Arduino IDE → PlatformIO + IDE

Arduino, what is it

- It is the name for the hardware (atmel based..)
- It is the name for the IDE
- It is the name for the FrameWork (libs+code)
- It’s a set of tools to make it easy for a novice to start making a chip do useful things

What does it do (hardware)

- It handles all the interfacing with your microcontroller, as easily as possible
- Hardware comes with a bootloader (no special programmer required)
- Hardware is ‘abstracted’; pin mappings/functions along edge of board
- It provides compatibility(ish)

What does it do (IDE)

- It knows the types of boards out there and how to upload software to them
- It provides a way to manage compatible libraries
- It makes it easy to start experimenting (example sketches)
- It abstracts a number of programming concepts away from the user; turns ‘Arduino code’ into C

What does it do (framework)

- It provides ‘higher level’ functions, abstracting away the ‘nitty gritty’ of an architecture
- It is essentially a ton of C-files defining
- Pin-definitions, useful functions (pinMode, DigitalWrite, analogRead, etc)
- It provides a way for code to work over multiple ‘Architectures’ (started with Atmel, but others available)

What does it do , badly

- It is very bad at handling larger projects (IDE)
- IDE is clunky and slow (JAVA)
- Hard to work with other architectures (esp8266/esp32)
- It has bad library handling (dependencies are global)

Introducing Platform IO (in IDE)

- Lives on PlatformIO.org
- Open source
- “Professional collaborative platform for embedded development”
- “PlatformIO is written in pure Python and doesn't depend on any additional libraries/tools from an operation system. It allows you to use PlatformIO beginning from PC and ending with credit-card sized computers (like Raspberry Pi, BeagleBone, CubieBoard)

PlatformIO, what does it do

- Separates ‘platform’ , ‘architecture’, ‘board’ and ‘Programming Environment’ from eachother
- Platform: arduino, esp-idf, RTOS, etc
- Architecture: atmel, stm32, espressif8266/esp32
- Board: a board may be useable for multiple platforms
- Programming-Environment: IDE, or not to IDE (or basis of larger project: esphome.io, etc)

PlatformIO, the ‘core’

- The basis is a python tool ‘platformio’
- Can be used stand-alone (shell/makefile)
- Manages ‘projects’
- Communicates with the ‘registry’ for searching, installing and updates.
- Contains ‘platforms’,’frameworks’,’boards’ and ‘libraries’
- Can communicate with platformio for other tasks (remote ota)
- Has a smart way of handling dependencies (libraries)

PlatformIO, Project

- /src - Your stuff
- /include - To be included (.h)
- /lib - (private) libraries
- /test - unit tests (magic for pro’s)
- platformio.ini - config-file for awesome

Platformio, platforms

- Compiler for the chosen architecture
- Support-tools for uploading/debugging
- ‘atmelavr’, ‘atmelmegaavr’
- ‘espressif8266’
- ‘esp32’

PlatformIO, Frameworks

- ‘Arduino’ is a framework available over multiple ‘platforms’
- Allows for easy portability to other chips (and thus: boards)
- Others available

PlatformIO, boards

- A ‘board’ provides info about:
- What platform (cpu → compiler)
- What upload/debug tools (serial-usb , wifi/ota)
- What abstraction a framework should use , if any (pins)

Platformio, Libraries

- Pulled from the ‘registry’ or installed locally in project /lib
- ‘Library Dependency Finder’ → handles finding the right library, guided by hints in platformio.ini
- Come with example-code
- Pulled directly from upstream github repo (not mirrored)

Platformio.io – Less talk, more doing

- Install/Demo-time
- VsCode install
- Platformio install
- (intermezzo about Codium)

Platformio – Install VSCode/Codium

- Vscode: code.visualstudio.com
- Codium = VsCode with no/less telemetry
- Codium: vscodium.com
- Caveat: visualstudio marketplace is advised/required
- https://github.com/VSCodium/vscodium/blob/master/DOCS.md

Platformio, install extension

- https://platformio.org/install/ide?install=vscode
- 1 Open Vscode Extension manager
- 2 Search for Official Platform IDE extension
- 3 Install PlatformIO IDE
- …. set some settings...
- 4 Profit (and/or ‘read quickstart guide’)

Platformio , config. vscode

- PlatformIO basedir = ~/Documents/PlatformIO
- Projects: ~/Documents/PlatformIO/Projects
- You know better ? :
- Sidebar ‘PlatformIO’ logo → New Terminal
- ‘pio settings set projects_dir */new/path/projects/dir’*

PlatformIO, setting up for Arduino

- Find ‘home screen’ (bottom-left if not open by default)
- Start new project
- Choose a name
- Choose a board : ‘arduino uno’
- MAGIC HAPPENS HERE
- Installed: atmelavr platform, arduino framework
- Created: project-dir + ‘platformio.ini’ file

PlatformIO, your first blink.ino

- *src*/main.cpp

```
#include <Arduino.h>

void setup() {
// initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                       // wait for a second
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                       // wait for a second
}
```

PlatformIO, blink.ino, continued

- ‘platformio.ini’

  [env:uno]

  platform = atmelavr

  board = uno

  framework = arduino

  upload_port = */dev/ttyUSB0*

  upload_speed = 115200

  monitor_port = /dev/ttyUSB0

  monitor_speed = 9600

PlatformIO, blink, Profit

- Bottom left:
- House: Home-page
- Checkmark: Build Code (CTRL-ALT-B)
- Arrow: Upload to board (CTRL-ALT-U)
- Garbage-bin: Clean project
- Plug: Serial Monitor (CTRL-ALT-S)
- Screen: open terminal

PlatformIO, blink.ino, aftermath

- YAY WE DID IT
- WORKSHOP OVER
- DONE!
- GRATUATED
- OMG SO SMARTS
- Hold your horses, young padawan.

PlatformIO, arduino framework info

- #include <Arduino.h> at top
- Define your functions before using them! (demo)
- … that’s it, afaik

Platformio, Libraries

- Refer to library with name or (registry) ID
- lib_deps =

  knolleary/PubSubClient

  bblanchon/ArduinoJson @ ~5,!=5.4

  https://github.com/gioblu/PJON.git#v2.0

  https://github.com/me-no-dev/ESPAsyncTCP.git

  https://github.com/adafruit/DHT-sensor-library/archive/master.zip

PlatformIO, Libraries cont.

- Library-dependencies travel with project
- Automatically pulled when building
- Checked for updates after install
- Can be finetuned further: lib_extra_dirs, lib_ignore, lib_ldf_mode, lib_compat_mode
- Check library registry online or in PlatformIO IDE

PlatformIO, extra things to note

- platform.ini file [env] (global)
- Build: pio run
- Upload: pio run -t upload
- [env:myEnv] : pio run -e myEnv
- Debug: define ‘debug=True’ in one env
- pio run debug

PlatformIO, advanced stuff

- Esphome.io showcase
- Home-automation devices ‘created’ in yaml-syntax
- Generates Arduino-code + platform.ini
- Uses platformio for all the heavy lifting
- DEMOTIME

PlatformIO, further reading

- Platformio ‘home’
- Docs.platformio.org, tutorials, examples
- Platformio.org/lib
- Library ‘examples’ tab

PlatformIO, conclusion

- Fixes shortcomings of Arduino IDE
- Allows greater platform/framework freedom
- Helps structurize projects
- Automates installation and maintenance of toolchains and libraries
- Integrates well with popular IDE’s (or not)

PlatformIO, the end

- Questions, comments, tomatoes
-
- A.P. “Justa” Marijnissen
- contact@sociallife.org