# Thing's Testing

## Unit Testing with PlatformIO

TQS – ProDEI
24/November/2017

João Pedro Dias
jpmdias@fe.up.pt
Researcher @ INESC TEC

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO

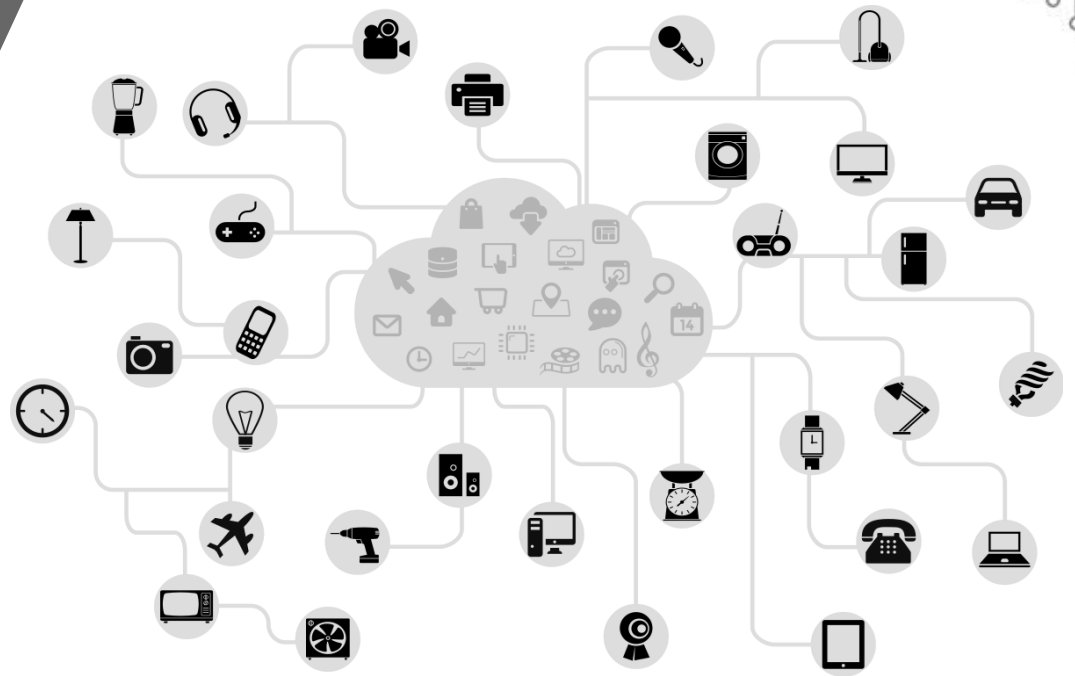
# Outline

- Contextualization
- Unit Testing
- Overview
- PlatformIO
- Demo & Tutorial
- Summary

# Contextualization
# *The Internet-of-Things*

- The Internet of Things (IoT) is the network of physical objects that contain embedded technology to communicate and sense or interact with their internal states or the external environment.

*-- Gartner – IT Glossary*



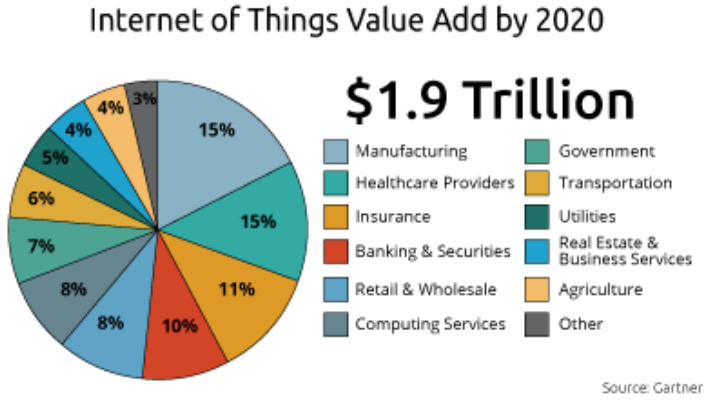
# Contextualization
# *Why IoT matters?*

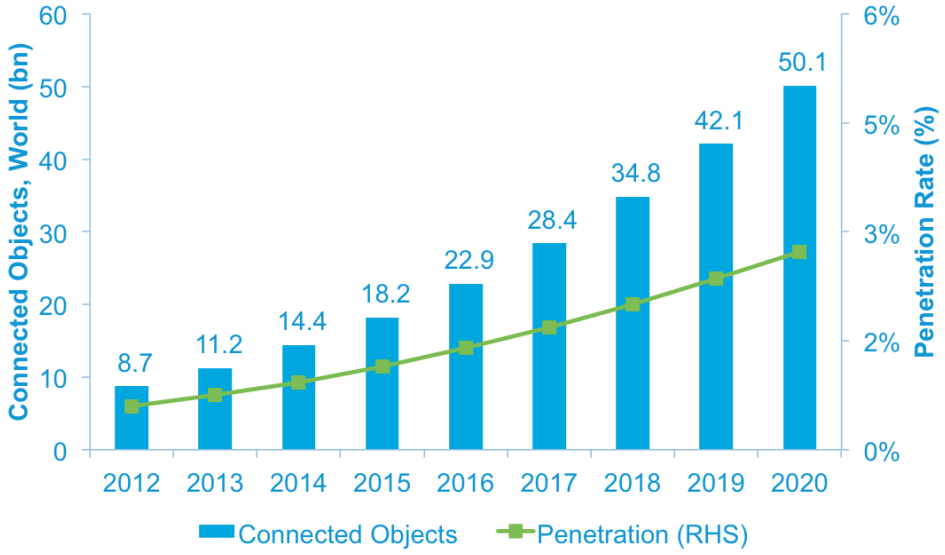
- IoT is the *infrastructure of the information society.*

  *– Global Standards Initiative on IoT (IoT-GSI)*

- "The IoT is quickly becoming something the world relies on to drive, report, and optimize performance from the timing of your run, to the gas mileage of your car. And it still needs to be tested."

  -- SmartBear Software *blog*



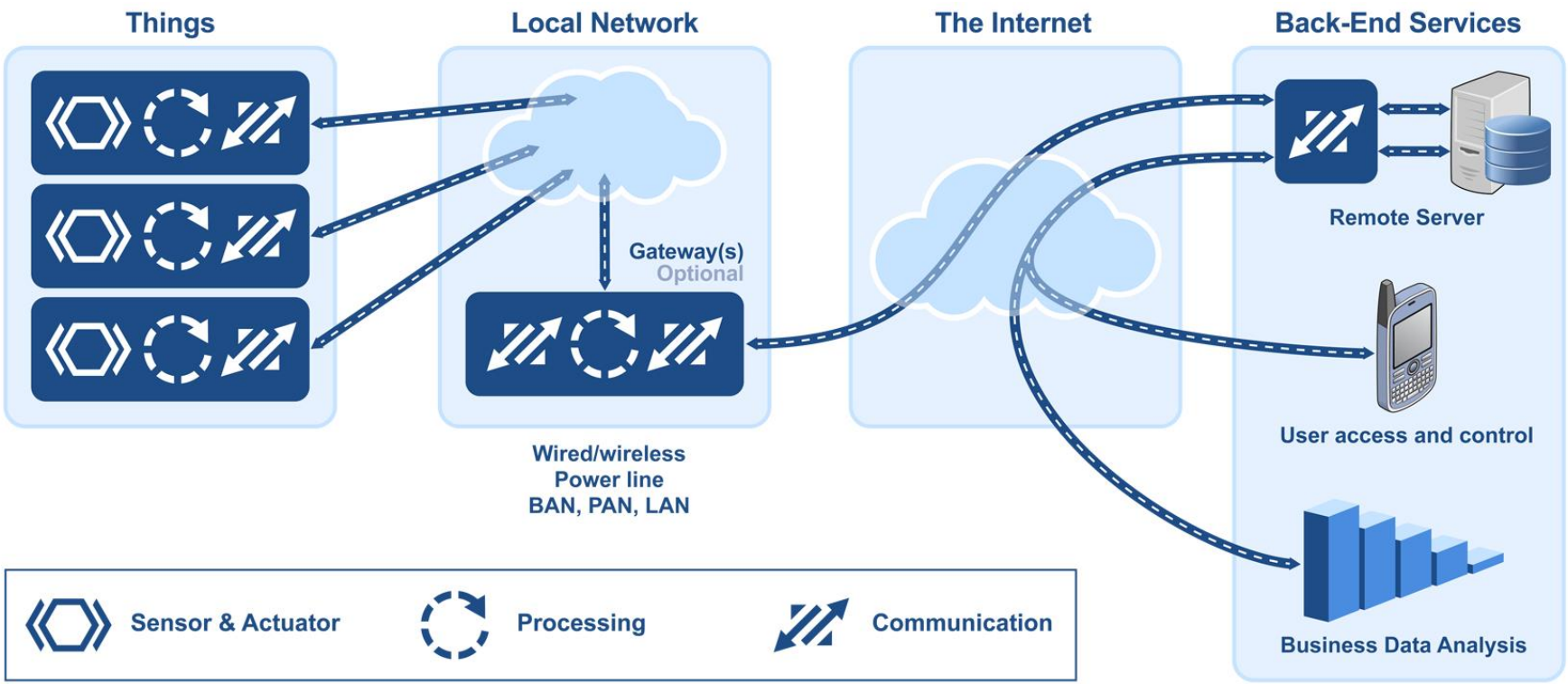




**Penetration of connected objects in total 'things' expected to reach 2.7% in 2020 from 0.6% in 2012**

Source: CCS, 2013

Contextualization

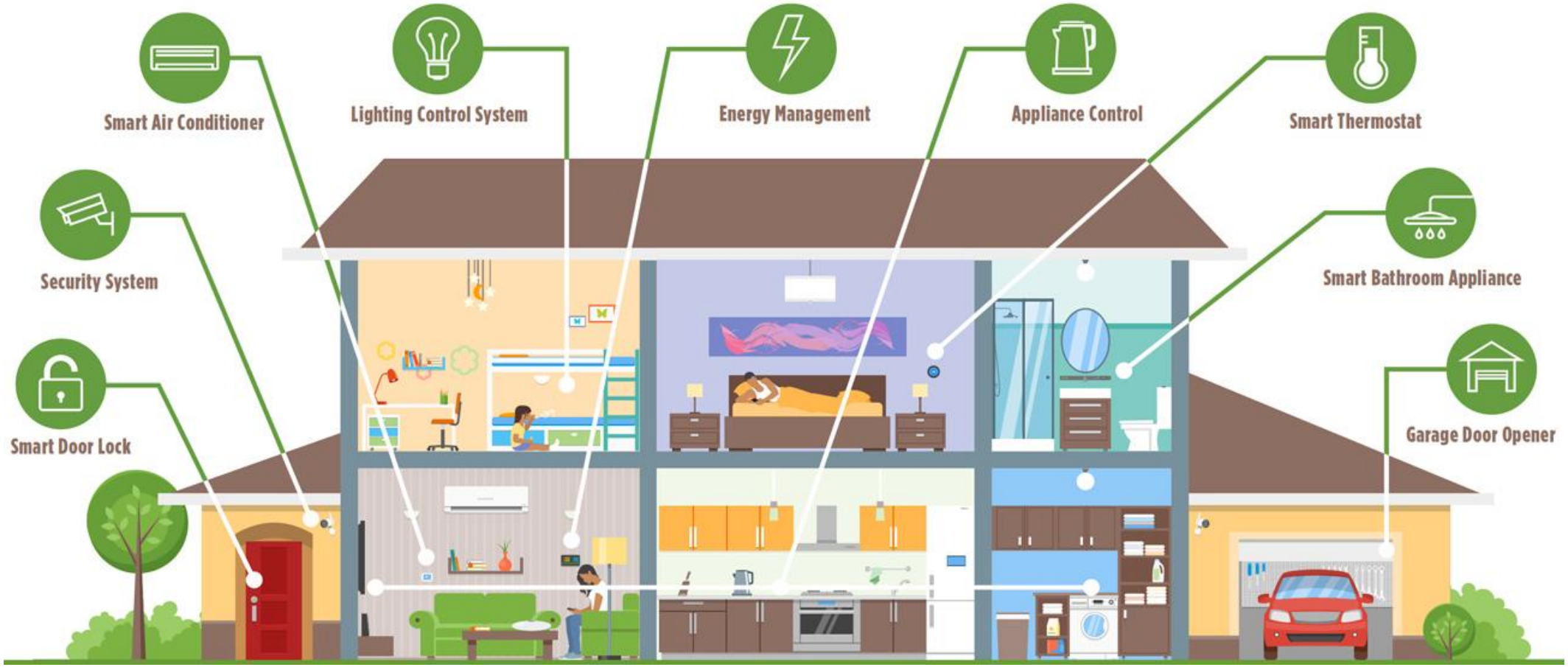
# *Testing Things*




- Different system layers lead to different needs of testing and validation.

Contextualization

# *Scenario: Smart Home*



Contextualization

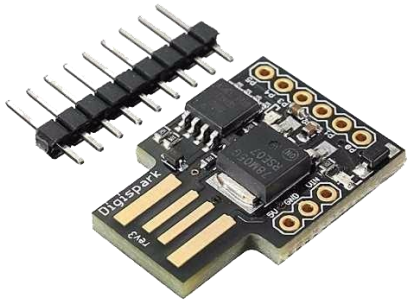
# *IoT Testing: What to test?*

- **Usability:** Device usability, related applications & analytics.
- **Security:** Network, application and hardware vulnerabilities. Privacy protection.
- **Connectivity:** Dependency in network availability and resilience to instability.
- **Performance:** Scalability (data collection, processing and decision) & power consumption
- **Compatibility Testing:** Multiple operating system versions, browser types and respective versions, generations of devices & communication modes.
- **Pilot Testing:** Make laboratories with real use scenarios to test the system correct functioning.
- **Regulatory Testing:** Depending on the IoT devices, multiple regulatory/compliance rules should be followed.
- **Upgrade Testing:** The heterogeneity of devices can lead to malfunctions on upgrades (e.g. firmware updates).

Contextualization

# *Testing Things*

- Focusing on the *things*
  - Mostly Microcontrollers
    - Arduino, ESP8266, ESP32, NodeMCU, Attiny, …
  - But also other devices (gateways, etc.)
    - Raspberry Pi, Asus Tinker Board, BeagleBoard, Parallela, …

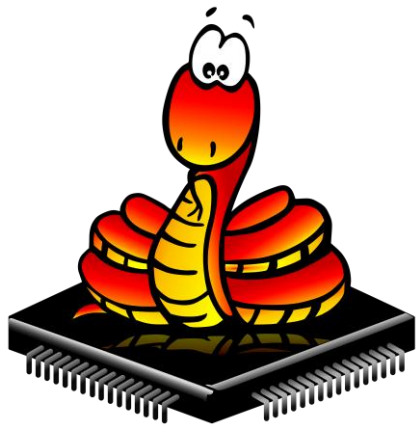




Contextualization

# *Testing Things*

- What language runs on *things*?
  - Lua, MicroPython, C/C++, Arduino Language, …
- Complementary libraries dedicated for hardware interaction
  - e.g. `Arduino.h`



Lua

Contextualization

# How to test on embedded devices / *things*?

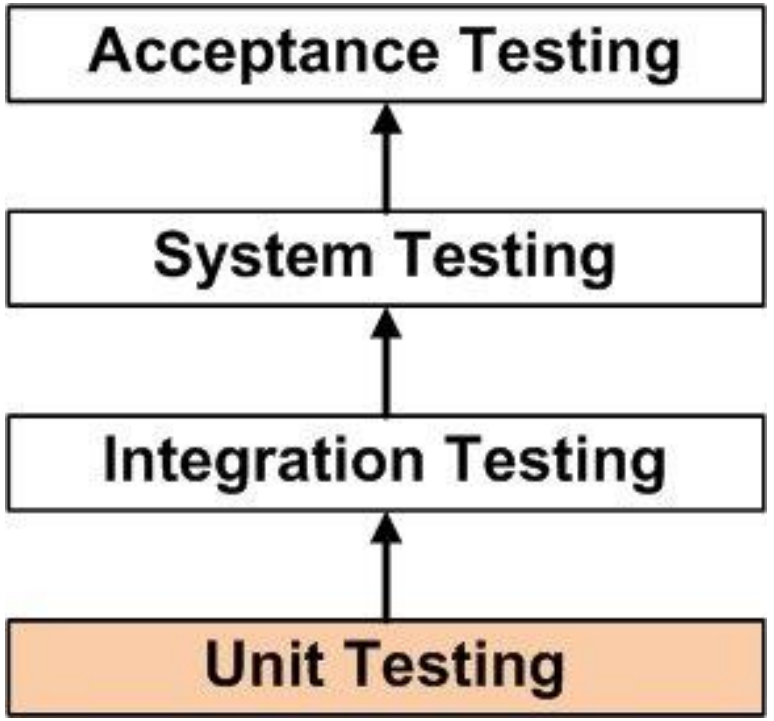
- Use a non-embedded platform to validate code that does not depend on the hardware inputs
  - e.g. algorithms, API's, …
- Run the code to test on the embedded devices and check hardware responses
  - e.g. Serial Port monitoring
- Use device virtual representations
  - Emulators, Simulators, ...
- Use Unit Testing (Locally and on-board)
  - Limitations due to memory constrains when testing on-board.

# Unit Testing

- Unit testing (UT) is the testing of individual software components in isolation in order to glean early and frequent feedback.

-- Gartner, Unit Testing

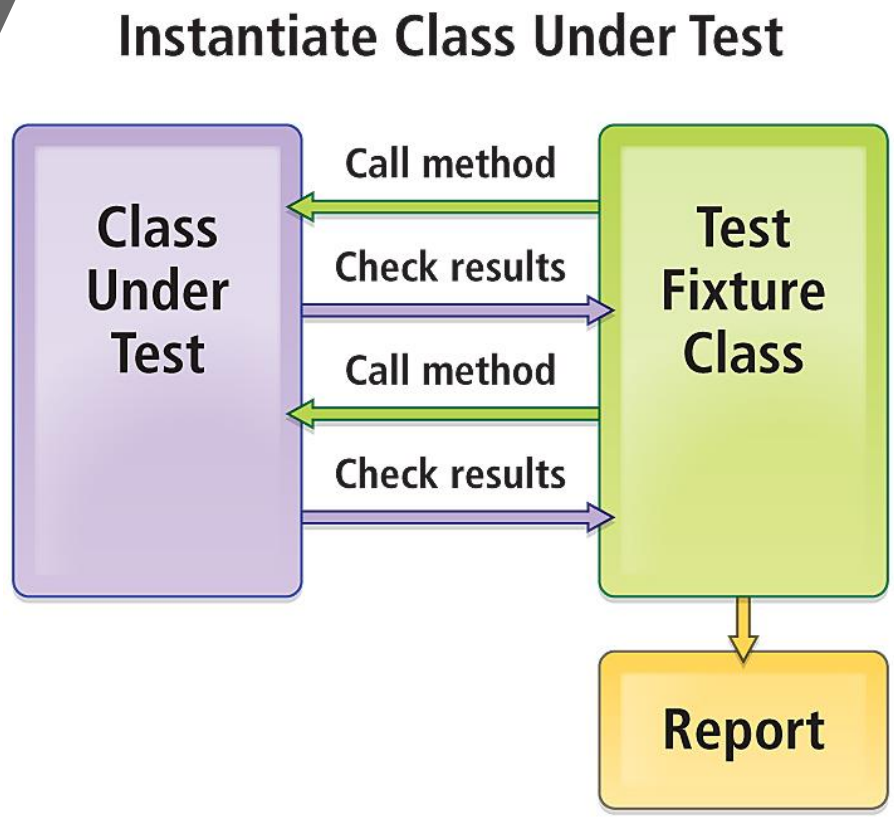
- Unit testing is a core component of the Test Driven Development (TDD)



# Advantages of Unit testing

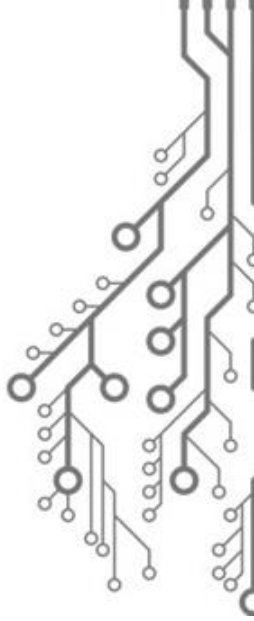
1. Issues are found at early stage.
2. Unit testing helps in maintaining and changing the code.
3. Since the bugs are found early in unit testing hence it also helps in reducing the cost of bug fixes.
4. Unit testing helps in simplifying the debugging process.

-- What is Unit testing?, ISTQB



# Tools

- ArduinoUnit
  - Unit testing framework for Arduino code & libraries.
- UNITY by ThrowTheSwitch.org
  - Unit Testing for C (especially Embedded Software)
- Cmock by ThrowTheSwitch.org
  - Automagical generation of stubs and mocks for Unity Tests
- PlatformIO: PIO Unit Testing
  - Allows testing on the local host machine (native), on the multiple local embedded devices/boards (connected to local host machine), or in both cases.



# Demo

Hands-on Tutorial on PlatformIO

http://docs.platformio.org

PIO Unit Testing

# Key Concepts

- **Test Runner**: allows to process specific environments or ignore tests using "Glob patterns".
- **Local Testing**: Allows you to run tests on a host machine or on target devices (boards) that are directly connected to the host machine.
- **Remote Testing:** Allows you to run tests on a remote machine or remote target devices (boards) without having to depend on OS software, extra software, SSH, VPN or opening network ports.
- **Unity Test API**: Tests are written based on the syntax of UNITY by ThrowTheSwitch.

# PIO Unit Testing Engine Design

1. **Main Program:** Contains the independent modules, procedures, functions or methods that will be the target candidates (TC) for testing.
2. **Unit Test:** A small independent program that is intended to re-use TC from the main program and apply tests to them.
3. **Test Processor:** The set of approaches and tools that will be used to apply tests for the environments from Project Configuration File `platformio.ini`

PIO Home - ws_unittests - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS

PIO Home

WS_UNITTESTS

DOCKER

PIO Home ×

Have an account? Log in

Home

Account

Libraries

Boards

Platforms

Devices

# Welcome to PlatformIO

Show at startup

Home `0.3.2` · Core `3.5.0b3`

## Quick Access

- New Project
- Import Arduino Project
- Open Project
- Project Examples

Web · Open Source · Get Started · Docs · News · Community · Report an Issue · Donate · Contact

## Recent Projects

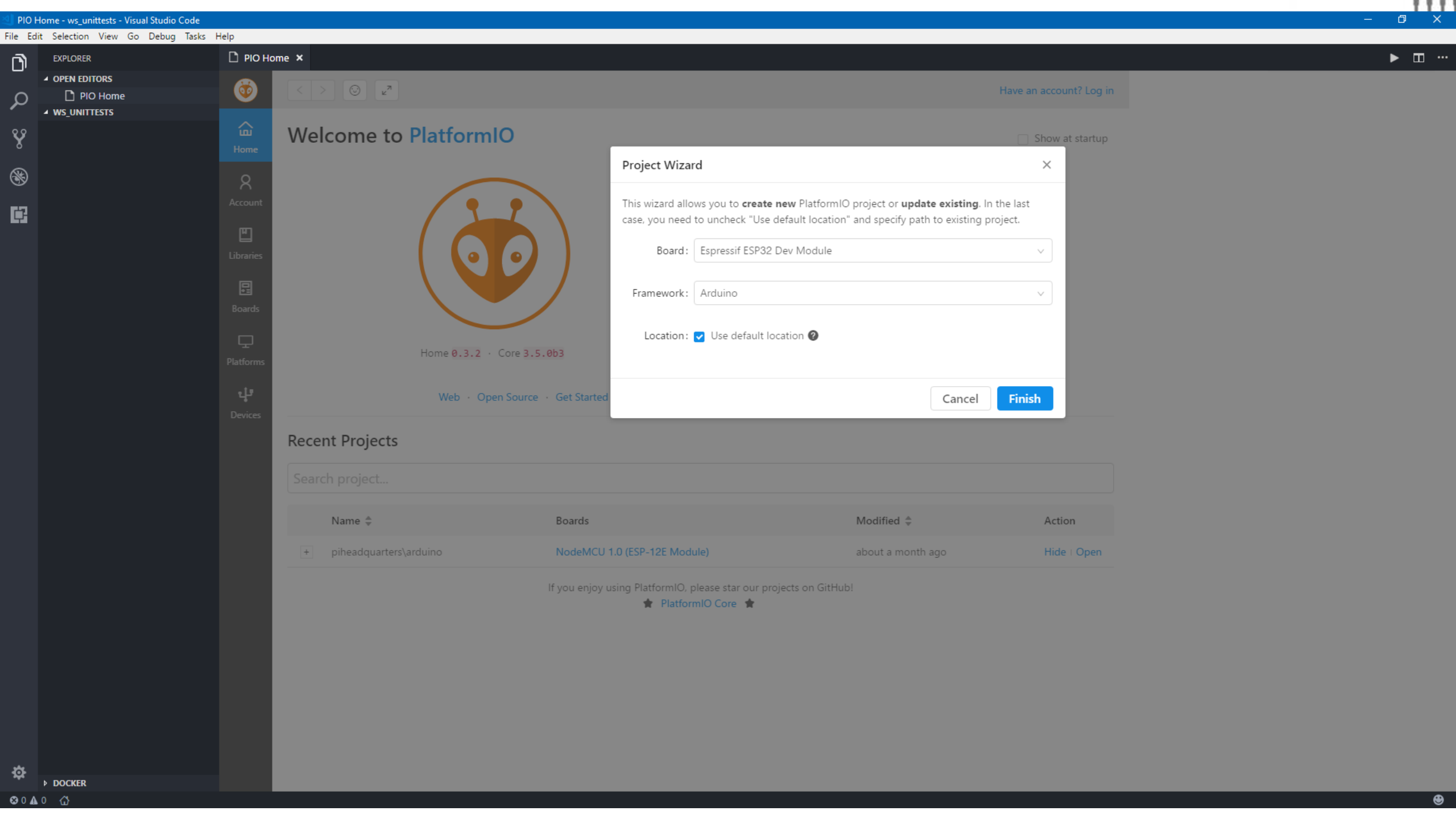
Search project...

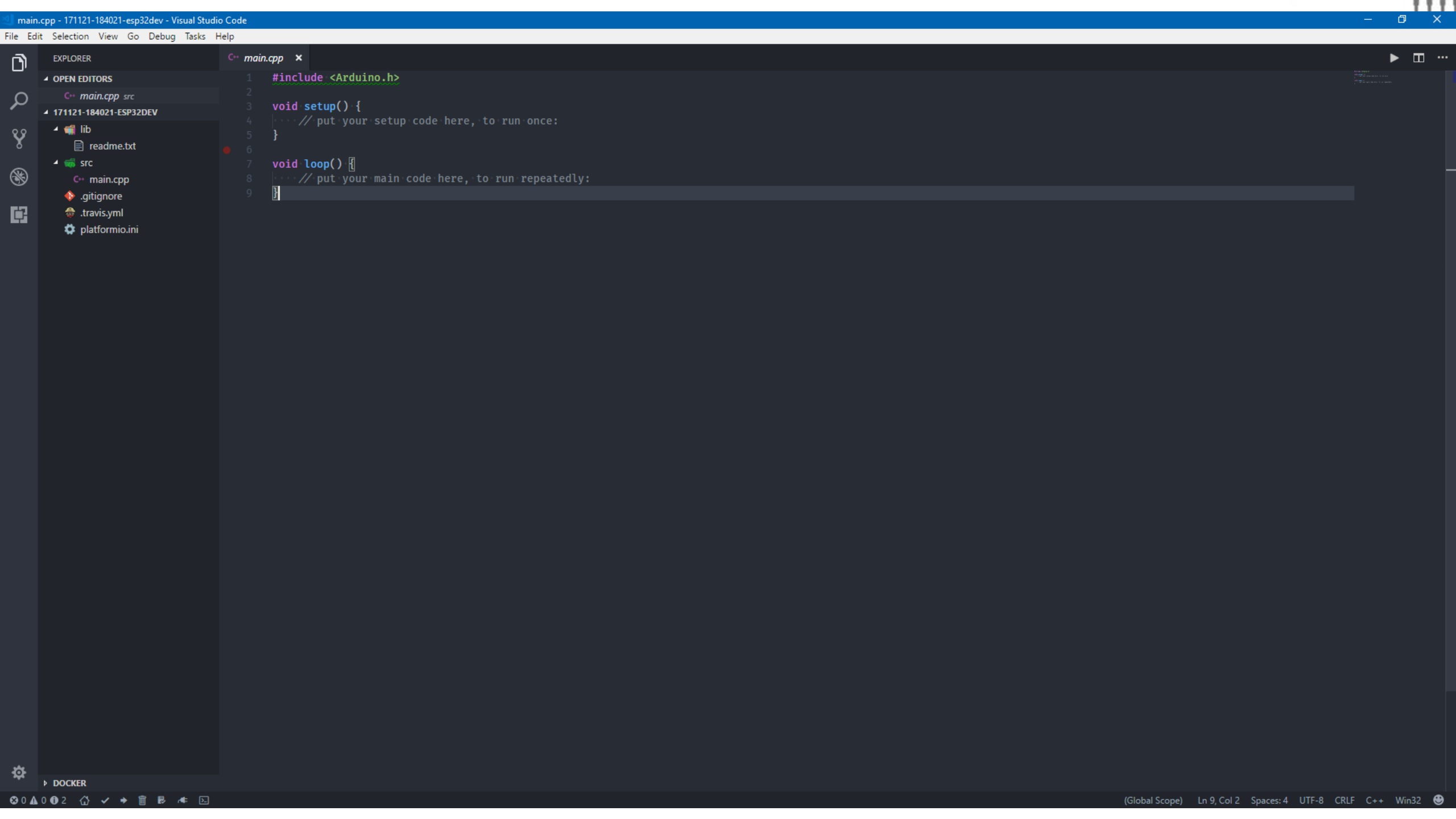
| | Name | Boards | Modified | Action |
|---|---|---|---|---|
| + | piheadquarters\arduino | NodeMCU 1.0 (ESP-12E Module) | about a month ago | Hide \| Open |

If you enjoy using PlatformIO, please star our projects on GitHub!

★ PlatformIO Core ★

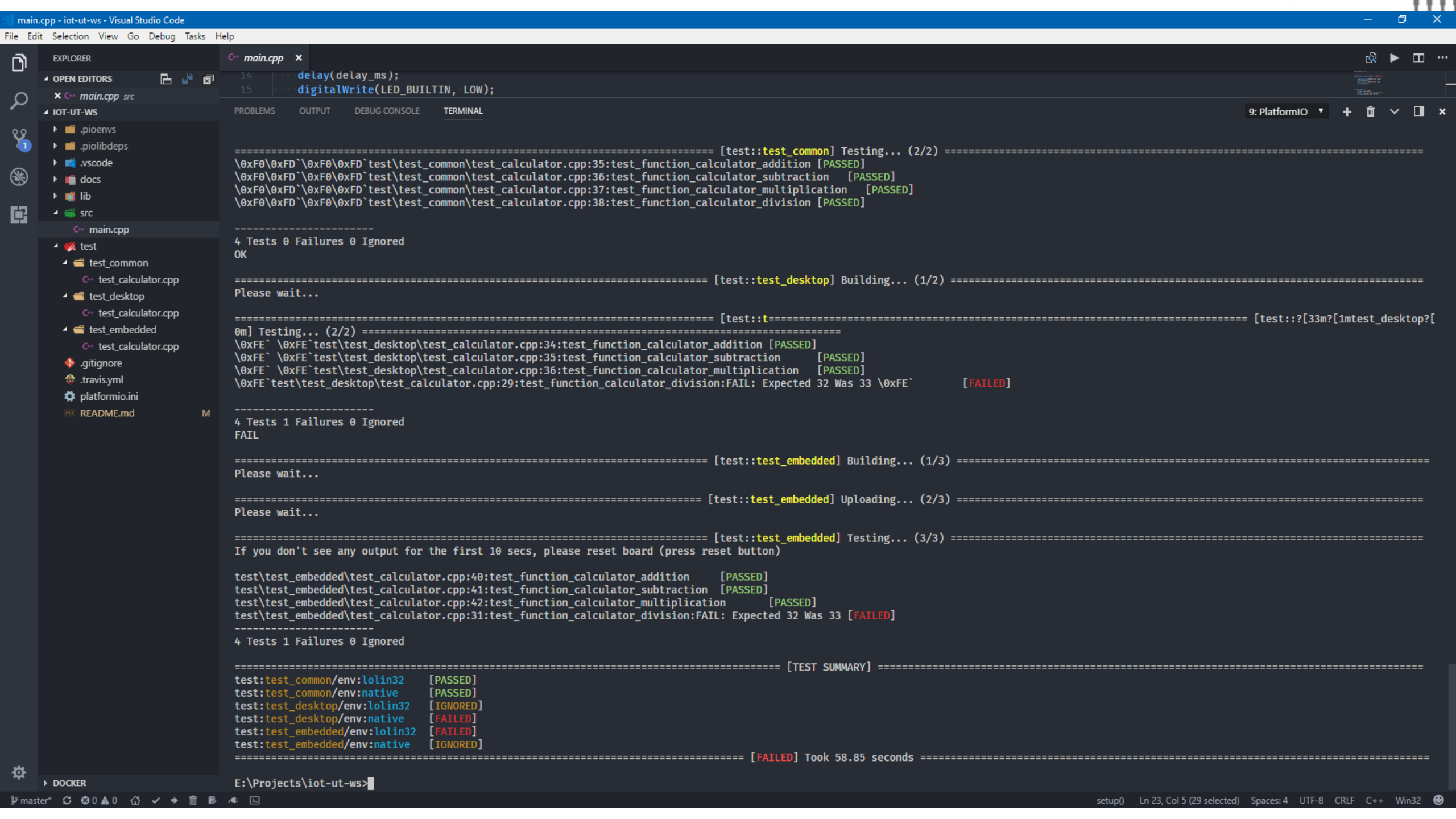
0 ⚠ 0


PIO Home - ws_unittests - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
PIO Home
WS_UNITTESTS
DOCKER
PIO Home
Have an account? Log in
Welcome to PlatformIO
Show at startup
Home
Account
Libraries
Boards
Platforms
Devices
Home 0.3.2 · Core 3.5.0b3
Web · Open Source · Get Started
Project Wizard
This wizard allows you to create new PlatformIO project or update existing. In the last case, you need to uncheck "Use default location" and specify path to existing project.
Board: Espressif ESP32 Dev Module
Framework: Arduino
Location: Use default location
Cancel
Finish
Recent Projects
Search project...
Name
Boards
Modified
Action
piheadquarters\arduino
NodeMCU 1.0 (ESP-12E Module)
about a month ago
Hide | Open
If you enjoy using PlatformIO, please star our projects on GitHub!
PlatformIO Core


main.cpp - 171121-184021-esp32dev - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
main.cpp src
171121-184021-ESP32DEV
lib
readme.txt
src
main.cpp
.gitignore
.travis.yml
platformio.ini
DOCKER
main.cpp
#include <Arduino.h>
void setup() {
// put your setup code here, to run once:
}
void loop() {
// put your main code here, to run repeatedly:
}
(Global Scope) Ln 9, Col 2 Spaces: 4 UTF-8 CRLF C++ Win32

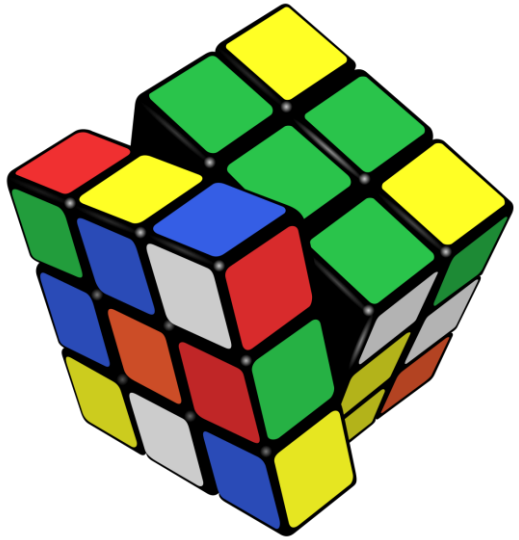
# Walkthrough

```
# Clone repository
> git clone https://github.com/jpdias/iot-ut-ws.git

# Change directory to example
> cd iot-ut-ws

# Open VS Code
> code .

# Test project
> platformio test

# Test specific environment
> platformio test -e uno

# Process test on native desktop machine
> platformio test -e native

# Deploy
> platformio run -e lolin32 -t upload
```


main.cpp - iot-ut-ws - Visual Studio Code
File Edit Selection View Go Debug Tasks Help
EXPLORER
OPEN EDITORS
main.cpp src
IOT-UT-WS
.pioenvs
.piolibdeps
.vscode
docs
lib
src
main.cpp
test
test_common
test_calculator.cpp
test_desktop
test_calculator.cpp
test_embedded
test_calculator.cpp
.gitignore
.travis.yml
platformio.ini
README.md M
DOCKER
main.cpp
delay(delay_ms);
digitalWrite(LED_BUILTIN, LOW);
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
9: PlatformIO
[test::test_common] Testing... (2/2)
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:35:test_function_calculator_addition [PASSED]
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:36:test_function_calculator_subtraction [PASSED]
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:37:test_function_calculator_multiplication [PASSED]
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:38:test_function_calculator_division [PASSED]
4 Tests 0 Failures 0 Ignored
OK
[test::test_desktop] Building... (1/2)
Please wait...
[test::t [test::?[33m?[1mtest_desktop?[0m] Testing... (2/2)
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:34:test_function_calculator_addition [PASSED]
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:35:test_function_calculator_subtraction [PASSED]
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:36:test_function_calculator_multiplication [PASSED]
\0xFE`test\test_desktop\test_calculator.cpp:29:test_function_calculator_division:FAIL: Expected 32 Was 33 \0xFE` [FAILED]
4 Tests 1 Failures 0 Ignored
FAIL
[test::test_embedded] Building... (1/3)
Please wait...
[test::test_embedded] Uploading... (2/3)
Please wait...
[test::test_embedded] Testing... (3/3)
If you don't see any output for the first 10 secs, please reset board (press reset button)
test\test_embedded\test_calculator.cpp:40:test_function_calculator_addition [PASSED]
test\test_embedded\test_calculator.cpp:41:test_function_calculator_subtraction [PASSED]
test\test_embedded\test_calculator.cpp:42:test_function_calculator_multiplication [PASSED]
test\test_embedded\test_calculator.cpp:31:test_function_calculator_division:FAIL: Expected 32 Was 33 [FAILED]
4 Tests 1 Failures 0 Ignored
[TEST SUMMARY]
test:test_common/env:lolin32 [PASSED]
test:test_common/env:native [PASSED]
test:test_desktop/env:lolin32 [IGNORED]
test:test_desktop/env:native [FAILED]
test:test_embedded/env:lolin32 [FAILED]
test:test_embedded/env:native [IGNORED]
[FAILED] Took 58.85 seconds
E:\Projects\iot-ut-ws>
master* 0 0
setup() Ln 23, Col 5 (29 selected) Spaces: 4 UTF-8 CRLF C++ Win32

# Exercise

1. Explore the code and folder structure.
2. Find the bugs in the test cases.
3. Run the tests and deploy to board when correct.



test_calculator.cpp - iot-ut-ws - Visual Studio Code

File Edit Selection View Go Debug Tasks Help

EXPLORER

OPEN EDITORS
- test_calculator.cpp test\te... M
- test_calculator.cpp test\te... M

IOT-UT-WS
- .pioenvs
- .piolibdeps
- .vscode
- docs
- lib
- src
  - main.cpp
- test
  - test_common
    - test_calculator.cpp
  - test_desktop
    - test_calculator.cpp M
  - test_embedded
    - test_calculator.cpp M
- .gitignore
- .travis.yml
- platformio.ini
- README.md M

DOCKER

test_calculator.cpp ...\test_desktop | test_calculator.cpp ...\test_embedded

```
12    // }
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL

9: PlatformIO

```
Please wait...

============================================================================= [test::test_common] Testing... (2/2) ==============================================================
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:35:test_function_calculator_addition [PASSED]
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:36:test_function_calculator_subtraction   [PASSED]
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:37:test_function_calculator_multiplication   [PASSED]
\0xF0\0xFD`\0xF0\0xFD`test\test_common\test_calculator.cpp:38:test_function_calculator_division [PASSED]

----------------------
4 Tests 0 Failures 0 Ignored
OK

============================================================================= [test::test_desktop] Building... (1/2) ============================================================
Please wait...

============================================================================= [test::test_desktop] Testing... (2/2) =============================================================
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:34:test_function_calculator_addition [PASSED]
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:35:test_function_calculator_subtraction      [PASSED]
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:36:test_function_calculator_multiplication   [PASSED]
\0xFE` \0xFE`test\test_desktop\test_calculator.cpp:37:test_function_calculator_division [PASSED]

----------------------
4 Tests 0 Failures 0 Ignored
OK

============================================================================= [test::test_embedded] Building... (1/3) ===========================================================
Please wait...

============================================================================= [test::test_embedded] Uploading... (2/3) ==========================================================
Please wait...

============================================================================= [test::test_embedded] Testing... (3/3) ============================================================
If you don't see any output for the first 10 secs, please reset board (press reset button)

test\test_embedded\test_calculator.cpp:40:test_function_calculator_addition     [PASSED]
test\test_embedded\test_calculator.cpp:41:test_function_calculator_subtraction  [PASSED]
test\test_embedded\test_calculator.cpp:42:test_function_calculator_multiplication       [PASSED]
test\test_embedded\test_calculator.cpp:43:test_function_calculator_division     [PASSED]
----------------------
4 Tests 0 Failures 0 Ignored

================================================================================ [TEST SUMMARY] =================================================================================
test:test_common/env:lolin32    [PASSED]
test:test_common/env:native     [PASSED]
test:test_desktop/env:lolin32   [IGNORED]
test:test_desktop/env:native    [PASSED]
test:test_embedded/env:lolin32  [PASSED]
test:test_embedded/env:native   [IGNORED]
================================================================================ [PASSED] Took 61.17 seconds ====================================================================

E:\Projects\iot-ut-ws>
```

master* 0 0 | test_function_calculator_division(void) Ln 31, Col 25 Spaces: 4 UTF-8 CRLF C++ Win32

# Further Reading

- PIO Unit Testing: http://docs.platformio.org/en/latest/plus/unit-testing.html
- throwtheswitch Tools: https://www.throwtheswitch.org/
- Testing the Internet of Things - Test and Verification Solutions: https://www.testandverification.com/wp-content/uploads/Testing%20the%20Internet%20of%20Things.pdf
- Embedded Software Testing - CMU: https://www.ece.cmu.edu/~ece649/lectures/10_testing.pdf

# Thank you.

## TQS – ProDEI
## 24/November/2017

João Pedro Dias

jpmdias@fe.up.pt

Researcher @ INESC TEC

U.PORTO
FEUP FACULDADE DE ENGENHARIA
UNIVERSIDADE DO PORTO