Home Forums What's new Members

Log in Register Search

Home > Forums > Main Category > **General Discussion** >

[Tutorial] How to use PlatformIO / Visual Code Studio for Teensy

flocked · Mar 21, 2021

F

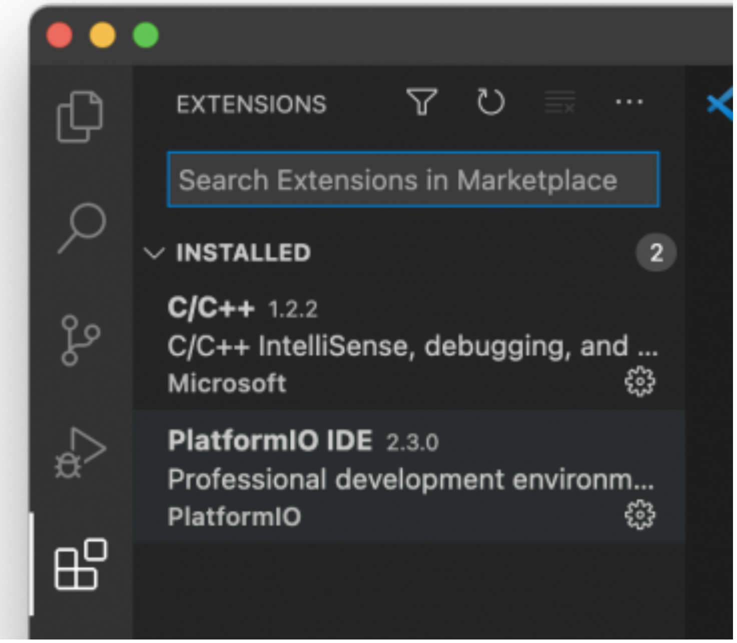
flocked
Active member

Mar 21, 2021 #1

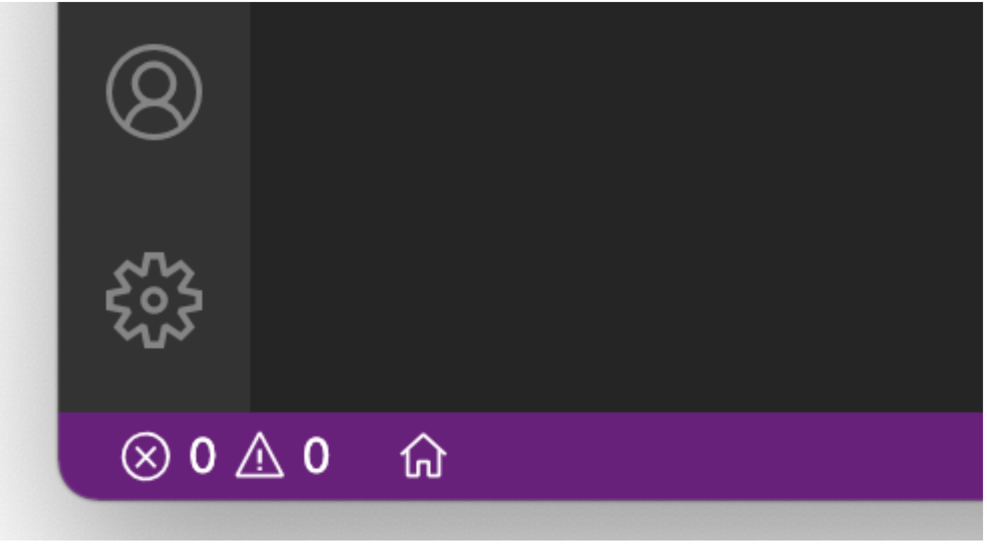
I only recently discovered PlatformIO and I’m so happy about the improved workflow and features compared to the editor of Arduino. No more slow app launch, having auto complete, much better UI and being able to install useful editor extensions is awesome.

I stupidly always thought PlatformIO would be hard to install, configurate and therefor never tried it out. I think it’s mainly the websites fault. But turns out: The opposite is true. It couldn’t be easier to install. Here a small tutorial that hopefully motivates more people to try it out.

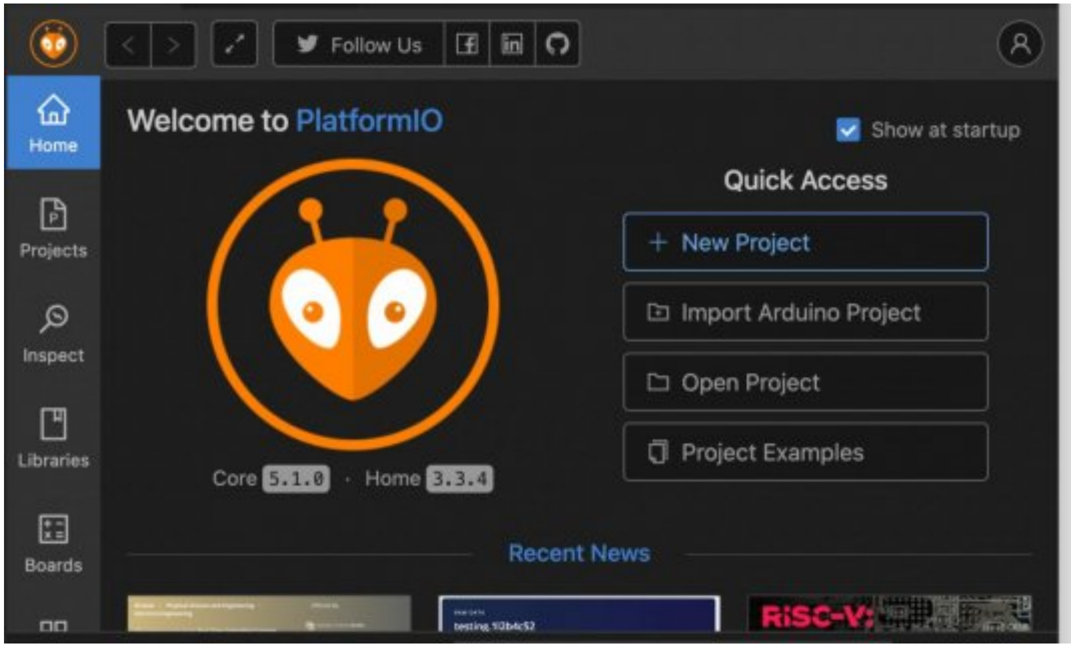
Installation
1) Get Visual Code Studio. It’s free and open source.
2) Go to the extension library (the 4th icon on the sidebar), search for PlatformIO and install it. Congrats you have PlatformIO installed.


EXTENSIONS
Search Extensions in Marketplace
INSTALLED
2
C/C++ 1.2.2
C/C++ IntelliSense, debugging, and ...
Microsoft
PlatformIO IDE 2.3.0
Professional development environm...
PlatformIO

PlatformIO main page
PlatformIO adds a purple/blue toolbar to the bottom of the window. The home button opens PlatformIO.

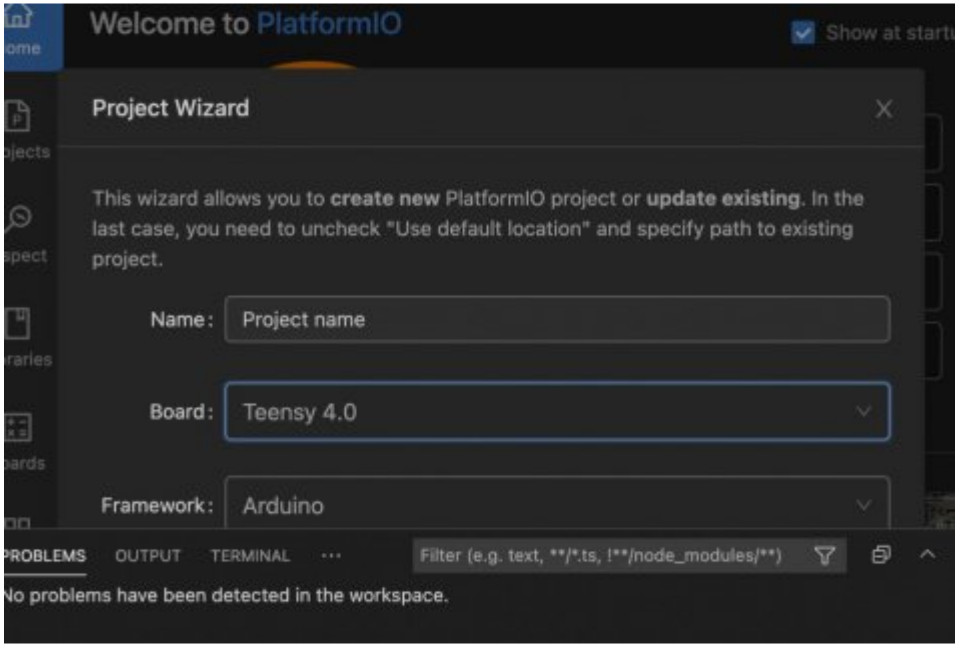


The main page allows to browse all your projects, create new ones (even import Arduino ones), browse libraries and find documentation about boards.



Creating a new project

To create a new project simply press on „New project“ on the main page. Select the board you want to use and it automatically creates a project.



Welcome to PlatformIO

☑ Show at startu

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.

Name: Project name

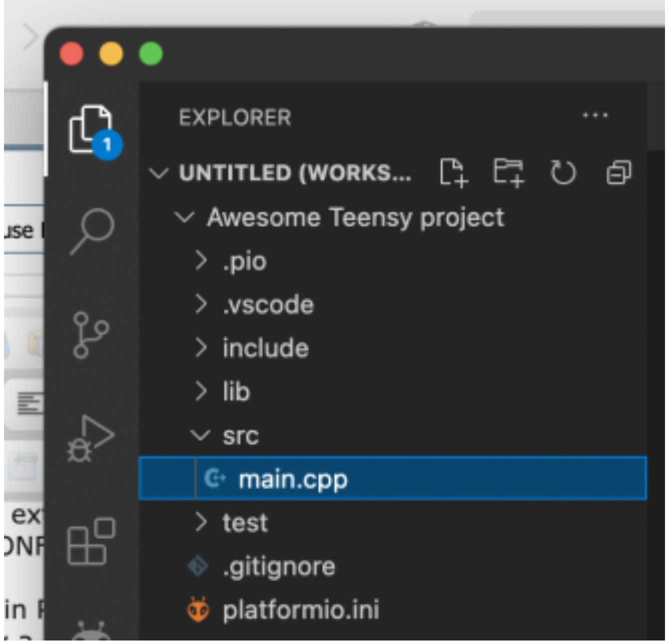
Board: Teensy 4.0

Framework: Arduino

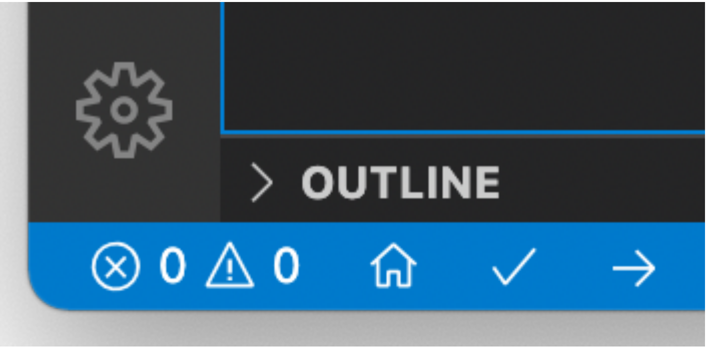
No problems have been detected in the workspace.

Project

The main file is underneath src and is called main.cpp. All other files on the projects sidebar don’t matter to us.

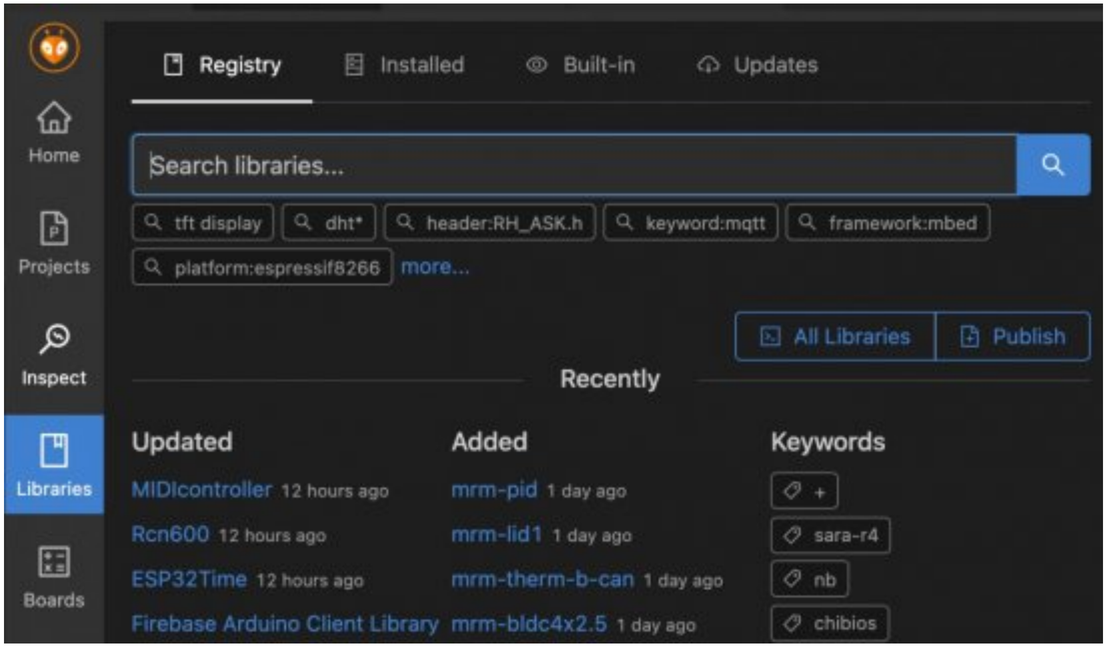


To compile a project click the checkmark and to upload the arrow on the bottom toolbar.



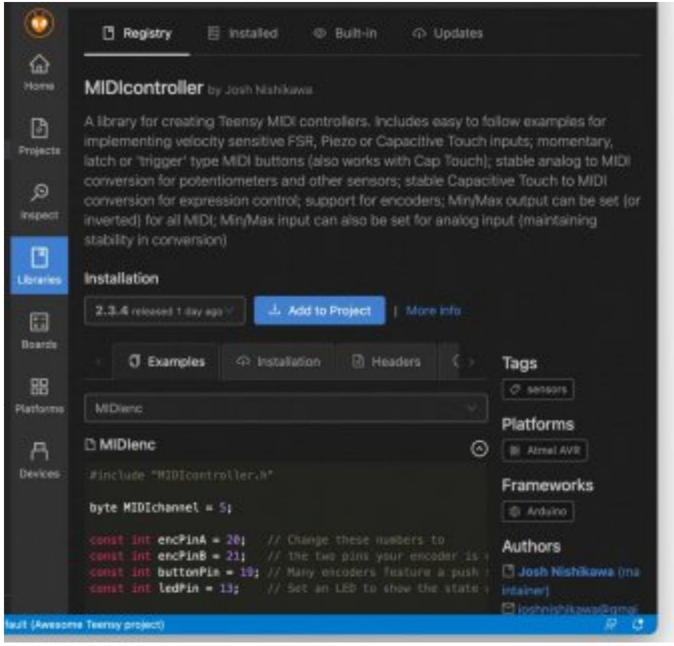
Add libraries to projects

If you want to add additional libraries PlatformIO provides it's own browser for libraries. Go to PlatformIOs main page via blue home button and click on libraries. It provides all libraries that Arduino also provides in it's editor plus additional.



The overview of each library is much more detailed and better compared to Arduino. You can already take a look inside the files before installing, see changelogs and take a look at examples. To install an extension simply click on

„add to project“



MIDIcontroller by Josh Nishikawa

A library for creating Teensy MIDI controllers. Includes easy to follow examples for implementing velocity sensitive FSR, Piezo or Capacitive Touch inputs; momentary, latch or 'trigger' type MIDI buttons (also works with Cap Touch); stable analog to MIDI conversion for potentiometers and other sensors; stable Capacitive Touch to MIDI conversion for expression control; support for encoders; Min/Max output can be set (or inverted) for all MIDI; Min/Max input can also be set for analog input (maintaining stability in conversion)

MIDIenc

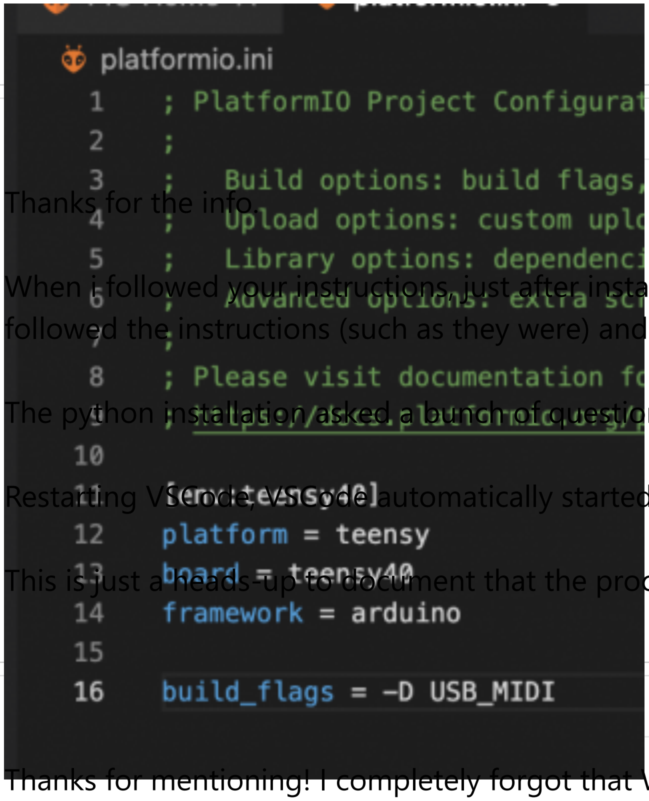
```
#include "MIDIcontroller.h"

byte MIDIchannel = 5;

const int encPinA = 20;   // Change these numbers to
const int encPinB = 21;   // the two pins your encoder is
const int buttonPin = 19; // Many encoders feature a push
const int ledPin = 13;    // Set an LED to show the state
```

Teensy USB type
Open the PlatformIO.ini file from the projects sidebar. And add this entry:

Code:

```
build_flags = -D USB_MIDI
```

The correct build_flag depends on the usb mode you want. The teensy platform documentation inside the platforms browser lists all.
Yes this step is unfortunately a bit less comfortable compared to Arduino. But at least the the selected usb type gets saved inside the projects.



```
platformio.ini
; PlatformIO Project Configurat
;
;   Build options: build flags,
;   Upload options: custom uplo
;   Library options: dependenci
;   Advanced options: extra scr
;
; Please visit documentation fo
; https://docs.platformio.org/

[env:teensy40]
platform = teensy
board = teensy40
framework = arduino

build_flags = -D USB_MIDI
```

C

chipaudette
Well-known member

#2

Thanks for the info.

When i followed your instructions, just after installing the PlatformIO extension, it said that it needed Python. I followed the instructions (such as they were) and chose to install the most recent Python release from python.org.

The python installation asked a bunch of questions that I hope I got right.

Restarting VSCode, VSCode automatically started installing some additional Python stuff.

This is just a heads-up to document that the process may have additional steps for some users.

Last edited: Mar 22, 2021

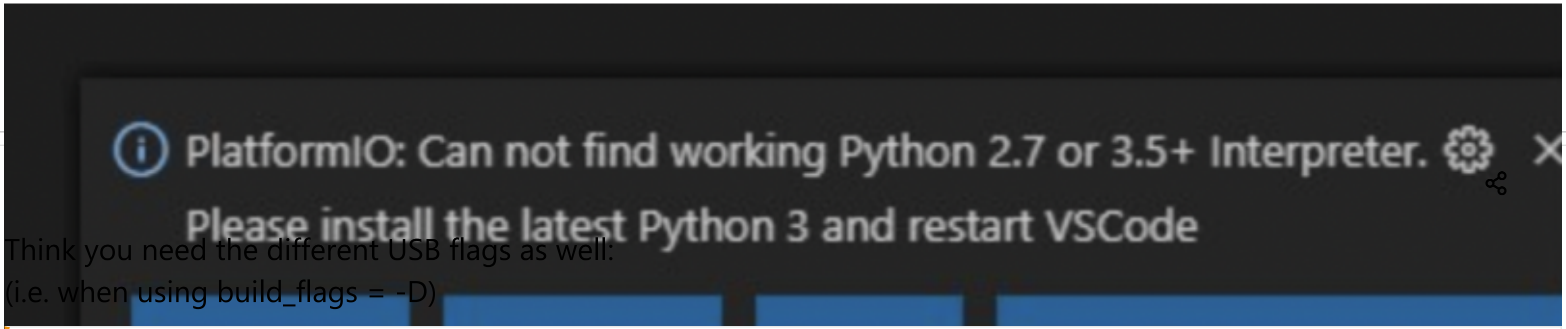
F

flocked
Active member

#3

Thanks for mentioning! I completely forgot that Windows doesn't have Python preinstalled. At least Visual Code Studio notifies during PlatformIO installation if Python is missing and offers to install it.
Where are the code examples for boards and libraries?
Arduino editor provides examples for its boards and installed libraries can also provide examples.
In PlatfromIO examples for platforms/boards can be found by selecting the specific platform inside the platforms browser. Examples for libraries can be found by selecting a specific library in the libraries browser.

Attachments



PlatformIO: Can not find working Python 2.7 or 3.5+ Interpreter.
Please install the latest Python 3 and restart VSCode

manicksan
Well-known member

Think you need the different USB flags as well:
(i.e. when using build_flags = -D)

Code:

```
Build-flag                | Description (as in Arduino IDE)
--------------------------|--------------------------
USB_SERIAL                | Serial
USB_DUAL_SERIAL           | Dual Serial
USB_TRIPLE_SERIAL         | Triple Serial
USB_KEYBOARDONLY          | Keyboard
USB_TOUCHSCREEN           | Keyboard + Touch Screen
USB_HID_TOUCHSCREEN       | Keyboard + Mouse + Touch Screen
USB_HID                   | Keyboard + Mouse + Joystick
USB_SERIAL_HID            | Serial + Keyboard + Mouse + Joystick
USB_MIDI                  | MIDI
USB_MIDI4                 | MIDIx4
USB_MIDI16                | MIDIx16
USB_MIDI_SERIAL           | Serial + MIDI
USB_MIDI4_SERIAL          | Serial + MIDIx4
USB_MIDI16_SERIAL         | Serial + MIDIx16
USB_AUDIO                 | Audio
USB_MIDI_AUDIO_SERIAL     | Serial + MIDI + Audio
USB_MIDI16_AUDIO_SERIAL   | Serial + MIDIx16 + Audio
USB_MTPDISK               | MTP Disk (Experimental)
USB_RAWHID                | Raw HID
USB_FLIGHTSIM             | Flight Sim Controls
USB_FLIGHTSIM_JOYSTICK    | Flight Sim Controls + Joystick
USB_EVERYTHING            | All of the Above (only available @ teensy 3.1 3.5 3.6)
USB_DISABLED              | No USB
```

note. the flags cannot be mixed and only one can be active at a time.

example:
build_flags = -D USB_MIDI_SERIAL

also the USB_EVERYTHING is not supported by teensy 4.x
because of hardware limits.
https://forum.pjrc.com/threads/6627...-on-Teensy-4-0?p=270343&viewfull=1#post270343

Last edited: Mar 22, 2021

R

RoSchmi
Member

Mar 22, 2021 #5

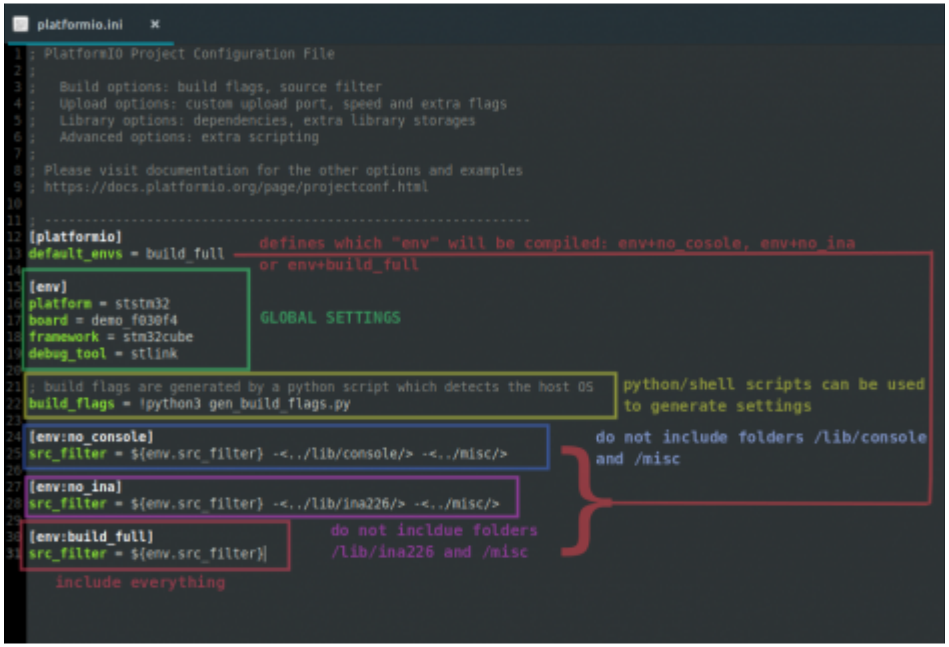
Thanks for the nice tutorial. I can confirm that PlatformIo is really convenient.
I think that hardly anyone who has used it for a time will go back to the Arduino IDE.
Especially I like the Auto Complete function and the possibility of debugging, for some boards direct, for many other boards with e.g. a jlink-debugger (unfortunately not yet my Teensy 4.1).
Setting breakpoints and inspecting variables by positioning the cursor on them makes programming so much easier.
Worth to mention is that VS Code is available for MAC and Linux as well (even though I never tried)

P

Pio
Well-known member

Mar 22, 2021 #6

What i find extremely useful in Platformio:
1. local to the project libraries placed in the */lib* folder. Compiler will look into this folder first and if an included library is not found, only then look in the global path. This way i can have my customized versions of libraries per project, not interfering with the globally installed/stock ones. Packaging a complete project is much simpler if all required libs are stored locally.
2. things the plarformio.ini file can do, and there is a lot. As an example:



```
; PlatformIO Project Configuration File
;
;   Build options: build flags, source filter
;   Upload options: custom upload port, speed and extra flags
;   Library options: dependencies, extra library storages
;   Advanced options: extra scripting
;
; Please visit documentation for the other options and examples
; https://docs.platformio.org/page/projectconf.html

; ------------------------------------------------------------
[platformio]
default_envs = build_full

[env]
platform = ststm32
board = demo_f030f4
framework = stm32cube
debug_tool = stlink

; build flags are generated by a python script which detects the host OS
build_flags = !python3 gen_build_flags.py

[env:no_console]
src_filter = ${env.src_filter} -<../lib/console/> -<../misc/>

[env:no_ina]
src_filter = ${env.src_filter} -<../lib/ina226/> -<../misc/>

[env:build_full]
src_filter = ${env.src_filter}
```

defines which "env" will be compiled: env+no_cosole, env+no_ina or env+build_full

GLOBAL SETTINGS

python/shell scripts can be used to generate settings

do not include folders /lib/console and /misc

do not incldue folders /lib/ina226 and /misc

include everything

It could be very useful when working on new library versions, comparing them against the old, currently installed within the framework ones.
Simply add the new version in the /lib folder and create a conditional build, ie:

Code:

```
[platformio]
default_builds = stock_lib
[env]
; define board, framework, platform etc here
[env:stock_build]
src_filter = ${env.src_filter} -<../lib/new_lib_test_version/>
```

If the *default_libs* is set to *stock_lib*, the local library folder will not be included and the compiler will grab+link the global version.

G

GrosDede
Member

Hello,

Thanks for the tutorial, I have dropped Arduino IDE. (Maybe arduino 2.0 would be better)

3. The fact i can use the same VScode for many other non arduino projects.

Next step would be to have debugging available, to avoid "println" debug messages The PlatformIO single way to debug teensy costs more than I can afford.

One important fact not yet mentioned is platformio requires the functions to be declared, just as with normal c/c++.

Simply copying the ino file into main will most likely fail to compile due to lack of function declarations.

Seems that Visual Micro already has debugging support.

Edit : Maybe found a solution to test : https://github.com/ftrias/TeensyDebug

J

joepasquariello
Well-known member

Jun 27, 2022 #8

I've tried PlatformIO a couple of times, but ended up going back to Arduino. Arduino has lots of issues, but I do understand how to install it and maintain it. Can you explain where in the PlatformIO process was TeensyDuino installed, and how can you update to a new or beta version of TeensyDuino?

G

GrosDede
Member

Jun 27, 2022 #9

I have installed latest stable Arduino IDE (1.8.19) and Teensyduino (1.56). The problems I have with Arduino IDE is productivity : code edition, selection, completion, navigation... For me, VSCode is really faster. I did not searched for an alternative, but I was curious what platformio was, so I tried. I saw vscode extension, with teensy integration. I installed the vscode extension and tried an empty project.

One feature I would love to see is debugging (breakpoints, see variables contents...).

For example, I am developping a small device based on teensy, that can be configured using USB. So I have a golang app that detects teensy serial ports, and then interract with it though commands. It acts as wrapper with teensy serial port and provides rest apis, for an angular front-end. I have some data exchanged as JSON. I can have vscode as debugger for backend and frontend, but no debugger for teensy (yet).

BriComp
Well-known member

Jun 28, 2022 #10

I would recommend Visual Micro with Visual Studio. It gives the stability/understanding of the Arduino system with the IDE of visual studio and Teensy Debugging.
I use it all the time and would not go back. I have used the Arduino IDE and PlatformIO but as you say not impressed.
Any libraries or code added to Arduino fall straight through to VisualMicro/Visual Studio.
See here.

joepasquariello
Well-known member

Jun 28, 2022 #11

> BriComp said:
>
> I would recommend Visual Micro with Visual Studio. It gives the stability/understanding of the Arduino system with the IDE of visual studio and Teensy Debugging.
> I use it all the time and would not go back. I have used the Arduino IDE and PlatformIO but as you say not impressed. Any libraries or code added to Arduino fall straight through to VisualMicro/Visual Studio.
> See here.

Thanks for pointing out that VM/VS uses what has already been installed for Arduino. I find that much preferable to the PlatformIO approach, and I'll try it again.

Lestra
Member

Jul 28, 2022 #12

I am in VS + VM now and can not find Teensy Examples. Please help.

Lestra
Member

Jul 29, 2022 #13

Found it, everything arduino/teensy is under vMicro toolbar, of course.

Oct 27, 2022 #14

Attempting PlatformIO on some existing T3 code. Get:

> Code:

bicycleguy
Well-known member

```
C/C++ IntelliSense service does not support .INO files. It might lead to the spurious problems with code co
```

A link suggests to add declarations to all my functions, lots of them. Am I missing something?

W

WMXZ
Well-known member

Oct 27, 2022 #15

@bicycleguy
if you wanted to use the advantage of VSCode (as I do), simply move all code from ino to a cxx file and let ino be empty.
Obviously, your new cxx file must be logically correct an include all declaration int the right order (better cxx programming anyway).

Arduino converts ino file to temporary ino.cxx files that are cxx correct, so ino cannot be understood by intellisense.

Do so you can use VSCode and if you want also Arduino for compiling. Arduino is happy if there is an ino file even if empty.

D

DroneOverLord
Active member

Nov 5, 2022 #16

I'm trying out PlatformIO. My program compiles and loads to a Teensy 4.1 with no issues.

Is there a serial monitor function in VS/PlatformIO that I can use or are people just using Putty or something?

S

shawn
Well-known member

Nov 9, 2022 #17

I use Serial from Decisive Tactics (https://www.decisivetactics.com/products/serial/). PlatformIO does have a “monitor” task (or “upload and monitor”), but I prefer Serial.

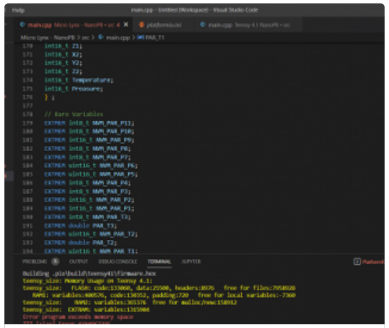
D

DroneOverLord
Active member

Nov 14, 2022 #18

shawn, thanks for the reply. I'm using the built-in serial monitor in platformio, it's the icon near the home button that looks like a plug.

Attachments


Help
main.cpp - Untitled (Workspace) - Visual Studio Code
main.cpp Micro Lynx - NanoPB • src 4
platformio.ini
main.cpp Teensy 4.1 NanoPB • src
Micro Lynx - NanoPB > src > main.cpp > PAR_T1
PROBLEMS 5
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
PlatformIO

```
int16_t Z1;
int16_t X2;
int16_t Y2;
int16_t Z2;
int16_t Temperature;
int16_t Pressure;
} ;

// Baro Variables
EXTMEM int8_t NVM_PAR_P11;
EXTMEM int8_t NVM_PAR_P10;
EXTMEM int16_t NVM_PAR_P9;
EXTMEM int8_t NVM_PAR_P8;
EXTMEM int8_t NVM_PAR_P7;
EXTMEM uint16_t NVM_PAR_P6;
EXTMEM uint16_t NVM_PAR_P5;
EXTMEM int8_t NVM_PAR_P4;
EXTMEM int8_t NVM_PAR_P3;
EXTMEM int16_t NVM_PAR_P2;
EXTMEM int16_t NVM_PAR_P1;
EXTMEM int8_t NVM_PAR_T3;
EXTMEM double PAR_T3;
EXTMEM uint16_t NVM_PAR_T2;
EXTMEM double PAR_T2;
EXTMEM uint16_t NVM_PAR_T1;
```

```
Building .pio\build\teensy41\firmware.hex
teensy_size: Memory Usage on Teensy 4.1:
teensy_size:   FLASH: code:133060, data:25500, headers:8976   free for files:7958920
   RAM1: variables:400576, code:130352, padding:720   free for local variables:-7360
teensy_size:   RAM2: variables:365376  free for malloc/new:158912
teensy_size:  EXTRAM: variables:1315904
Error program exceeds memory space
```

Capture.PNG

M

mad4linux
New member

Apr 9, 2024 #19

I'm using platformio with qt creator on linux. Does someone know, how i could start teensy loader automatically after building my hex-file for teensy? I could add a custom build step. Would something line "<path to>/teensy <path to>/firmware.hex" work?
Cannot test right now (missing cable) but should probably work...
After testing this command, it seems that the teensy program doesn't open te file given on the command line. So I can start the loader, but have to select the file manually.

Last edited: Apr 23, 2024

You must log in or register to reply here.

Share:

Home > Forums > Main Category > **General Discussion** >

Terms and rules Privacy policy Help Home