Using Bluetooth Low Energy with publish/subscribe

In this lab we will transmit measurement data using Bluetooth Low Energy capabilities of the Arduino. Received data on the laptop will be made available in a publish subscribe system with PyQt5 user interface.

Start by installing the **ArduinoBLE** library in the Arduino environment and the **bleak** library for Python.

Part 1.1. Finding your board in the Bluetooth chaos

Many devices use bluetooth, so when we scan the environment you will notice that perhaps 50 or more devices will be around. It's therefore important that you can filter output your own board and not read measurements or try to control the board of your neighbour. First have a look at the Arduino ble_led.ino file.
There are several important sections to be noticed:
At the top is a service defined with a large 128 bit hexadecimal code which is reserved identifier for the LED on the board.
The second identifier with a slightly different value is used to control the LED.

Another important line is **BLE.setLocalName("BLE-LAB2");**
This is advertised name you will see when doing a scan for Bluetooth devices. **Change this name to "BLE-AR##" where ## is the Arduino number you are using**. Do not be childish, so use strings like above instead of stupid names,

Your smartphone or laptop app is normally not capable to detect the Arduino, but you can use the python application Discover.py which is provided on canvas (part of the tarball of lab2).
You will see a large number of devices including your Arduino. Check if it's really your board and not the one of your neighbour!!
Also walk through the Python code of this application.
Note especially the usage of the asyncio library with await contructions and the BleakScanner discover function. Check the online API on internet of Bleak.

Part 1.2. Controlling the LED using BLE

Use the 2 templates (lab2_led.ino and lab2_led_template.py). Extend the Python file in order to control the led. (turn the LED 10 times on and off as stated in the template file)
Save the file as **lab2_led.py**

Part 2. Reading the acceleration sensor using BLE

Check the directory ble_accel directory with the Arduino file which is more or less a finished file. As you can see in this file there is a line BLECharacteristic…. which contains the words BLERead | BLENotify which means that the Arduino will "advertise" that new values are available.
The loop will write continuously new values coming from the acceleration sensor. Check also if the values are correct by turning the board in different directions.

Finish the lab2_sensor_template.py in order to read the 3 sensor values from the Arduino. Save the file as **lab2_sensor.py**.

Part 3.1 Creating a Publish / Subscribe environment
You now have a Python application which receives the data from the nearby Arduino. Suppose we would have several of such configurations in different locations and would like to combine these as a network of IoT devices.
A good method is using the MQTT standard. However most environments are large and complicated to install on your laptop. We will use a publish-subscribe system which has basically the same functionality and is easy to work with. The library is called Zero Message Queue (**zeromq.org**) which is available for a large number of languages. There are different forks of the library, but we will use the Python version of zmq. Install this on your system.
First play around with the publisher and subscriber file which you can find in
https://github.com/zeromq/pyzmq/tree/main/examples/pubsub
Run the publisher and subscriber in different terminals using as “connect-to” argument tcp://127.0.0.1:<some_port_number>. The number 127.0.0.1 is an ip address which is called localhost and refers to you own system.

Part 3.2 Using Zmq to send the data to a remote client
Copy your lab2_sensor.py to **lab2_publisher.py** and add publisher functionality into the file. Create a l**ab2_subscriber.py** which receives the data.

Part 3.3 Layered subscriptions / Gyro sensor.
Extend the functionality with separate subscriptions for the axes and add functionality for the gyro sensor of the Arduino.

Part 4: PyQt5 user interface for the client.
Modify your lab2_subscriber with a PyQt5 interface for the system above. The application should show a graph with the received data. It should be able to subscribe or unsubscribe to specific data from the publisher, i.e. if you want the accelerometer data, gyro sensor and which parts of the data (add selection user interface elements).