Advanced Software Engineering (LAB)

Stefano Forti

name.surname@di.unipi.it

Department of Computer Science @ University of Pisa

Step 0

We will work in 5 groups.

It suffices 1 computer per group, with:

- Linux/Windows installed natively, and
- npm and nodeJS installed.

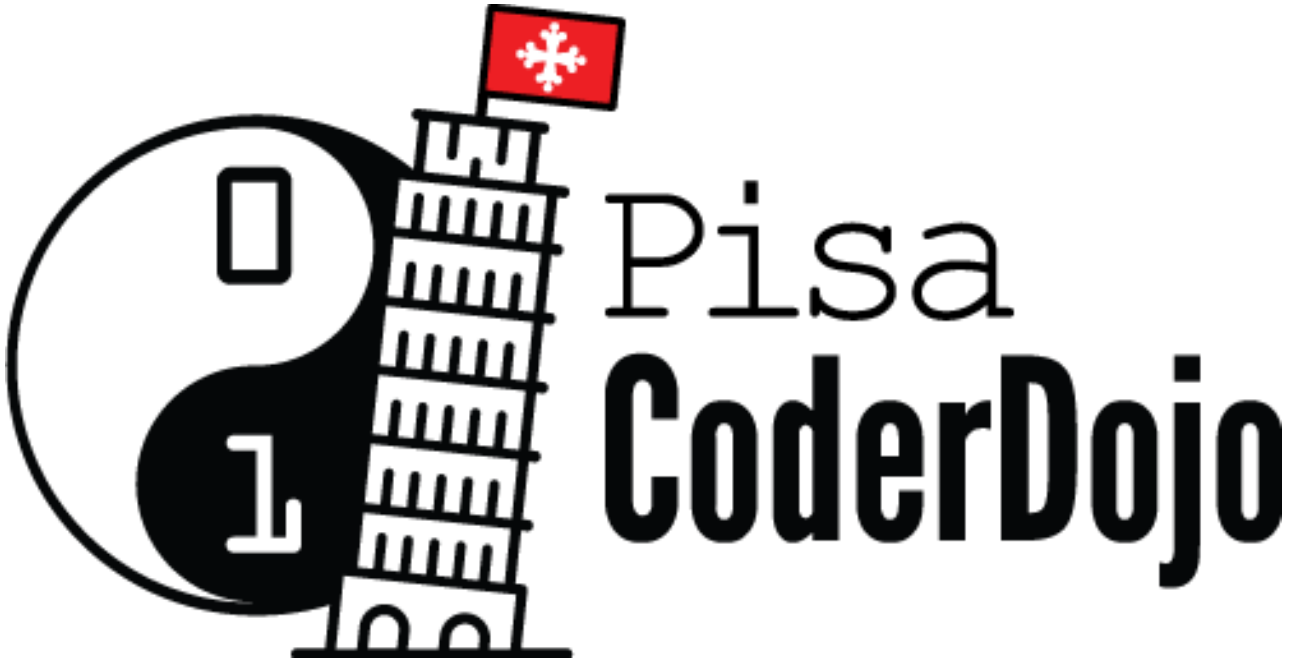
Select the computer(s) you will use and run:

```
sudo npm cache clean –f
sudo npm install -g n
sudo n 8.9.2
```

What will I do?

- Code an application to monitor soil moisture.
- Code an application to visualise data.
- Stream data to Cloud for further processing.

Special Thanks



pisa.coderdojo.it

Micro USB
MSC, UART, CMSIS-DAP
Drag-and-drop programming

5x5 LED Matrix

Digital/analog IO
Muxable to SPI, UART, I2C

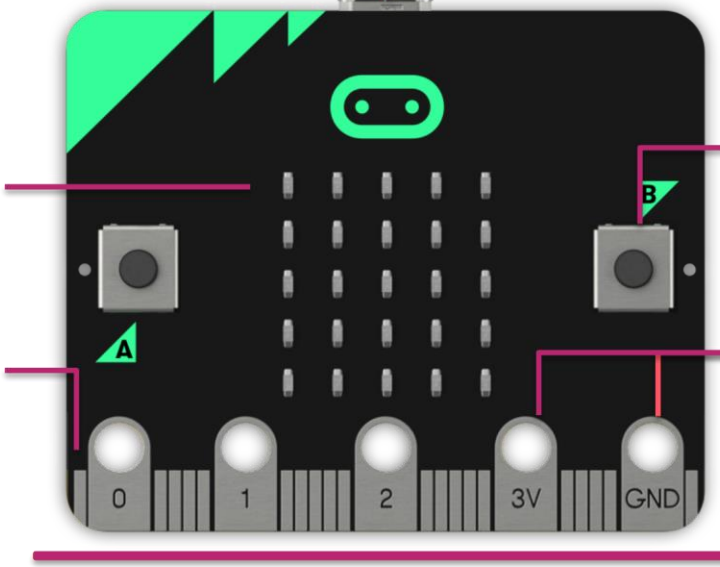
Pads for crocodile clips

Holes for banana plugs

User buttons

External supply
Regulated 3.3V in or battery out

Edge Connector



2.4GHZ Antenna
Bluetooth low energy
Gazell

Nordic nRF51822

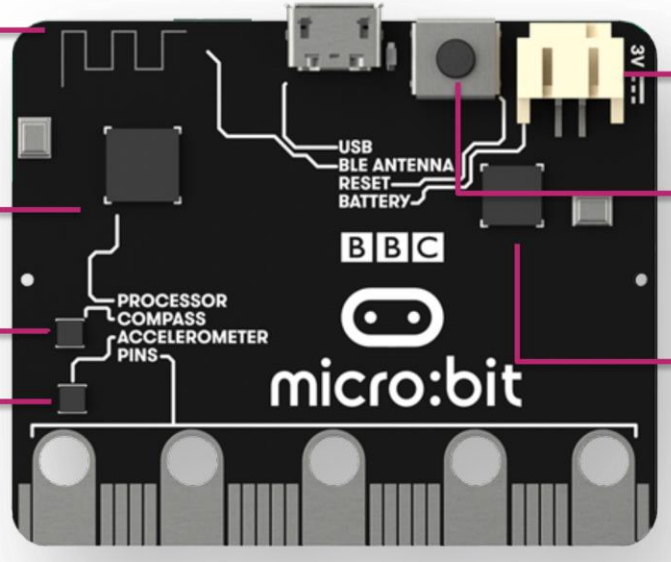
Magnetometer
Freescale MMA8652

Acceleromater
Freescale MAG3110

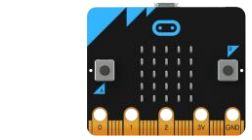
Battery connector
JST connection for 3V

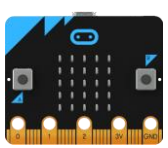
Reset Button

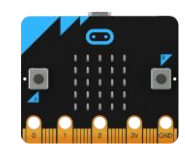
Freescale KL26Z
USB Interface chip



Bonsai Fog







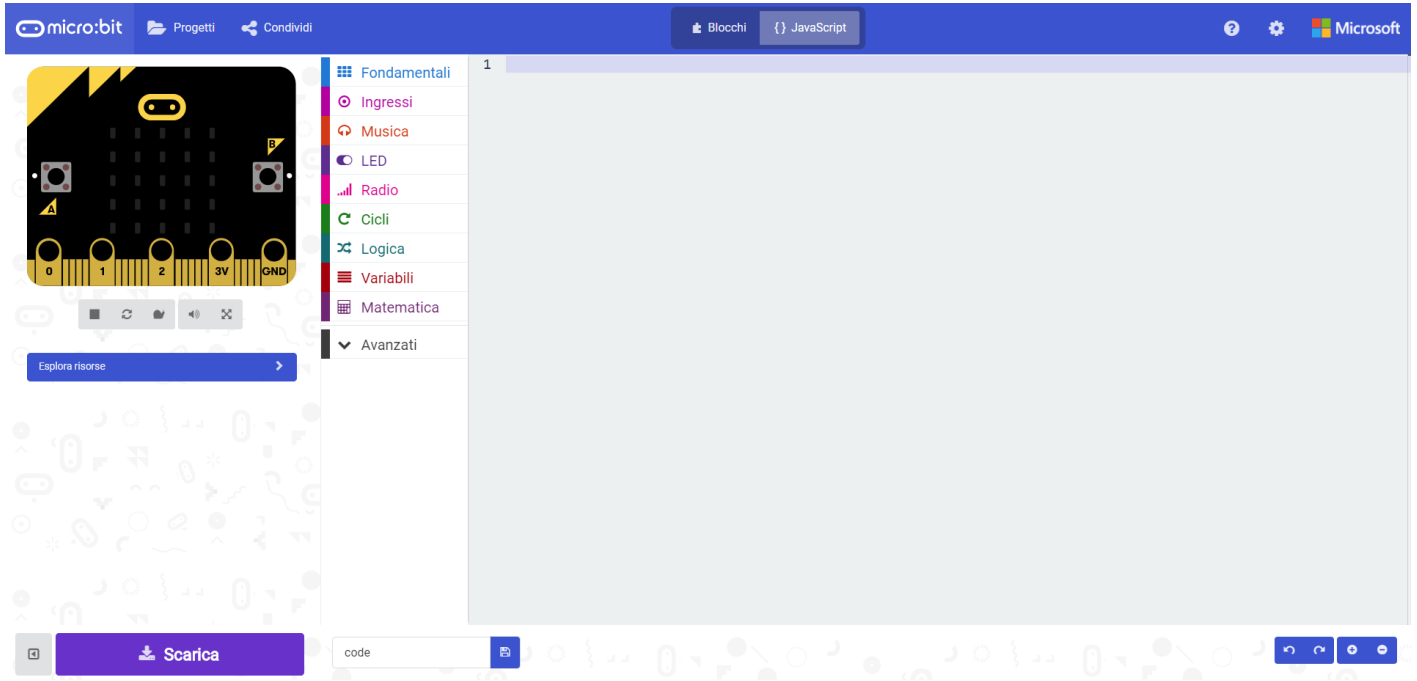
Cloud

Fog

IoT

Make code!

- To code a micro:bit, you need to navigate to [https://makecode.microbit.org]
- Then, select the JavaScript editor (indicated by the curly braces):



Measuring Moisture

- We start coding a moisture meter using two paper clips.
- The soil itself has some electrical resistance which depends on the amount of water and nutrients in it.

```
basic.forever(() => {
    led.plotBarGraph(
        pins.analogReadPin(AnalogPin.P0),
        1023
    )
})
```

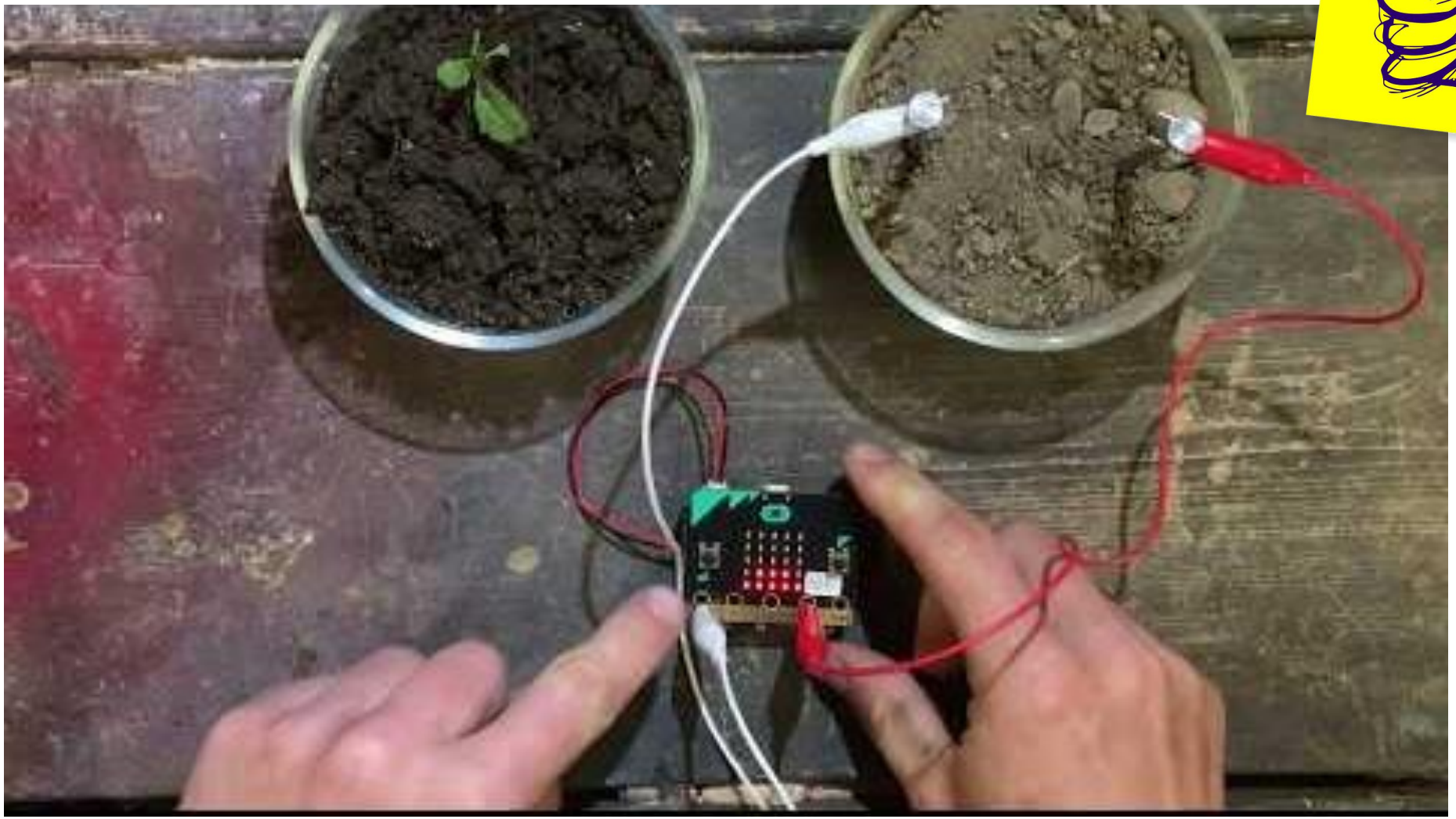
We read the voltage on pin **P0** to measure the soil electrical resistance. The returned value is among **0** and **1023**.

Sensor data values

- In the previous program, we only have a rough idea of what the sensor value is.
- Let's add code that displays the current reading when button **A** is pressed.

```
let reading = 0
basic.forever(() => {
    reading = pins.analogReadPin(AnalogPin.P0)
    led.plotBarGraph(
        reading,
        1023
    )
    if (input.buttonIsPressed(Button.A)) {
        basic.showNumber(reading)
    }
})
```

Try it!



Energy-aware

- We want our soil probes to work for a long time and to save our **battery power**, so we need to tweak our code so our **moisture sensor** doesn't use too much energy.

```
led.setBrightness(64)
let reading = 0
basic.forever(() => {
    pins.analogWritePin(AnalogPin.P1, 1023)
    reading = pins.analogReadPin(AnalogPin.P0)
    pins.analogWritePin(AnalogPin.P1, 0)
    led.plotBarGraph(
        reading,
        1023
    )
    if (input.buttonIsPressed(Button.A)) {
        basic.showNumber(reading)
    }
    basic.pause(1000)
})
```

Connect to the Dashboard

- Use the radio to send the current moisture level to a dashboard micro:bit.
- The dashboard will display one LED per micro:bit.

```
radio.setTransmitSerialNumber(true)
radio.setGroup(4)
led.setBrightness(64)
let reading = 0
basic.forever(() => {
    pins.analogWritePin(AnalogPin.P1, 1023)
    reading = pins.analogReadPin(AnalogPin.P0)
    radio.sendNumber(reading / 4);
    pins.analogWritePin(AnalogPin.P1, 0)
    led.plotBarGraph(
        reading,
        1023
    )
    if (input.buttonIsPressed(Button.A)) {
        basic.showNumber(reading)
    }
    basic.pause(1000)
})
```

Write to serial

- Write to the serial port to which micro:bit is connected.

```
radio.setTransmitSerialNumber(true)
radio.setGroup(4)
led.setBrightness(64)
let reading = 0
basic.forever(() => {
    pins.analogWritePin(AnalogPin.P1, 1023)
    reading = pins.analogReadPin(AnalogPin.P0)
    radio.sendNumber(reading / 4);
    pins.analogWritePin(AnalogPin.P1, 0)
    led.plotBarGraph(
        reading,
        1023
    )
    if (input.buttonIsPressed(Button.A)) {
        basic.showNumber(reading)
    }
    serial.writeValue("moisture", reading)
    basic.pause(1000)
})
```

Setup

- Create an account on [https://thingspeak.com/]

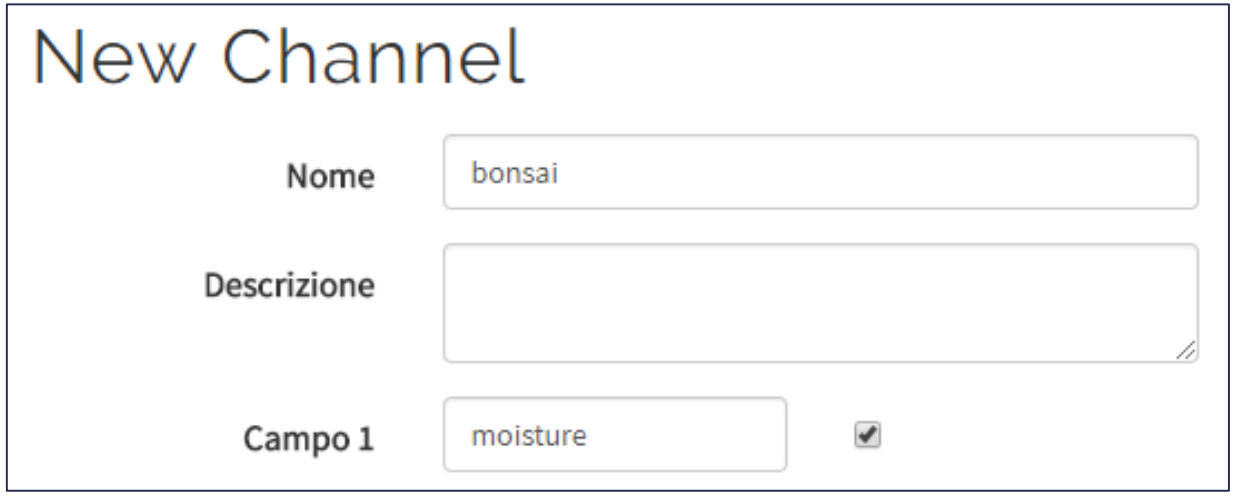
- Install `node.js` [https://nodejs.org/it/download/]
- Connect your micro:bit to your computer via USB.
- Download the `fog_primer.zip` from the Moodle and unzip it.
- Move to the folder where you unzipped it and run:

```
npm install
```

- This will download all that is needed to read from micro:bit's serial port.

Create a channel

- Enter *ThingSpeak* and create a new channel.
- You should only fill the blanks indicated below.

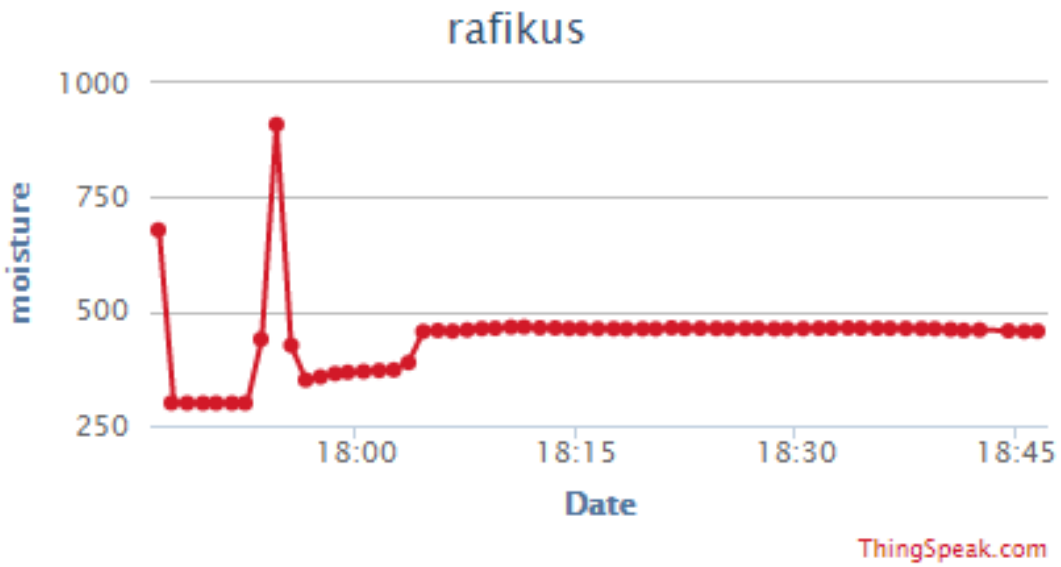


- Move to API keys and copy the URL in Update a Channel Feed. E.g.,

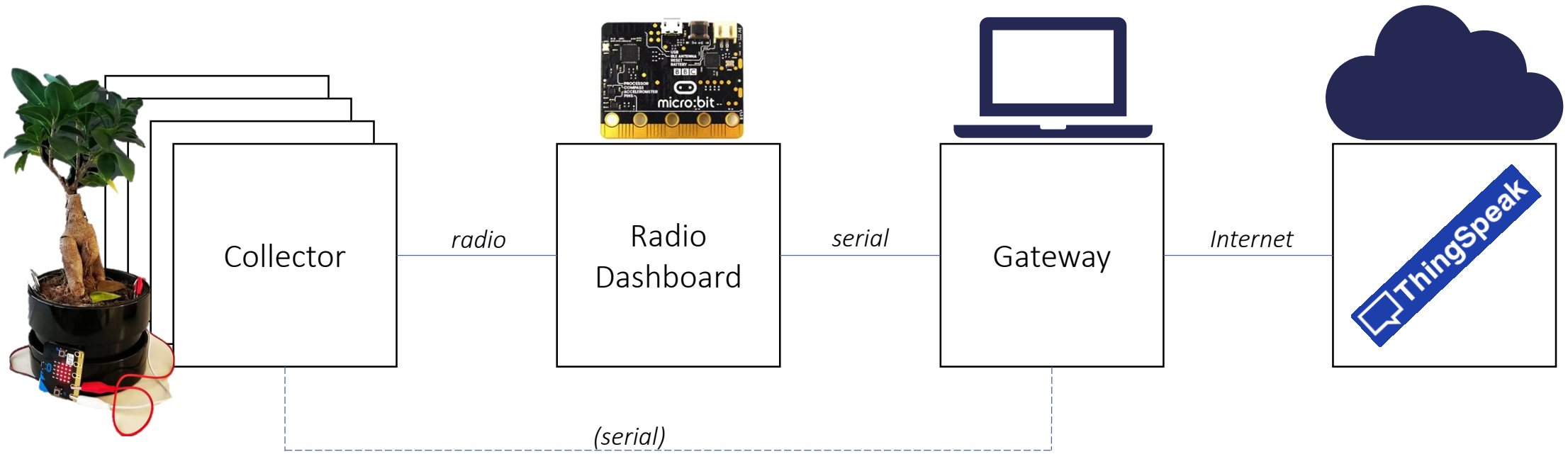
```
https://api.thingspeak.com/update?api_key=<API_KEY>
```

Main

- We will first deploy an IoT+Cloud application, in which data is streamed every second to *ThingSpeak*. Complete **`main_template.js`**.
- Burn the firmware into the microbit and open *ThingSpeak*!
- Run `node main.js`

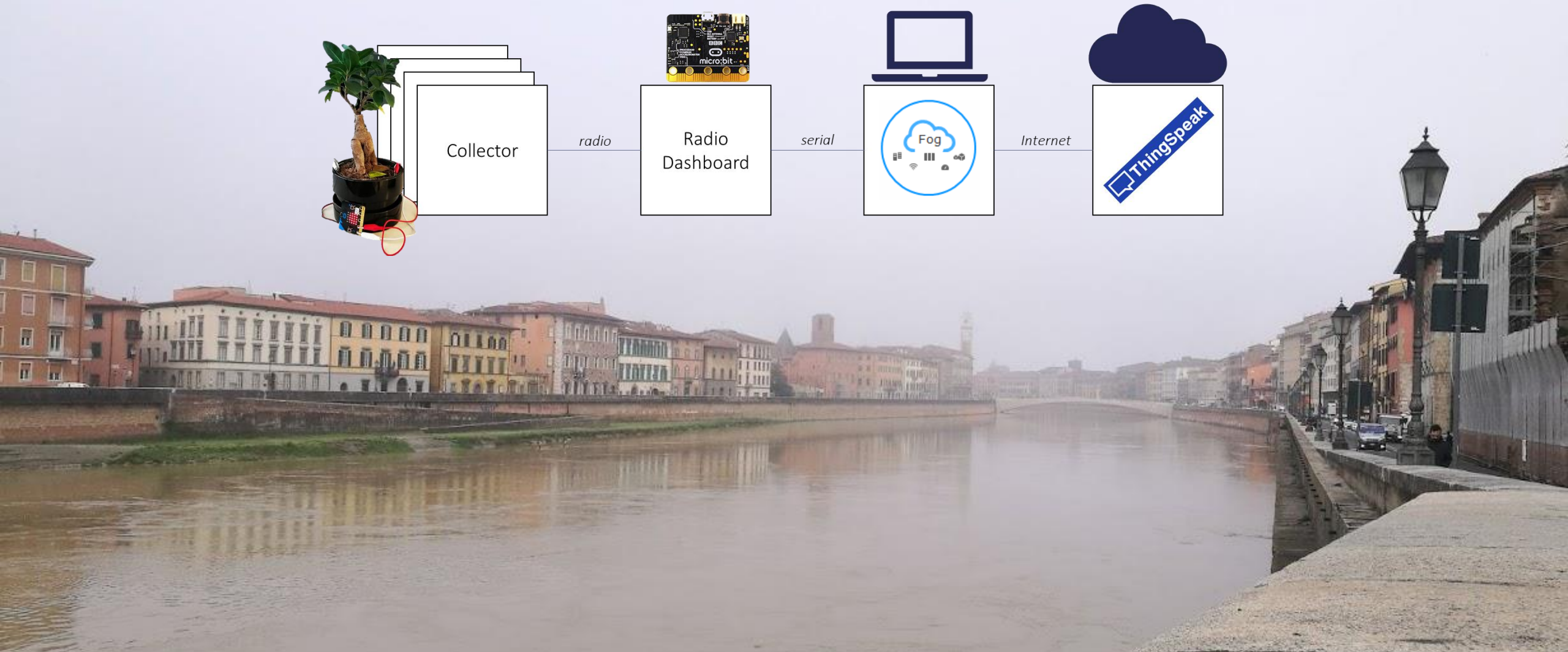


BonsaiFog App

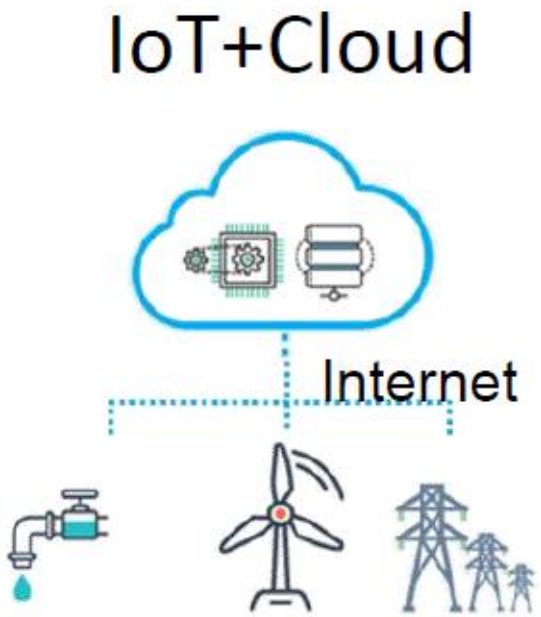


https://github.com/di-unipi-socc/bonsaifog

Where is the Fog?

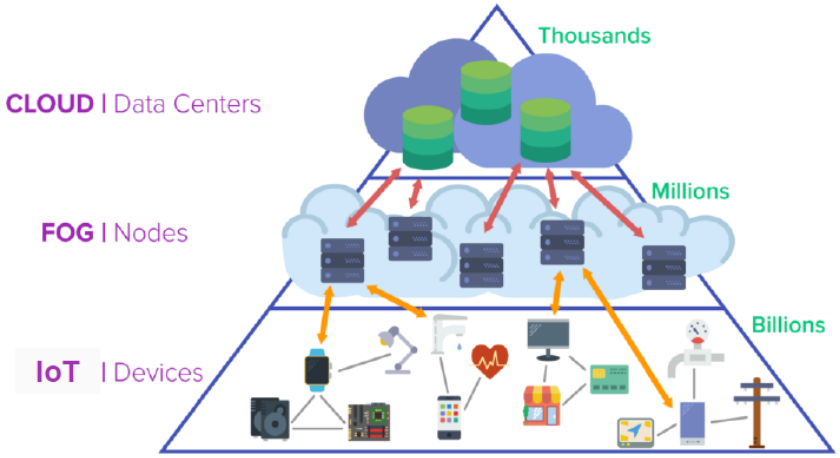


Recap



- Send data to Cloud for processing
- Huge computing power but
- Mandatory connectivity
- High latencies
- Bandwidth bottleneck

- The Cloud alone cannot support the **IoT momentum**
- Need to **filter** and **process** data *before* the Cloud



Fog computing aims at extending the Cloud towards the IoT to better support **latency-sensitive** and **bandwidth-hungry** IoT applications

What can we do?

Idea #1

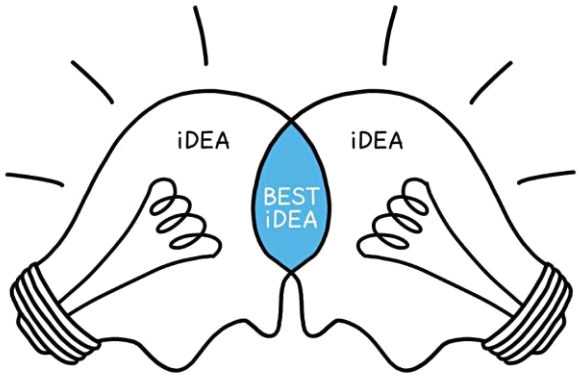
Aggregate data and send an average every 10 seconds.

Implement Idea #1

- Write a new `fogFun()` that is input to `startFog(f)` and uses push data to send to the Cloud the average of the measured data in the last UPDATE_TIME interval.
- Modify the receive handler to accumulate values in `sum` and count the number `n` of received values.

```
function fogFun() {
//send the sum of received values divided by number n of values
//reset variables sum and n
}
```

Idea #2



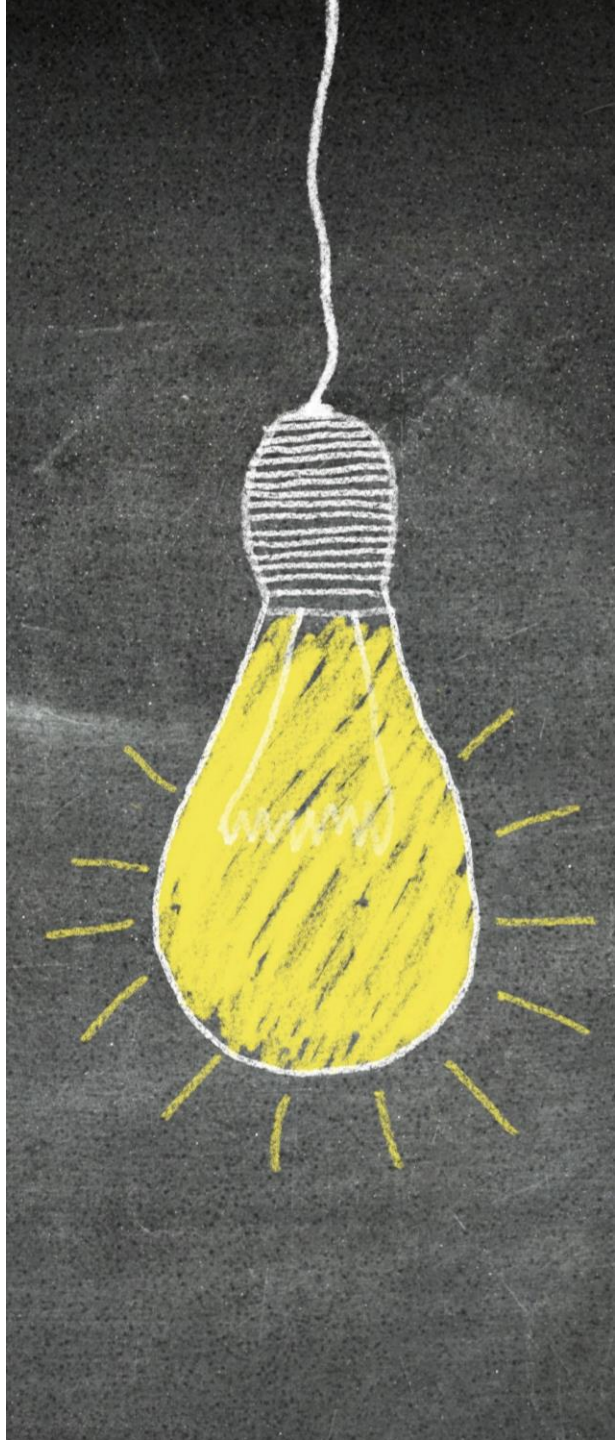
Send data only if the difference between previous average is greater than a threshold.

Problem…

- Even though no big change is registered, maybe we want to keep our plot up-to-date.
- Then what?

Idea #3

Send data only if the difference between previous average is greater than a threshold. Send it anyhow, if no data hasn’t been sent for 1 hour.



Concluding Remarks

First ***hands-on experience*** and ***active learning***

- Practically understand Fog computing.
- Show differences with alternative deployment models.

Context- & location-awareness

Low latency & bandwidth savings

Pervasiveness & geo-distribution

Fog & Things Mobility

Heterogeneity of devices