Universidade do Minho
Departamento de Informática

SENSORIZAÇÃO E AMBIENTE

MESTRADO EM ENGENHARIA INFORMÁTICA, 1º ANO - Perfil SI

Universidade do Minho
Departamento de Informática

Soft/Physical Sensors

Agenda

- Soft Sensors
  - Mobile
- Physical Sensors
  - Arduino-type Boards
- Hands On

Universidade do Minho
Departamento de Informática

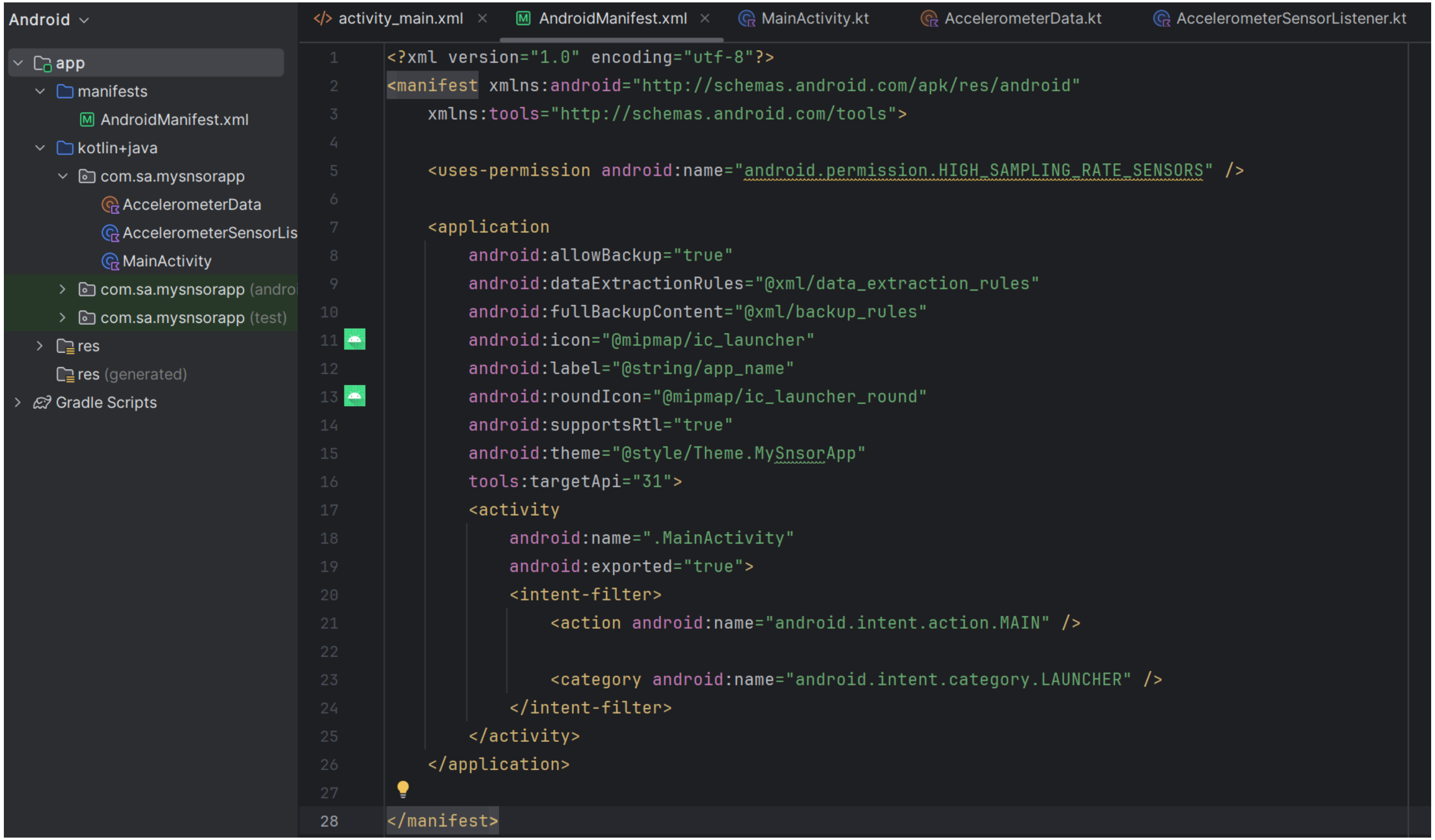
Soft Sensors

Mobile

Android App for Sensors

- Problems with AndroidStudio emulators? Edit user's permissions in `AndroidManifest.xml` with

  `<uses-permission android:name="android.permission.HIGH_SAMPLING_RATE_SENSORS"/>`

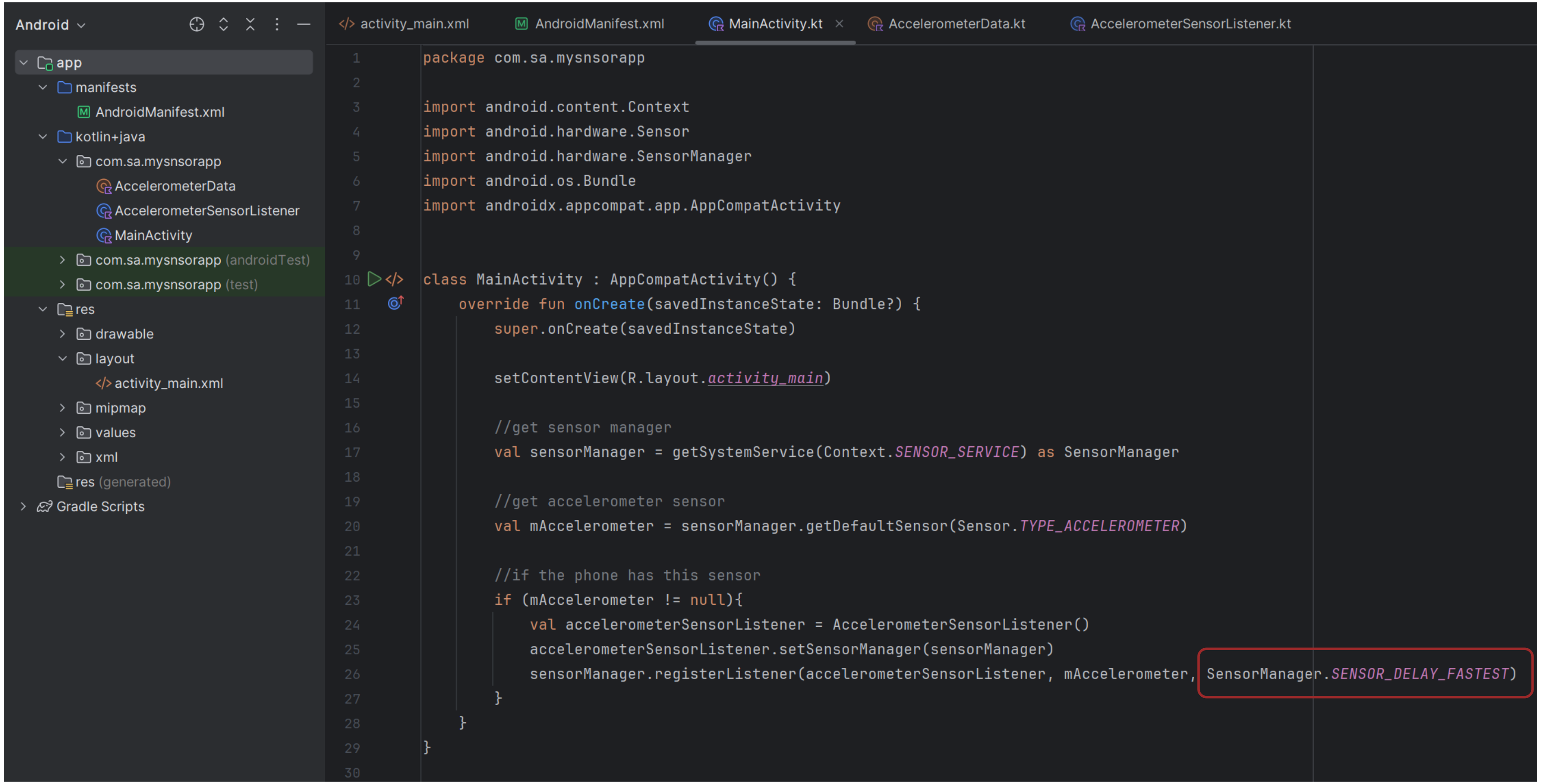


```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:tools="http://schemas.android.com/tools">

    <uses-permission android:name="android.permission.HIGH_SAMPLING_RATE_SENSORS" />

    <application
        android:allowBackup="true"
        android:dataExtractionRules="@xml/data_extraction_rules"
        android:fullBackupContent="@xml/backup_rules"
        android:icon="@mipmap/ic_launcher"
        android:label="@string/app_name"
        android:roundIcon="@mipmap/ic_launcher_round"
        android:supportsRtl="true"
        android:theme="@style/Theme.MySnsorApp"
        tools:targetApi="31">
        <activity
            android:name=".MainActivity"
            android:exported="true">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER" />
            </intent-filter>
        </activity>
    </application>

</manifest>
```

@Guilheme Barbosa

Point I - Sampling Rate

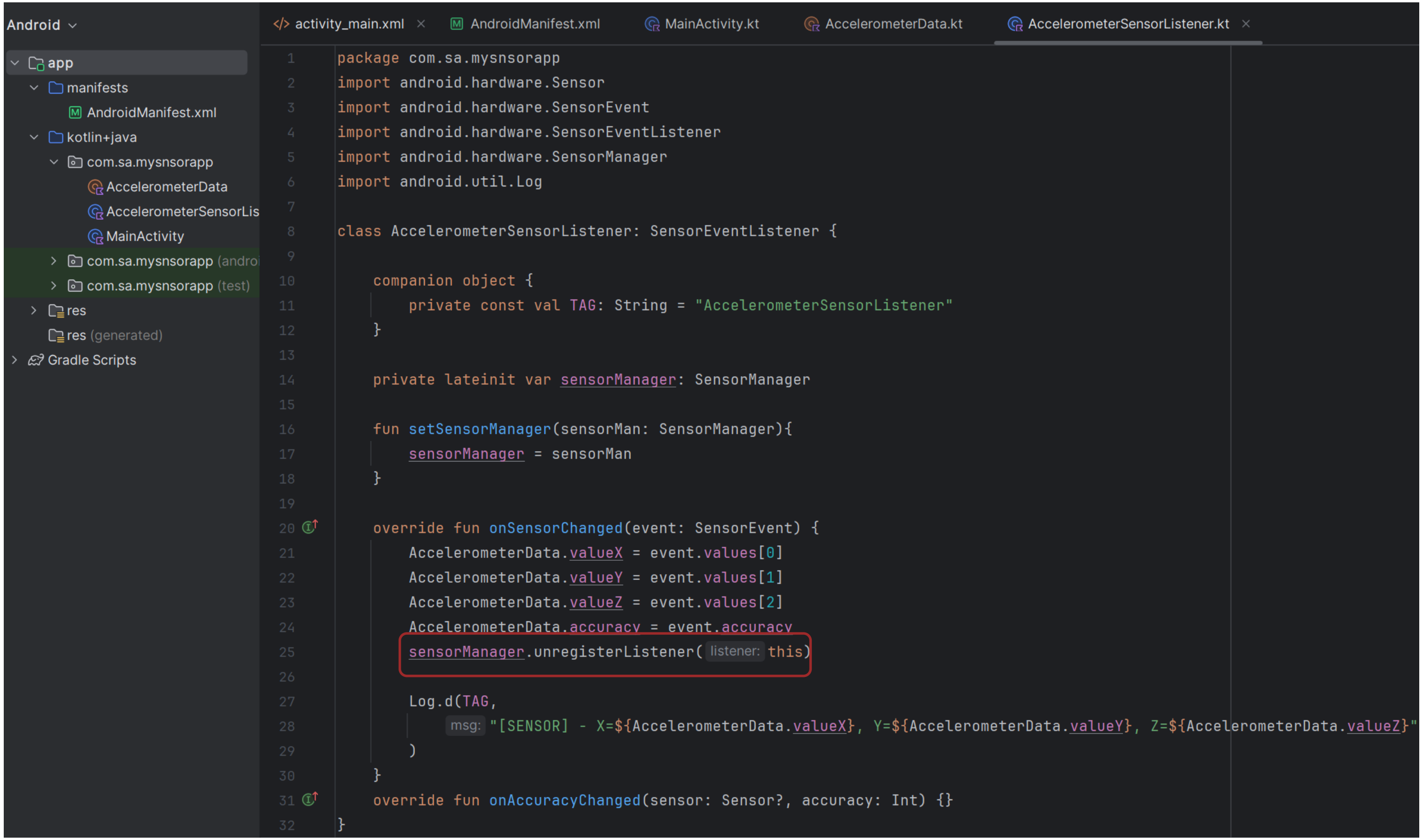
- **SensorManager** lets you access the device's sensors. The `SENSOR_DELAY_FASTEST` tells the app to get sensor data as fast as possible - you need user permission to use such sampling rate. Change to `SENSOR_DELAY_NORMAL`, for example.



```kotlin
package com.sa.mysnsorapp

import android.content.Context
import android.hardware.Sensor
import android.hardware.SensorManager
import android.os.Bundle
import androidx.appcompat.app.AppCompatActivity


class MainActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

        setContentView(R.layout.activity_main)

        //get sensor manager
        val sensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager

        //get accelerometer sensor
        val mAccelerometer = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)

        //if the phone has this sensor
        if (mAccelerometer != null){
            val accelerometerSensorListener = AccelerometerSensorListener()
            accelerometerSensorListener.setSensorManager(sensorManager)
            sensorManager.registerListener(accelerometerSensorListener, mAccelerometer, SensorManager.SENSOR_DELAY_FASTEST)
        }
    }
}
```

Point II - Single sample collection

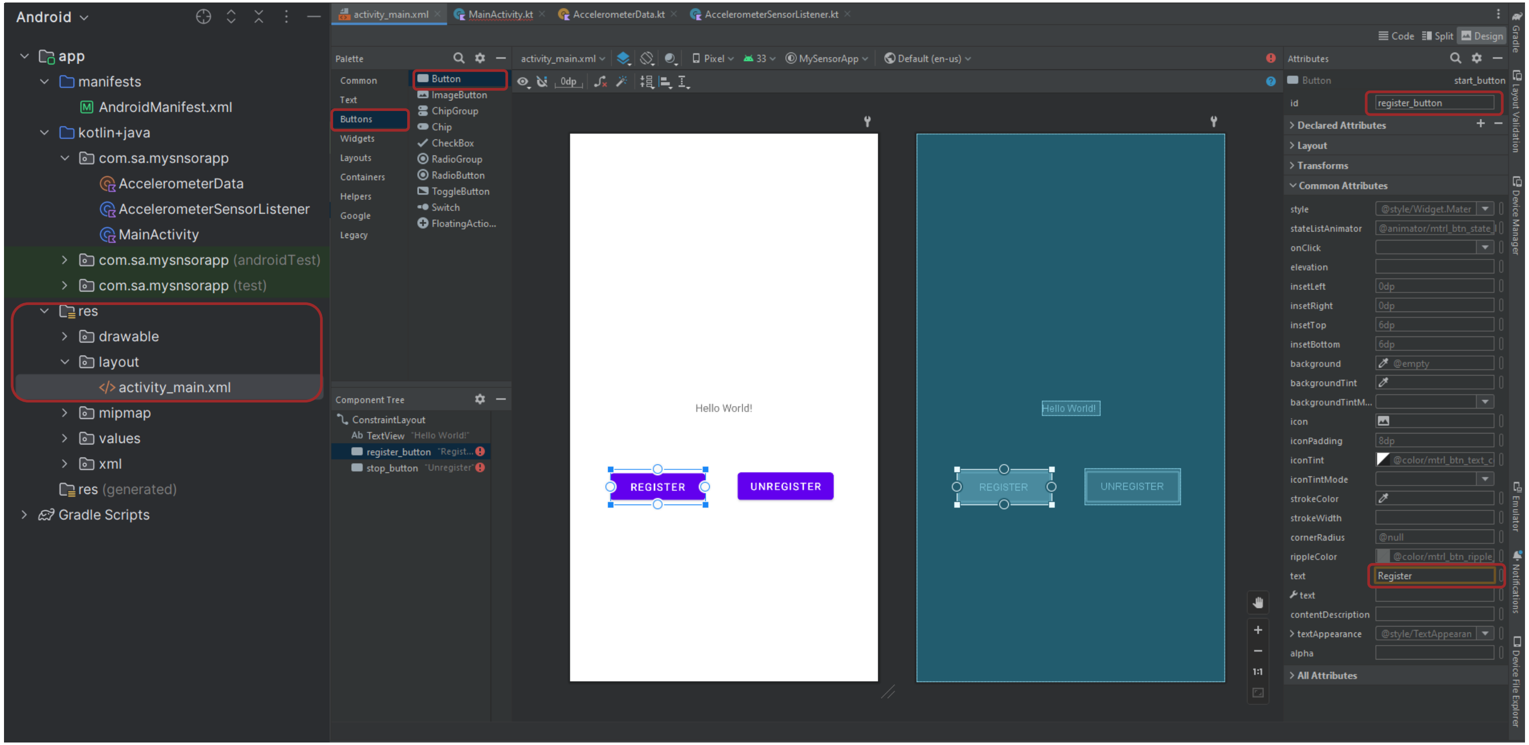
- The app will only collect **one single sample** from the accelerometer sensor. Why? Because we are unregistering the listener as soon as a sample is obtained.



```kotlin
package com.sa.mysnsorapp
import android.hardware.Sensor
import android.hardware.SensorEvent
import android.hardware.SensorEventListener
import android.hardware.SensorManager
import android.util.Log

class AccelerometerSensorListener: SensorEventListener {

    companion object {
        private const val TAG: String = "AccelerometerSensorListener"
    }

    private lateinit var sensorManager: SensorManager

    fun setSensorManager(sensorMan: SensorManager){
        sensorManager = sensorMan
    }

    override fun onSensorChanged(event: SensorEvent) {
        AccelerometerData.valueX = event.values[0]
        AccelerometerData.valueY = event.values[1]
        AccelerometerData.valueZ = event.values[2]
        AccelerometerData.accuracy = event.accuracy
        sensorManager.unregisterListener( listener: this)

        Log.d(TAG,
            msg: "[SENSOR] - X=${AccelerometerData.valueX}, Y=${AccelerometerData.valueY}, Z=${AccelerometerData.valueZ}"
        )
    }
    override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {}
}
```

Point III - Enabling/Disabling Sensors

- To solve this, create **two new buttons** in your **main activity**: one to **register** and another to **unregister** the listener.



Button start_button
id: register_button

style: @style/Widget.Mater
stateListAnimator: @animator/mtrl_btn_state
onClick:
elevation:
insetLeft: 0dp
insetRight: 0dp
insetTop: 6dp
insetBottom: 6dp
background: @empty
backgroundTint:
backgroundTintM...:
icon:
iconPadding: 8dp
iconTint: @color/mtrl_btn_text_c
iconTintMode:
strokeColor:
strokeWidth:
cornerRadius: @null
rippleColor: @color/mtrl_btn_ripple
text: Register
text:
contentDescription:
textAppearance: @style/TextAppearan
alpha:

Point III - Enabling/Disabling Sensors

- In your **main activity**, respond to click events. Something such as:

```
//…

    // inside the onCreate method
    findViewById<Button>(R.id.register_button).setOnClickListener {
        Log.d("BUTTON", "User clicked the register button.")
        //register the listener
        //…
    }
    findViewById<Button>(R.id.unregister_button).setOnClickListener {
        Log.d("BUTTON", "User clicked the unregister button.")
        //unregister the listener
        //…
    }

//…
```

https://developer.android.com/develop/ui/views/components/button

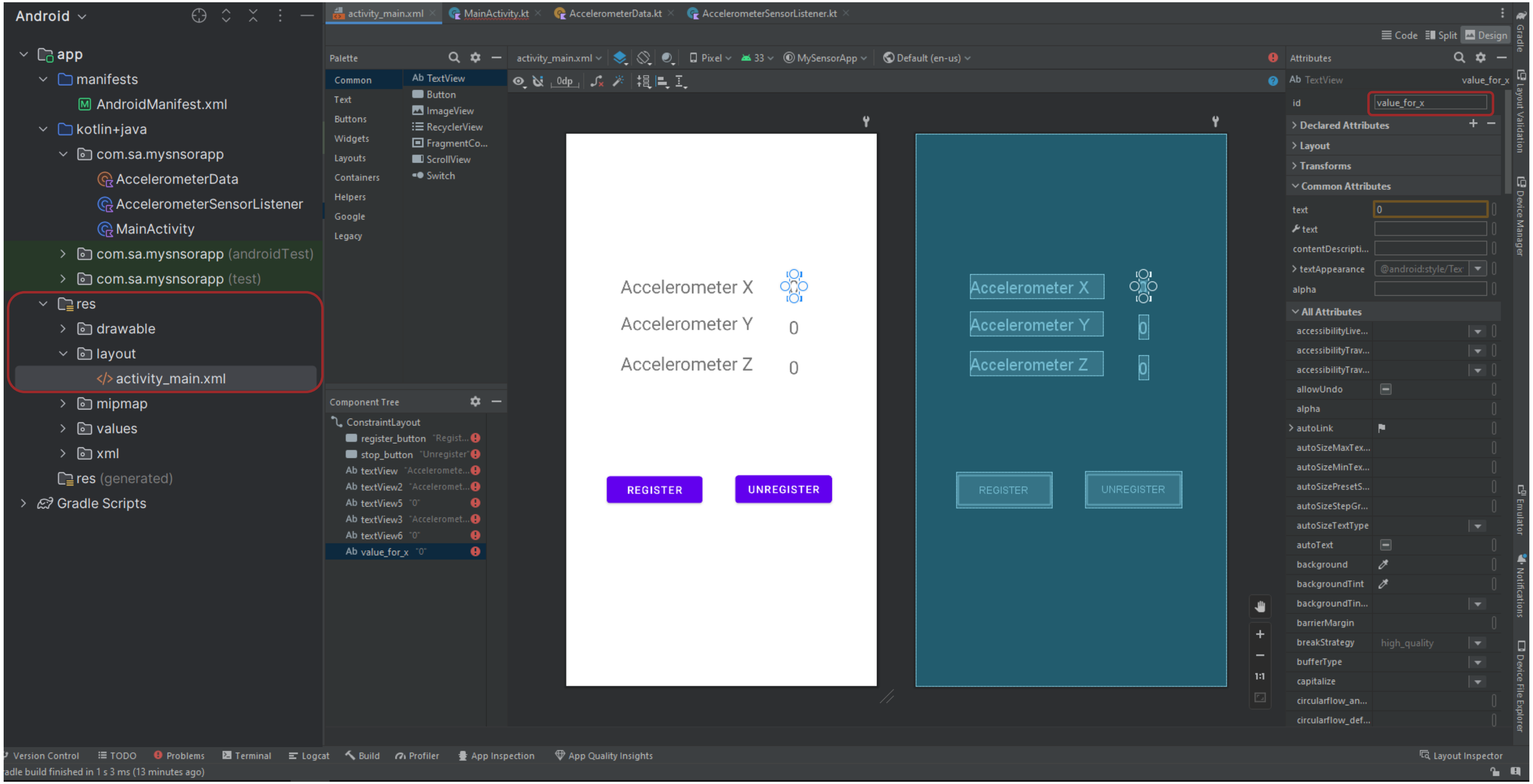
Point III - Enabling/Disabling Sensors

```
// inside the onCreate method
    findViewById<Button>(R.id.register_button).setOnClickListener {
        // get the sensor manager
        sensorManager = getSystemService(Context.SENSOR_SERVICE) as SensorManager
        // get the accelerometer sensor
        val mAccelerometer = sensorManager.getDefaultSensor(Sensor.TYPE_ACCELEROMETER)
        // if the smartphone has this sensor
        if (mAccelerometer != null){
            accelerometerSensorListener = AccelerometerSensorListener()
            accelerometerSensorListener.setSensorManager(sensorManager)
            sensorManager.registerListener(AccelerometerSensorListener, mAccelerometer,
                                                                SensorManager.SENSOR_DELAY_NORMAL)
            }
    }
    findViewById<Button>(R.id.unregister_button).setOnClickListener {
        sensorManager.unregisterListener(accelerometerSensorListener)
    }
```

https://developer.android.com/develop/ui/views/components/button

Point IV - LiveData/ViewModel

- How to **see the values of the accelerometer** in our main activity? Add some **TextViews** and update its value whenever new data is collected.



id: value_for_x

text: 0

adle build finished in 1 s 3 ms (13 minutes ago)

Point IV - LiveData/ViewModel

- Then use **LiveData** to listen to updates to the UI. First our `ViewModel` class:

```
class AccelerometerViewModel: ViewModel() {

    // Create a LiveData object with a AccelerometerData object
    val currentAccelerometerData: MutableLiveData<AccelerometerData> by lazy {
        MutableLiveData<AccelerometerData>()
    }

}
```

https://developer.android.com/topic/libraries/architecture/livedata

Point IV - LiveData/ViewModel

- Then change the value of our **LiveData** (the var `currentAccelerometerData`) every time there is new data obtained from the sensor. Hence, in our `AccelerometerSensorListener` class, we must receive our `ViewModel` and update its value `onSensorChanged`. As such:

```
class AccelerometerSensorListener: SensorEventListener {

    companion object {
        private const val TAG: String = "AccelerometerSensorListener"
    }

    private lateinit var sensorManager: SensorManager
    private lateinit var ourAccelerometerViewModel: AccelerometerViewModel

    fun setSensorManager(sensorMan: SensorManager, aViewModel: AccelerometerViewModel) {
        sensorManager = sensorMan
        ourAccelerometerViewModel = aViewModel
    }
    //...
}
```

https://developer.android.com/topic/libraries/architecture/livedata

Point IV - LiveData/ViewModel

- Then change the value of our **LiveData** (the var `currentAccelerometerData`) every time there is new data obtained from the sensor. Hence, in our `AccelerometerSensorListener` class, we must receive our `ViewModel` and update its value `onSensorChanged`. As such:

```
class AccelerometerSensorListener: SensorEventListener {

    //...
    override fun onSensorChanged(event: SensorEvent) {
        AccelerometerData.valueX = event.values[0]
        AccelerometerData.valueY = event.values[1]
        AccelerometerData.valueZ = event.values[2]
        AccelerometerData.accuracy = event.accuracy
        ourAccelerometerViewModel.currentAccelerometerData.value = AccelerometerData
    }

    override fun onAccuracyChanged(sensor: Sensor?, accuracy: Int) {}

}
```

https://developer.android.com/topic/libraries/architecture/livedata

Point IV - LiveData/ViewModel

- Then use **LiveData** to listen to updates to the UI. Then, look to our **LiveData**.

```
// in build.gradle.kts file, go to the dependencies (at the bottom) and add the following lib:
implementation("androidx.fragment:fragment-ktx:1.8.6")


// in MainActivity.kt
class MainActivity : AppCompatActivity() {

  private val aViewModel: AccelerometerViewModel by viewModels()
  //…
    // … then, inside the onCreate method
    // create the observer which updates the UI.
    val accelerometerObserver = Observer<AccelerometerData> { accSample ->
      findViewById<TextView>(R.id.textview_x).text = accSample.valueX.toString()
      findViewById<TextView>(R.id.textview_y).text = accSample.valueY.toString()
      findViewById<TextView>(R.id.textview_z).text = accSample.valueZ.toString()
    }
    // observe the LiveData, passing in this activity as the LifecycleOwner and the observer
    aViewModel.currentAccelerometerData.observe(this, accelerometerObserver)
  }
}
```

https://developer.android.com/topic/libraries/architecture/livedata

Universidade do Minho
Departamento de Informática

Physical Sensors

Arduino-type Boards

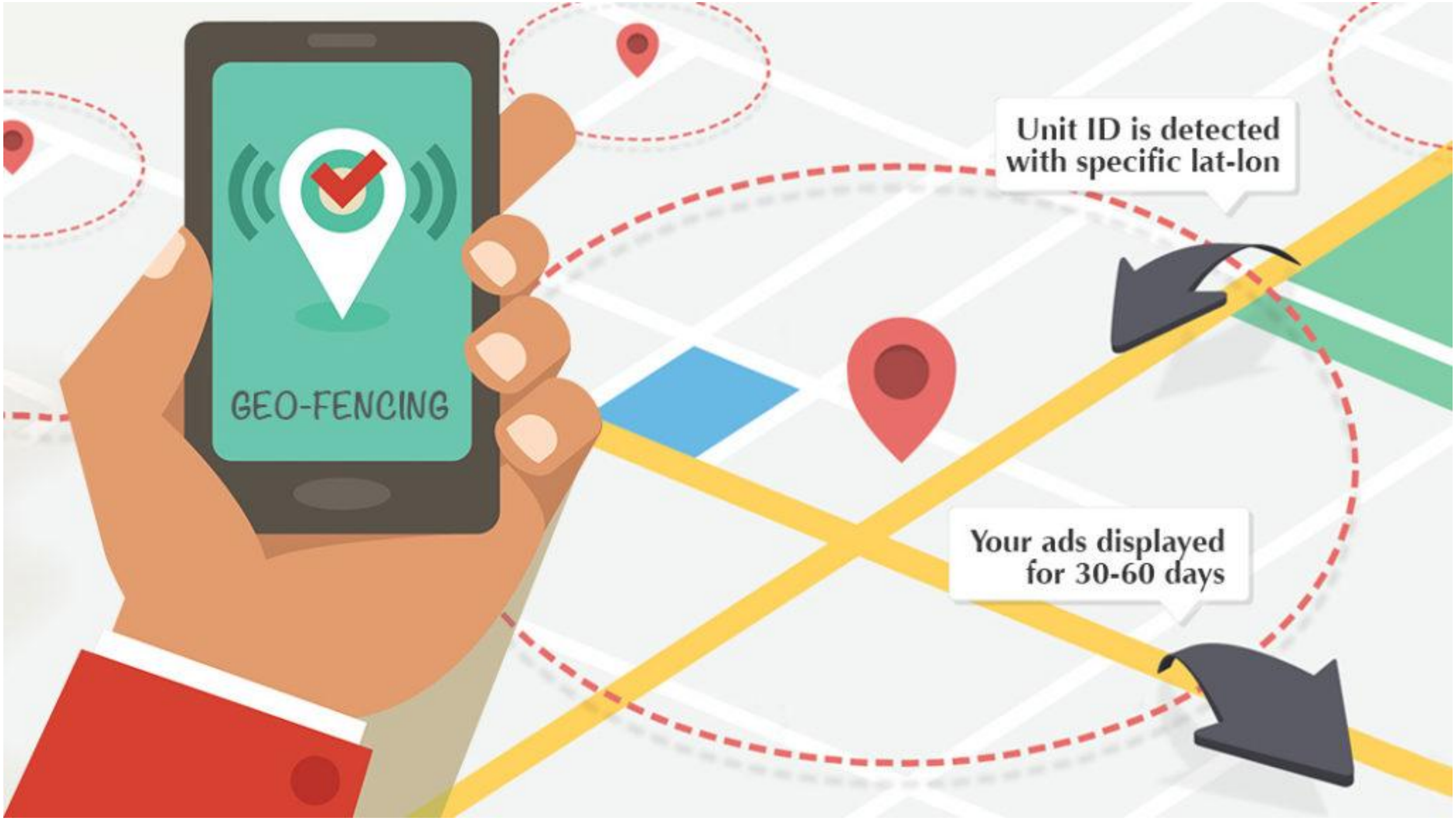
Beacons

- It is a **Bluetooth-based sensor** with low-cost and low-power transmitters (a Bluetooth Smart/LE signal), which **notify Bluetooth devices** of one's presence.

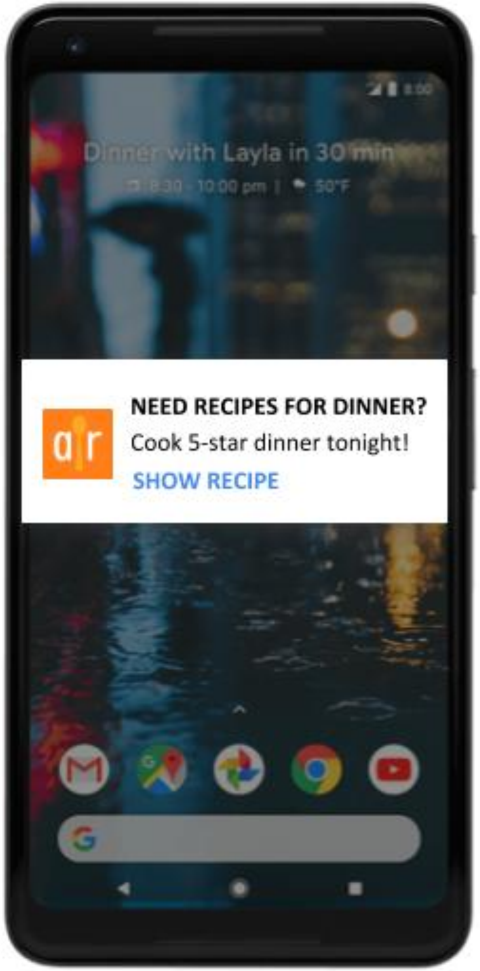
- This signal makes it **possible to identify the beacon** as well as other telemetry information about the receiving device. It has **no user interface or GPS** capabilities.

- The beacon works as such:
  - periodically wakes up;
  - transmits a Bluetooth Low Energy (BLE) signal;
  - returns to a low-power state.

Beacon: Case Studies


GEO-FENCING
Unit ID is detected with specific lat-lon
Your ads displayed for 30-60 days

Beacon: Case Studies



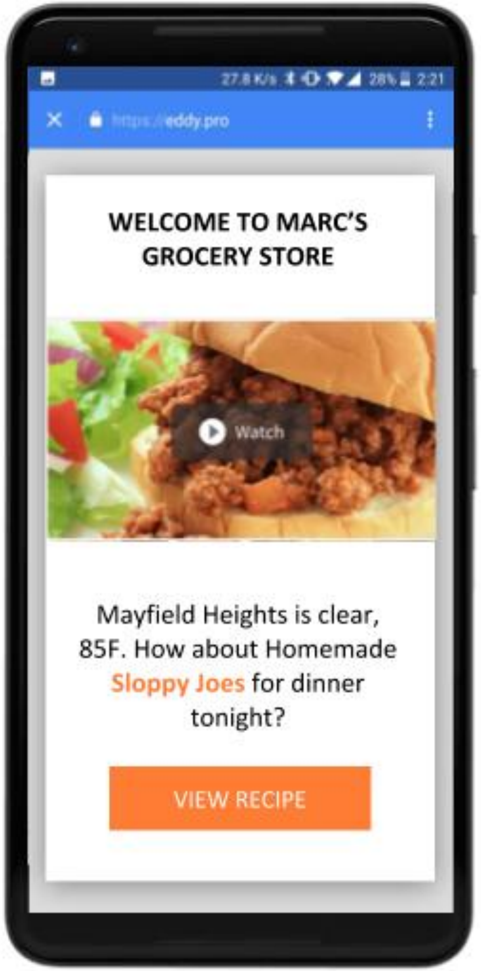
Dinner with Layla in 30 min

8:30 - 10:00 pm | 50°F

NEED RECIPES FOR DINNER?

Cook 5-star dinner tonight!

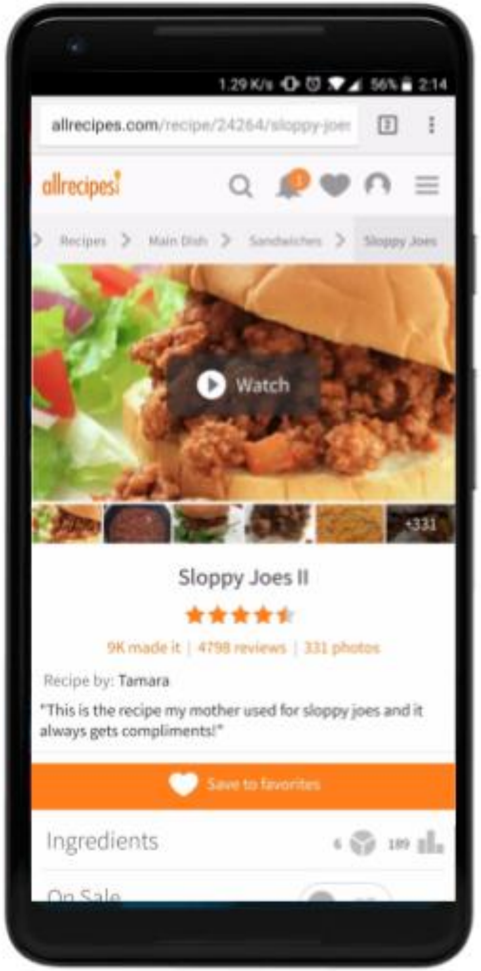
SHOW RECIPE



WELCOME TO MARC'S GROCERY STORE

Mayfield Heights is clear, 85F. How about Homemade Sloppy Joes for dinner tonight?

VIEW RECIPE



Sloppy Joes II

9K made it | 4798 reviews | 331 photos

Recipe by: Tamara

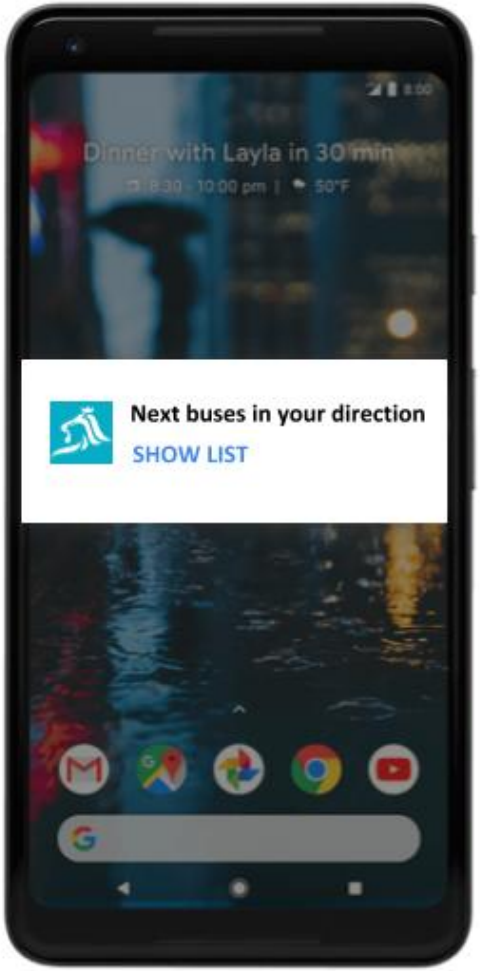
"This is the recipe my mother used for sloppy joes and it always gets compliments!"

Save to favorites

Ingredients

On Sale

Beacon: Case Studies

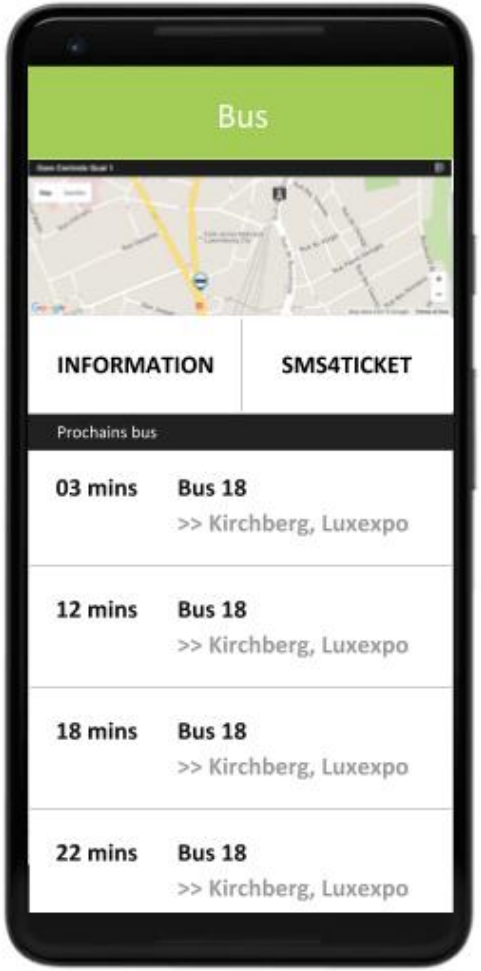


Dinner with Layla in 30 min

8:30 - 10:00 pm | 50°F

Next buses in your direction

SHOW LIST



Bus

INFORMATION | SMS4TICKET

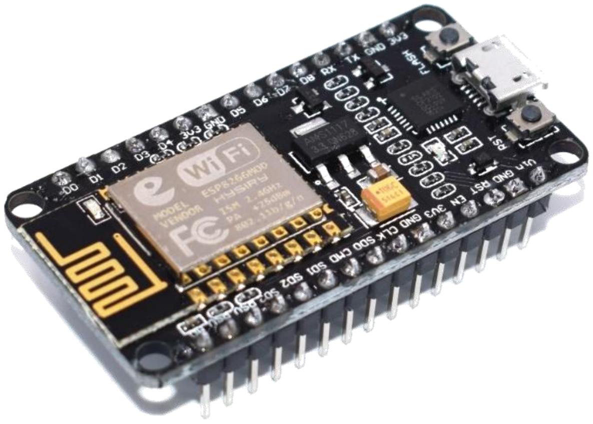
Prochains bus

03 mins	Bus 18 >> Kirchberg, Luxexpo
12 mins	Bus 18 >> Kirchberg, Luxexpo
18 mins	Bus 18 >> Kirchberg, Luxexpo
22 mins	Bus 18 >> Kirchberg, Luxexpo

ESP8266

- It is a **low-power Arduino type board** suitable for IoT that can facilitate the **bridge towards Smart Cities**, removing the need for wired communication and processing.

- It has a very interesting set of features:
  - Wi-Fi capability (2.4 GHz band)
  - 4 MB of flash memory
  - a micro-USB interface
  - a built-in antenna
  - open-source
  - small dimensions (4.8x2.4x0.5cm)
  - low weight (109g)
  - Ultra-Low Power Consumption



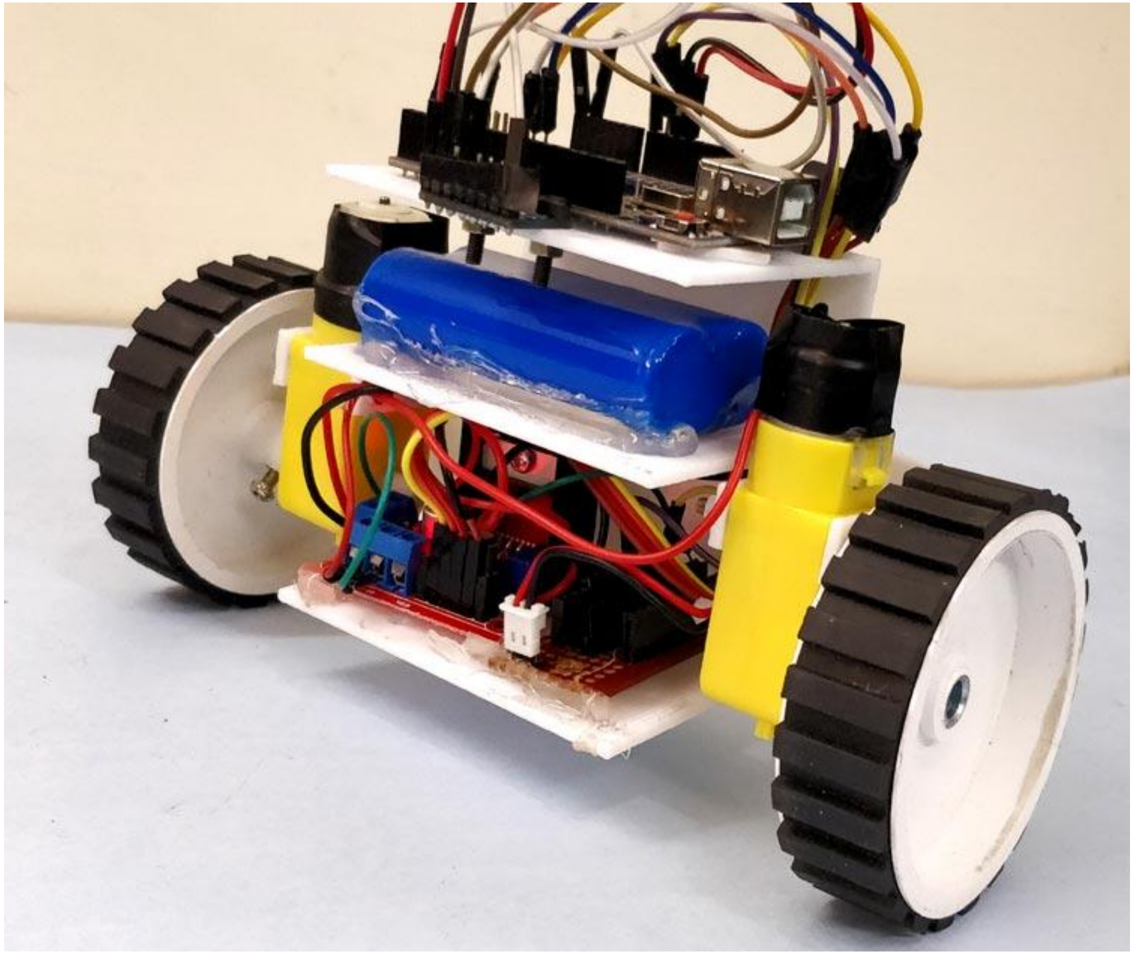
ESP32

- Similar to ESP8266 but with additional features:
  - BLE connectivity (Hybrid Wi-Fi & Bluetooth Chip)
  - Dual-core

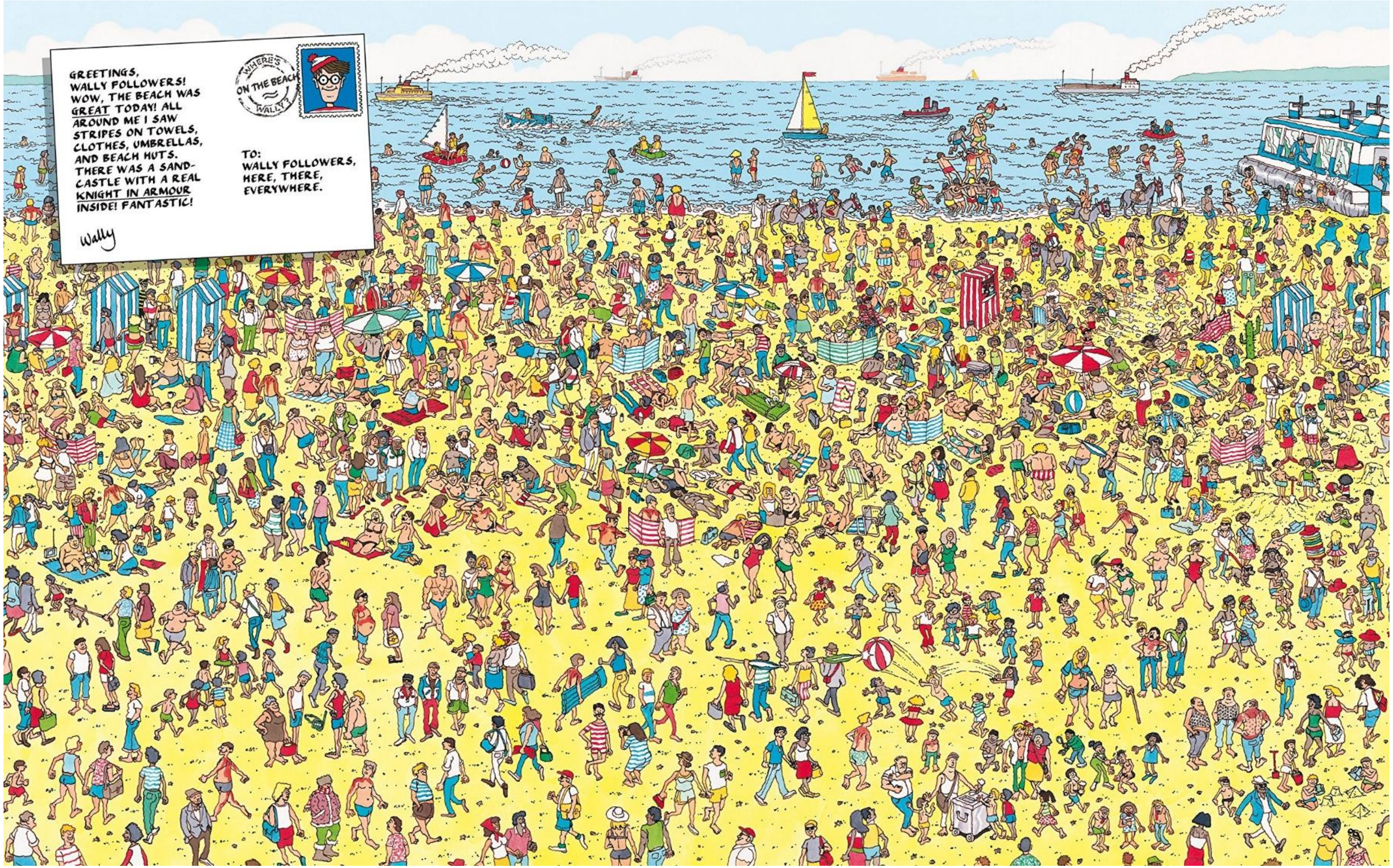
- However, the availability (and documentation) of libraries for the ESP32 is **significantly lower** when compared to the ESP8266.

Arduino(-type) Boards: Case Studies

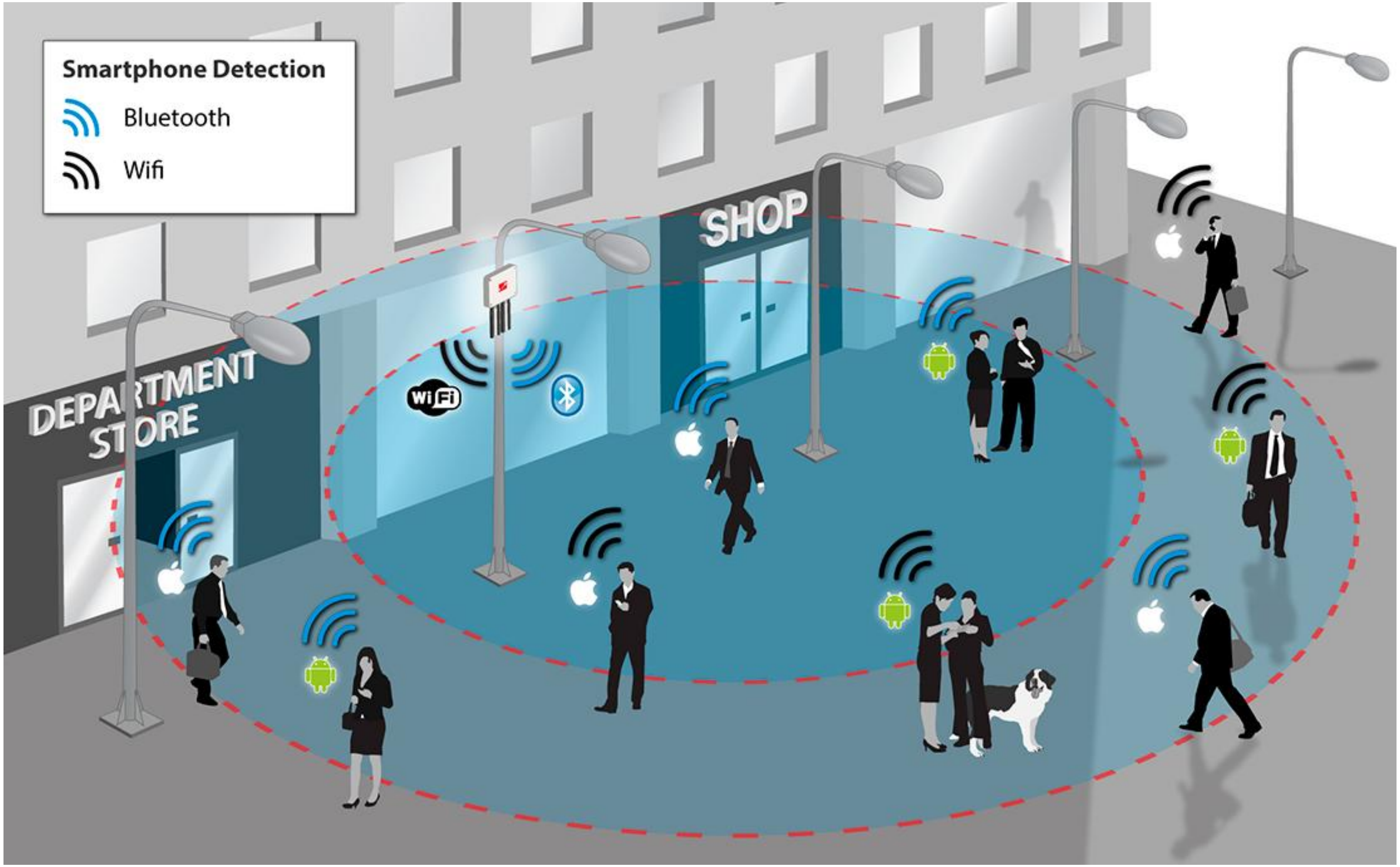
Arduino(-type) Boards: Case Studies



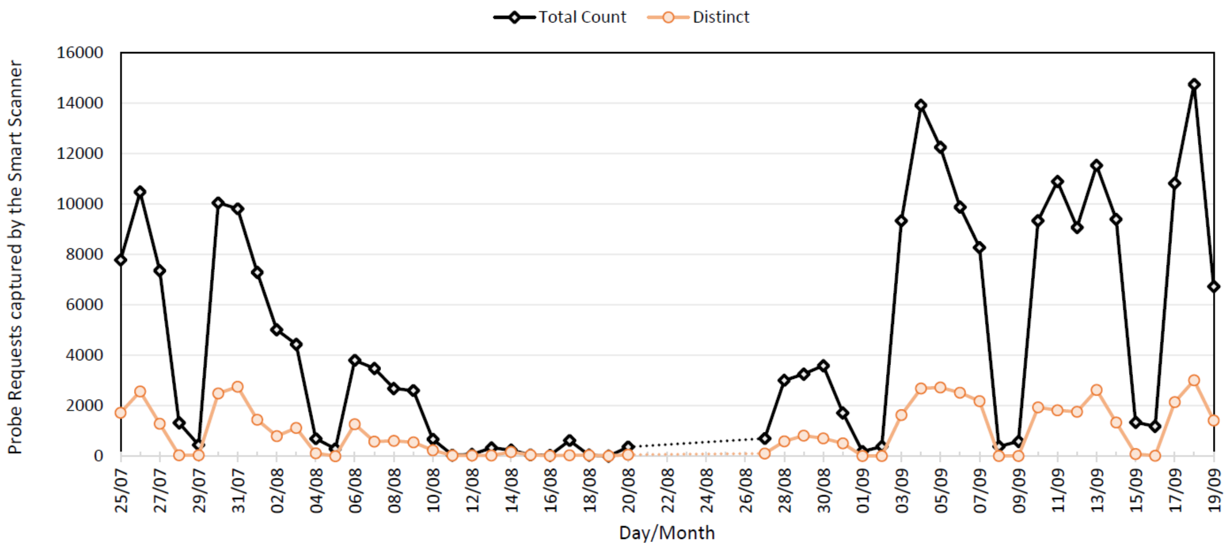
Arduino(-type) Boards: Case Studies

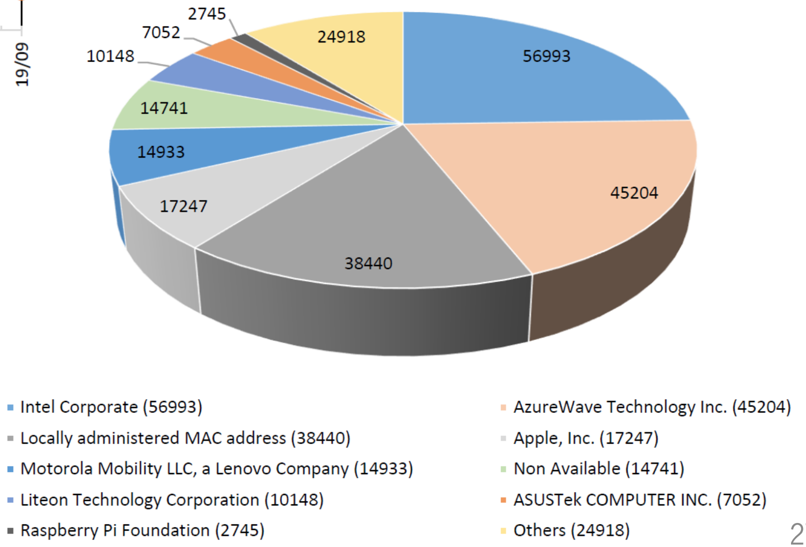


Arduino(-type) Boards: Case Studies



Arduino(-type) Boards: Case Studies

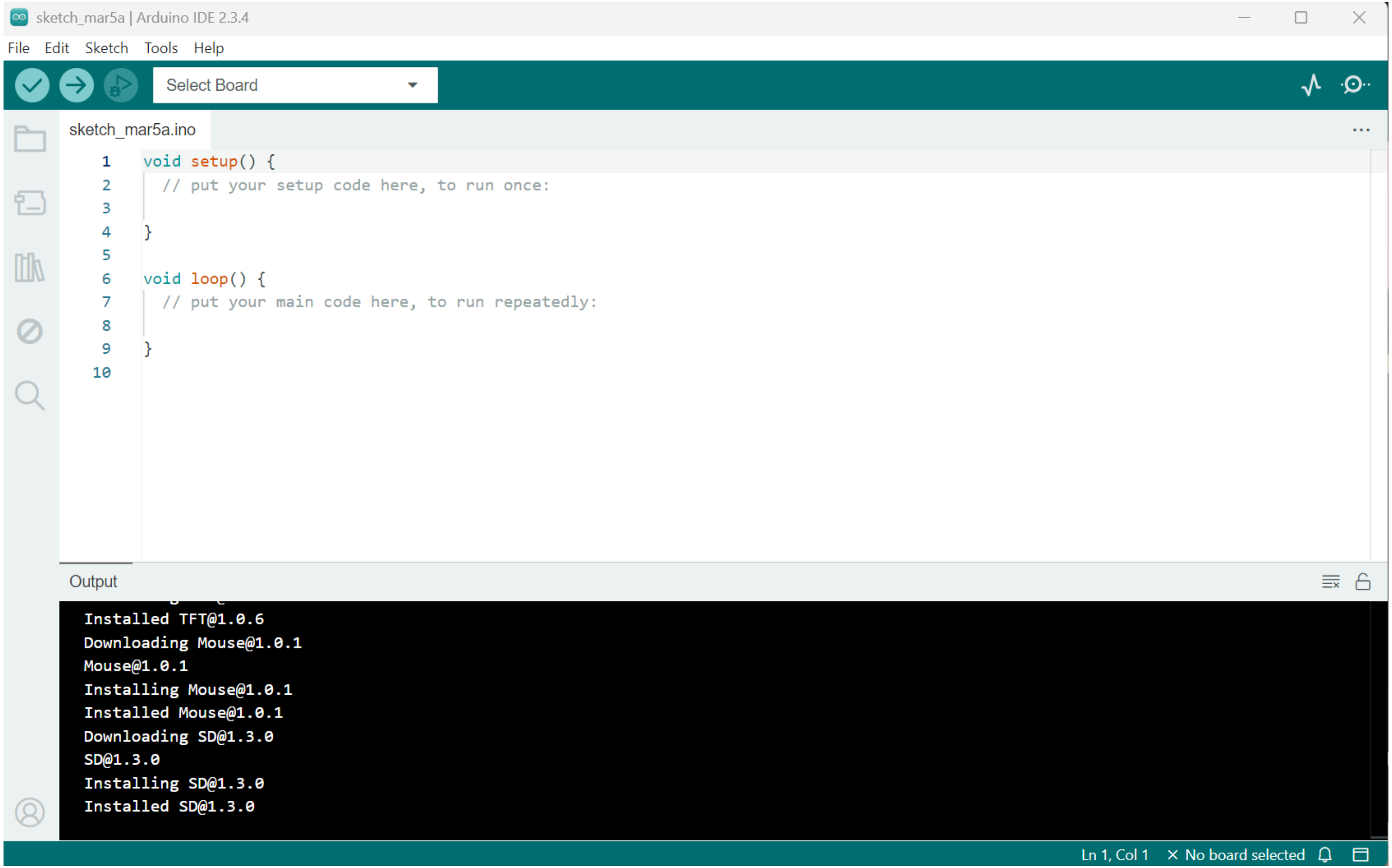




Arduino(-type) Boards: Case Studies

- But also, at…
  - Concerts
  - Races
  - Stores
  - Libraries
  - Football Games
  - Etc.

Arduino IDE: How To



```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}

```

```
Installed TFT@1.0.6
Downloading Mouse@1.0.1
Mouse@1.0.1
Installing Mouse@1.0.1
Installed Mouse@1.0.1
Downloading SD@1.3.0
SD@1.3.0
Installing SD@1.3.0
Installed SD@1.3.0
```

https://www.arduino.cc/en/software

Arduino IDE: How To

- Makes it very easy to develop code and upload it to a board;

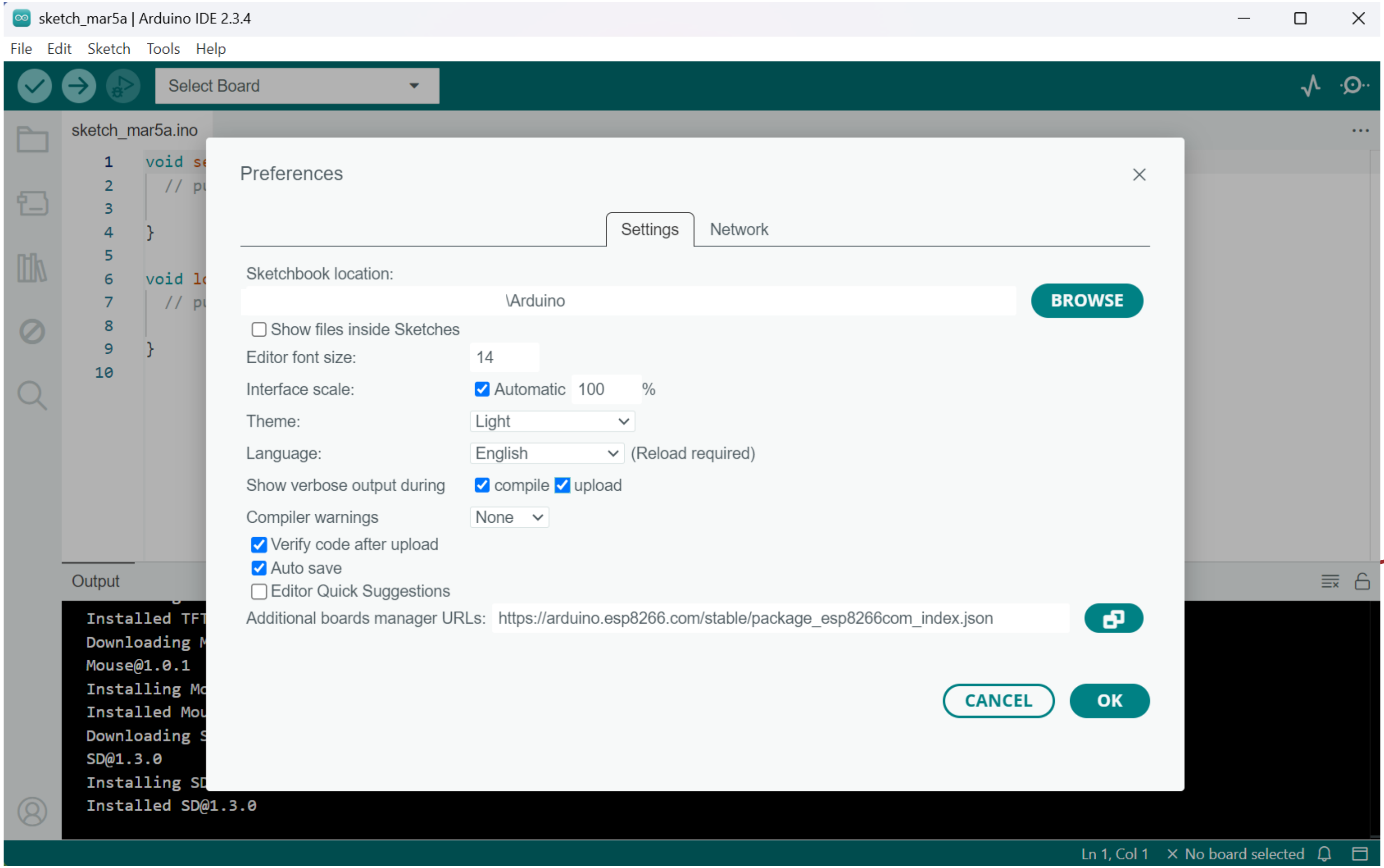
- Provides basic **one-click mechanisms** to compile and **upload sketches**:
  - **Sketch** is the name given to a program developed with this IDE (written in C/C++)

- The nature of the Arduino project facilitated the release of many open-source libraries;

- Programming in the Arduino IDE requires the developer to define, at least, two functions:
  - `setup()` - called once when a sketch starts after powering up or resetting, being used to initialize variables, input and output pin modes, and other libraries required by the sketch;
  - `loop()` - repeatedly executed in the main program until the board is powered off or reset.

Arduino IDE: How To

- Install the board (ESP8266) in Arduino IDE:



sketch_mar5a | Arduino IDE 2.3.4
File Edit Sketch Tools Help
Select Board
sketch_mar5a.ino
Output
Ln 1, Col 1 ✕ No board selected

```
void se
  // pu

}

void lo
  // pu

}
```

```
Installed TFT
Downloading M
Mouse@1.0.1
Installing Mo
Installed Mou
Downloading S
SD@1.3.0
Installing SD
Installed SD@1.3.0
```

Preferences

Settings | Network

Sketchbook location: \Arduino BROWSE

□ Show files inside Sketches

Editor font size: 14

Interface scale: ☑ Automatic 100 %

Theme: Light

Language: English (Reload required)

Show verbose output during ☑ compile ☑ upload

Compiler warnings None

☑ Verify code after upload

☑ Auto save

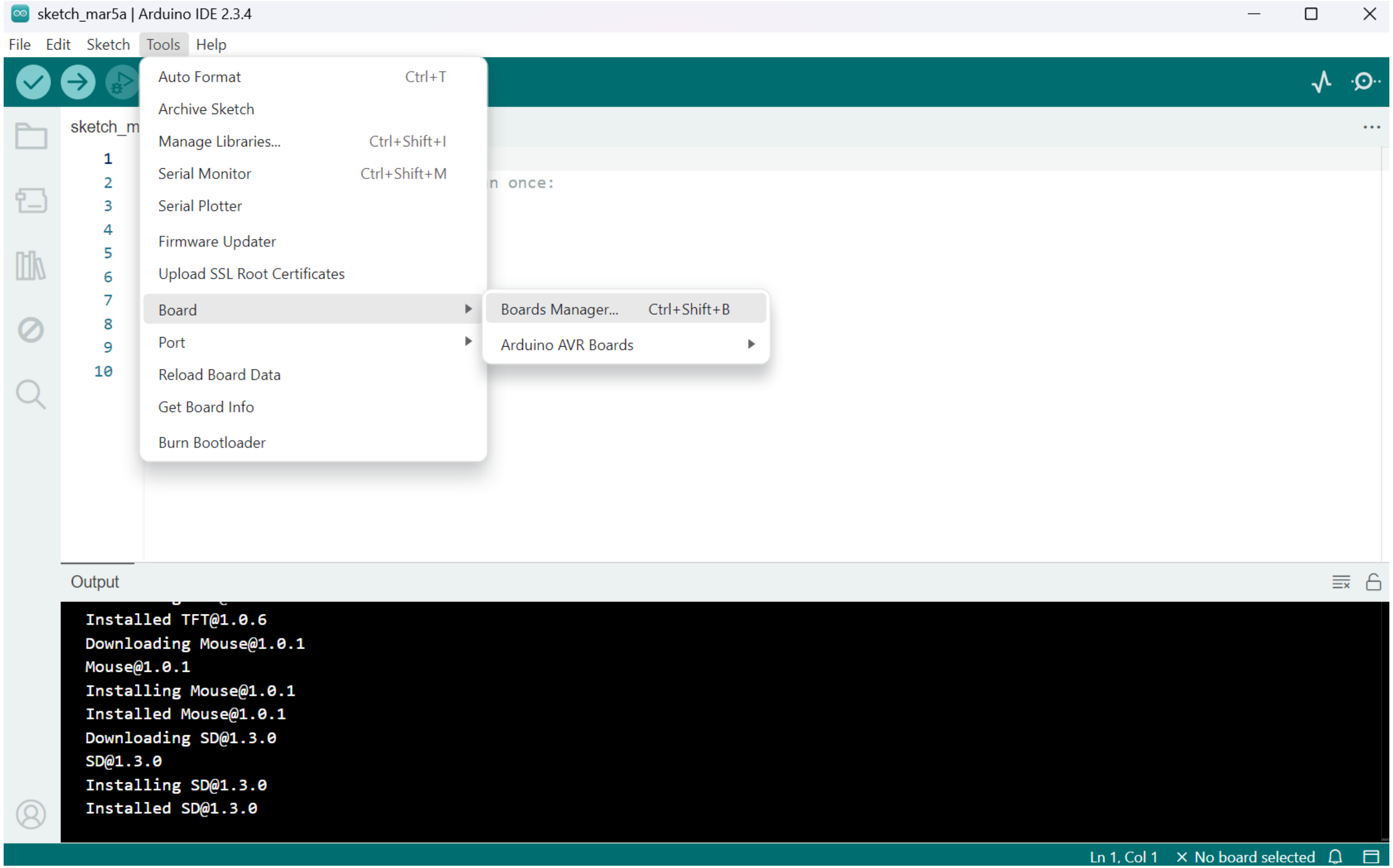
□ Editor Quick Suggestions

Additional boards manager URLs: https://arduino.esp8266.com/stable/package_esp8266com_index.json

CANCEL OK

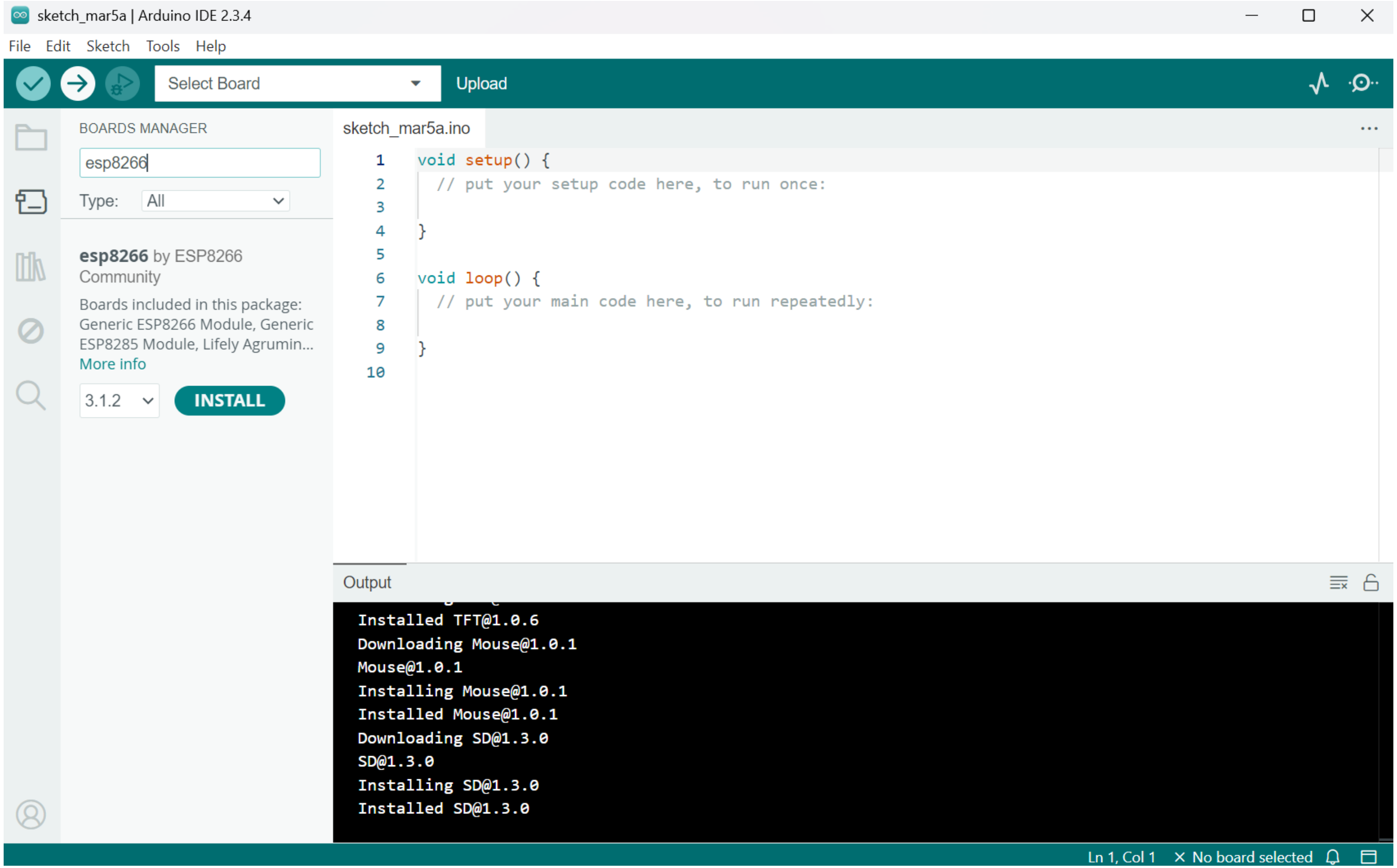
https://arduino.esp8266.com/stable/package_esp8266com_index.json

Arduino IDE: How To



```
Installed TFT@1.0.6
Downloading Mouse@1.0.1
Mouse@1.0.1
Installing Mouse@1.0.1
Installed Mouse@1.0.1
Downloading SD@1.3.0
SD@1.3.0
Installing SD@1.3.0
Installed SD@1.3.0
```

Arduino IDE: How To



esp8266

Type: All

esp8266 by ESP8266 Community

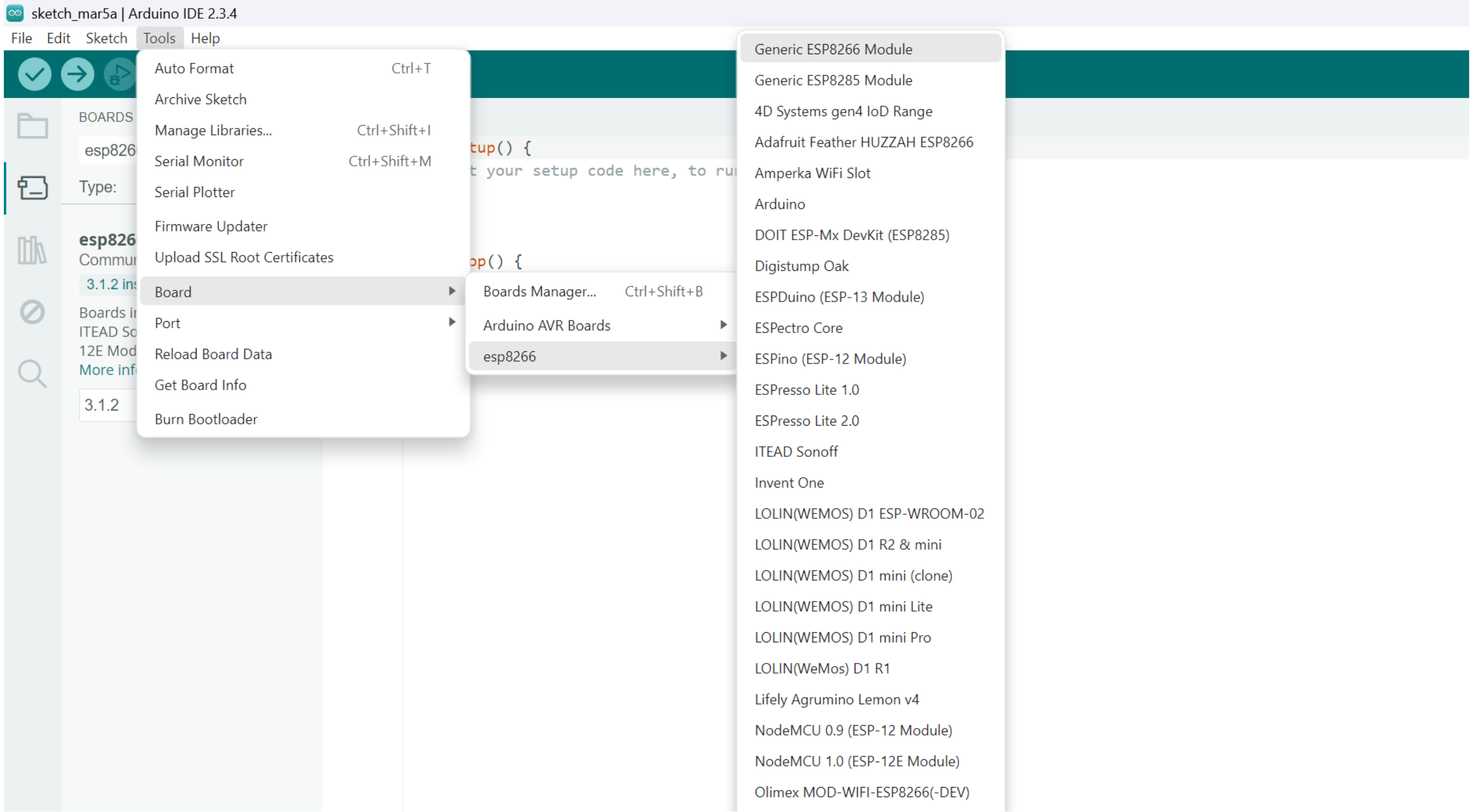
Boards included in this package: Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumin...

3.1.2

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}

```

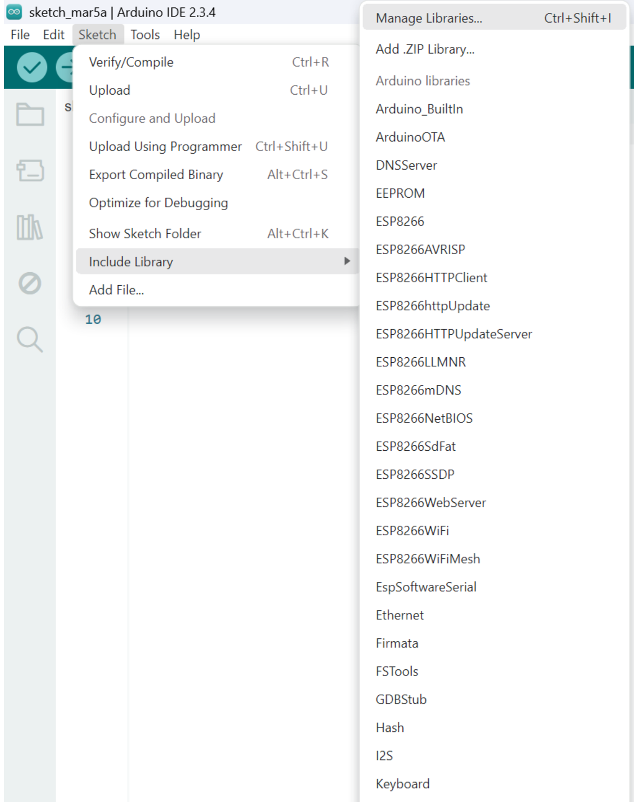
```
Installed TFT@1.0.6
Downloading Mouse@1.0.1
Mouse@1.0.1
Installing Mouse@1.0.1
Installed Mouse@1.0.1
Downloading SD@1.3.0
SD@1.3.0
Installing SD@1.3.0
Installed SD@1.3.0
```

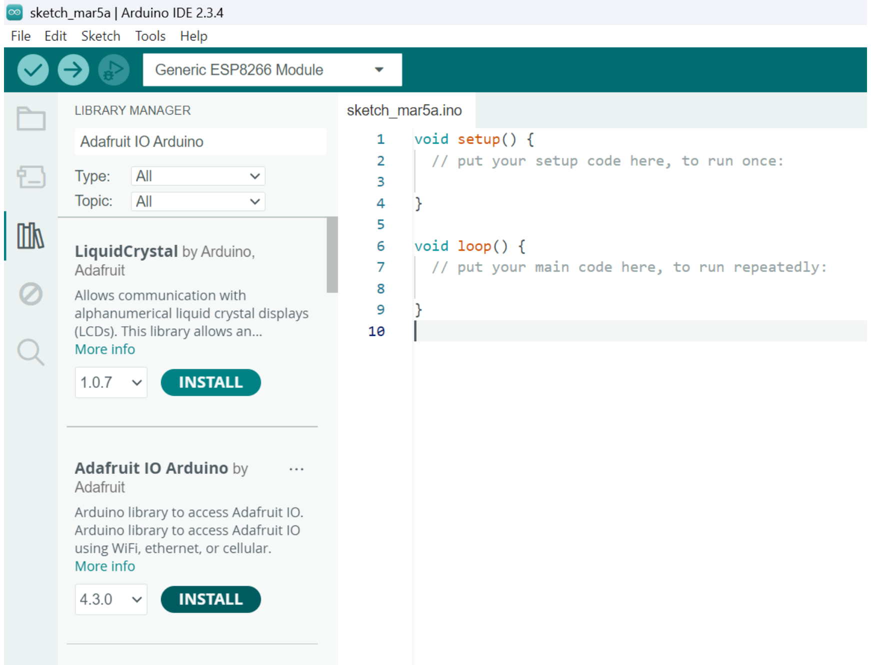
Arduino IDE: How To



Arduino IDE: How To

- Install some libraries: ArduinoHttpClient, Adafruit IO Arduino, Adafruit MQTT, PubSubClient, ArduinoJson, Firebase Arduino





LiquidCrystal by Arduino, Adafruit

Allows communication with alphanumerical liquid crystal displays (LCDs). This library allows an... More info

Adafruit IO Arduino by Adafruit

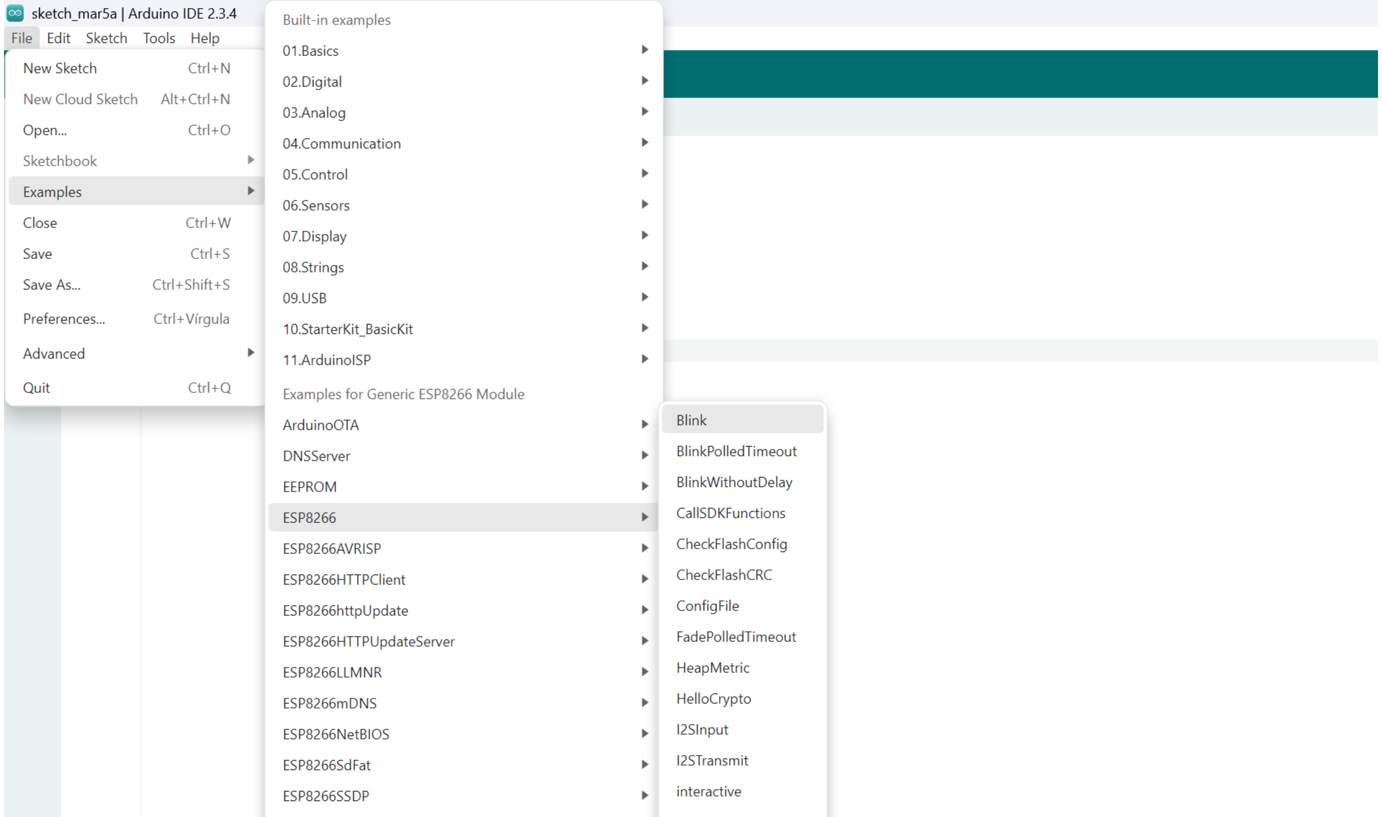
Arduino library to access Adafruit IO. Arduino library to access Adafruit IO using WiFi, ethernet, or cellular. More info

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}

```

https://github.com/googlesamples/firebase-arduino/archive/master.zip

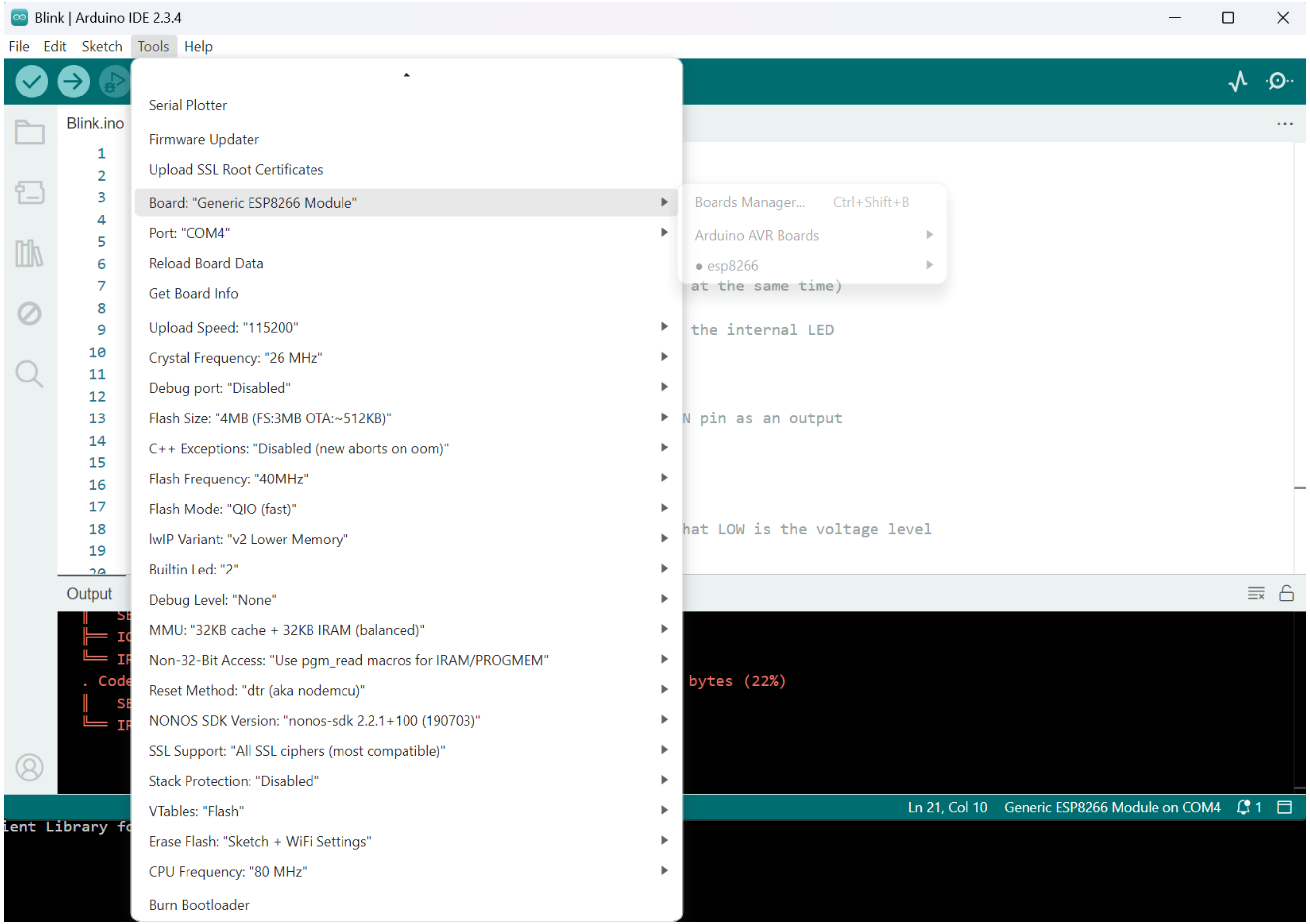
Arduino IDE: How To

- Try a few sketch examples: *Blink the Led*



Arduino IDE: How To

- Connect the board to the PC and set the correct board parameters



Serial Plotter

Firmware Updater

Upload SSL Root Certificates

Board: "Generic ESP8266 Module"

Port: "COM4"

Reload Board Data

Get Board Info

Upload Speed: "115200"

Crystal Frequency: "26 MHz"

Debug port: "Disabled"

Flash Size: "4MB (FS:3MB OTA:~512KB)"

C++ Exceptions: "Disabled (new aborts on oom)"

Flash Frequency: "40MHz"

Flash Mode: "QIO (fast)"

lwIP Variant: "v2 Lower Memory"

Builtin Led: "2"

Debug Level: "None"

MMU: "32KB cache + 32KB IRAM (balanced)"

Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM"

Reset Method: "dtr (aka nodemcu)"

NONOS SDK Version: "nonos-sdk 2.2.1+100 (190703)"

SSL Support: "All SSL ciphers (most compatible)"

Stack Protection: "Disabled"

VTables: "Flash"

Erase Flash: "Sketch + WiFi Settings"

CPU Frequency: "80 MHz"

Burn Bootloader

Boards Manager... Ctrl+Shift+B

Arduino AVR Boards

● esp8266

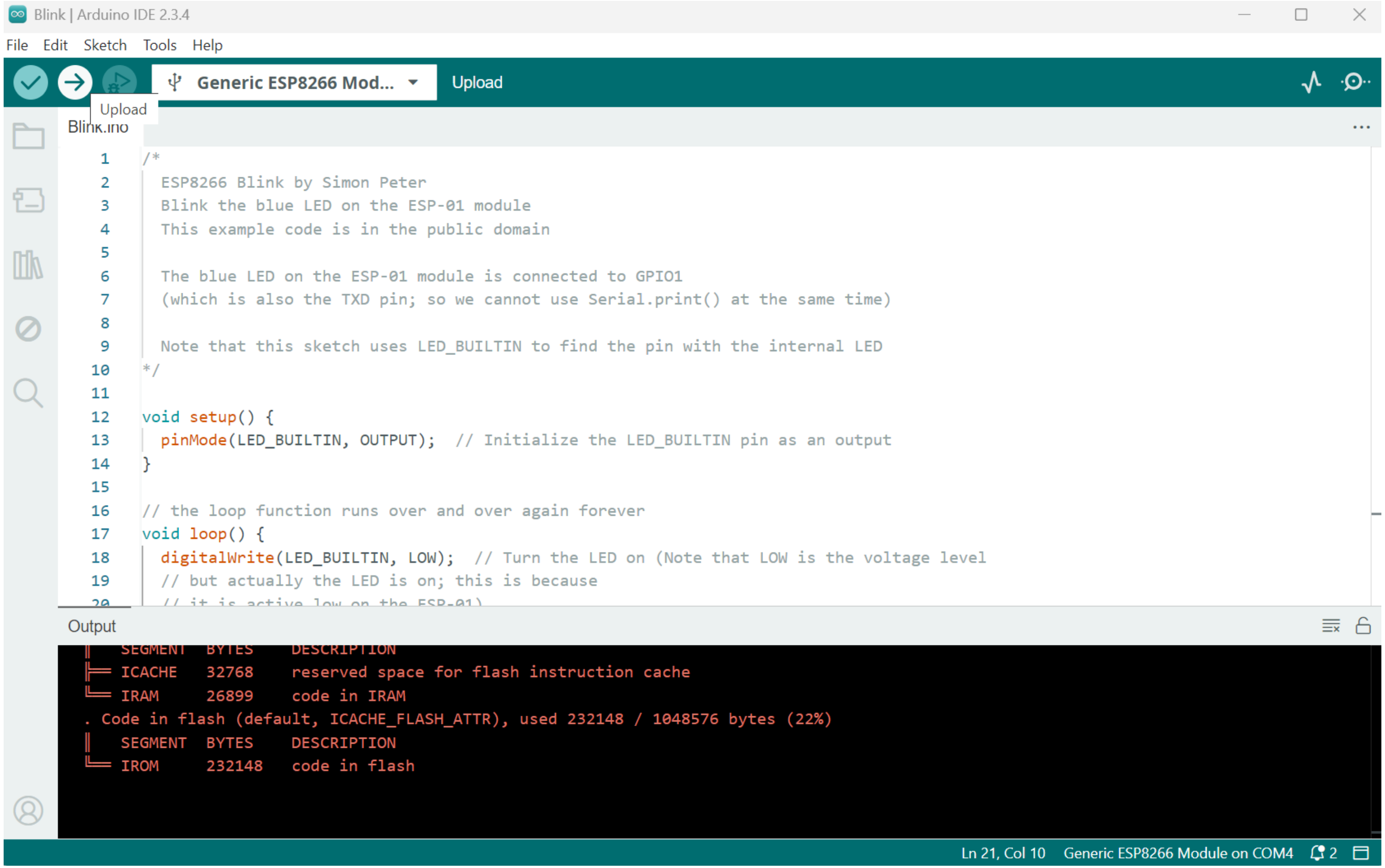
```
at the same time)
the internal LED
N pin as an output
hat LOW is the voltage level
bytes (22%)
ient Library fo
```

Ln 21, Col 10   Generic ESP8266 Module on COM4

https://arduino.esp8266.com/stable/package_esp8266com_index.json

Arduino IDE: How To

- Verify, compile and upload it

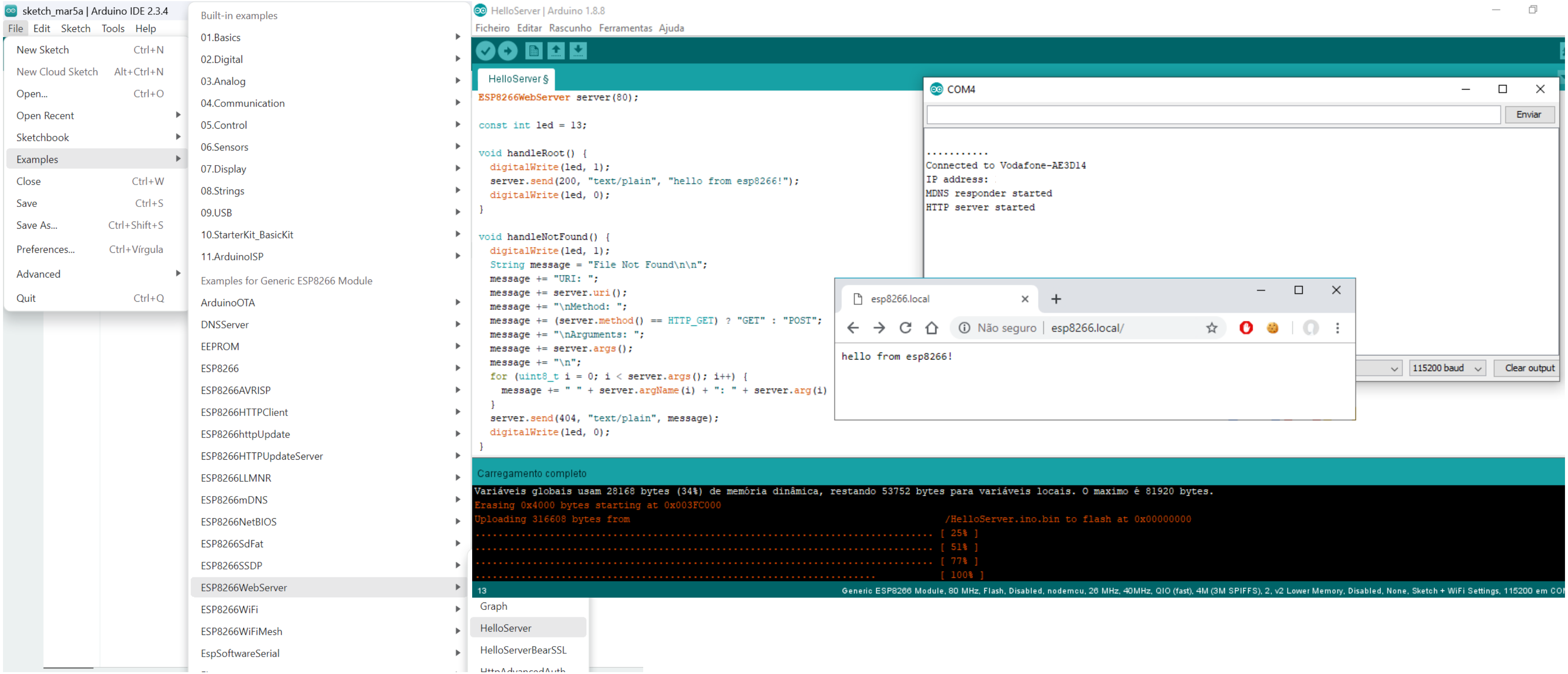


```
/*
  ESP8266 Blink by Simon Peter
  Blink the blue LED on the ESP-01 module
  This example code is in the public domain

  The blue LED on the ESP-01 module is connected to GPIO1
  (which is also the TXD pin; so we cannot use Serial.print() at the same time)

  Note that this sketch uses LED_BUILTIN to find the pin with the internal LED
*/

void setup() {
  pinMode(LED_BUILTIN, OUTPUT);  // Initialize the LED_BUILTIN pin as an output
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, LOW);  // Turn the LED on (Note that LOW is the voltage level
  // but actually the LED is on; this is because
  // it is active low on the ESP-01)
```

```
║   SEGMENT  BYTES    DESCRIPTION
╠══ ICACHE   32768    reserved space for flash instruction cache
╚══ IRAM     26899    code in IRAM
. Code in flash (default, ICACHE_FLASH_ATTR), used 232148 / 1048576 bytes (22%)
║   SEGMENT  BYTES    DESCRIPTION
╚══ IROM     232148   code in flash
```

Arduino IDE: How To

- Try other examples: *HelloServer*



```
ESP8266WebServer server(80);

const int led = 13;

void handleRoot() {
  digitalWrite(led, 1);
  server.send(200, "text/plain", "hello from esp8266!");
  digitalWrite(led, 0);
}

void handleNotFound() {
  digitalWrite(led, 1);
  String message = "File Not Found\n\n";
  message += "URI: ";
  message += server.uri();
  message += "\nMethod: ";
  message += (server.method() == HTTP_GET) ? "GET" : "POST";
  message += "\nArguments: ";
  message += server.args();
  message += "\n";
  for (uint8_t i = 0; i < server.args(); i++) {
    message += " " + server.argName(i) + ": " + server.arg(i)
  }
  server.send(404, "text/plain", message);
  digitalWrite(led, 0);
}
```

```
...........
Connected to Vodafone-AE3D14
IP address:
MDNS responder started
HTTP server started
```

```
hello from esp8266!
```

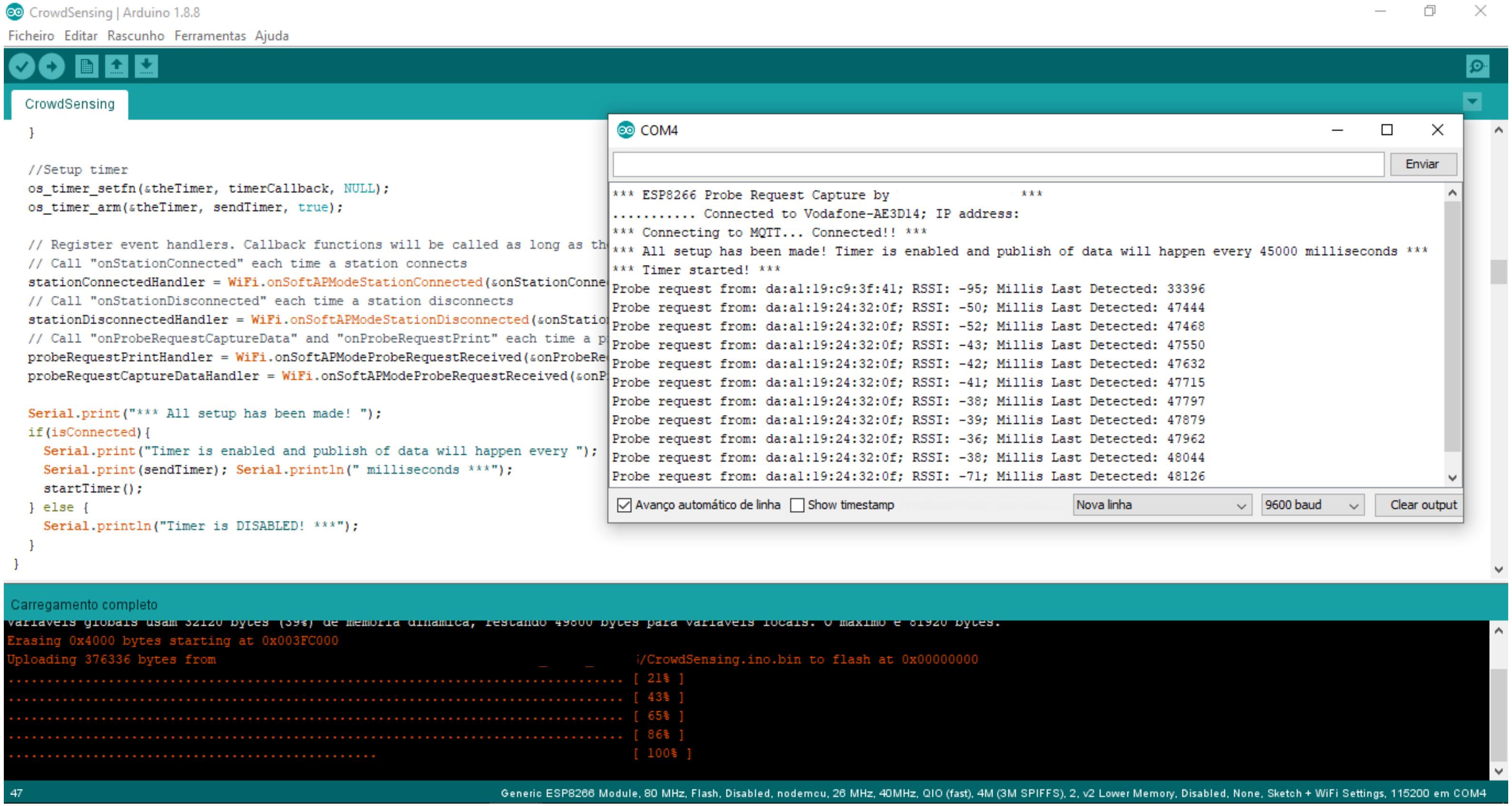
Carregamento completo

```
Variáveis globais usam 28168 bytes (34%) de memória dinâmica, restando 53752 bytes para variáveis locais. O maximo é 81920 bytes.
Erasing 0x4000 bytes starting at 0x003FC000
Uploading 316608 bytes from                                   /HelloServer.ino.bin to flash at 0x00000000
................................................................................ [ 25% ]
................................................................................ [ 51% ]
................................................................................ [ 77% ]
..........................................................................       [ 100% ]
```

13 Generic ESP8266 Module, 80 MHz, Flash, Disabled, nodemcu, 26 MHz, 40MHz, QIO (fast), 4M (3M SPIFFS), 2, v2 Lower Memory, Disabled, None, Sketch + WiFi Settings, 115200 em CO

Arduino IDE: How To

- Try other examples: *CrowdSensing* (compatible with ArduinoJson v5)



CrowdSensing | Arduino 1.8.8
Ficheiro Editar Rascunho Ferramentas Ajuda
CrowdSensing
COM4
Enviar
☑ Avanço automático de linha
☐ Show timestamp
Nova linha
9600 baud
Clear output
Carregamento completo
47

```
  }

  //Setup timer
  os_timer_setfn(&theTimer, timerCallback, NULL);
  os_timer_arm(&theTimer, sendTimer, true);

  // Register event handlers. Callback functions will be called as long as th
  // Call "onStationConnected" each time a station connects
  stationConnectedHandler = WiFi.onSoftAPModeStationConnected(&onStationConne
  // Call "onStationDisconnected" each time a station disconnects
  stationDisconnectedHandler = WiFi.onSoftAPModeStationDisconnected(&onStatio
  // Call "onProbeRequestCaptureData" and "onProbeRequestPrint" each time a p
  probeRequestPrintHandler = WiFi.onSoftAPModeProbeRequestReceived(&onProbeRe
  probeRequestCaptureDataHandler = WiFi.onSoftAPModeProbeRequestReceived(&onP

  Serial.print("*** All setup has been made! ");
  if(isConnected){
    Serial.print("Timer is enabled and publish of data will happen every ");
    Serial.print(sendTimer); Serial.println(" milliseconds ***");
    startTimer();
  } else {
    Serial.println("Timer is DISABLED! ***");
  }
}
```

```
*** ESP8266 Probe Request Capture by [redacted] ***
........... Connected to Vodafone-AE3D14; IP address:
*** Connecting to MQTT... Connected!! ***
*** All setup has been made! Timer is enabled and publish of data will happen every 45000 milliseconds ***
*** Timer started! ***
Probe request from: da:al:19:c9:3f:41; RSSI: -95; Millis Last Detected: 33396
Probe request from: da:al:19:24:32:0f; RSSI: -50; Millis Last Detected: 47444
Probe request from: da:al:19:24:32:0f; RSSI: -52; Millis Last Detected: 47468
Probe request from: da:al:19:24:32:0f; RSSI: -43; Millis Last Detected: 47550
Probe request from: da:al:19:24:32:0f; RSSI: -42; Millis Last Detected: 47632
Probe request from: da:al:19:24:32:0f; RSSI: -41; Millis Last Detected: 47715
Probe request from: da:al:19:24:32:0f; RSSI: -38; Millis Last Detected: 47797
Probe request from: da:al:19:24:32:0f; RSSI: -39; Millis Last Detected: 47879
Probe request from: da:al:19:24:32:0f; RSSI: -36; Millis Last Detected: 47962
Probe request from: da:al:19:24:32:0f; RSSI: -38; Millis Last Detected: 48044
Probe request from: da:al:19:24:32:0f; RSSI: -71; Millis Last Detected: 48126
```

```
variaveis globais usam 32120 bytes (39%) de memoria dinamica, restando 49800 bytes para variaveis locais. O maximo e 81920 bytes.
Erasing 0x4000 bytes starting at 0x003FC000
Uploading 376336 bytes from [redacted]/CrowdSensing.ino.bin to flash at 0x00000000
................................................................................ [ 21% ]
................................................................................ [ 43% ]
................................................................................ [ 65% ]
................................................................................ [ 86% ]
................................................                                 [ 100% ]
```

Generic ESP8266 Module, 80 MHz, Flash, Disabled, nodemcu, 26 MHz, 40MHz, QIO (fast), 4M (3M SPIFFS), 2, v2 Lower Memory, Disabled, None, Sketch + WiFi Settings, 115200 em COM4

https://goo.gl/RoPJM7

Resources

- When there is no board available, try an emulator:
  - https://wokwi.com/
    - https://wokwi.com/projects/380479029459892225
    - https://docs.wokwi.com/pt-BR/vscode/getting-started
  - https://blog.adafruit.com/2023/06/21/an-esp8266-simulator-in-javascript-emulation-wokwimakes/
  - https://github.com/afnid/espsim
  - https://hackaday.io/project/183023-esp8266-pc-xt-emulator
  - https://docs.zephyrproject.org/1.12.0/boards/xtensa/xt-sim/doc/xt-sim.html
  - https://github.com/OSLL/qemu-xtensa/tree/xtensa-esp8266
  - https://github.com/witnessmenow/esp8266-alexa-wemo-emulator
  - https://www.qemu.org/docs/master/index.html

- Useful links about connecting ESP8266 and Arduino:
  - https://tttapa.github.io/ESP8266/Chap01%20-%20ESP8266.html
  - https://github.com/esp8266/Arduino
  - https://doc.riot-os.org/group__cpu__esp8266.html