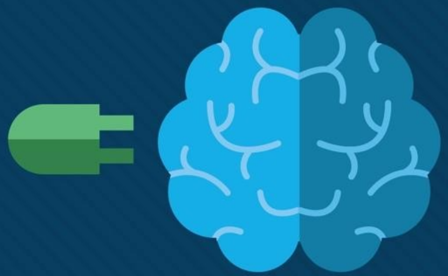
Embedded Machine Learning



Tassadaq Hussain Cheema

Professor EE Department NAMAL University

February 27, 2018

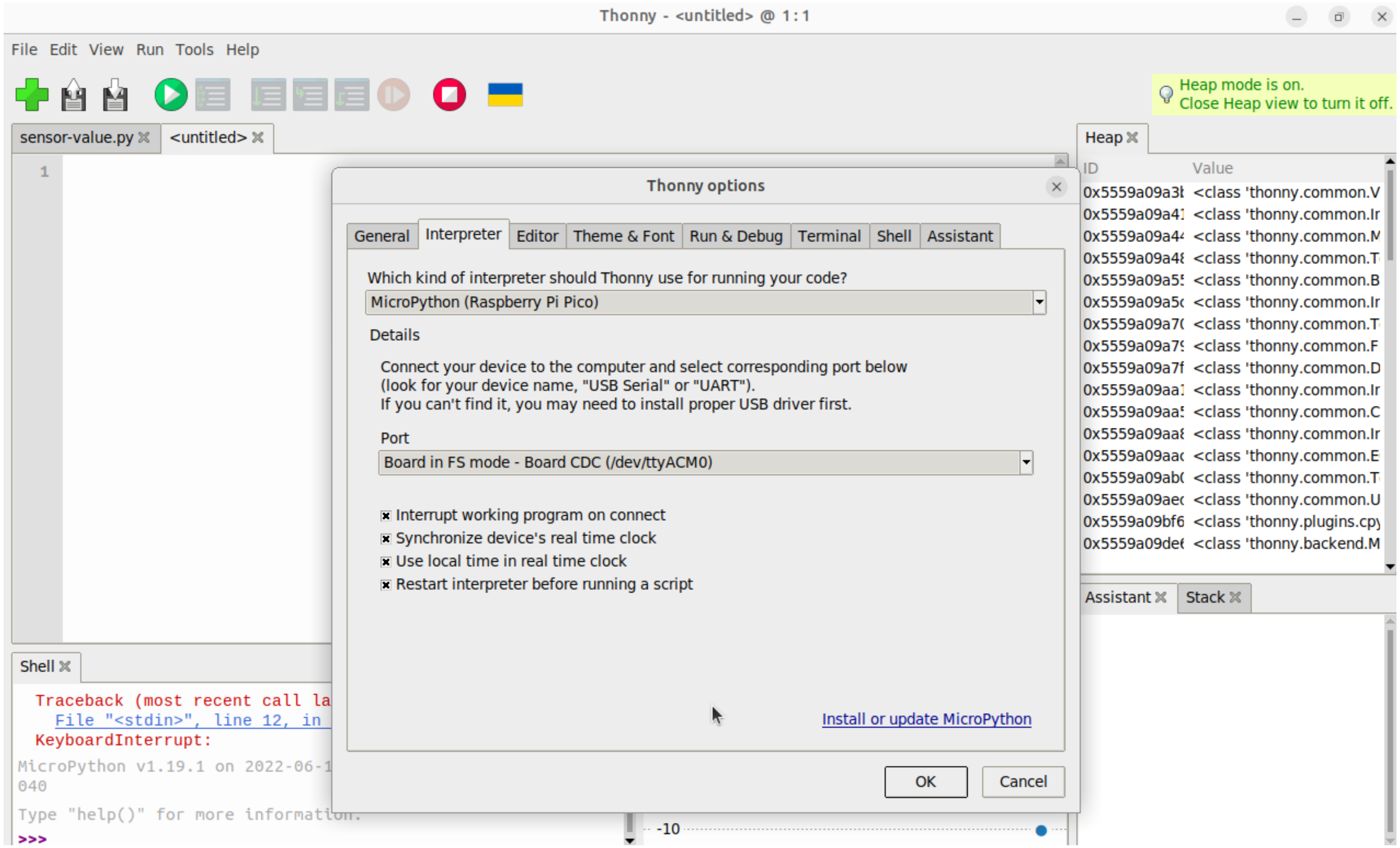
Topics

- MicroPython
- Thonny
- TinyML
- Tensorflow Lite
- Microsoft Azure Sphere

MicroPython

- MicroPython is a software implementation of the Python programming language designed for microcontrollers and embedded systems.
- It is a lean and efficient version of Python that can run on devices with limited resources such as low-power microcontrollers, development boards, and other small computing devices.

- MicroPython supports a wide range of microcontrollers, including popular platforms like Arduino, ESP32, ESP8266, Raspberry Pi Pico, and many more.
- It provides access to hardware peripherals and I/O pins, allowing developers to control sensors, actuators, and other components connected to the microcontroller.



Which kind of interpreter should Thonny use for running your code?

MicroPython (Raspberry Pi Pico)

Details

Connect your device to the computer and select corresponding port below
(look for your device name, "USB Serial" or "UART").
If you can't find it, you may need to install proper USB driver first.

Port

Board in FS mode - Board CDC (/dev/ttyACM0)

- [x] Interrupt working program on connect
- [x] Synchronize device's real time clock
- [x] Use local time in real time clock
- [x] Restart interpreter before running a script

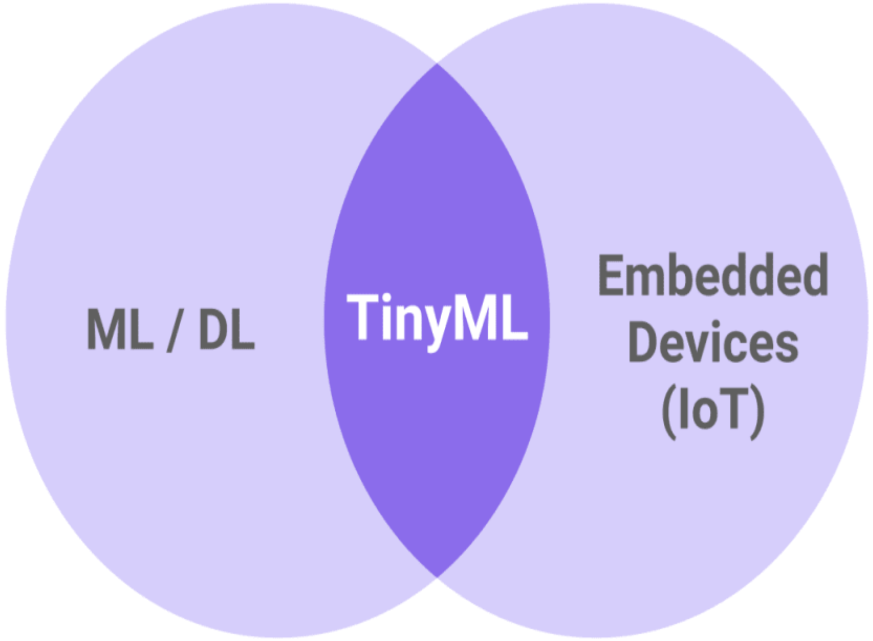
ID	Value
0x5559a09a3b	<class 'thonny.common.V
0x5559a09a41	<class 'thonny.common.Ir
0x5559a09a44	<class 'thonny.common.M
0x5559a09a48	<class 'thonny.common.T
0x5559a09a55	<class 'thonny.common.B
0x5559a09a5c	<class 'thonny.common.Ir
0x5559a09a70	<class 'thonny.common.T
0x5559a09a79	<class 'thonny.common.F
0x5559a09a7f	<class 'thonny.common.D
0x5559a09aa1	<class 'thonny.common.Ir
0x5559a09aa5	<class 'thonny.common.C
0x5559a09aa8	<class 'thonny.common.Ir
0x5559a09aac	<class 'thonny.common.E
0x5559a09ab0	<class 'thonny.common.T
0x5559a09aec	<class 'thonny.common.U
0x5559a09bf6	<class 'thonny.plugins.cpy
0x5559a09de6	<class 'thonny.backend.M

```
Traceback (most recent call la
  File "<stdin>", line 12, in
KeyboardInterrupt:
MicroPython v1.19.1 on 2022-06-1
040
Type "help()" for more information.
>>>
```

-10

TinyML

- TinyML is a type of machine learning that allows models to run on smaller, less powerful devices.
- It involves hardware, algorithms, and software that can analyze sensor data on these devices with very low power consumption, making it ideal for always-on use-cases and battery-operated devices.



Why TinyML

- **Bandwidth** - As an example, a device at 100Hz sampling rate produces 360,000 data points each hour. Now imagine the amount of data produced by a fleet of these devices.
- **Latency** - "time between when a system takes in a sensory input and responds to it". In case of conventional ML deployment data must be first sent to an ML application. This increases the time in which an edge device can take action as it waits for the response.
- **Economics** - Cloud is cheap but not so cheap. It still costs money to ingest large amounts of data, especially if it must happen in real-time.
- **Reliability** - In case of high-frequency sampling, it might be hard to ensure that data arrives to a target in the same order as it was produced by an edge device.
- **Privacy** - TinyML processes data on-device and is not sent through network. This reduces the surface for data abuse.

Assignment 2

- For the semester Complex Engineering Problem.
- Submit Front-end, Data, Software and Hardware Architectures. Along with short description of each architecture.