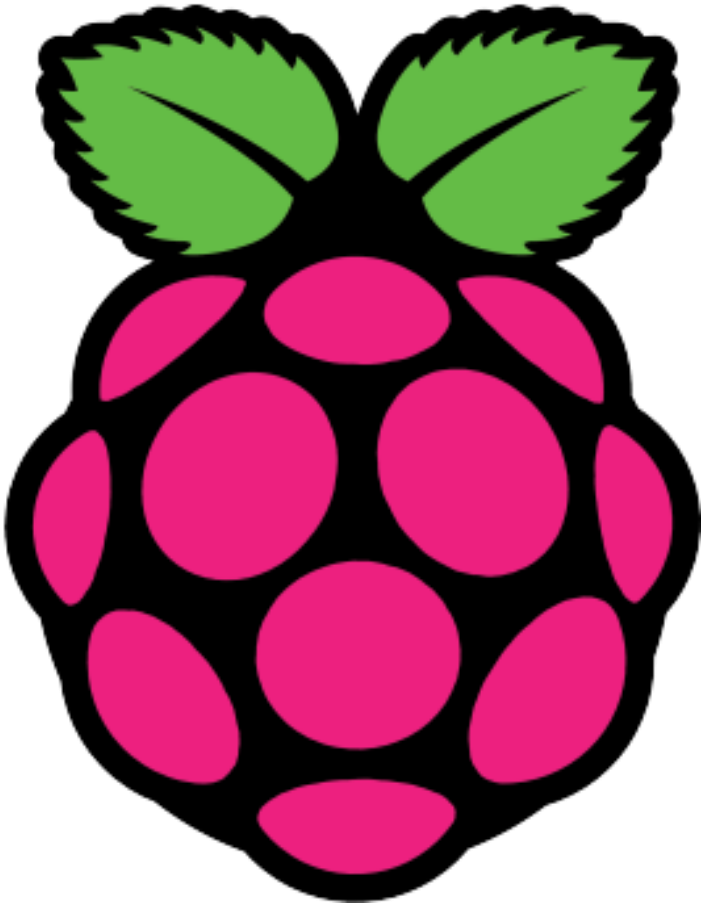
“Raspberry pi course”

ENG: AHMED MUBARAK

01032414034



SESSION NO.“9”

- BLYNK APP
- SONOFF
- ALEXA ECHO DOT

ENG.AHMED MUBARAK

01032414034

BLYNK APP

SETUP BLYNK LIBRARY TO RASPBERRY PI

1

Download Blynk app

Blynk app for iOS and Android is the easiest way to build your own mobile app that work with the hardware of your choice.

No iOS or Android coding required.

2

Install Blynk Library

Blynk Library is an extension that runs on top of your hardware application. It handles all the connection routines and data exchange between your hardware, Blynk Cloud, and your app project.

- C++, C#
- JavaScript
- Python, MicroPython
- HTTP RESTful API
- Node.js
- Lua
- OpenWrt
- MBED
- Node-RED

and others

3

Connect hardware

To get your hardware online and connect it to Blynk Cloud, you would need a device Authentication Token.

Once you download the app you will be able to generate Auth Token for every device.

4

Enjoy Blynking

After your hardware is connected spend some time learning Blynk basics. It will help you to easily build new projects or integrate Blynk into your existing project.

We prepared a lot of examples and tutorials to get you started

- **Download Blynk App :**
  Android: https://play.google.com/store/apps/details?id=cc.blynk&hl=en_US
  Ios: https://itunes.apple.com/us/app/blynk-iot-for-arduino-esp32/id808760481?mt=8

1. Create New Account in Blynk app

Account is needed to save your projects and provide access from any smartphone you have.

☝Use a valid email address as it will be later used often

2. Create New Project

• Create New Project and choose the hardware you use.
• If you can't find the hardware you use – select Generic Board
• Choose what type of connectivity you use?
• Choose Dark or Light UI interface

3. Get Auth Token

Check your inbox to see if you get an email from Blynk with the Auth Token. You will need it later

- **Install Blynk Library on your raspberry and run it**
- **sudo apt-get install git-core**
- **git clone https://github.com/blynkkk/blynk-library.git**
- **cd blynk-library/linux**
- **ls**
- **make clean all target=raspberry**
- **./build.sh raspberry**
- **sudo ./blynk --token=…….. (REPLACE …….. BY YOUR AUTH TOKEN)**

ADFRUIT IO

```
sudo pip3 install adafruit-blinka

sudo pip3 install adafruit-io
```

FIRS PROJECT TO CONTROL RELAY USING ADFRUIT IO

```
nano adfruit_C.py
```

```
# Import standard python modules.
import sys

# Import blinka python modules.
import board
import digitalio

# This example uses the MQTTClient instead of the REST client
from Adafruit_IO import MQTTClient

# Set to your Adafruit IO key.
# Remember, your key is a secret,
# so make sure not to publish it when you publish this code!
ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'
```

```
# Set to your Adafruit IO username.
# (go to https://accounts.adafruit.com to find your username)
ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'

# Set to the ID of the feed to subscribe to for updates.
FEED_ID = 'digital'

relay = digitalio.DigitalInOut(board.D5)
relay.direction = digitalio.Direction.OUTPUT

# Define callback functions which will be called when certain events happen.
def connected(client):
    """Connected function will be called when the client is connected to
    Adafruit IO.This is a good place to subscribe to feed changes.  The client
    parameter passed to this function is the Adafruit IO MQTT client so you
    can make calls against it easily.
    """
    # Subscribe to changes on a feed named Counter.
    print('Subscribing to Feed {0}'.format(FEED_ID))
    client.subscribe(FEED_ID)
    print('Waiting for feed data…')

def disconnected(client):
    """Disconnected function will be called when the client disconnects."""
    sys.exit(1)

def message(client, feed_id, payload):
    """Message function will be called when a subscribed feed has a new value.
    The feed_id parameter identifies the feed, and the payload parameter has
    the new value.
    """
    print('Feed {0} received new value: {1}'.format(feed_id, payload))

    if payload == "OFF":
        print("Turn off relay!")
        relay.value = False
    elif payload == "ON":
        print("Turn on relay!")
        relay.value = True

# Create an MQTT client instance.
client = MQTTClient(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)

# Setup the callback functions defined above.
client.on_connect = connected
client.on_disconnect = disconnected
client.on_message = message

# Connect to the Adafruit IO server.
```

```
client.connect()

# The first option is to run a thread in the background so you can continue
# doing things in your program.
client.loop_blocking()
```

```
python3 adfruit_C.py
```

SECOND PROGRAM TO READ SENSOR VALUE ON ADFRUIT IO

```
nano adfruit_R.py
```

```
# import standard python modules.
import time

# import adafruit dht library.
import Adafruit_DHT

# import Adafruit IO REST client.
from Adafruit_IO import Client, Feed

# Delay in-between sensor readings, in seconds.
DHT_READ_TIMEOUT = 5

# Pin connected to DHT22 data pin
DHT_DATA_PIN = 26

# Set to your Adafruit IO key.
# Remember, your key is a secret,
# so make sure not to publish it when you publish this code!
ADAFRUIT_IO_KEY = 'YOUR_AIO_KEY'

# Set to your Adafruit IO username.
# (go to https://accounts.adafruit.com to find your username).
ADAFRUIT_IO_USERNAME = 'YOUR_AIO_USERNAME'

# Create an instance of the REST client.
aio = Client(ADAFRUIT_IO_USERNAME, ADAFRUIT_IO_KEY)

# Set up Adafruit IO Feeds.
temperature_feed = aio.feeds('temperature')
humidity_feed = aio.feeds('humidity')

# Set up DHT22 Sensor.
dht11_sensor = Adafruit_DHT.DHT11
```

```python
while True:
    humidity, temperature = Adafruit_DHT.read_retry(dht11_sensor, DHT_DATA_PIN)
    if humidity is not None and temperature is not None:
        print('Temp={0:0.1f}*C Humidity={1:0.1f}%'.format(temperature, humidity))
        # Send humidity and temperature feeds to Adafruit IO
        temperature = '%.2f'%(temperature)
        humidity = '%.2f'%(humidity)
        aio.send(temperature_feed.key, str(temperature))
        aio.send(humidity_feed.key, str(humidity))
    else:
        print('Failed to get DHT22 Reading, trying again in ', DHT_READ_TIMEOUT, 'seconds')
    # Timeout to avoid flooding Adafruit IO
    time.sleep(DHT_READ_TIMEOUT)
```

```
python3 adfruit_R.py
```

ALEXA ECHO DOT RUN WITH SONOFF AND ADFRUIT USING RASSPBERY

THIS IS ONLY PRACTICAL

With my best wishes:

ENG : AHMED MUBARAK
