

# Installer les logiciels

Pour envoyer le code à la carte M5StickC, nous avons besoin d'un **environnement de développement**, dans notre cas ce sera **VSCODE** :

- <https://code.visualstudio.com/download>



Les codes sont stockés sur Github. Pour télécharger et utiliser le code nous auront besoin du logiciel **Git** :

- <https://git-scm.com/downloads>

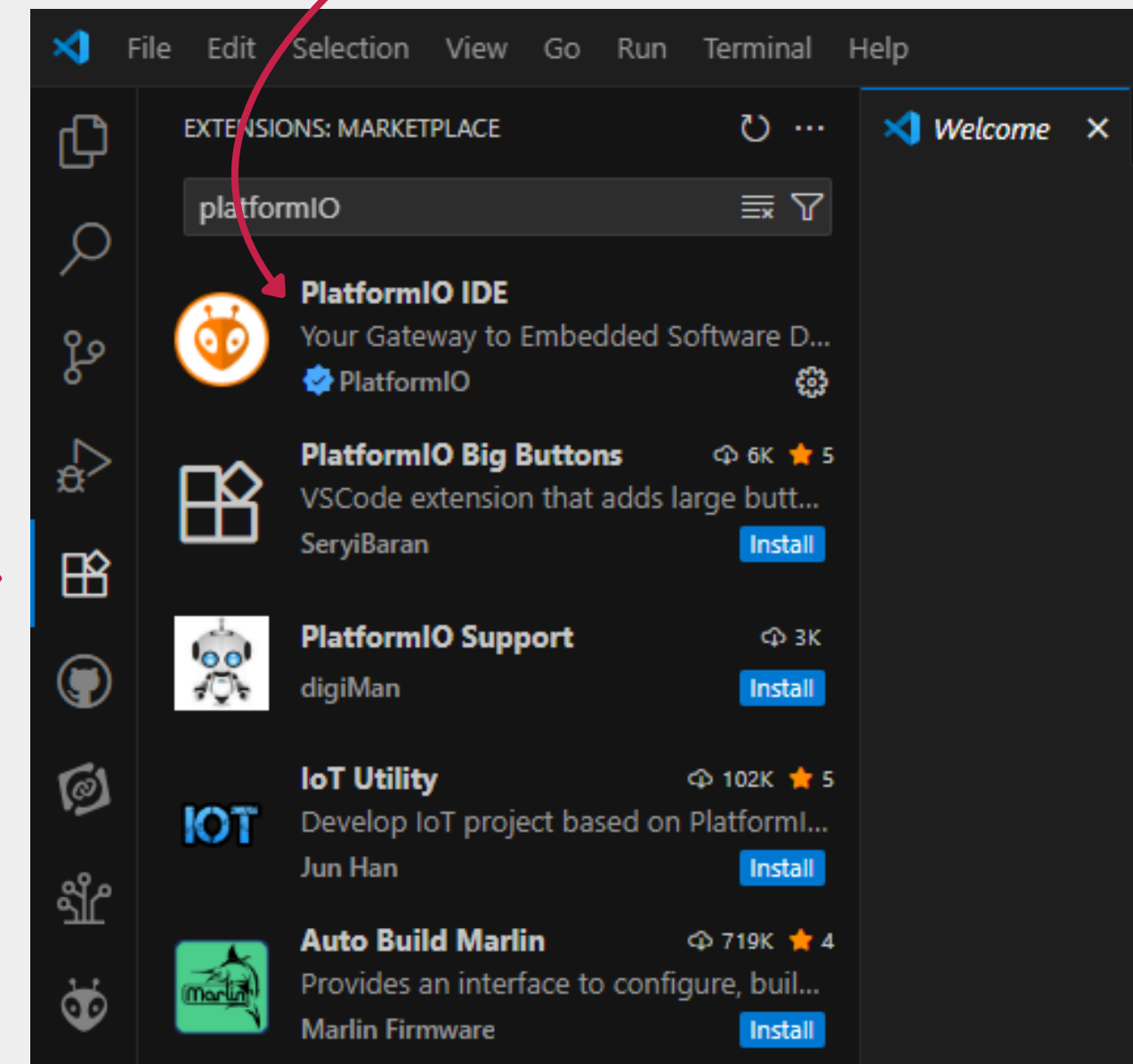


La carte M5StickC nécessite un driver pour être programmé :  
<https://www.silabs.com/developer-tools/usb-to-uart-bridge-vcp-drivers?tab=downloads>

# Installer les extensions

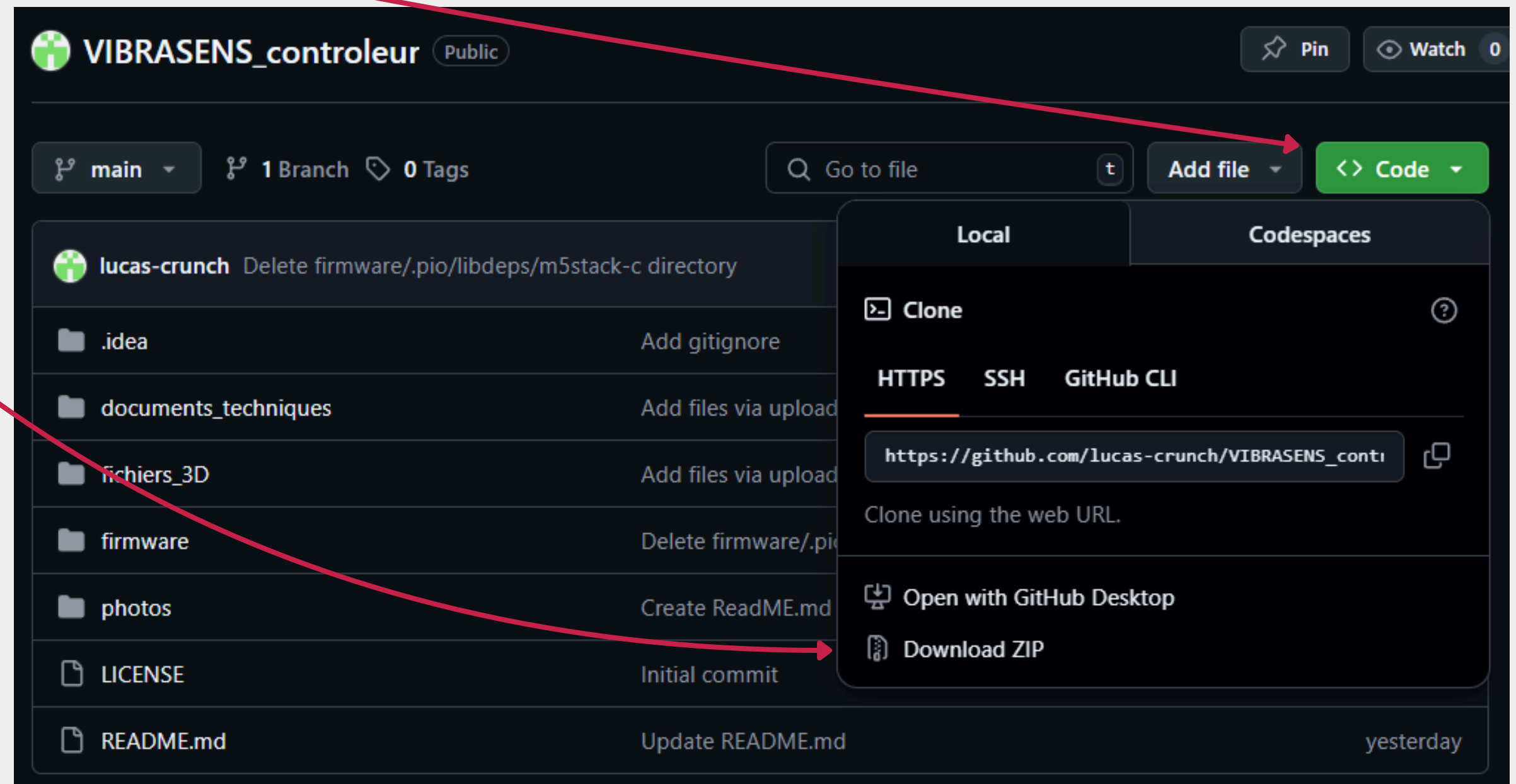
Vous pouvez maintenant ouvrir VSCode, chercher **“PlatformIO”** dans les extensions et l’installer.

L’installation peut prendre 5 minutes.



# Télécharger le code

Sur le Github [https://github.com/lucas-crunch/VIBRASENS\\_controleur](https://github.com/lucas-crunch/VIBRASENS_controleur), vous trouverez un bouton **"Code"** et vous pouvez le télécharger en cliquant sur **"Download ZIP"**

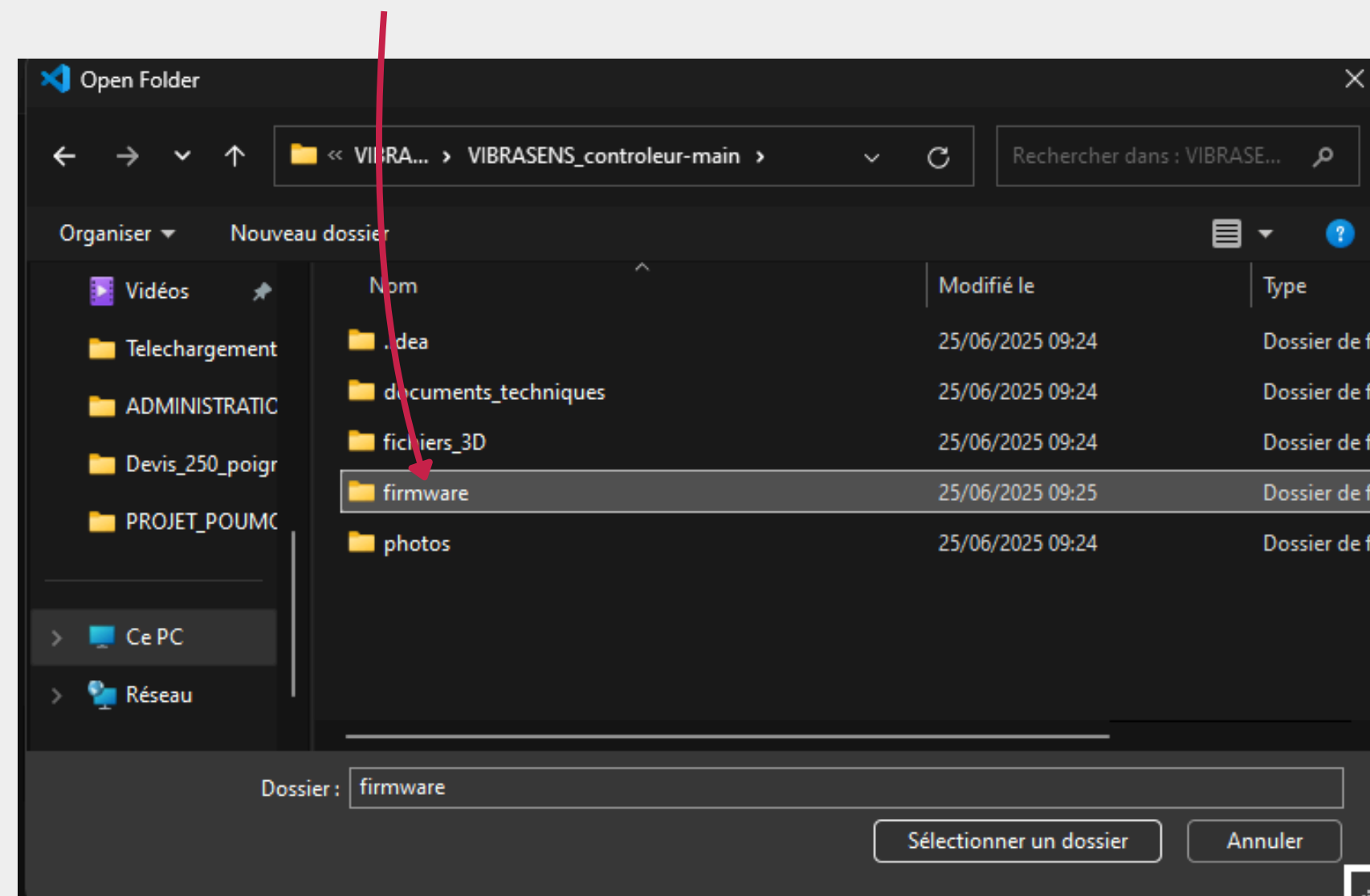
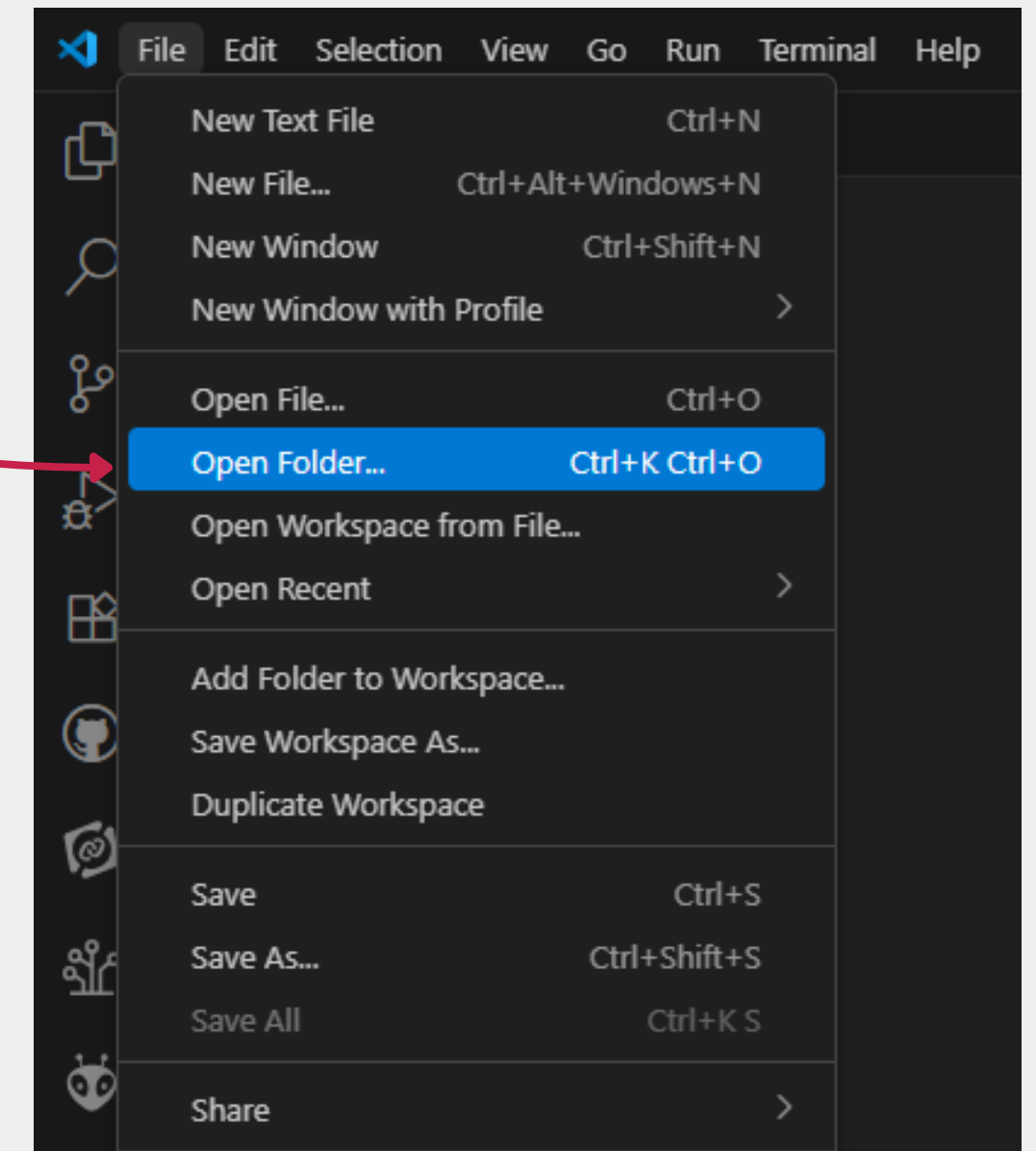


# Importer le code dans votre environnement de développement

Dans un premier temps, **dézippez** le fichier obtenu.

Dans VSCode, cliquez sur **"File"** puis **"Open Folder"**.

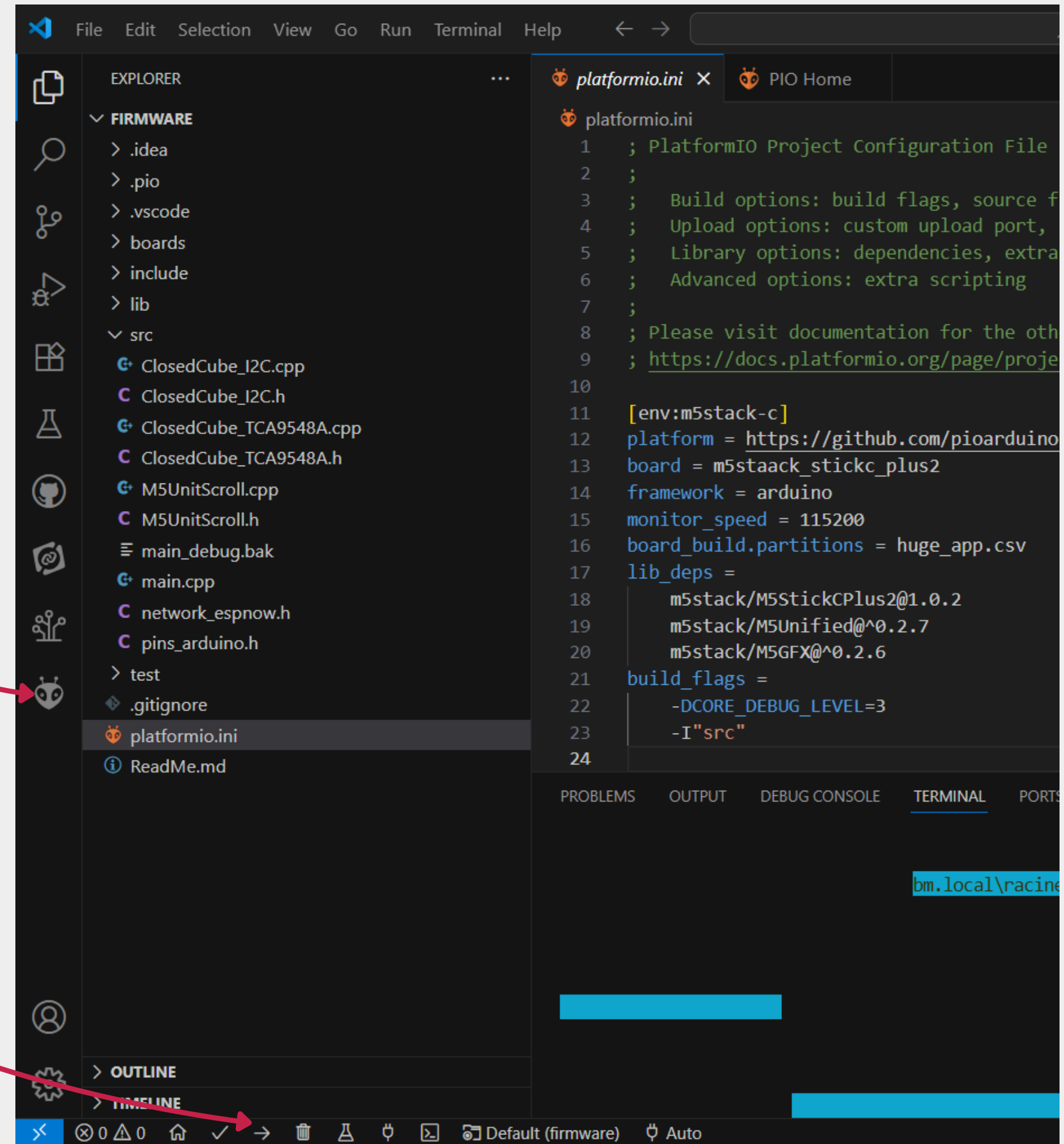
Choisissez le dossier **"Firmware"**.



# Importer le code dans votre environnement de développement

Cette fenêtre doit s'ouvrir.

Si le **bandeau de contrôle** n'est pas présent alors cliquez sur le **logo platformIO**



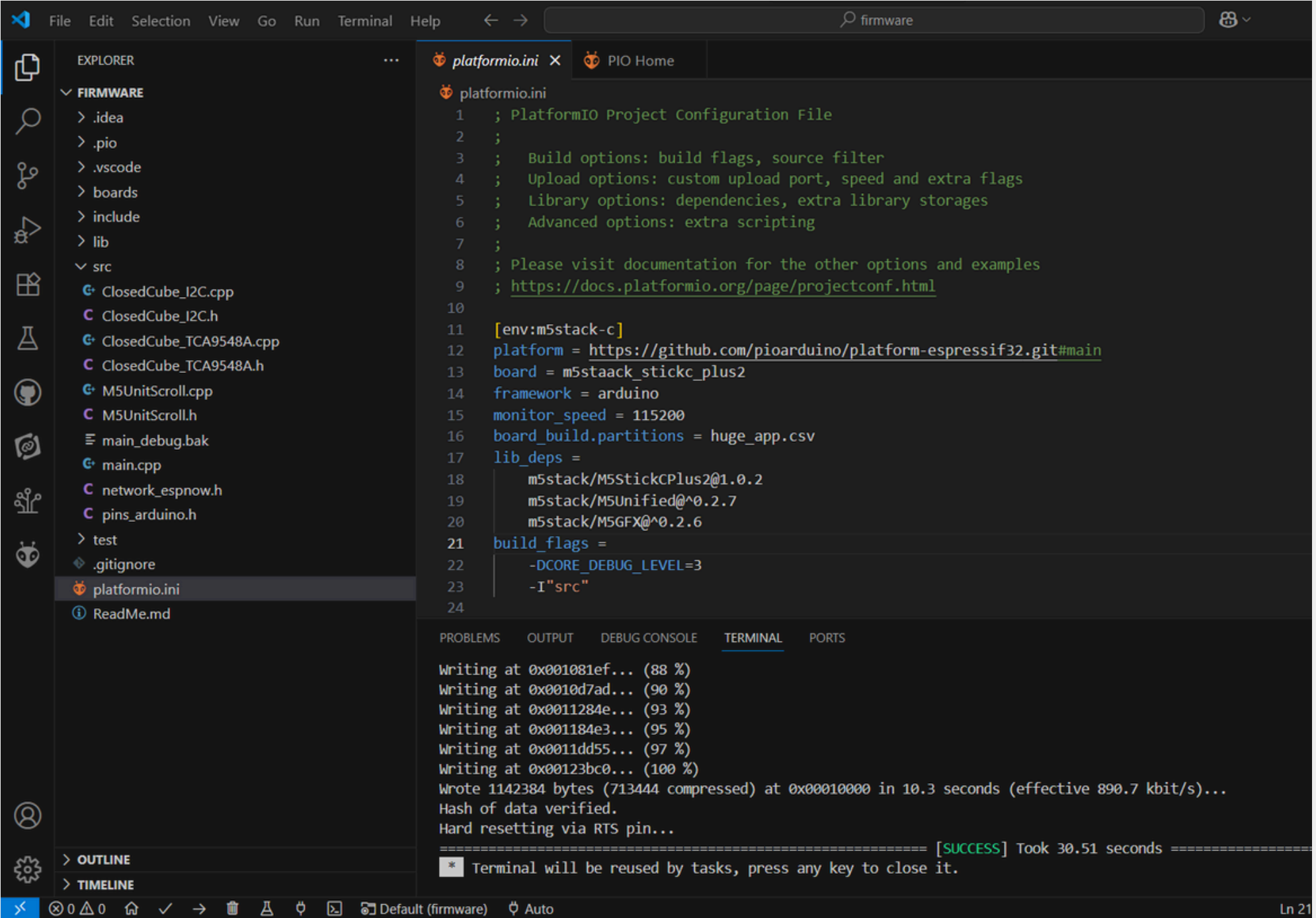


# Envoyer le code à la carte

Reliez la carte M5StickC à l'aide d'un USB-C à l'ordinateur.

Puis cliquez sur ➡ du **bandeau de commande**.

Le code devrait s'envoyer à la carte et vous afficher **"Success"**.



The screenshot shows the Visual Studio Code interface with the PlatformIO extension. The Explorer panel on the left shows a project structure with a 'platformio.ini' file highlighted. The main editor displays the contents of 'platformio.ini', which is a PlatformIO Project Configuration File. The file includes build options, upload options, library options, and advanced options. The board is configured as 'm5staack\_stickc\_plus2' using the 'arduino' framework. The terminal at the bottom shows the output of the upload process, including writing to memory, verifying the hash, and hard resetting via RTS pin. The final status is '[SUCCESS] Took 30.51 seconds'.

```
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:m5stack-c]
12 platform = https://github.com/pioarduino/platform-espessif32.git#main
13 board = m5staack_stickc_plus2
14 framework = arduino
15 monitor_speed = 115200
16 board_build.partitions = huge_app.csv
17 lib_deps =
18   m5stack/M5stickCPlus2@1.0.2
19   m5stack/M5Unified@^0.2.7
20   m5stack/M5GFX@^0.2.6
21 build_flags =
22   -DCORE_DEBUG_LEVEL=3
23   -I"src"
24
```

Writing at 0x001081ef... (88 %)  
Writing at 0x0010d7ad... (90 %)  
Writing at 0x0011284e... (93 %)  
Writing at 0x001184e3... (95 %)  
Writing at 0x0011dd55... (97 %)  
Writing at 0x00123bc0... (100 %)  
Wrote 1142384 bytes (713444 compressed) at 0x00010000 in 10.3 seconds (effective 890.7 kbit/s)...  
Hash of data verified.  
Hard resetting via RTS pin...  
===== [SUCCESS] Took 30.51 seconds =====  
\* Terminal will be reused by tasks, press any key to close it.