Automonous Systems

INTERNET OF THINGS

Fábio Silva
fabiosilva@di.uminho.pt

Index

Platforms

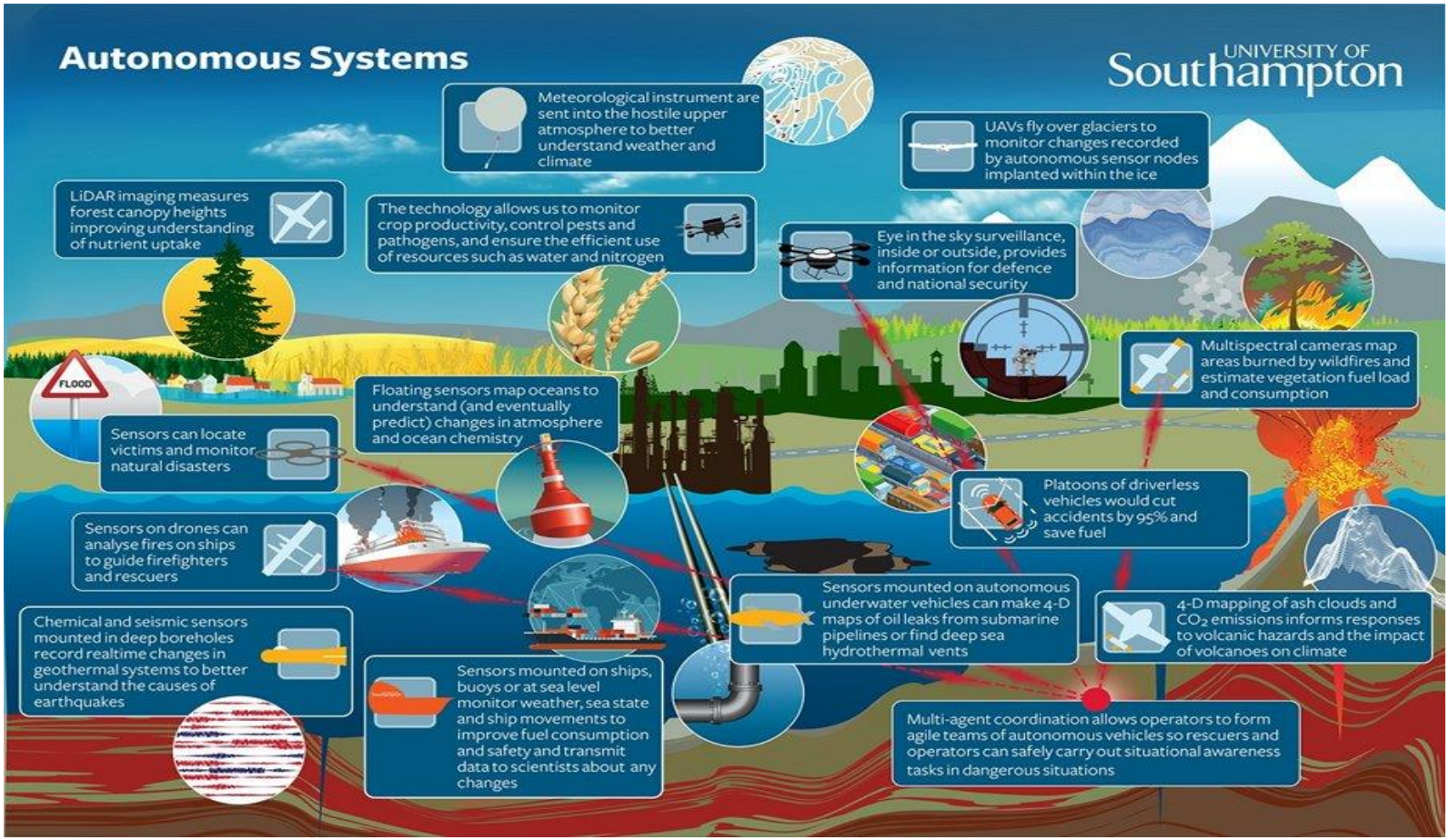
Actuator plaform perform actions on the behalf of the user

They decide each action based on triggers, conditions that define a context, observed through services, sensors and inputs

Traditional platforms are reative in nature and automize tasks.

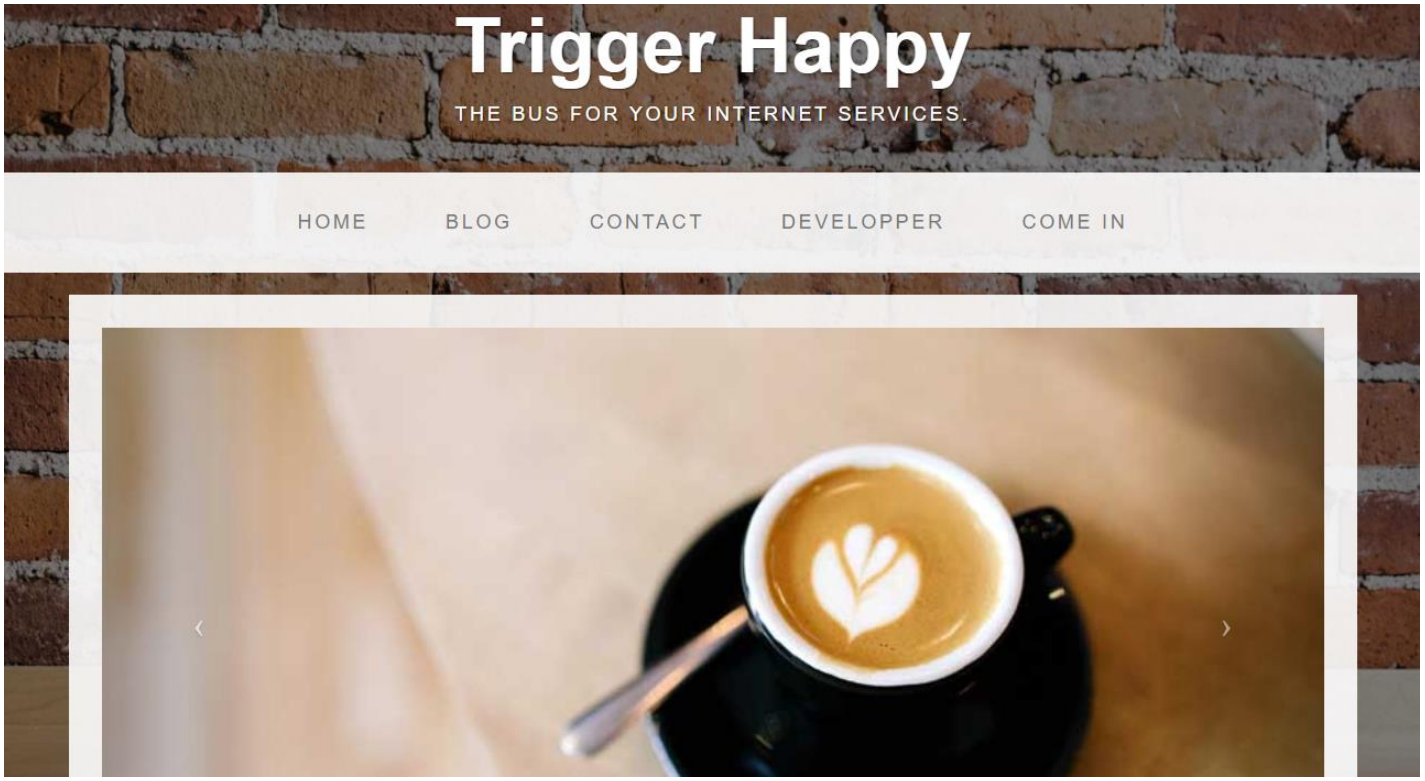
Better implementations should make use of deliberative and hybrid processes

Platforms



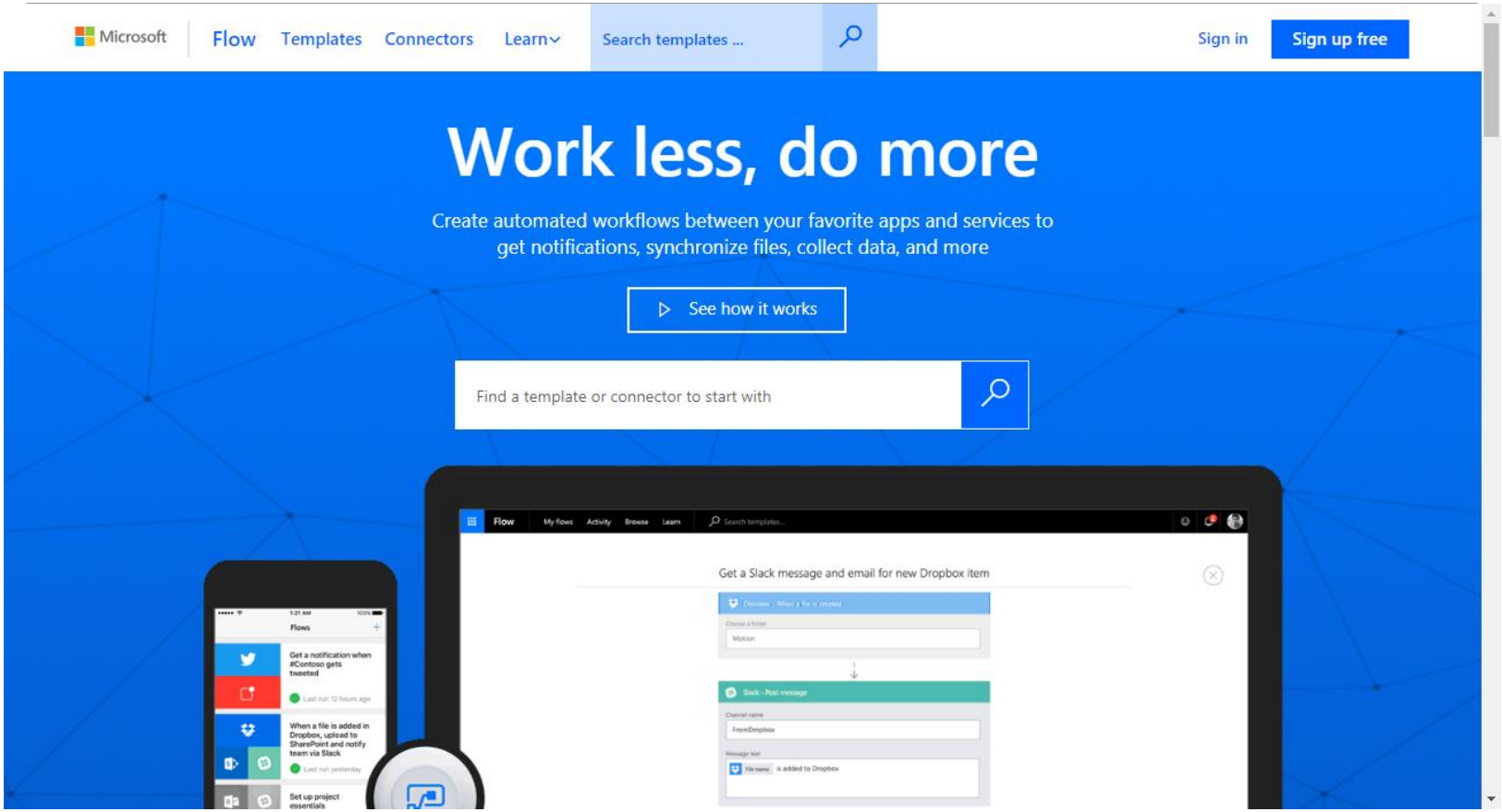
Source: https://www.southampton.ac.uk/autonomous-systems/index.page

Platforms



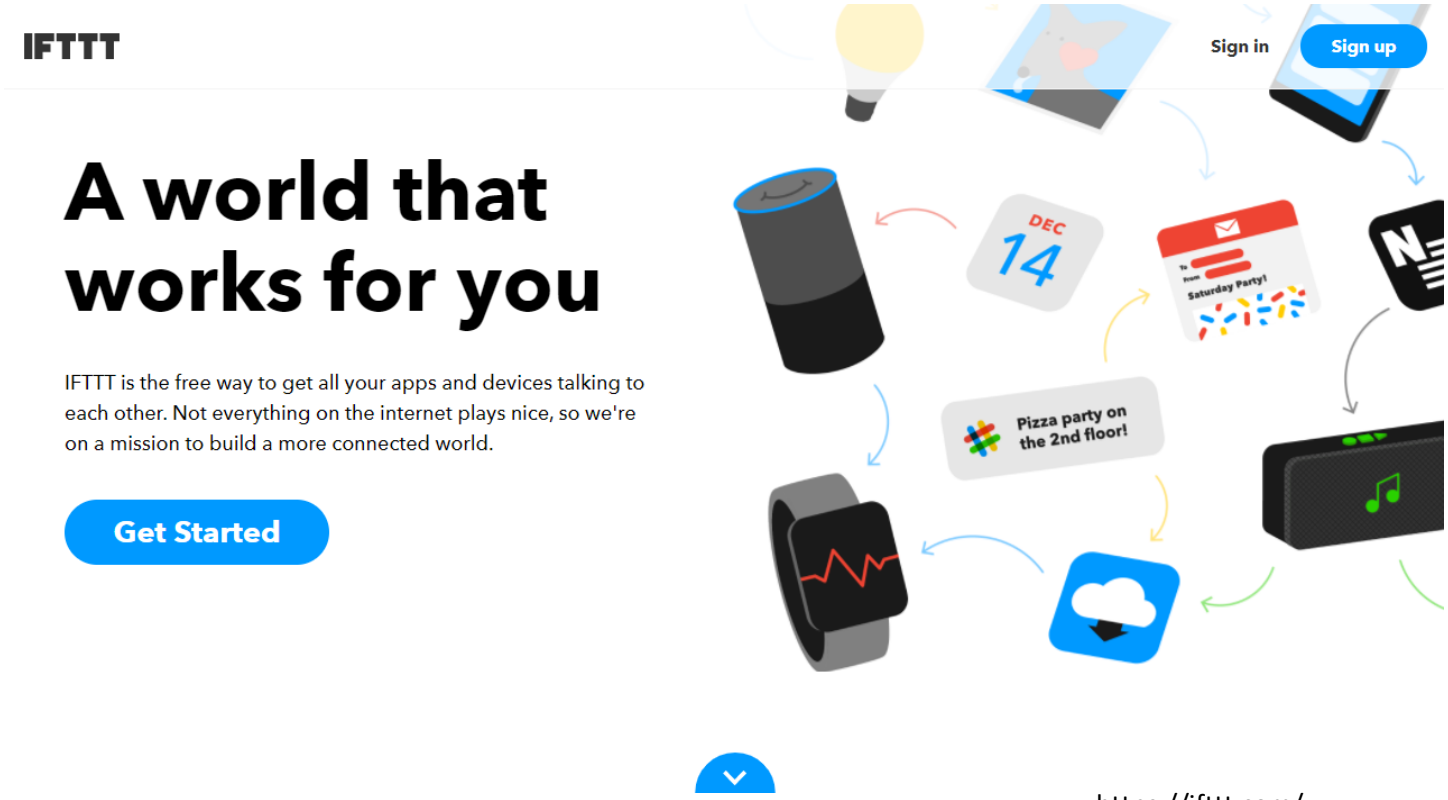
https://trigger-happy.eu/

Platforms



https://flow.microsoft.com/en-us/

Platforms



https://ifttt.com/

IFTTT

IFTTT - If This Then That

- Service to create chains of simple conditional statements, called applets
- Is triggered by changes that occur within other web services
- After a trigger executes an actionable service in the platform

AdafruitIO

MQTT

- MQTT (Message Queue Telemetry Transport) was originally developed out of IBM's pervasive computing team and their work with partners in the industrial sector.

- Lightweight message protocol:
  - Connecting to a server only takes about 80 bytes
  - Push data from server to device is about 20 bytes

- MQTT messages are sent to feeds in an MQTT broker such as AdafruitIO which then distributes them through the devices which subscribed feeds

- http://mqtt.org/

AdafruitIO and IFTTT Tutorial

In this tutorial we will develop an IFTTT applet that reacts to values in AdafruitIO feeds

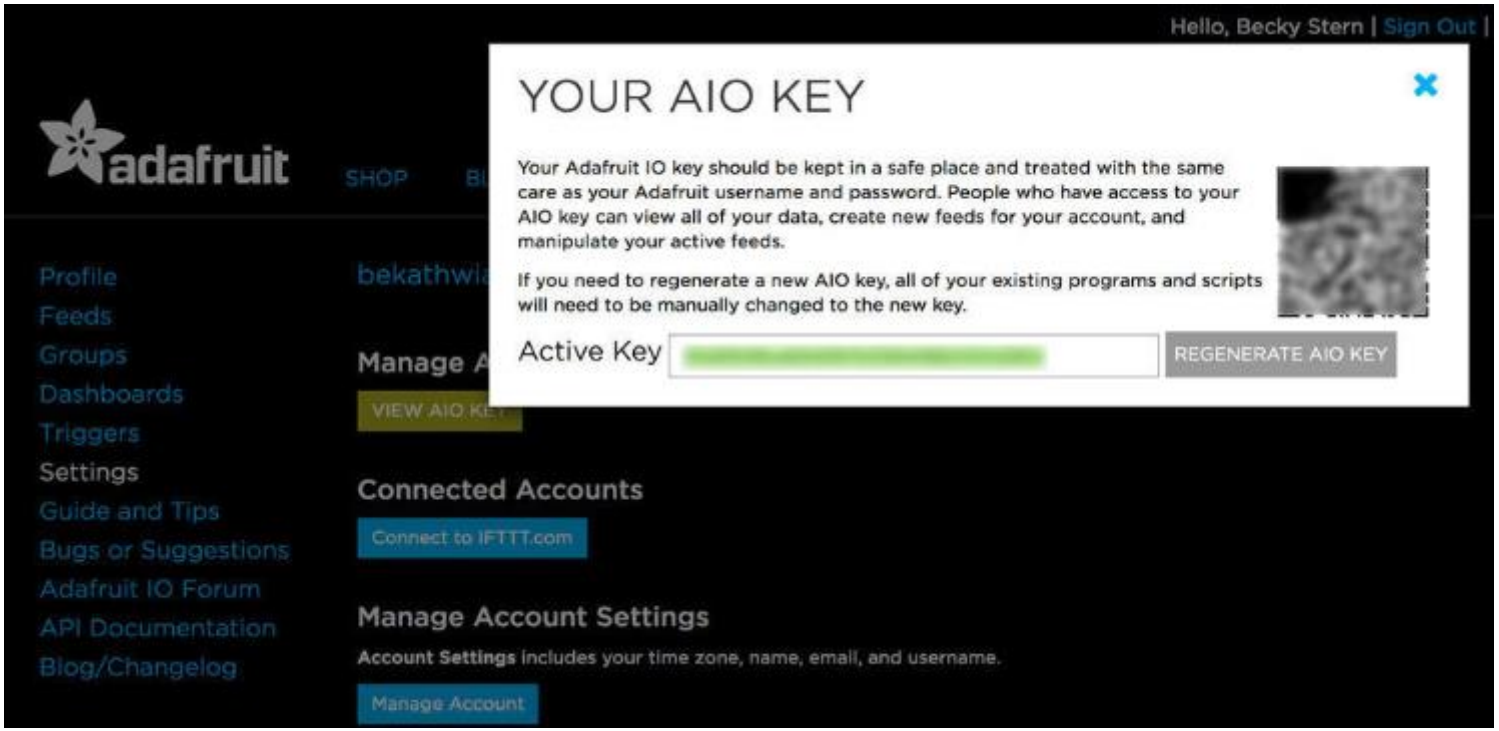
The objective is to monitor sensor values sent to AdafruitIO feeds and take actions in some contexts

In order to complete this tutorial it is needed:

- An IFTTT account
- An AdafruiIO account
- A smartphone with the IFTTT application installed

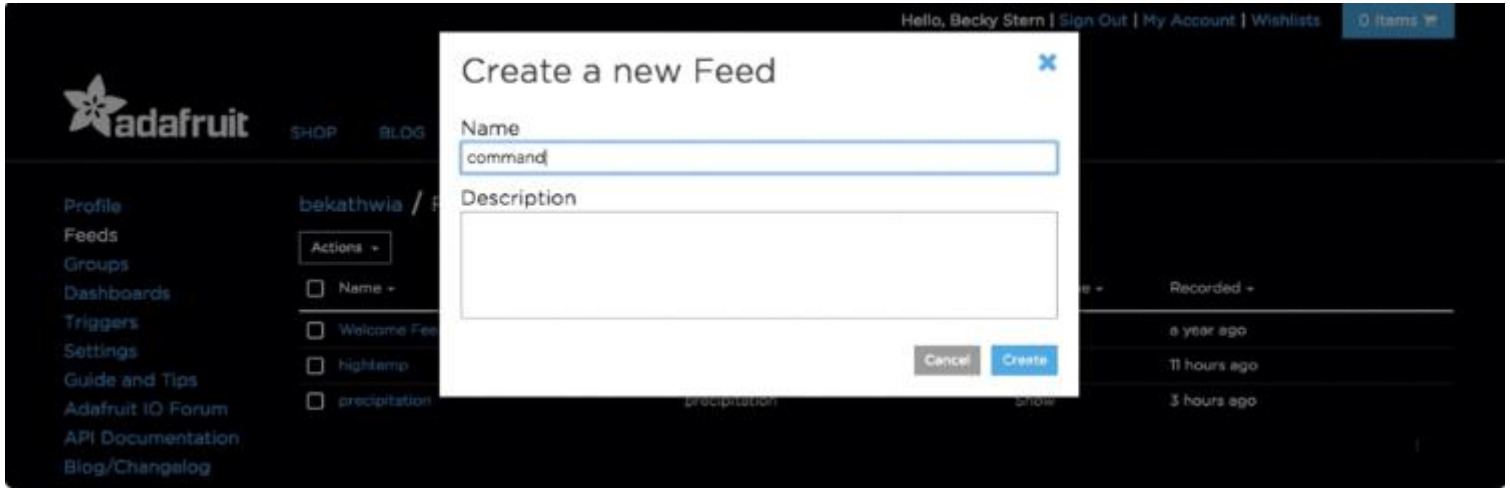
AdafruitIO and IFTTT Tutorial

Take note of your username and AIO Key



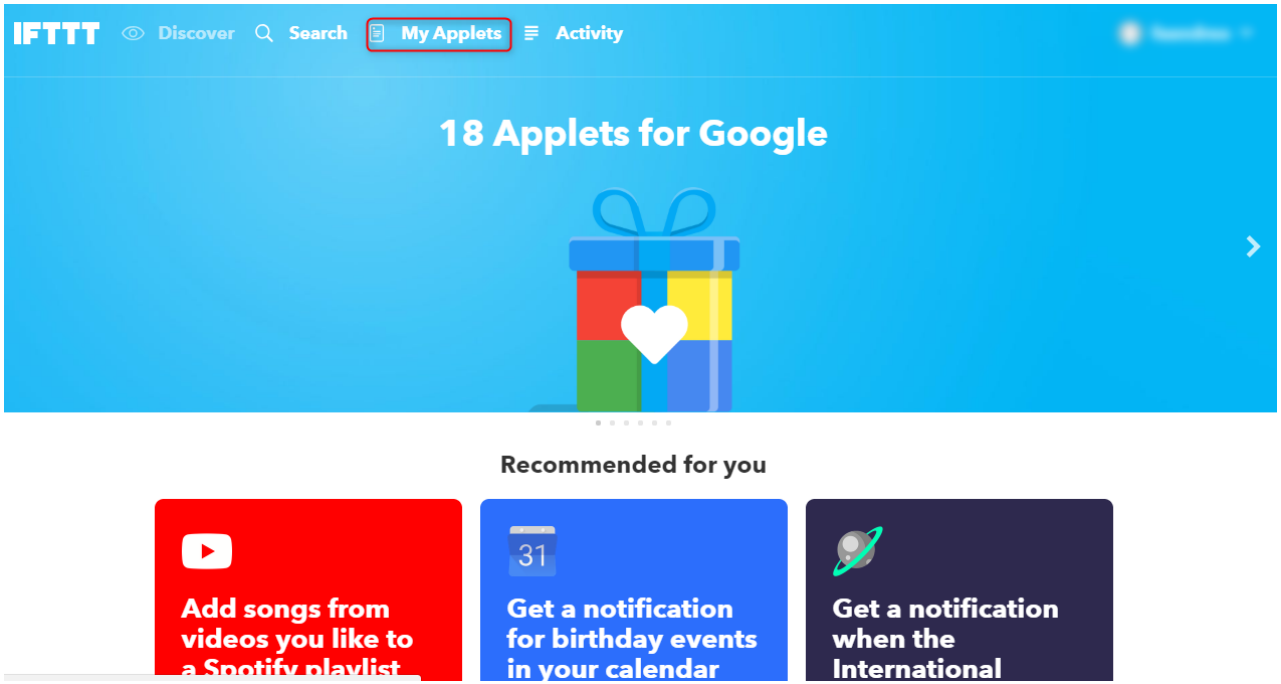
AdafruitIO and IFTTT Tutorial

Create a new feed named command in adafruitIO



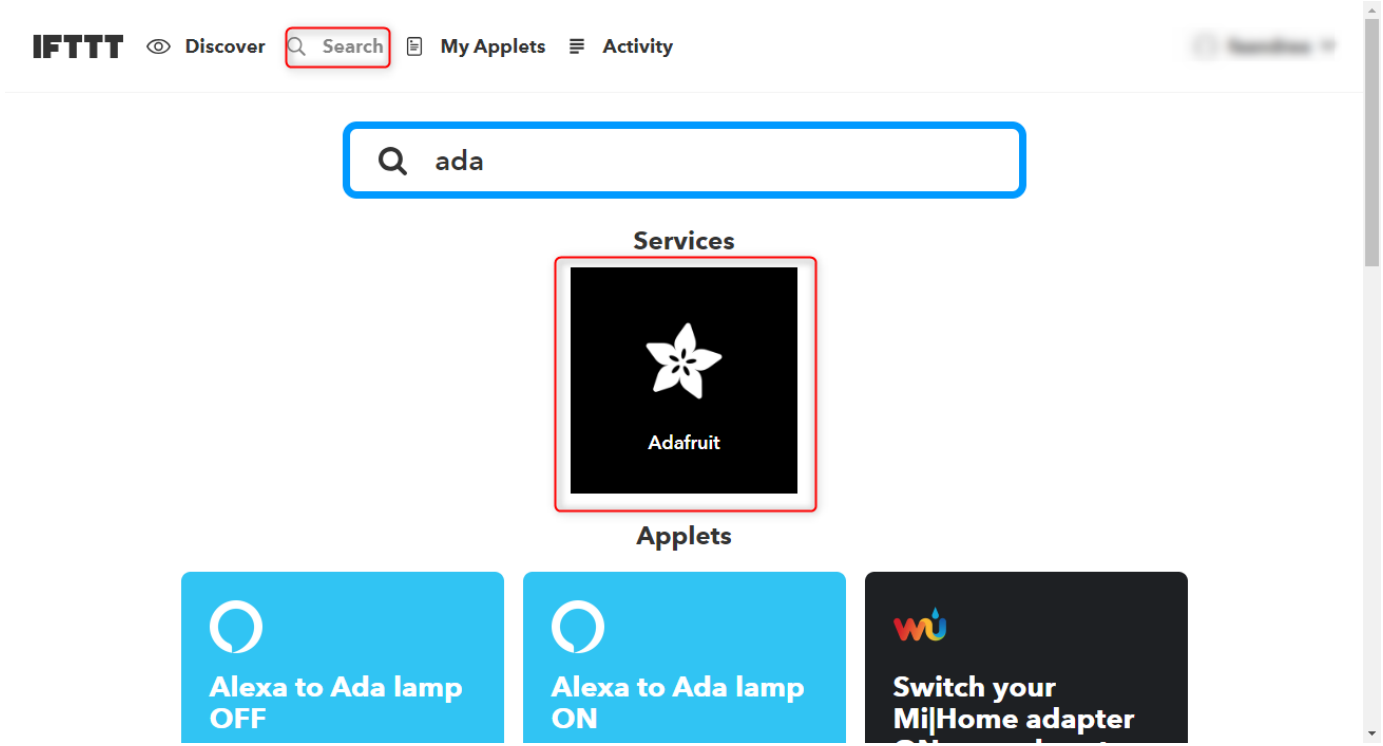
AdafruitIO and IFTTT Tutorial

Log in to the IFTTT platform



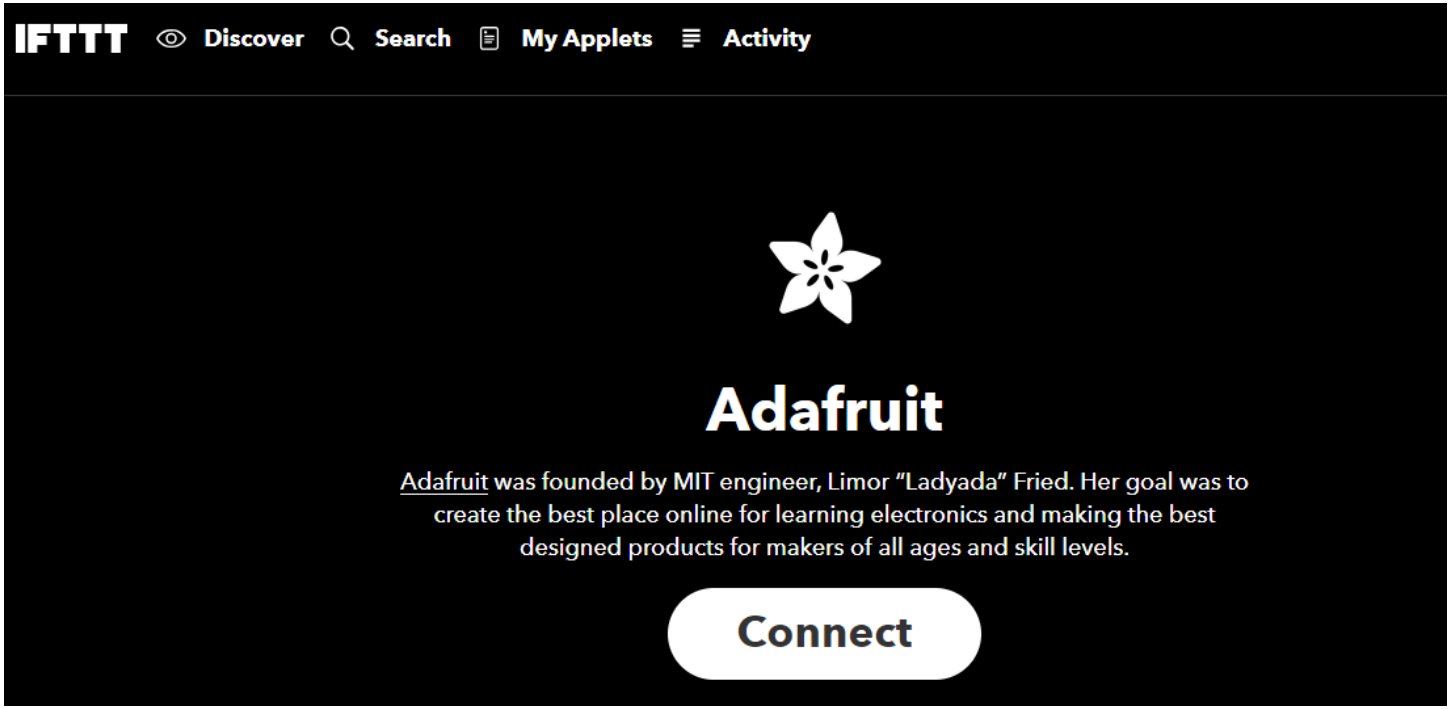
AdafruitIO and IFTTT Tutorial

Search for the Adafruit service



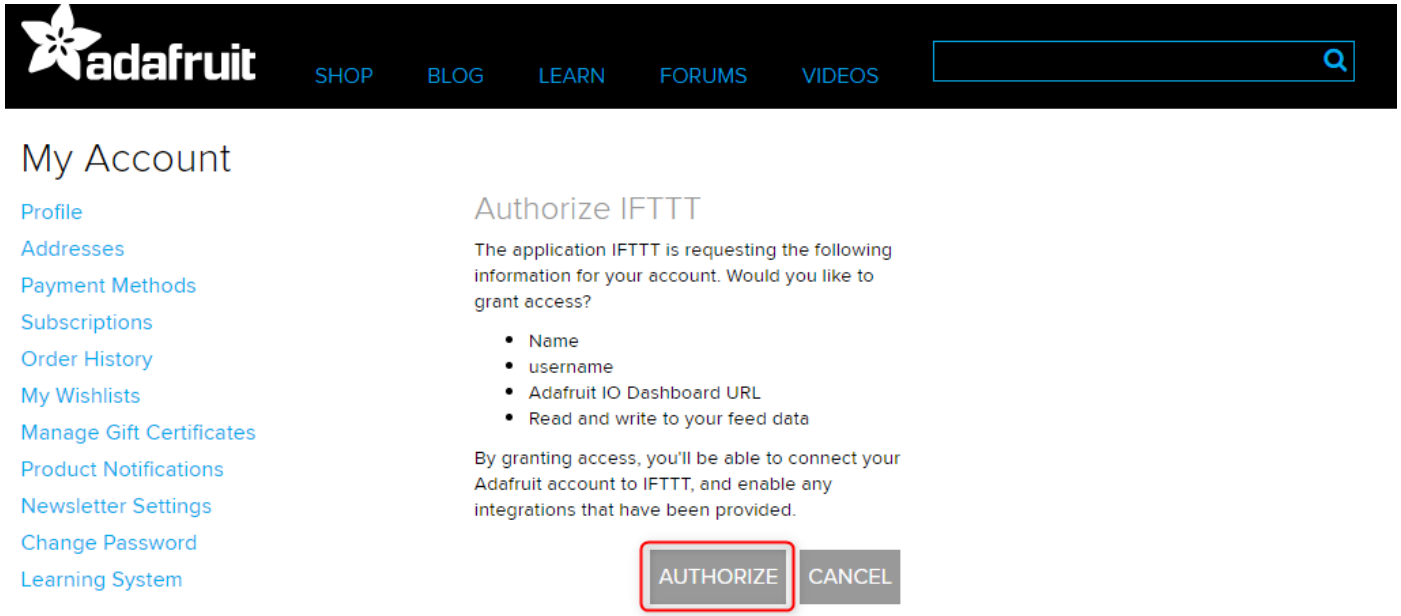
AdafruitIO and IFTTT Tutorial

Connect IFTTT to the Adafruit platform



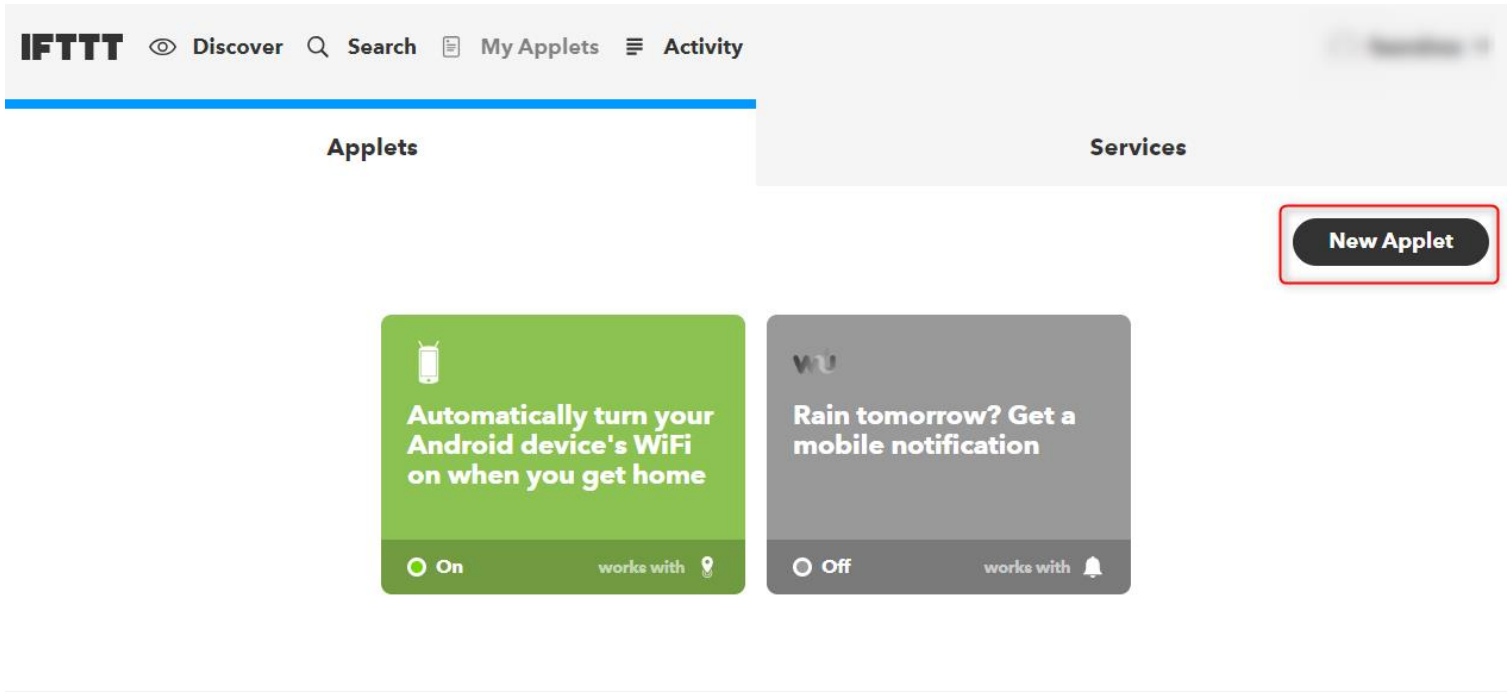
AdafruitIO and IFTTT Tutorial

Authorize IFTTT access



AdafruitIO and IFTTT Tutorial

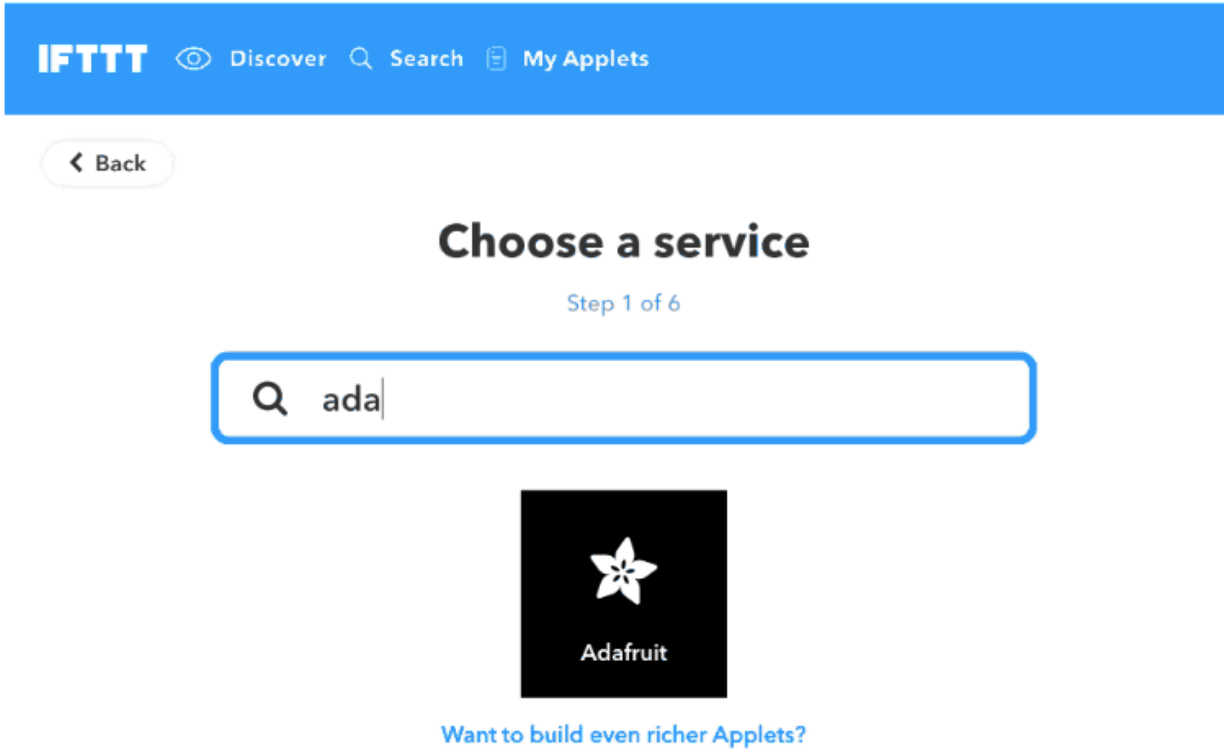
On the IFTTT platform lets create a new applet



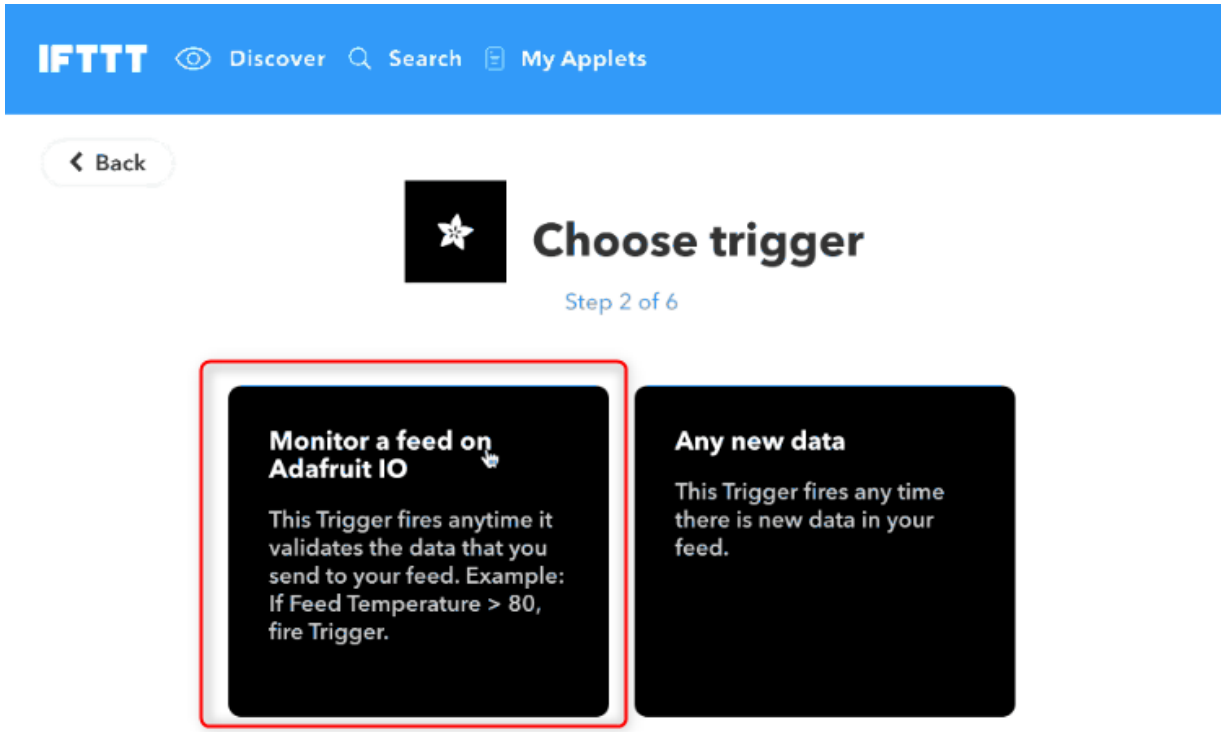
AdafruitIO and IFTTT Tutorial

AdafruitIO and IFTTT Tutorial

Choose the adafruit service



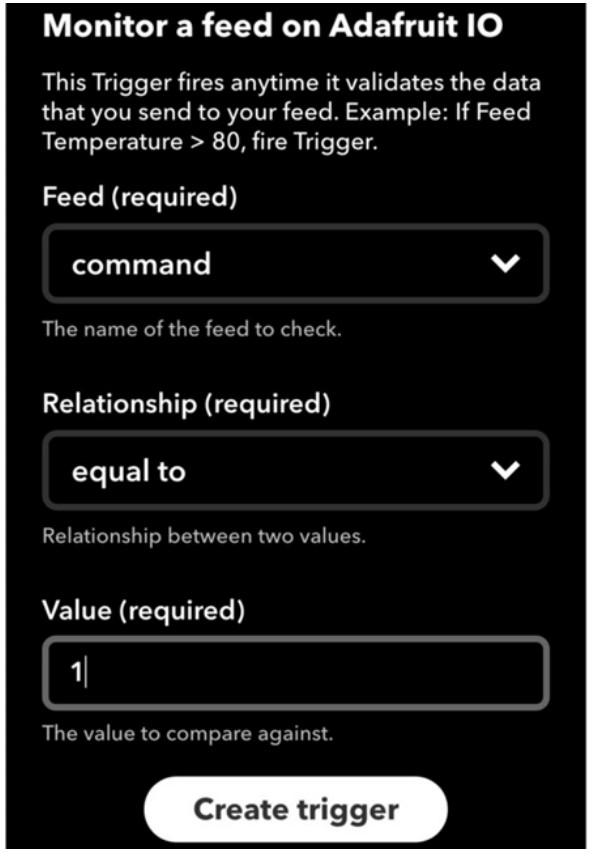
AdafruitIO and IFTTT Tutorial



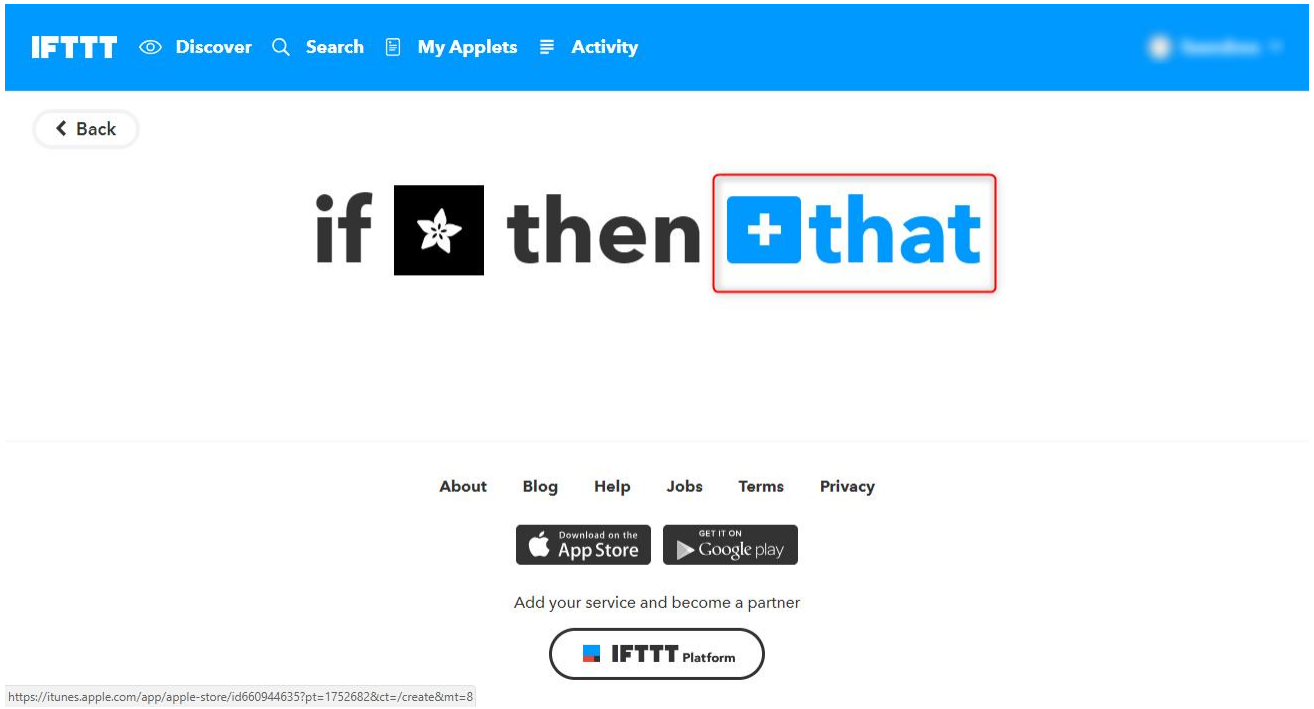
AdafruitIO and IFTTT Tutorial

Select the command feed created earlier

Create a trigger after the value 1 is sent to the command feed

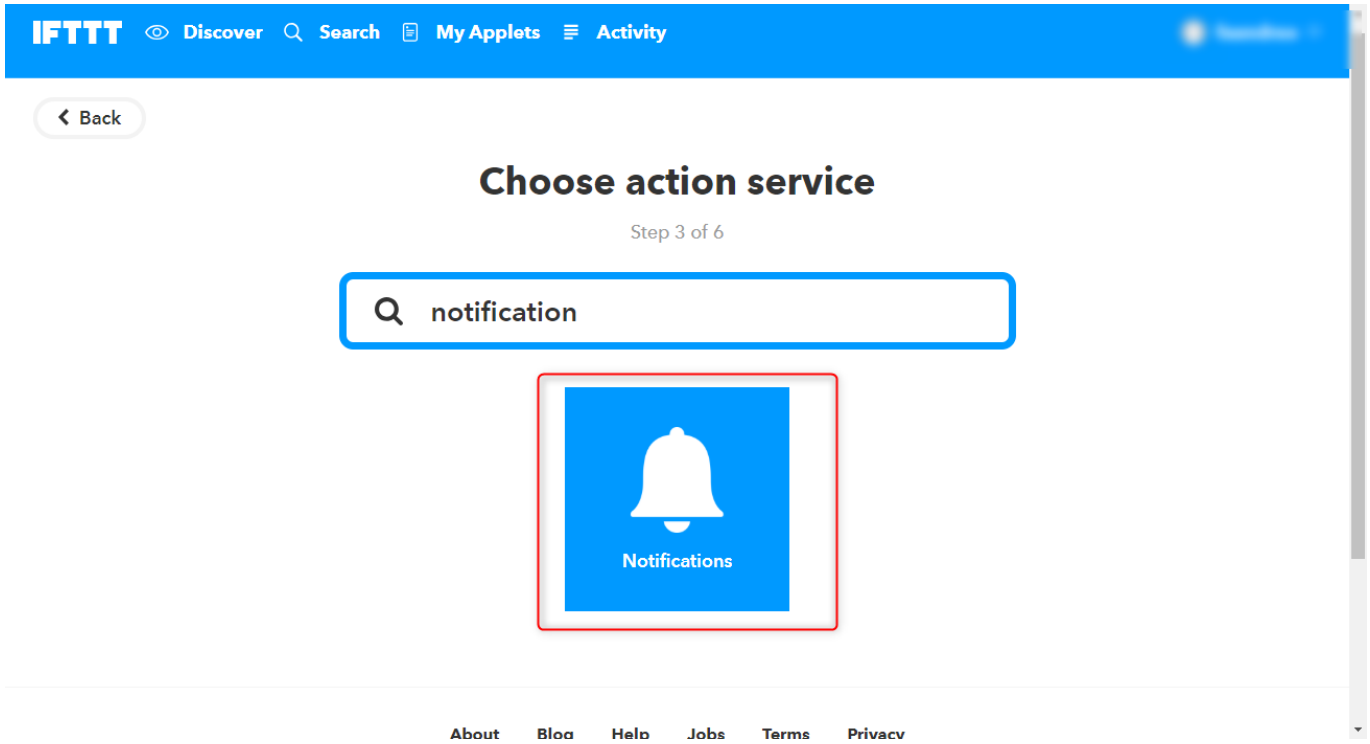


AdafruitIO and IFTTT Tutorial

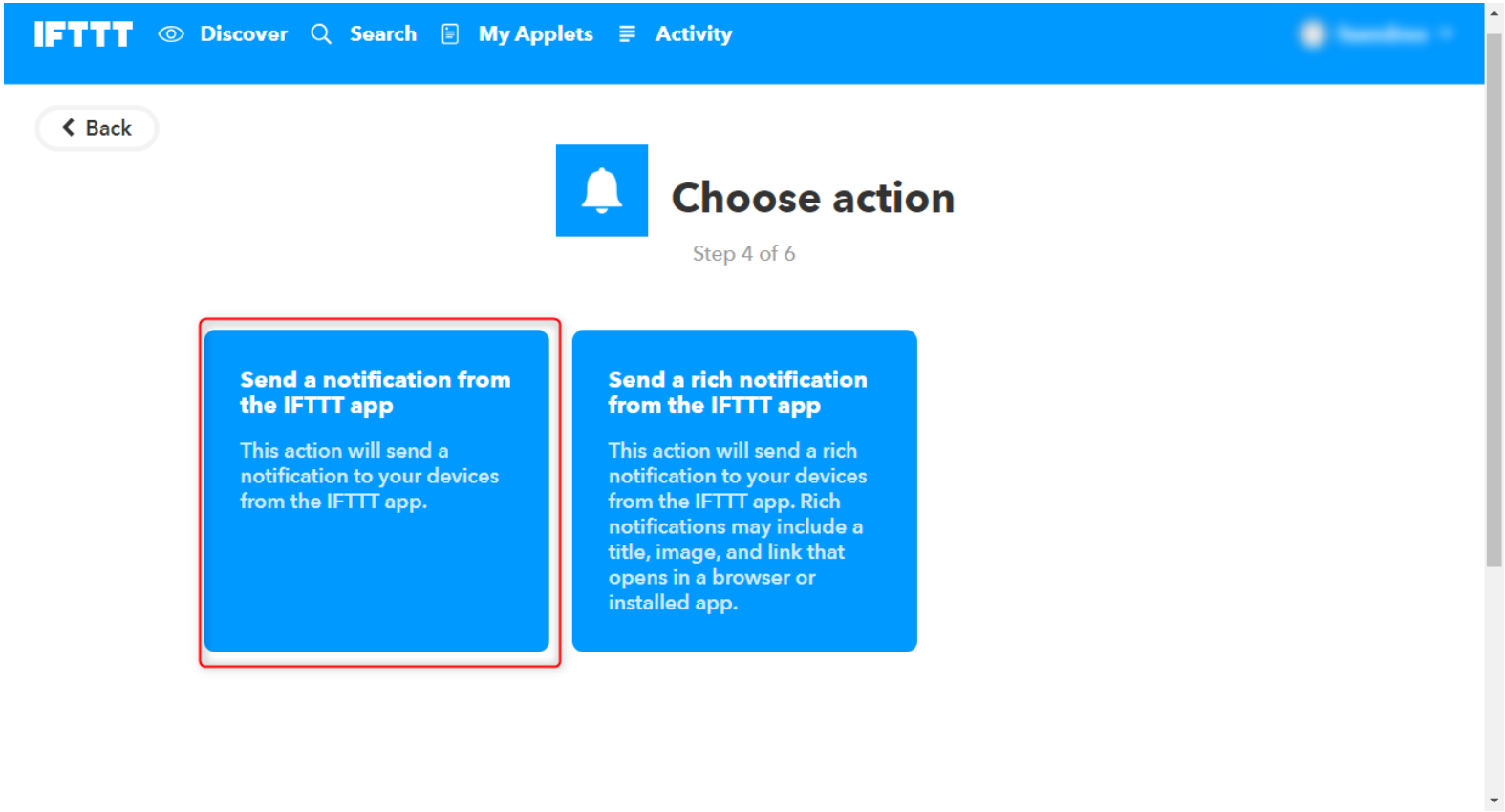


AdafruitIO and IFTTT Tutorial

Choose a notification service for the action



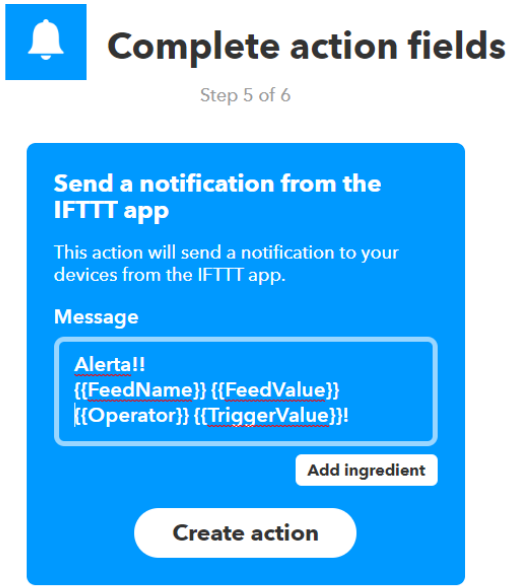
AdafruitIO and IFTTT Tutorial



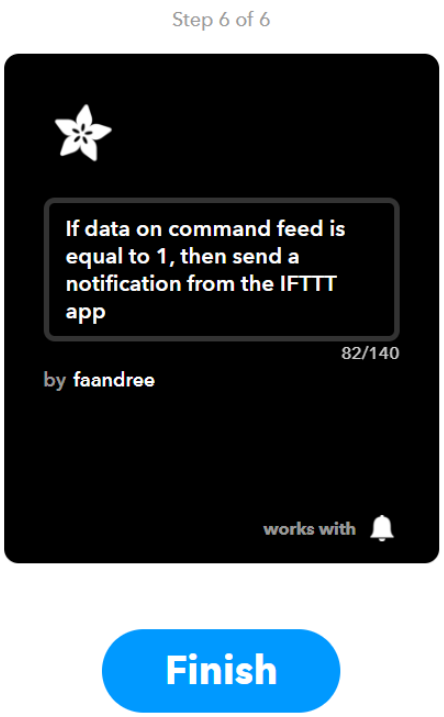
AdafruitIO and IFTTT Tutorial

Create a custom notification to display in device where the IFTTT app is installed

Inside the double brackets “{{ … }}”, are the values obtained from the trigger we created earlier



AdafruitIO and IFTTT Tutorial

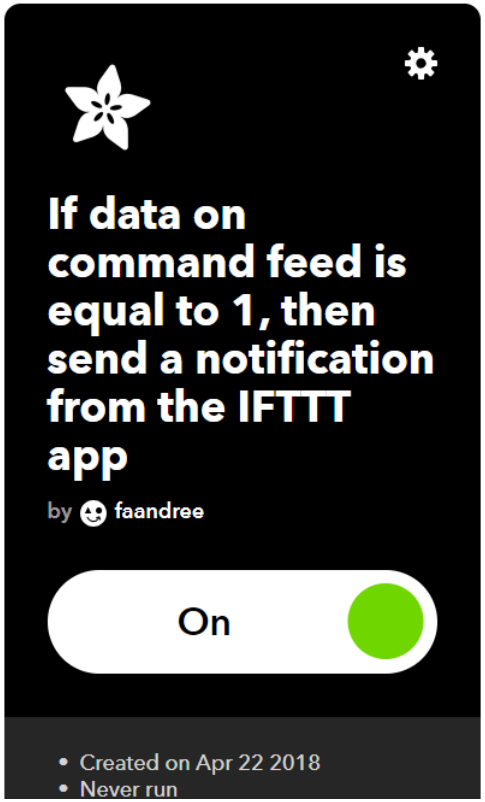


Review the rule created

If all is well we can finish the process

AdafruitIO and IFTTT Tutorial

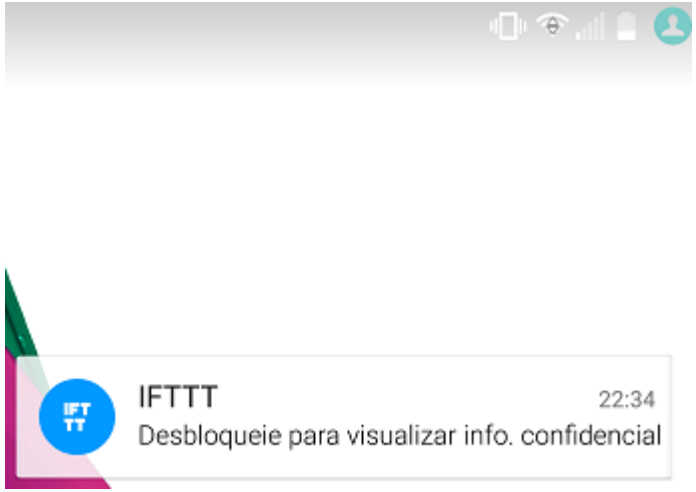
Active the rule created

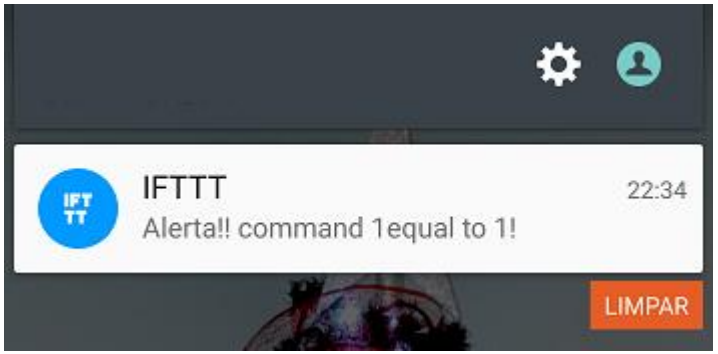


AdafruitIO and IFTTT Tutorial

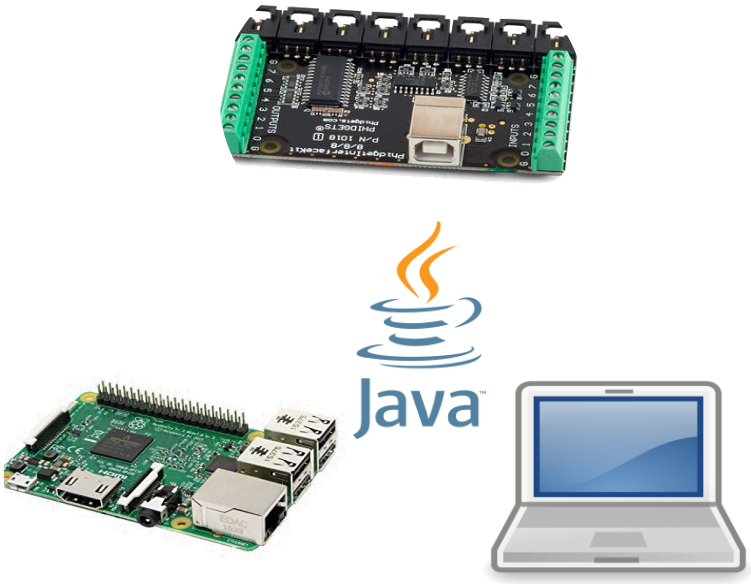
AdafruitIO and IFTTT Tutorial

Notification on the smartphone





Java and AdafruitIO



Java and AdafruitIO

A sample Java Program to test interation between AdafruitIO and a Java Program

```
import org.eclipse.paho.client.mqttv3.MqttClient;
import org.eclipse.paho.client.mqttv3.MqttConnectOptions;
import org.eclipse.paho.client.mqttv3.MqttException;
import org.eclipse.paho.client.mqttv3.MqttMessage;
import org.eclipse.paho.client.mqttv3.persist.MemoryPersistence;

public class MQTT_Test {

public static void main(String[] args) {
	String topic       = “topic";
	String content     = "Message from MqttPublishSample";
	int qos            = 2;
	String broker      = "tcp://io.adafruit.com:1883";
	String clientId    = "JavaSample";
	MemoryPersistence persistence = new MemoryPersistence();
                                                        (…)
```

Java and AdafruitIO

```
(…)
try {
        MqttClient sampleClient = new MqttClient(broker, clientId, persistence);
        MqttConnectOptions connOpts = new MqttConnectOptions();
        connOpts.setCleanSession(true);
        connOpts.setUserName(“username");
        connOpts.setPassword(“key".toCharArray());
        connOpts.setSSLProperties(new Properties());
        System.out.println("Connecting to broker: "+broker);
        sampleClient.connect(connOpts);
        System.out.println("Connected");
        System.out.println("Publishing message: "+content);
        MqttMessage message = new MqttMessage(content.getBytes());
        message.setQos(qos);
        sampleClient.publish(topic, message);
        System.out.println("Message published");
        sampleClient.disconnect();
        System.out.println("Disconnected");
        System.exit(0);
(…)
```

Java and AdafruitIO

```
                                                    (…)
} catch(MqttException me) {
        System.out.println("reason "+me.getReasonCode());
        System.out.println("msg "+me.getMessage());
        System.out.println("loc "+me.getLocalizedMessage());
        System.out.println("cause "+me.getCause());
        System.out.println("excep "+me);
        me.printStackTrace();
    }
  }
```

Java and AdafruitIO

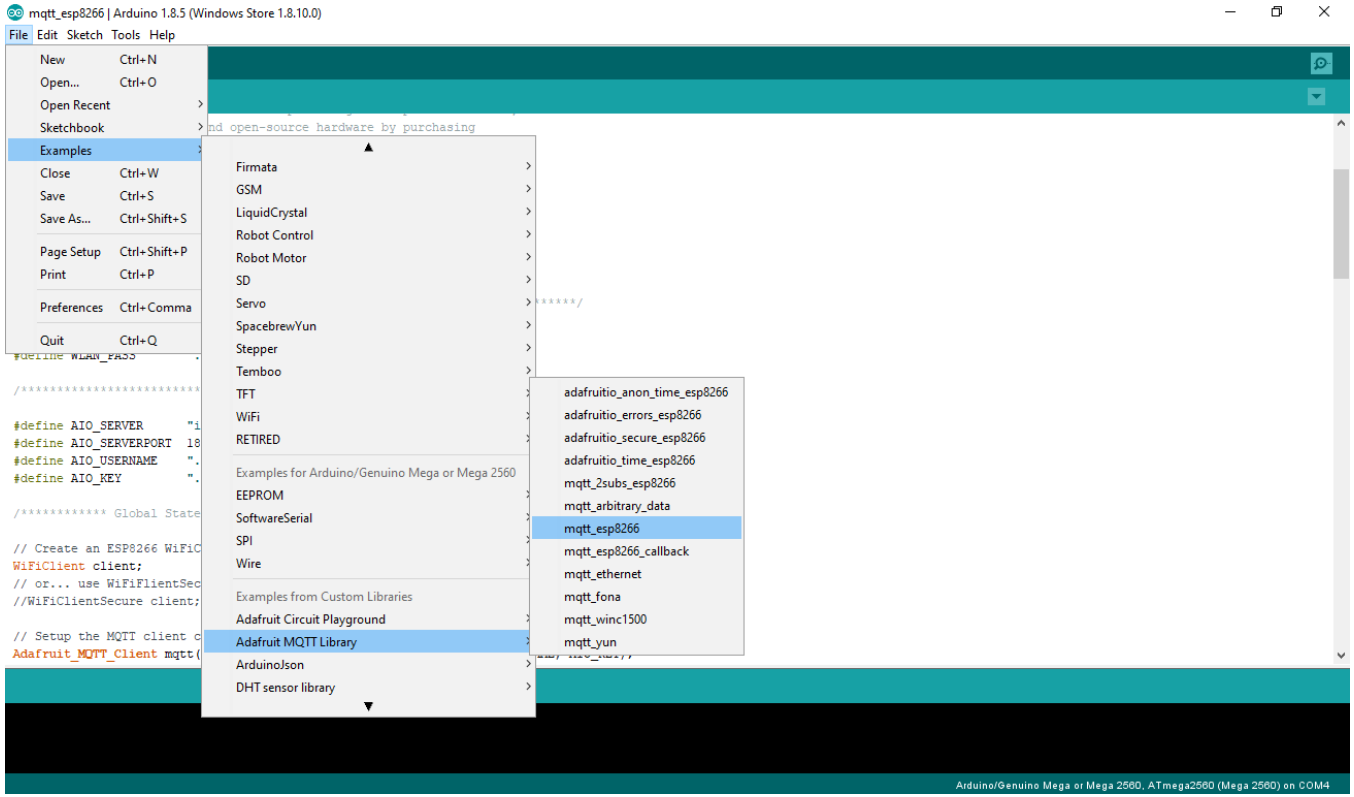
References:

- https://www.eclipse.org/paho
- http://wiki.eclipse.org/Paho
- http://www.eclipse.org/paho/clients/java

Arduino and AdafruitIO

Arduino and AdafruitIO

Use an arduino studio IDE example to get started



Arduino and AdafruitIO

```
#include <ESP8266WiFi.h>
#include "Adafruit_MQTT.h"
#include "Adafruit_MQTT_Client.h"

/************************* WiFi Access Point *********************************/

#define WLAN_SSID       "...your SSID..."
#define WLAN_PASS       "...your password..."

/************************* Adafruit.io Setup *********************************/

#define AIO_SERVER      "io.adafruit.com"
#define AIO_SERVERPORT  1883                   // use 8883 for SSL
#define AIO_USERNAME    "...your AIO username (see https://accounts.adafruit.com)..."
#define AIO_KEY         "...your AIO key..."

/************ Global State (you don't need to change this!) ******************/

// Create an ESP8266 WiFiClient class to connect to the MQTT server.
WiFiClient client;
// or... use WiFiFlientSecure for SSL
//WiFiClientSecure client;

// Setup the MQTT client class by passing in the WiFi client and MQTT server and login details.
Adafruit_MQTT_Client mqtt(&client, AIO_SERVER, AIO_SERVERPORT, AIO_USERNAME, AIO_KEY);
```

Arduino and AdafruitIO

```
….
/****************************** Feeds ***************************************/
Adafruit_MQTT_Publish commandpublish = Adafruit_MQTT_Publish(&mqtt, AIO_USERNAME "/feeds/command");
Adafruit_MQTT_Subscribe commandsubscribe = Adafruit_MQTT_Subscribe(&mqtt, AIO_USERNAME "/feeds/command");
/*************************** Sketch Code ************************************/
// Bug workaround for Arduino 1.6.6, it seems to need a function declaration
// for some reason (only affects ESP8266, likely an arduino-builder bug).
void MQTT_connect();

void setup() {
  Serial.begin(115200);
  delay(10);
  Serial.println(F("Adafruit MQTT demo"));
  // Connect to WiFi access point.
  Serial.println(); Serial.println();
  Serial.print("Connecting to ");
  Serial.println(WLAN_SSID);
  WiFi.begin(WLAN_SSID, WLAN_PASS);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
  Serial.println("WiFi connected");
  Serial.println("IP address: "); Serial.println(WiFi.localIP());
  // Setup MQTT subscription for onoff feed.
  mqtt.subscribe(&commandsubscribe);
}
…..
```

Arduino and AdafruitIO

```
….
uint32_t x=0;

void loop() {
  // Ensure the connection to the MQTT server is alive (this will make the first
  // connection and automatically reconnect when disconnected).  See the MQTT_connect
  // function definition further below.
  MQTT_connect();

Adafruit_MQTT_Subscribe *subscription;
  while ((subscription = mqtt.readSubscription(5000))) {
    if (subscription == &commandsubscribe) {
      Serial.print(F("Got: "));
      Serial.println((char *)commandsubscribe.lastread);
    }
  }
  // Now we can publish stuff!
  Serial.print(F("\nSending val "));
  Serial.print(x);
  Serial.print("...");
  if (! commandpublish.publish(x++)) {
    Serial.println(F("Failed"));
  } else {
    Serial.println(F("OK!"));
  }
}
 …..
```

Arduino and AdafruitIO

```
….
// Function to connect and reconnect as necessary to the MQTT server.
// Should be called in the loop function and it will take care if connecting.
void MQTT_connect() {
  int8_t ret;

  // Stop if already connected.
  if (mqtt.connected()) {
    return;
  }

  Serial.print("Connecting to MQTT... ");

  uint8_t retries = 3;
  while ((ret = mqtt.connect()) != 0) { // connect will return 0 for connected
       Serial.println(mqtt.connectErrorString(ret));
       Serial.println("Retrying MQTT connection in 5 seconds...");
       mqtt.disconnect();
       delay(5000);  // wait 5 seconds
       retries--;
       if (retries == 0) {
         // basically die and wait for WDT to reset me
         while (1);
       }
  }
  Serial.println("MQTT Connected!");
}
```

Arduino and AdafruitIO

References

- https://learn.adafruit.com/mqtt-adafruit-io-and-you/intro-to-adafruit-mqtt
- https://learn.adafruit.com/mqtt-adafruit-io-and-you/overview

Javascript and AdafruitIO

Javascript and AdafruitIO

A sample html web page with a javascript script to test interation between AdafruitIO and javascript

```
<html>
	<head>
		<script src="https://cdnjs.cloudflare.com/ajax/libs/paho-mqtt/1.0.1/mqttws31.js" type="text/javascript"></script>
	</head>
	<body>
		<h1> test </h1>
		<script>

		// Create a client instance
		client = new Paho.MQTT.Client("io.adafruit.com", Number(443), "wwww");

		// set callback handlers
		client.onConnectionLost = onConnectionLost;
		client.onMessageArrived = onMessageArrived;

		// connect the client
		client.connect({onSuccess:onConnect, userName:"username", password:"AIOKey", useSSL : true, mqttVersion: 4 });

….
```

Javascript and AdafruitIO

```
…
                // called when the client connects
                function onConnect() {
                  // Once a connection has been made, make a subscription and send a message.
                  console.log("onConnect");
                  client.subscribe("username/feeds/command");
                  message = new Paho.MQTT.Message("1");
                  message.destinationName = "username/feeds/command";
                  client.send(message);
                }

                // called when the client loses its connection
                function onConnectionLost(responseObject) {
                  if (responseObject.errorCode !== 0) {
                        console.log("onConnectionLost:"+responseObject.errorMessage);
                  }
                }

                // called when a message arrives
                function onMessageArrived(message) {
                  console.log("onMessageArrived:"+message.payloadString);
                }

                </script>
        </body>
</html>
```

Javascript and AdafruitIO

References:

- https://www.eclipse.org/paho
- http://wiki.eclipse.org/Paho
- http://www.eclipse.org/paho/clients/js

Exercises

Integrate your sensors from last class and publish their values in an AdafruitIO feed

Create an IFTTT applet to react to diferente contexts based on your sensor values and send notfications to your smartphone.

Examples:

- Send a notification when the temperature is uncomfortable
- Send an email when movement is detected
- Send an email and notification when a flame is detected

Exercises

Subscribe another group sensor values and light a led on your arduino according to certain range conditions

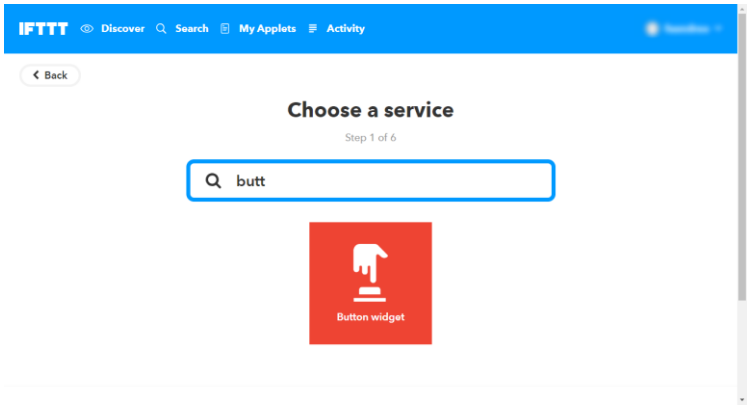
Use exclusively the AdafruitIO platform

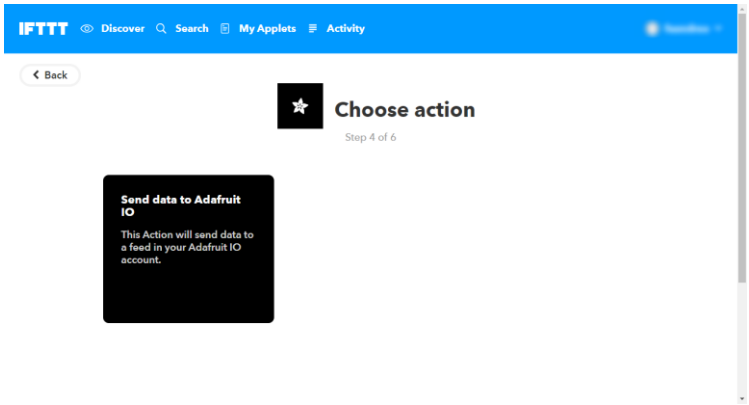
Exercises

Create a button trigger and an AdafruitIO action within an IFTTT applet

When the button is clicked on the IFTTT application data is submitted adafruit feed

Use an AdafruitIO client react to changes on the feed (Ex: light a led, buzz a buzzer, etc)





Exercises

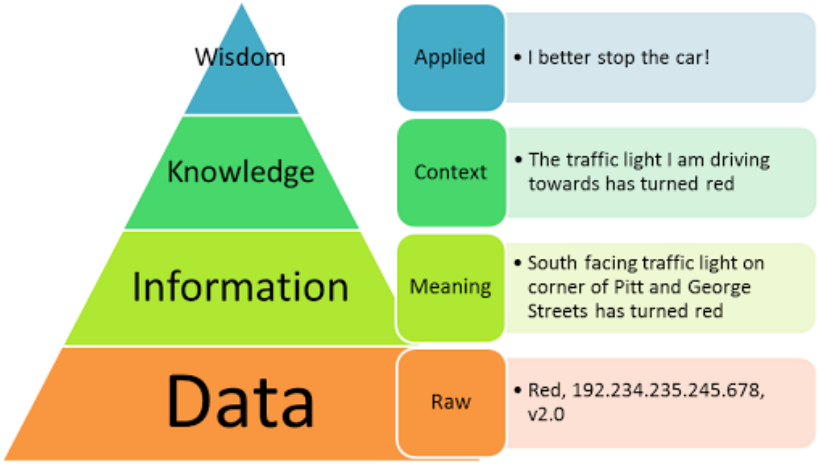
Remember the data pyramid

An efficient actuator perform its taks after data and information processing

Need to get information about contexts for deploying actions in automonous system

Actions can be:

- Reactive
- Deliberative
- Hybrid

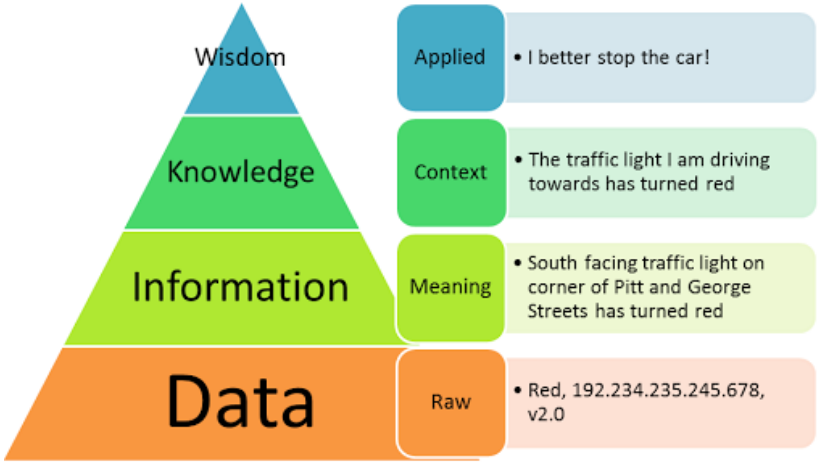


Exercises

Create a Java program that assess sensor values and computes context based on data and information fusion

Examples:

- Thermal sensation (temperature, humidity, wind, solar radiation, clothing, etc)
- Noise sensation (exposure, noise level, etc)
- Safety indicator (exposure, level)
- Smart alert system (location, time, tasks, etc)



Automonous Systems

INTERNT OF THINGS

Fábio Silva
fabiosilva@di.uminho.pt