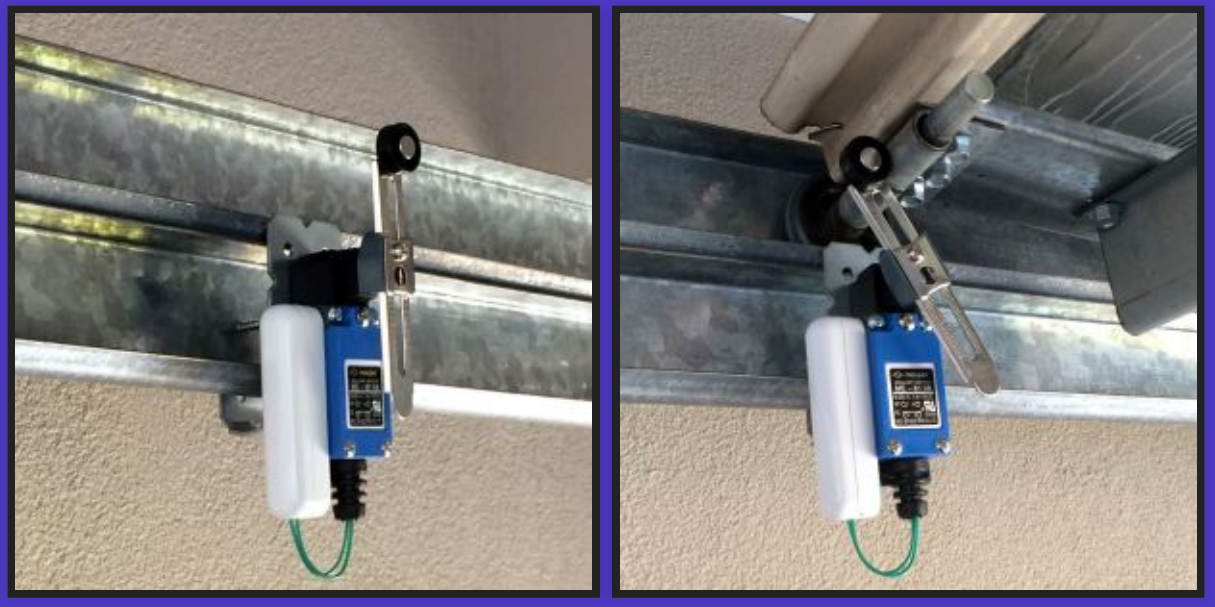
Dumb things (sensors/lights/relays)

can now have Internet Connectivity

Why a button?

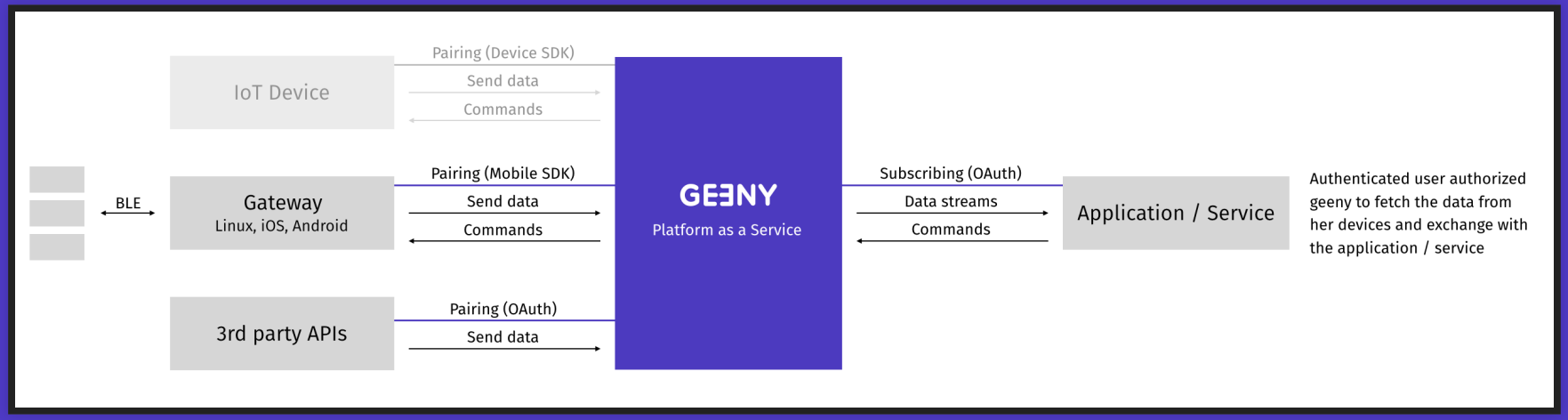

durex
EPSON
EXCEED YOUR VISION
WALKERS
RedBull

Simple Mechanical Sensors are Buttons




DANGER!

Why Geeny?



Hardware | Software

Workshop Time!

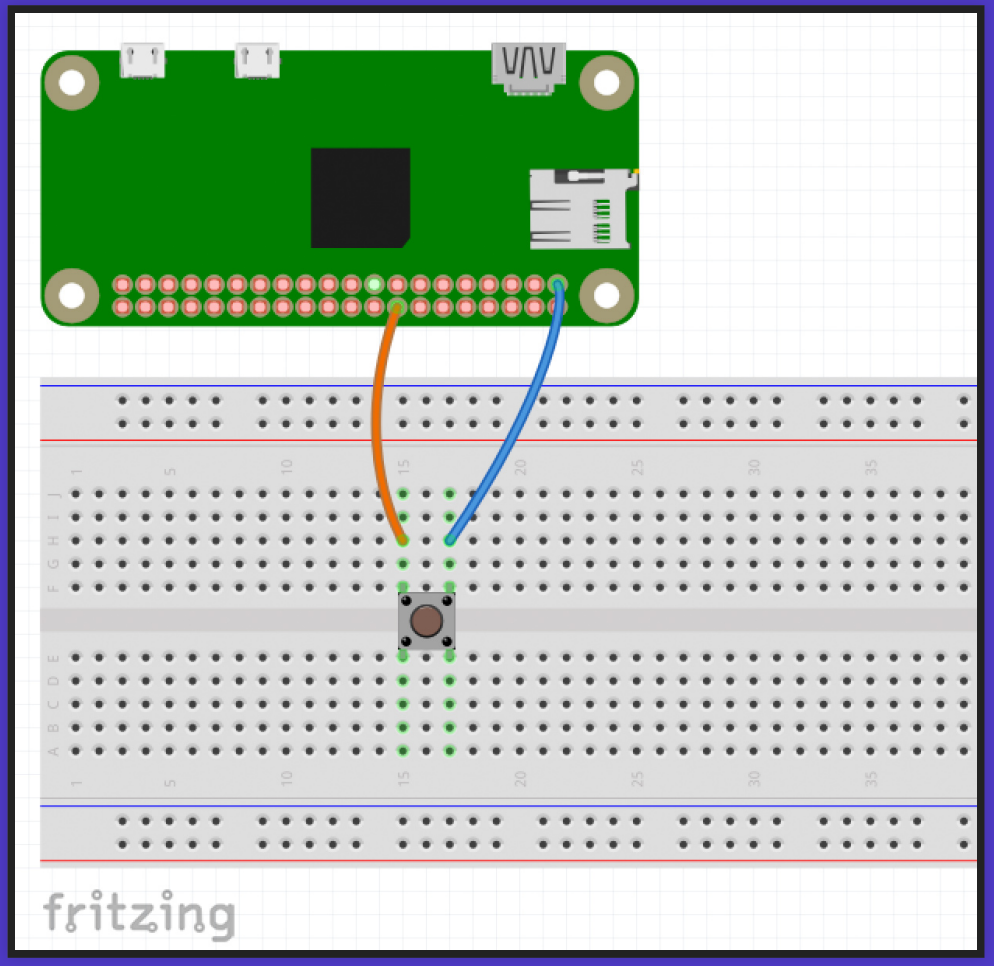
Materials at:

https://github.com/geeny/fu-workshop

Step 0: Connect to your Rpi

```
ssh pi@geeny1.local
curl -H "Content-Type: application/json" -X POST -d
'{"email":"","password":""}'
          http://localhost:9000/api/v1/login
>> success
```

Step 1: Plug Everything



Step 2: GPIO Intro

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

try:
  while True:
      if GPIO.input(23):
          print("Button 1 pressed")

except KeyboardInterrupt:
  pass
finally:
  print "Exit: Cleanup"
  GPIO.cleanup()
```

Gotcha, Too fast!

```
Button 1 pressed
Button 1 pressed
Button 1 pressed
Button 1 pressed
Button 1 pressed
Button 1 pressed
Button 1 pressed
...
```

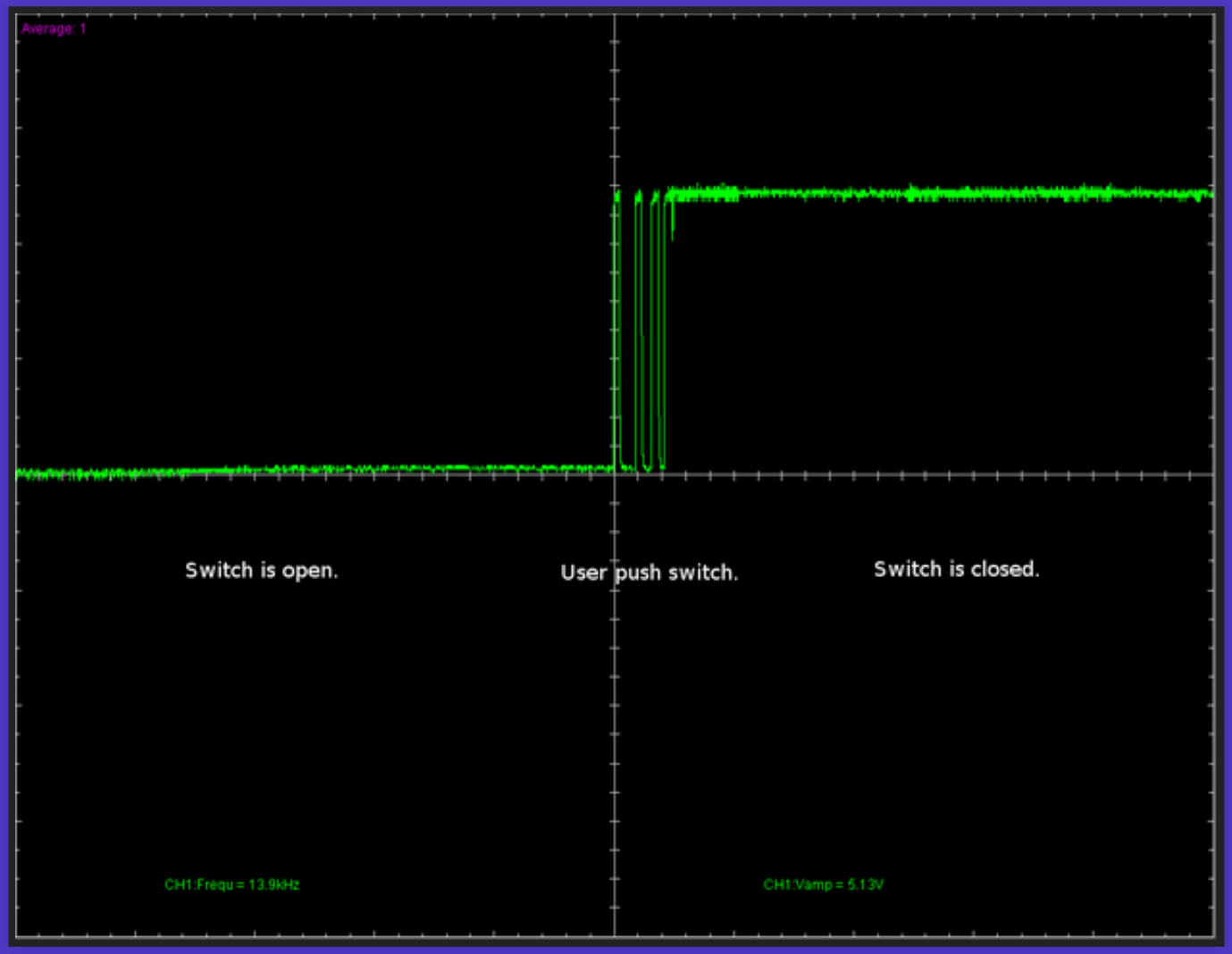
Step 3: Fixing Debouncing

```
import RPi.GPIO as GPIO

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

try:
  while True:
    if GPIO.input(23):
      print("Button pressed")
      while GPIO.input(23):
        pass
      print("Button released")

except KeyboardInterrupt:
  pass
finally:
  print "Exit: Cleanup"
  GPIO.cleanup()
```

Active Blocking

```
Button pressed
Button released
Button pressed
Button released
Button pressed
...
```

Bouncing (2)


Average: 1
Switch is open.
User push switch.
Switch is closed.
CH1:Frequ = 13.9kHz
CH1:Vamp = 5.13V

It Works!

```
import RPi.GPIO as GPIO
import time

GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

def pause():
    time.sleep(0.1)

try:
  while True:
    if GPIO.input(23):
      print("Button pressed")
      while GPIO.input(23):
        pass
      print("Button released")
    pause()

except KeyboardInterrupt:
  pass
finally:
  print "Exit: Cleanup"
  GPIO.cleanup()
```

Making an Emergency Button (1)

```
import RPi.GPIO as GPIO
import time
import os
from twilio.rest import Client

account_sid = os.environ["TWILIO_SID"]
auth_token  = os.environ["TWILIO_SECRET"]
client = Client(account_sid, auth_token)

def send_alert():
  message = client.messages.create(
    to="+4917627295457",
    from_="+18312221512",
    body="Diego! RED ALERT!")
  print("Sent: ", message.sid)
```

Making it an Emergency Button (2)

```
GPIO.setmode(GPIO.BCM)
GPIO.setup(23, GPIO.IN, pull_up_down=GPIO.PUD_DOWN)

try:
  while True:
    if GPIO.input(23):
      while GPIO.input(23):
        pass
      send_alert()
    time.sleep(0.1)

except KeyboardInterrupt:
  pass
finally:
  print "Exit: Cleanup"
  GPIO.cleanup()
```

Making it an Emergency Button (3)

```
TWILIO_SID=ACe9554c8d86227045b93590fe140a4d8c \
TWILIO_SECRET=139077208f75e79d76ddc8cd644358ae python button.py
```

Connect the Button to Geeny

Register Device

```
curl -X POST \
    -H 'Content-Type: application/json' \
    -H 'Accept: application/json' \
    -d '{
        "name": "<name-of-your-thing>",
        "serial_number": "123",
        "thing_type": "877827cc-0c78-4e55-80fe-2941479c681a"
        }' \
    'http://localhost:9000/api/v1/things' > thing.info
            </name-of-your-thing>
```

Login

Code

```
def login():
    url = 'http://localhost:9000/api/v1/login'
    payload = json.dumps({'email': EMAIL, 'password': PASSWORD})
    headers = {'Content-Type': 'application/json'}
    response = requests.post(url, data=payload, headers=headers)
    return response.text == "success"
```

Publish a Message

Code

```
def publish():
    print("publishing message...")
    url = 'http://localhost:9000/api/v1/messages/' + SERIAL_NUMBER
    payload = json.dumps({'msgs': []})
    headers = {'Content-Type': 'application/json'}
    response = requests.post(url, data=payload, headers=headers)
    print(response.text)
```

Summary

1. Getting Started with IoT is very Easy
2. IoT development is better using a platform like Geeny

The End

Questions and Feedback is always welcome: diego@geeny.io