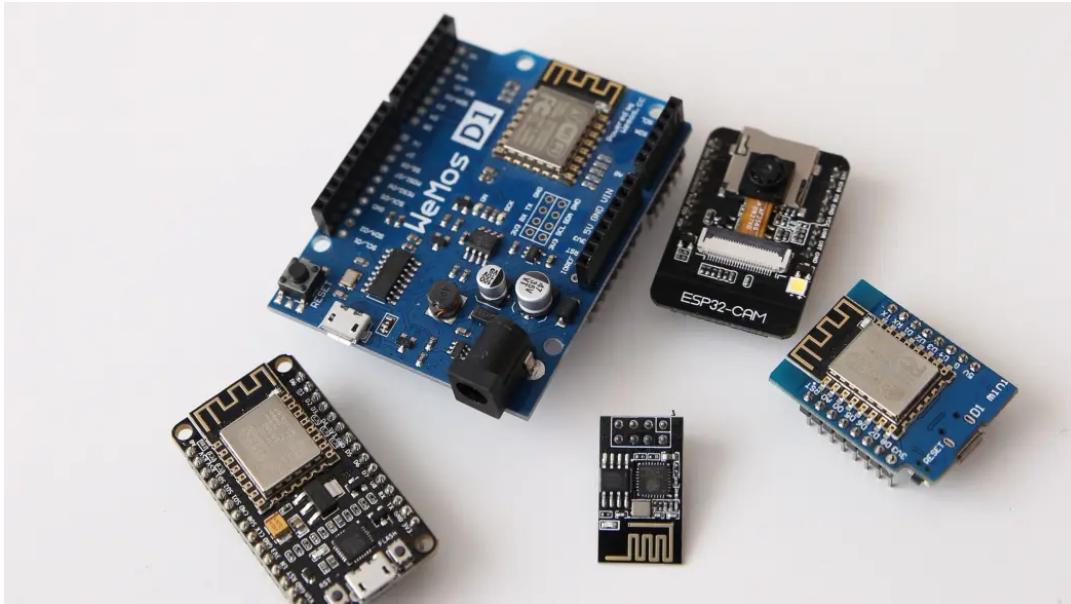


# Make:

## Mikrocontroller: Programmier-Umgebung PlatformIO installieren

06.01.2023 07:53 Uhr Heinz Behling



Die Arduino-IDE gelangt bei der Programmierung zahlreicher Mikrocontroller an Grenzen. Grund genug, sich mit etwas Leistungsfähigerem zu befassen: PlatformIO.

Die beliebte Arduino-IDE (Integrated Development Environment) gelangt nicht nur bei den neuesten Arduino- und ESP-Boards an ihre Grenzen: Zahlreiche andere moderne Controller-Boards, wie sie beispielsweise in 3D-Druckern sitzen, überfordern inzwischen die stets etwas stiefmütterlich gepflegte und so unkomfortabel gewordene Programmierumgebung (da kann auch die inzwischen **endlich veröffentlichte Version 2.0 [1]** das Ruder nicht komplett herumreißen). Grund genug, zum moderneren PlatformIO zu wechseln.

```
70 ArduinoOTA.onEnd([]() {
71   Serial.println("\nEnd");
72 });
73 ArduinoOTA.onProgress([](unsigned int progress, unsigned int total) {
74   Serial.printf("Progress: %u%%\r", (progress / (total / 100)));
75 });
76 ArduinoOTA.onError([](ota_error_t error) {
77   Serial.printf("Error[%u]: ", error);
78   if (error == OTA_AUTH_ERROR) Serial.println("Auth Failed");
79   else if (error == OTA_BEGIN_ERROR) Serial.println("Begin Failed");
80   else if (error == OTA_CONNECT_ERROR) Serial.println("Connect Failed");
81   else if (error == OTA_RECEIVE_ERROR) Serial.println("Receive Failed");
82   else if (error == OTA_END_ERROR) Serial.println("End Failed");
83 });
84 ArduinoOTA.begin();
85
86 // Internet-Zeit holen
87 configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
88
```

Die Arduino-IDE: Aufgeräumt, aber nur mit dem allerwichtigsten ausgestattet (hier noch im jahrelang vertrauten Erscheinungsbild der Versionen 1.X).

Im folgenden erfahren Sie, wie Sie die neue Entwicklungsumgebung inklusive wichtiger Erweiterungen installieren und bedienen.

[2]

## Was macht eine Programmier-Umgebung aus?

Programme in C++, Python oder anderen Programmiersprachen kann man selbst mit einfachen Texteditoren wie Notepad schreiben. Doch das genügt nicht, um daraus eine fehlerfreie Firmware für einen Mikrocontroller zu machen. So sollten beispielsweise Befehlsworte der jeweiligen Programmiersprache (meist farblich) hervorgehoben werden, damit man etwa Tippfehler leichter erkennt. Automatische Einrückungen einzelner Zeilen oder ganzer Absätze, die in Python beispielsweise sehr wichtig sind, findet man in einfachen Editoren ebenfalls nicht.

Grundlage der hier besprochenen IDE ist **Visual Studio Code (VS Code)**, eine kostenlose Opensource-Software von Microsoft. (Bitte nicht verwechseln mit Visual Studio, ebenfalls von Microsoft, aber nicht Open Source und vor allem als Entwicklungsumgebung für Windows- und Smartphone-Apps gedacht.)

Allein ist VS Code aber im Grunde nur ein Programmtext-Editor und ansonsten erst mal nutzlos. Schließlich muss aus dem Programmtext (dem Sourcecode) ja auch noch eine auf dem jeweiligen Board lauffähige Firmware produziert werden. Dazu müssen etwa Software-Bibliotheken mit eingebunden werden, die auf spezifische Bestandteile des jeweiligen Boards zugeschnitten sind. Erst damit ist dann die Ansteuerung von einzelnen Anschlusspins, Peripherie-Bausteinen oder Netzwerk-Chips möglich. Die Programmierumgebung sollte diese Bibliotheken bereitstellen, idealerweise automatisch zum jeweiligen Board passend. Die Arduino-IDE liefert bereits einige mit, die aber auf Arduinos zugeschnitten sind, denn nur diese Boards kennt die IDE von Haus aus. Selbst die Unterstützung für ESP-Boards muss von Hand eingerichtet werden. Und zahlreiche Bibliotheken für diese Boards muss man dann auch noch händisch installieren oder gar erst im Internet suchen.

Schließlich muss dann der Programmtext inklusive der aus den Bibliotheken übernommenen Bestandteile in eine funktionsfähige Firmware für den Mikrocontroller übersetzt werden. Dies

übernimmt ein Compiler. Programmier-Umgebungen enthalten immer mindestens einen davon, oft aber auch für jede benutzbare Programmiersprache je einen eigenen.

Den Compiler könnte man ohne weiteres händisch per Befehlszeile steuern. Das ist aber angesichts einer nahezu unüberblickbaren Anzahl von Einstellungsparametern recht mühsam und fehlerträchtig. Das sollte daher von der IDE übernommen werden. In dieser Anleitung übernimmt PlatformIO die Steuerung all dieser Aufgaben. Mit dieser Erweiterung wird VS Code erst zur PlatformIO-IDE.

Und noch eine Erweiterung werden wir in dieser Anleitung installieren: git. Das ist eine Software zur Versionsverwaltung von Dateien. Sie wird sich später nicht nur um die Versionen Ihres Software-Projekts kümmern, sondern vor allem um die Versionen der benutzten Software-Bibliotheken.

Falls Sie Erfahrungen mit der Arduino-IDE haben, kennen Sie sicher den Effekt, dass trotz Installation aller benötigten Bibliotheken der Kompiliervorgang mit etlichen Fehlermeldungen über fehlende Funktionen oder falsche Parameter abbricht. In den allermeisten Fällen liegt dies nicht an Ihrem Programmtext, sondern an Bibliotheks-Versionen, die im Funktionsumfang voneinander abweichen. Ihr Programm lässt sich dann nur mit einer bestimmten Versionsnummer der Bibliothek übersetzen.

In PlatformIO kann git sich um die Beschaffung der richtigen Bestandteile kümmern. Dies funktioniert in der Regel tadellos. Probleme gibt es dabei höchstens, wenn man ein Programm-Projekt von einem anderen Autor importiert, der zum Beispiel als Pfad zu einer Bibliothek einen Ordner seiner Festplatte benannt hat. Die kann dann natürlich auf einem anderen PC nicht gefunden werden.

Bei der folgenden Anleitung kann es übrigens zu Abweichungen gegenüber Ihrem Computer kommen. Grund dafür ist insbesondere, dass ein eventuell schon einmal auf dem Computer installiertes, aber wieder deinstalliertes VS Code Spuren hinterlässt, die das Verhalten und insbesondere das Aussehen der Programm-Oberfläche bei einer erneuten VS-Code-Installation beeinflussen. Diese Spuren sind meist in versteckten Ordnern gespeichert. Im Notfall hilft da eine Internet-Suche mit dem Stichwort *VSCode vollständig beseitigen*.

## Installation von VisualStudio Code

Wie gesagt, ist VS Code die Grundlage der PlatformIO-IDE. Daher wird es auch als erstes installiert. Auf der **Download-Seite [3]** stehen neben der Windows- auch Linux- und macOS-Versionen bereit. Im Folgenden verwenden wir die Windows-Ausgabe. Die Installation ist bei den anderen Betriebssystemen recht ähnlich.

The screenshot shows the official Visual Studio Code website. At the top, there's a navigation bar with links like 'Visual Studio Code', 'Docs', 'Updates', 'Blog', 'API', 'Extensions', 'FAQ', and 'Learn'. A search bar says 'Search Docs' and a 'Download' button is available. A message at the top states 'Version 1.74 is now available! Read about the new features and fixes from November.' Below this, a large banner with the text 'Code editing. Redefined.' and the subtitle 'Free. Built on open source. Runs everywhere.' is displayed. To the right of the banner is a 'Download for Windows' section with 'Stable Build' and 'Insiders' options. Below this are download links for macOS, Windows x64, and Linux x64, with 'Universal' and specific file formats (.deb, .rpm) listed. A 'Other downloads or open on web' link is also present. On the right side of the page, the VS Code interface is shown with several tabs open: 'serviceWorker.js - create-react-app - Visual Studio Code - In...', 'App.js', 'index.js', and 'serviceWorker.js'. The code editor shows some JavaScript code related to service workers. The bottom of the page includes a terminal window showing 'node' and some local network information.

Per Klick auf den Pfeil neben dem Download-Feld erreichen Sie die macOS- und Linuxversionen von VS Code.

Nach dem Start des Installationsprogramms müssen Sie zunächst der Lizenzvereinbarung zustimmen.

The screenshot shows the 'Setup - Microsoft Visual Studio Code (User)' window. The title bar has the Microsoft logo and the text 'Setup - Microsoft Visual Studio Code (User)'. The main area is titled 'Lizenzvereinbarung' (License Agreement). It contains a text box with the following content:

Diese Lizenz gilt für das Produkt „Visual Studio Code“. Der Quellcode für Visual Studio Code ist verfügbar unter <https://github.com/Microsoft/vscode> und unterliegt der MIT-Lizenzvereinbarung unter <https://github.com/microsoft/vscode/blob/main/LICENSE.txt>. Zusätzliche Lizenzinformationen finden Sie im Bereich „Häufig gestellte Fragen“ unter <https://code.visualstudio.com/docs/supporting/faq>.

**MICROSOFT-SOFTWARE-LIZENZBEDINGUNGEN**

**MICROSOFT VISUAL STUDIO CODE**

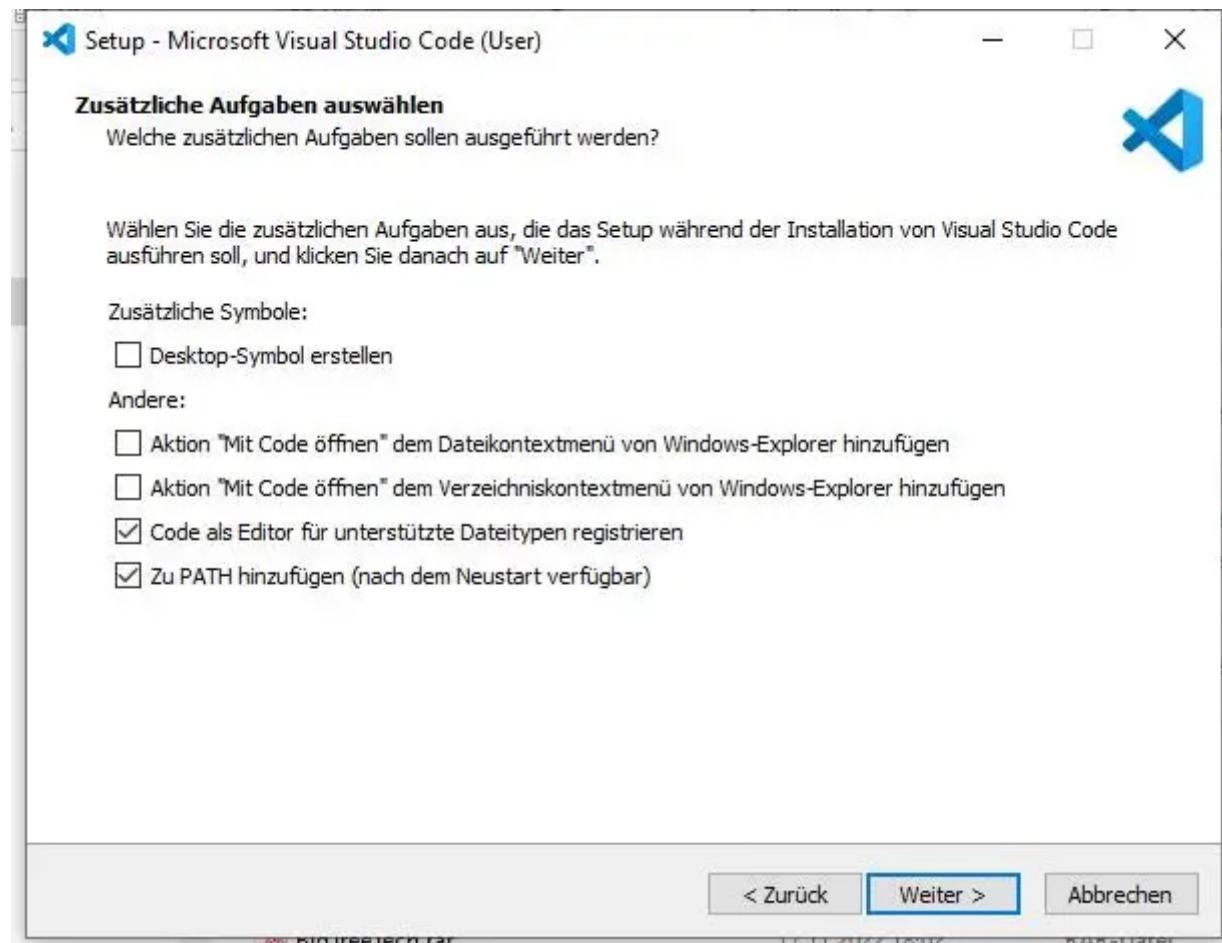
At the bottom, there are two radio buttons:

- Ich akzeptiere die Vereinbarung
- Ich lehne die Vereinbarung ab

At the very bottom are two buttons: 'Weiter >' and 'Abbrechen'.

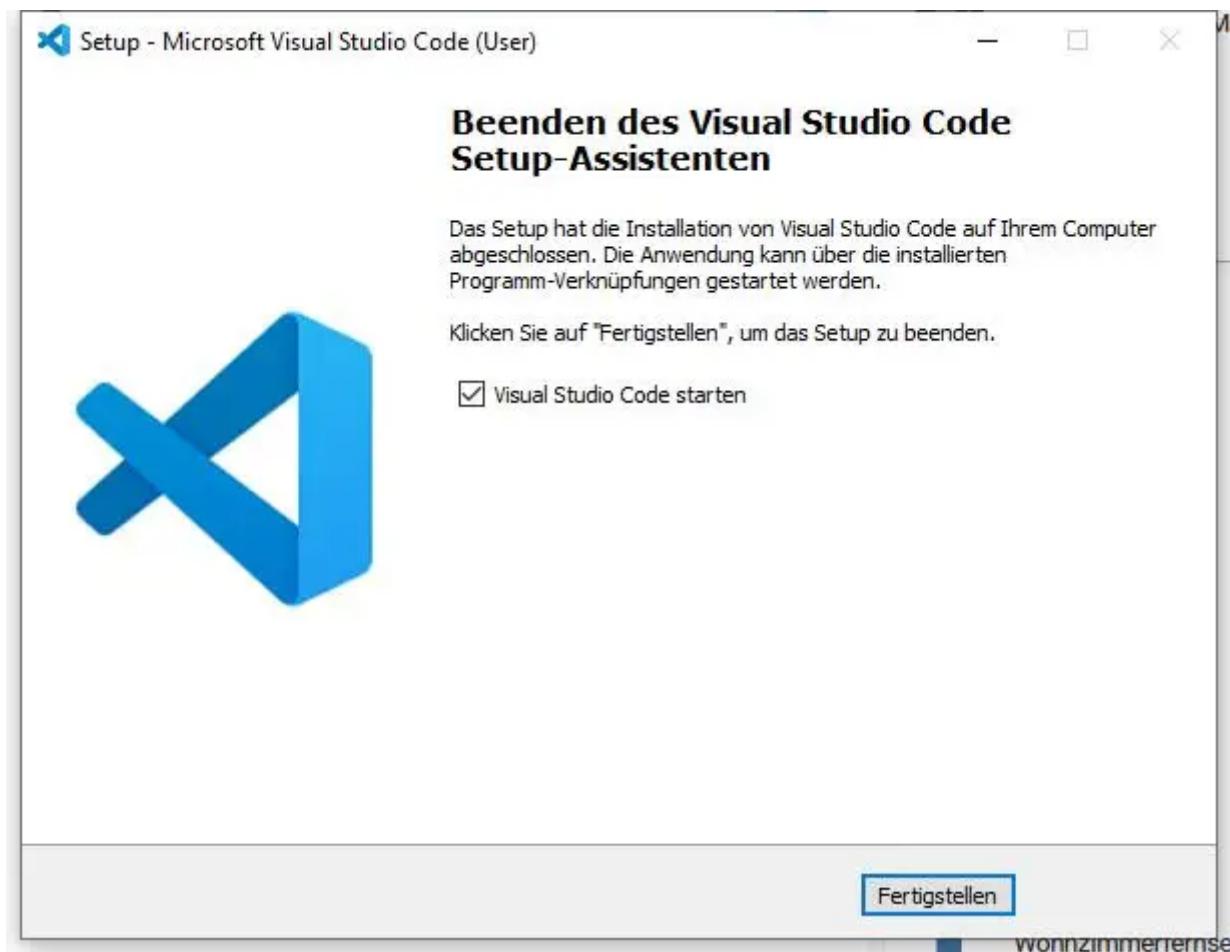
Um die Lizenzvereinbarung kommen Sie nicht herum.

Drei Klicks auf *Weiter* später (dabei zeigt Ihnen eines der Fenster den Pfad zum künftigen Projektverzeichnis an, bitte merken!) sollten Sie *Code als Editor für unterstützte Dateitypen registrieren* und *zum PATH hinzufügen*.



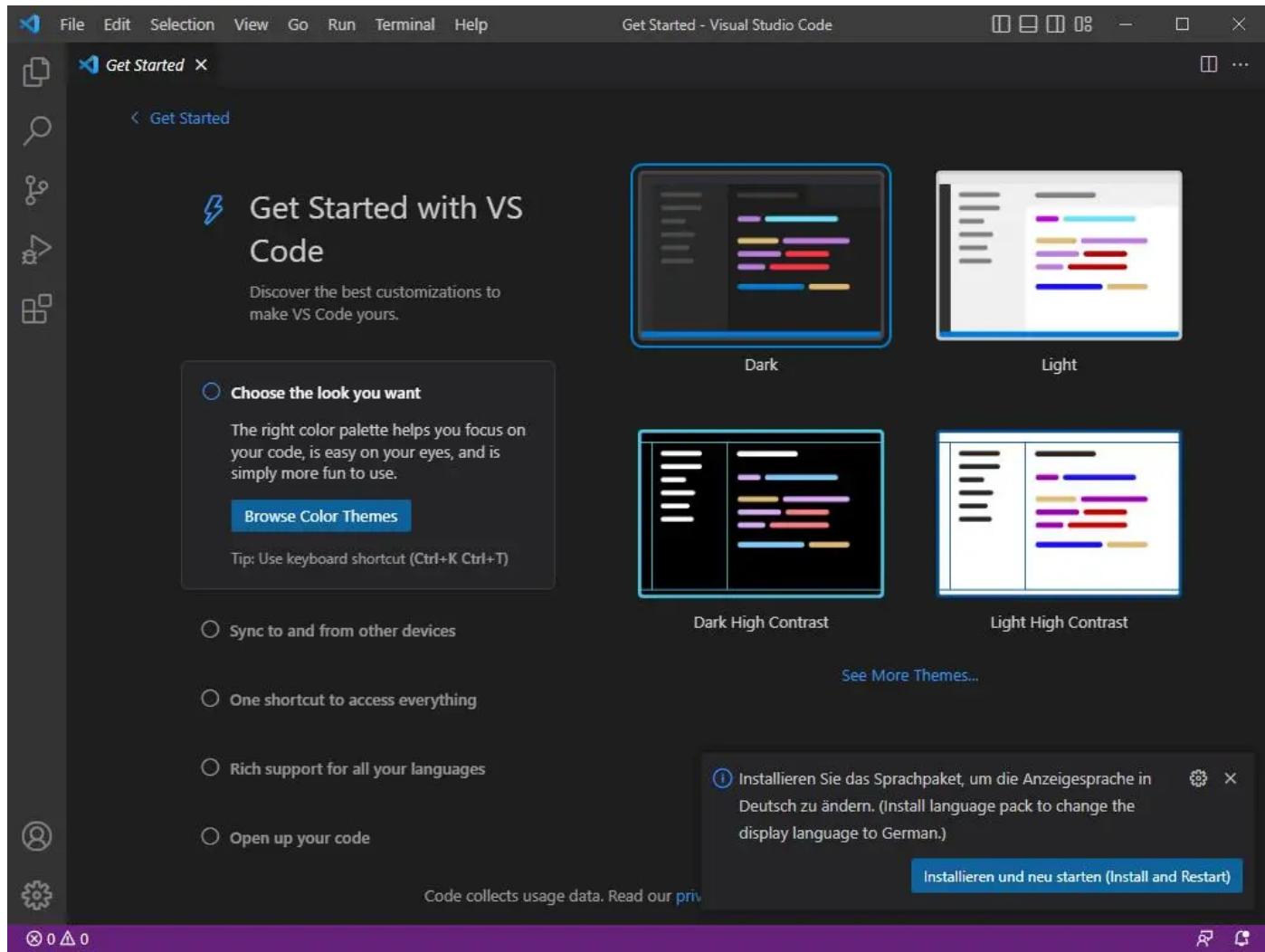
Diese beiden Häkchen sollten auf jeden Fall gesetzt sein.

Nochmals auf *Weiter* und dann auf *Installieren* klicken, und der Installationsvorgang beginnt. Kurze Zeit später erscheint die Abschlussmeldung, die mit einem Klick auf *Fertigstellen* wieder verschwindet.



Zum Schluss sollten Sie VS Code starten lassen.

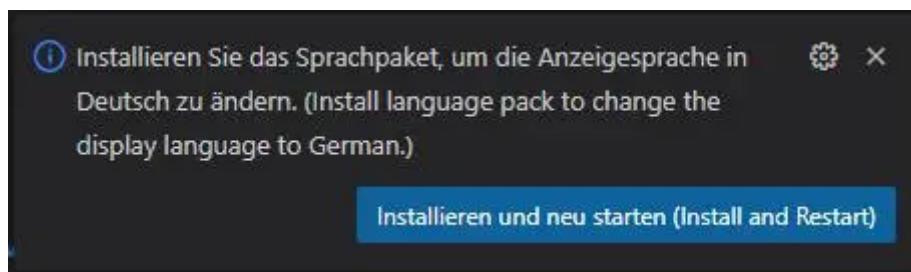
Daraufhin startet Visual Studio Code zum ersten Mal.



VS Code beim ersten Start noch ohne deutsches Sprachpaket

## Installation des deutschen Sprachpaketes

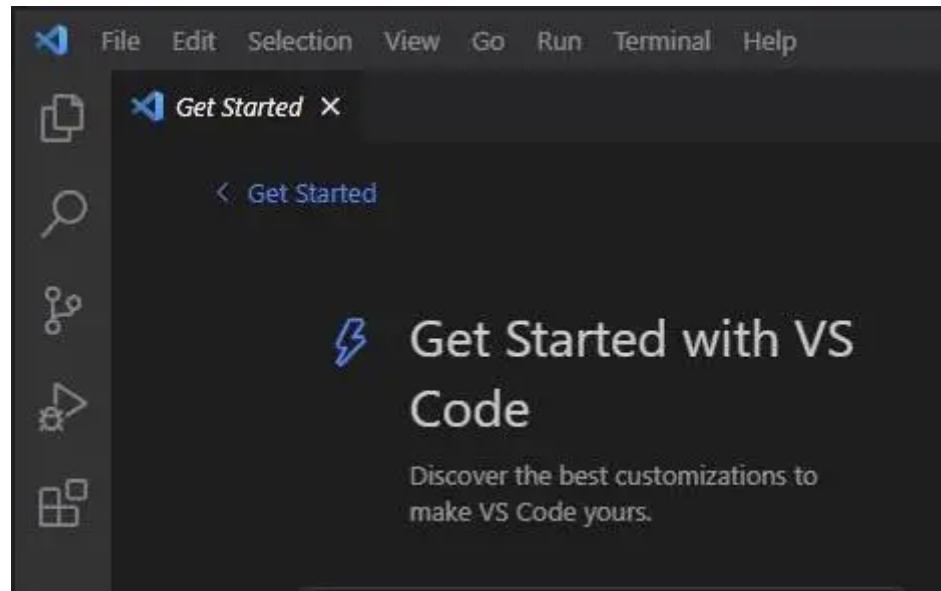
Falls auf dem Computer eine rein deutsche Version des Betriebssystems installiert ist, steht im VS-Code-Fenster nun die Frage, ob das deutsche Sprachpaket installiert werden soll.



Erkennt VS Code die Sprache des Betriebssystems eindeutig, erscheint diese Meldung.

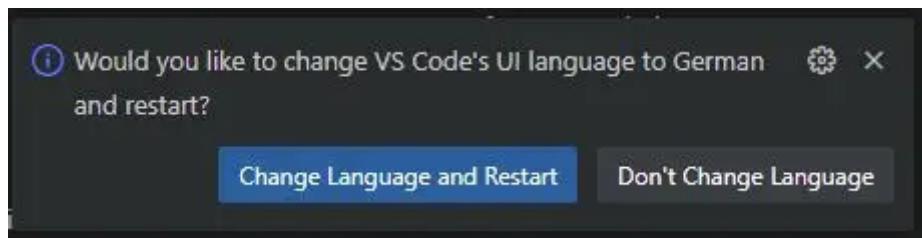
Dann genügt ein Klick auf *Installieren und neu starten*.

Bei mehrsprachigen Betriebssystemversionen funktioniert die Spracherkennung nicht immer. In diesem Fall müssen Sie nach einem Klick auf das *Extension*-Symbol (das ist das unterste in der Symbolleiste) ins Suchfeld *german* eingeben.



Das unterste Symbol in der Leiste führt zu den Erweiterungen (extensions).

Ein Klick auf *Install* beim deutschen Sprachpaket startet den Download und die Installation. In der abschließenden Meldung genügt ein Klick auf *Change Language and Restart*, um VS Code neu in deutscher Sprache zu starten.



Nur noch einen Klick von deutschsprachigen Menüs entfernt

## Versionsverwaltung git installieren

Größere Programmprojekte werden Sie in der Regel immer wieder weiterentwickeln. So entstehen schnell unterschiedliche Versionen. Damit Sie nicht durcheinanderkommen, gibt es ein praktisches Hilfsmittel in Visual Studio Code: git. Diese Erweiterung ermöglicht es, Übersicht zu behalten. Auch wenn Sie es vielleicht am Anfang bei kleinen Programmen nicht gleich einsetzen, sollten Sie es hier bereits installieren. In einer späteren Online-Anleitung zeigen wir Ihnen dann an einem konkreten Beispiel, wie Sie git einsetzen. Wer nicht so lange warten möchte, findet [hier \[4\]](#) eine deutschsprachige Anleitung.

Die git-Installation ist rasch erledigt: In der Menüzeile klicken Sie auf *Anzeigen* und *Quellcode*

**QUELLCODEVERWALTUNG**

Installieren Sie Git, ein beliebtes Quellcodeverwaltungssystem, um Codeänderungen nachzuverfolgen und mit anderen zusammenzuarbeiten. Weitere Informationen finden Sie in unseren [Git-Leitfäden](#).

**Git für Windows herunterladen**

Nach der Installation bitte [neu laden](#) (oder [Fehlerbehebung](#)). Zusätzliche Quellcodeverwaltungsanbieter können [aus dem Marketplace](#) installiert werden.

In der Quellcodeverwaltung steht das Download-Feld zur git-Installation.

verwaltung! (das ist dort in der aktuellen Version wirklich so geschrieben).

Dann folgt eine Sicherheitsabfrage, da der Download von einer externen Internet-Seite aus erfolgt.

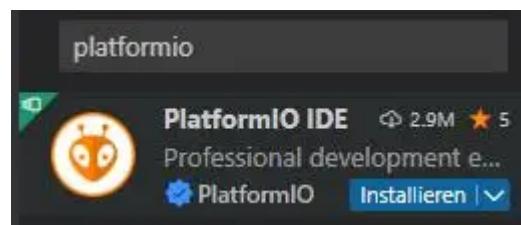


Die Sicherheitsfrage bitte mit *Öffnen* beantworten.

Nach dem Öffnen-Klick landen Sie auf der **Internet-Seite von git [5]**. Laden Sie die für Ihr Betriebssystem geeignete git-Version herunter und installieren Sie die Software wie üblich. Sie müssen also in den recht zahlreichen folgenden Fenster immer nur auf *Next* und schließlich auf *Finish* klicken. Das war es auch schon.

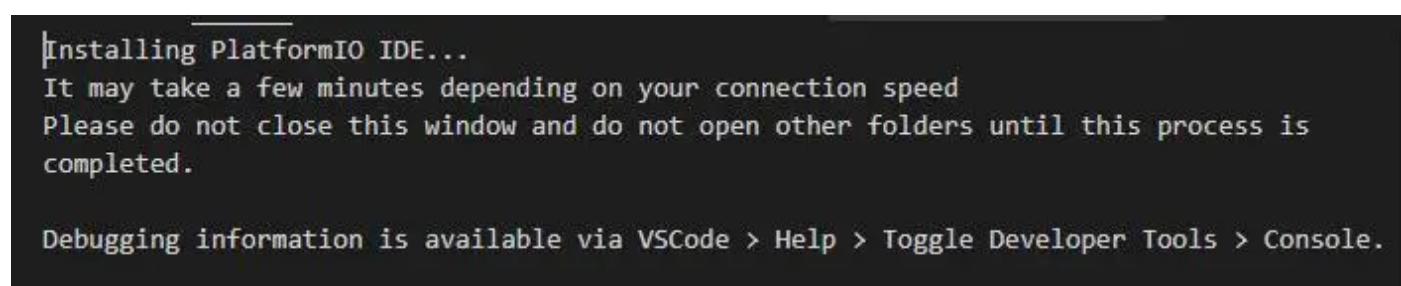
## Erweiterung mit PlatformIO

Jetzt kommen wir endlich zur eigentlichen Programmierumgebung PlatformIO. Auch die wird als Erweiterung in VS Code angeboten. Also klicken Sie wieder auf das Erweiterungssymbol und geben *platformio* ins Suchfeld ein. Sobald die *PlatformIO IDE* gefunden wird, klicken Sie dort auf *Installieren*.



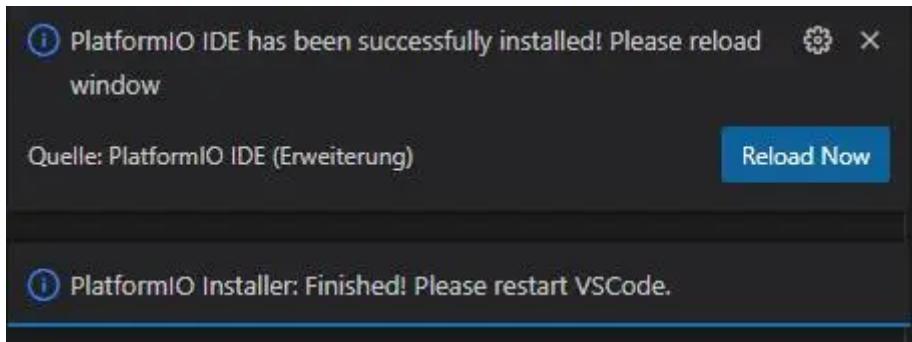
Hier klicken Sie auf *Installieren*.

Der Download kann ein paar Minuten dauern, je nach Geschwindigkeit der Internetverbindung.



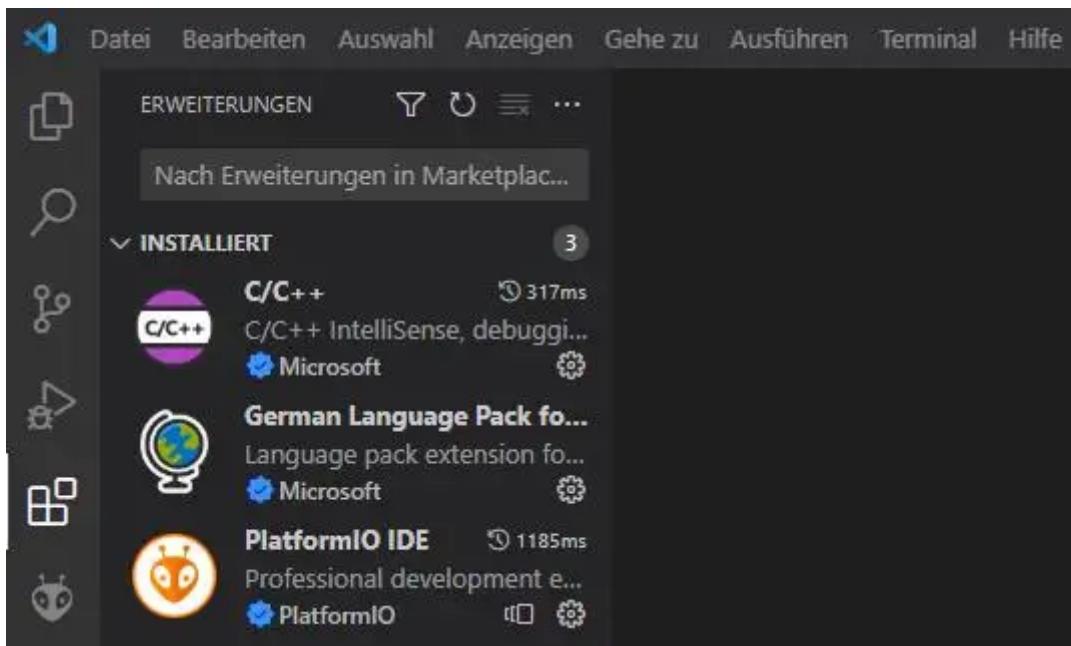
Solange diese Meldung angezeigt wird, sollten Sie VS Code nicht beenden!

Schließlich erscheinen zwei kleine Meldungsfenster, die Sie zum Neustart von VS Code auffordern.



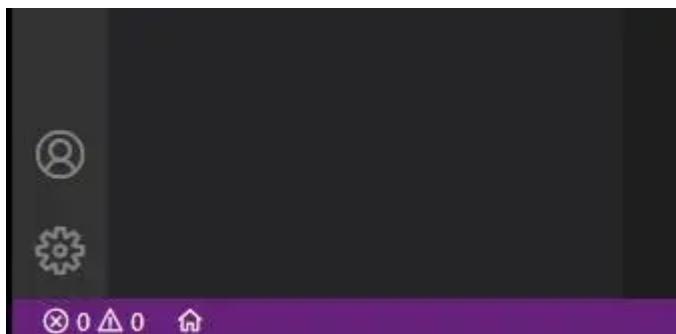
Ein Neustart ist fällig.

Nach dem Neustart wird Ihnen ein neues Symbol unter dem Erweiterungssymbol auffallen. Dieses an eine extraterrestrische Ameise erinnernde Zeichen ist der Zugang zu PlatformIO.



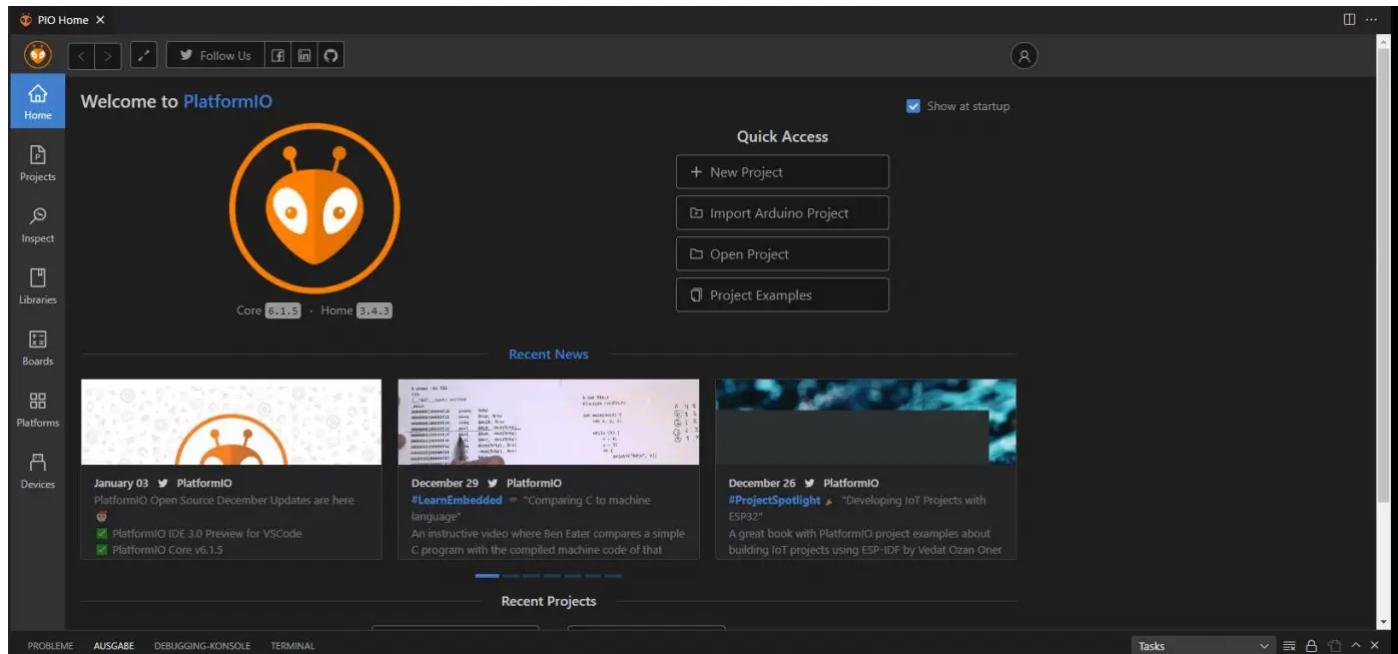
Links unten in der Symbolleiste erscheint nun das PlatformIO-Symbol.

Viel wichtiger ist aber ein anderes neues Symbol, das ganz unscheinbar am unteren linken Fensterrand dazugekommen ist: das Häuschen.



Ein Klick auf das kleine Haus in der unteren Symbolleiste führt Sie zum PlatformIO-Startfenster.

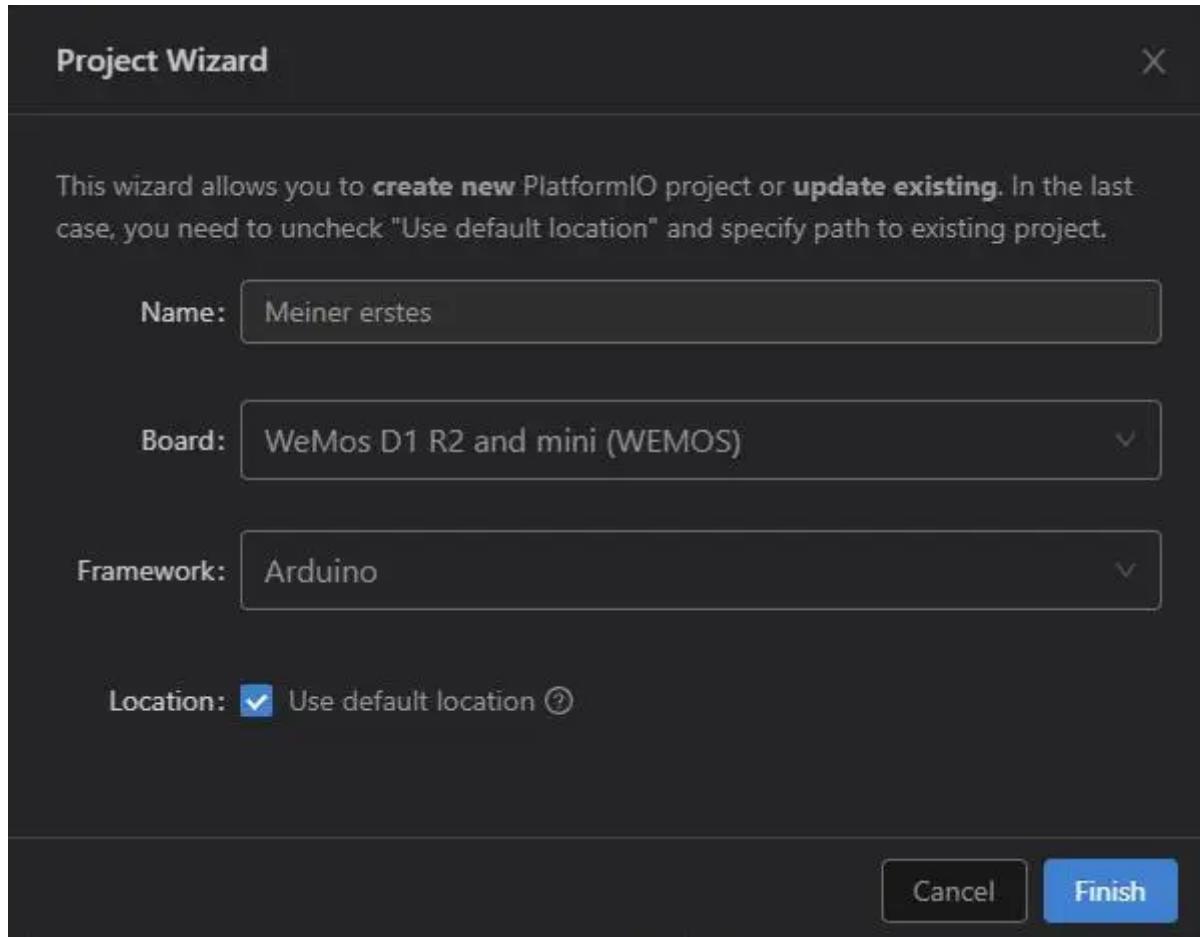
Damit erreichen Sie die Startseite der Programmierumgebung. Wie Sie Ihr erstes kleines Projekt damit anlegen, erfahren Sie auf der nächsten Seite.



Hier beginnt die Arbeit mit PlatformIO.

## Erstes Projekt in PlatformIO

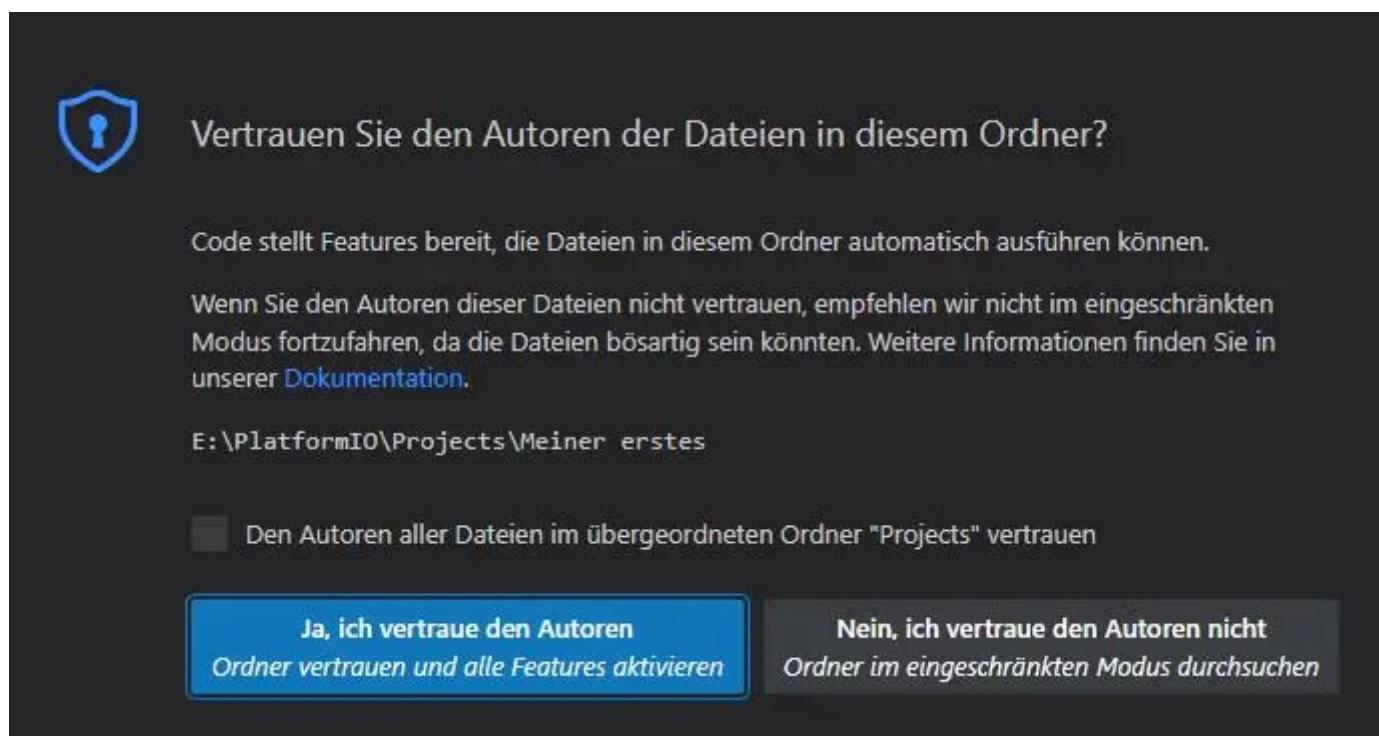
Für ein neues Projekt klicken Sie im Startbereich von PlatformIO auf *New Project*. Danach geben Sie Ihrem Projekt einen Namen und wählen das verwendete Board aus. PlatformIO kennt aktuell (Januar 2023) etwa 1500 Boards. Wählen Sie das passende einfach aus der Liste aus. Beim Framework belassen Sie es zunächst bei *Arduino*. Einige Boards benötigen andere Frameworks. Falls das mal bei einem Make-Projekt der Fall sein wird, werden Sie dort darauf hingewiesen.



Der Pfeil rechts im Board-Feld klappt bei Klick die Liste der bekannten Boards auf.

Mit *Finish* geht es weiter. Dann brauchen Sie beim ersten Projekt mit diesem Board etwas Geduld: PlatformIO lädt nämlich aus dem Internet alles nach, was es braucht. Solange erscheint ein blaues Feld *Please wait*.

Falls zwischendurch Sicherheitsfragen auftauchen, beantworten Sie die mit *Ja, ich vertraue den Autoren*. Möchten Sie nicht ständig wieder damit genervt werden, setzen Sie zuvor noch ein Häkchen vor *Den Autoren aller Dateien ...*



Bei der Installation neuer Software durch PlatformIO können solche Fragen auftauchen.

Wenn alles zusammengesammelt ist, erscheint ein Fenster mit dem Inhalt der Datei *platformio.ini* und links daneben eine Liste mit Ordnernamen.

```

platformio.ini - Meiner erstes - Visual Studio Code

File Edit View Insert Terminal Help

EXPLORER ... PIO Home platformio.ini
MEINER ERSTES
> .pio
> .vscode
> include
> lib
> src
> test
> .gitignore
platformio.ini

platformio.ini
1 ; PlatformIO Project Configuration File
2 ;
3 ; Build options: build flags, source filter
4 ; Upload options: custom upload port, speed and extra flags
5 ; Library options: dependencies, extra library storages
6 ; Advanced options: extra scripting
7 ;
8 ; Please visit documentation for the other options and examples
9 ; https://docs.platformio.org/page/projectconf.html
10
11 [env:d1_mini]
12 platform = espressif8266
13 board = d1_mini
14 framework = arduino
15

```

PlatformIO hat das neue Projekt angelegt inklusive aller Ordner und der Konfigurationsdatei *platformio.ini*.

Diese Datei ist im Grunde der Clou des ganzen: Hier wird alles vermerkt, was Ihr Projekt benötigt. Wenn Sie ein solches Projekt (das ist der komplette Projektordner inklusive aller darin enthalten Dateien und Unterordnern) später an andere weitergeben oder selbst auf einen anderen Computer übertragen, um es dort mit PlatformIO weiterzubearbeiten, wird die Programmierumgebung auf diesem Computer automatisch alles nachladen, was es für dieses Projekt braucht.

Aber wo landet nun der eigentliche Programmtext? Dazu gibt es den Ordner *src* (für sourcecode). Klicken Sie in der Ordnerliste auf *src* und Sie sehen, dass dort bereits eine Datei namens *main.cpp* enthalten ist. Diese Datei enthält den Anfang des Programms, das heißt, der Mikrocontroller beginnt die Programmausführung nach dem Kompilieren mit dem ersten Befehl in dieser Datei. Klicken Sie auf den Dateinamen und Sie sehen deren Inhalt.

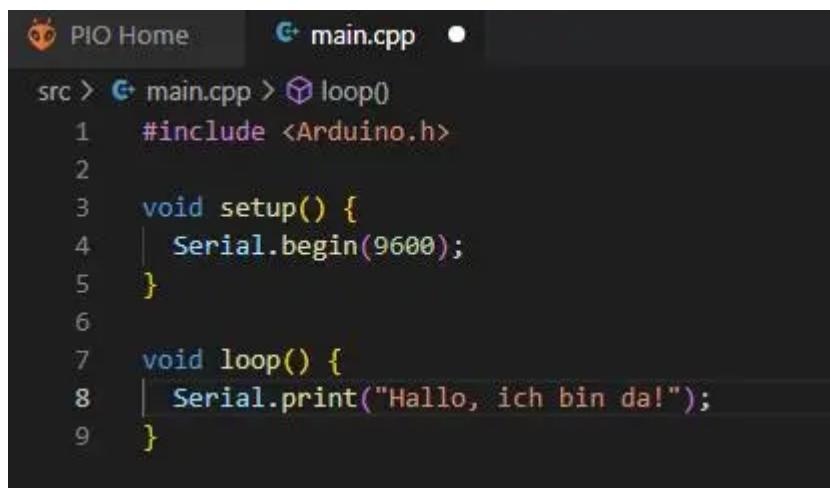


```
PIO Home   G main.cpp ×

src > G main.cpp > ...
1 #include <Arduino.h>
2
3 void setup() {
4     // put your setup code here, to run once:
5 }
6
7 void loop() {
8     // put your main code here, to run repeatedly:
9 }
```

Die Hauptdatei des Sourcecodes

Falls Ihnen das bekannt vorkommt: Das entspricht genau dem Inhalt der von der Arduino-IDE bekannten INO-Dateien. Und genauso können Sie dort Ihr Programm schreiben. Als einfaches Beispiel lassen wir nur über den seriellen Port eine Begrüßung ausgeben.



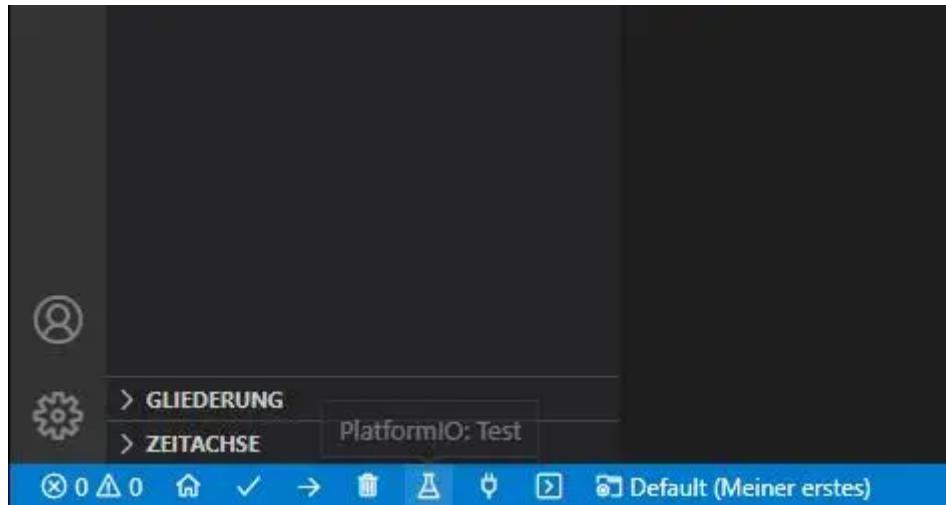
```
PIO Home   G main.cpp •

src > G main.cpp > loop()
1 #include <Arduino.h>
2
3 void setup() {
4     Serial.begin(9600);
5 }
6
7 void loop() {
8     Serial.print("Hallo, ich bin da!");
9 }
```

Das erste Programm

Nun wird es Zeit, dass Ihr Board etwas zu tun bekommt: Schließen Sie es per USB-Kabel an den Computer an. Klicken Sie dann in der blauen Leiste am unteren Fensterrand auf den nach rechts

zeigenden Pfeil.



In der blauen Leiste stecken nützliche Werkzeuge: Der Pfeil steht für Upload, das Steckersymbol startet den seriellen Monitor.

Jetzt beginnt die Programmier-Umgebung zu arbeiten: Zunächst werden eventuell noch benötigte Bestandteile (etwa der Compiler) nachgeladen. Anschließend erfolgt der Kompiliervorgang. Danach wird die fertige Firmware zunächst in die Datei *firmware.bin* geschrieben, die sich im Ordner *.pio\build\d1\_mini* befindet. Diese Datei ist wichtig für Boards, auf die man neue Firmware nicht über USB übertragen kann, sondern die ihr Update als Datei auf einer Speicherkarte verlangen.

Während der ganzen Zeit erscheinen zahlreiche Meldungen, hoffentlich ohne Fehler. Falls etwas bemängelt wird, dann bricht der Vorgang ab und Sie müssen den Sourcecode auf Tippfehler und ähnliches überprüfen.

The screenshot shows the PlatformIO IDE interface. In the top right, the title bar reads "Druckversion - Mikrocontroller: Programmier-Umgebung PlatformIO installieren | Make". The main area has several tabs: "EXPLORER", "PIO Home", "main.cpp", and "main.cpp > loop()". The "EXPLORER" tab shows a project structure with files like ".pio", ".vscode", "include", "lib", "src/main.cpp", "test", ".gitignore", and "platformio.ini". The "main.cpp" tab displays the following Arduino-style code:

```

1 #include <Arduino.h>
2
3 void setup() {
4     Serial.begin(9600);
5 }
6
7 void loop() {
8     Serial.print("Hallo, ich bin da!");
9 }

```

Below the code editor is a tab bar with "PROBLEME", "AUSGABE", "DEBUGGING-KONSOLE", and "TERMINAL". The "TERMINAL" tab is selected, showing the output of the compilation process:

```

Compiling .pio\build\d1_mini\FrameworkArduino\spiffs\spiffs_cache.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\spiffs\spiffs_check.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\spiffs\spiffs_gc.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\spiffs\spiffs_hydrogen.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\spiffs\spiffs_nucleus.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\spiffs\spiffs_api.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\sqrt32.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\stdlib_noniso.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\time.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\uart.cpp.o
Compiling .pio\build\d1_mini\FrameworkArduino\umm_malloc\umm_info.c.o

```

At the bottom of the terminal window, there are status indicators and a message: "Default (Meiner erstes)" and "Zeile 9, Spalte 2 Leerzeichen: 2 UTF-8 CRLF C++ PlatformIO".

Der Compiler bei der Arbeit

Weiter geht es dann mit dem Upload aufs Board. Im Gegensatz zur Arduino-IDE muss man sich hier nicht mit der Einstellung der richtigen Portnummer beschäftigen. PlatformIO erledigt das automatisch. Sollte das einmal nicht funktionieren, fehlt der für das jeweilige Board erforderliche USB-Treiber. Ist alles erfolgreich erledigt, kommt die Erfolgsmeldung.

The terminal window shows the progress of the upload:

```

Writing at 0x00024000... (76 %)
Writing at 0x00028000... (84 %)
Writing at 0x0002c000... (92 %)
Writing at 0x00030000... (100 %)
Wrote 269232 bytes (197667 compressed) at 0x00000000 in 17.5 seconds (effective 122.8 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
===== [SUCCESS] Took 25.89 seconds =====
* Das Terminal wird von Aufgaben wiederverwendet, drücken Sie zum Schließen eine beliebige Taste.

```

Die Erfolgsmeldung zeigt, dass alles in Ordnung ist.

Falls Sie das überprüfen möchten: Starten Sie den seriellen Monitor durch einen Klick auf das Steckersymbol in der blauen Fußleiste des Fensters. Dann sehen Sie die Begrüßungsmeldung des Mikrocontrollers.

```
ch bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo
, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Ha
llo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da
!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin
da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich
bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, i
ch bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo
, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Ha
llo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da
!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin
da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin
da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich bin da!Hallo, ich b
```

Sprachmodus aus

Der Prozessor arbeitet mit der neuen Firmware.

Herzlichen Glückwunsch! Sie haben Ihr erstes Mikrocontroller-Programm mithilfe von PlatformIO geschrieben, kompiliert und aufs Board übertragen.

## Vorhandene Projekte öffnen

Im Internet, insbesondere auf Github, finden Sie zahlreiche PlatformIO-Software-Projekte. Die können Sie nun problemlos auch auf Ihrem PC benutzen. Nach dem Download und dem Entpacken der ZIP-Datei sollten Sie den so erhaltenen Ordner ins *Projects*-Verzeichnis von PlatformIO kopieren. Den Namen haben Sie sich doch bei der Installation gemerkt? Falls nicht, zeigt PlatformIO Ihnen den Pfad nach Klicks auf *Datei* und *Ordner öffnen* an.

Mit *Datei* und *Ordner öffnen* können Sie dann das Projekt auch in PlatformIO laden, bearbeiten und kompilieren.

Und nun viel Spaß bei Ihren ersten eigenen Versuchen auf der neuen Plattform. Vermutlich werden Sie sie bald nicht mehr missen wollen! ([hgb \[6\]](#))

### URL dieses Artikels:

<https://www.heise.de/-7447240>

### Links in diesem Artikel:

- [1] <https://www.heise.de/news/Urgestein-der-Maker-IDE-in-Version-2-0-Arduino-mit-vielen-Verbesserungen-7264985.html>
- [2] <https://www.heise.de/make/>
- [3] <https://code.visualstudio.com/>
- [4] <https://www.digitalocean.com/community/tutorials/how-to-use-git-integration-in-visual-studio-code-de>
- [5] <https://git-scm.com/download/win>
- [6] <mailto:hgb@make-magazin.de>