

# PlatformIO avec Qt Creator

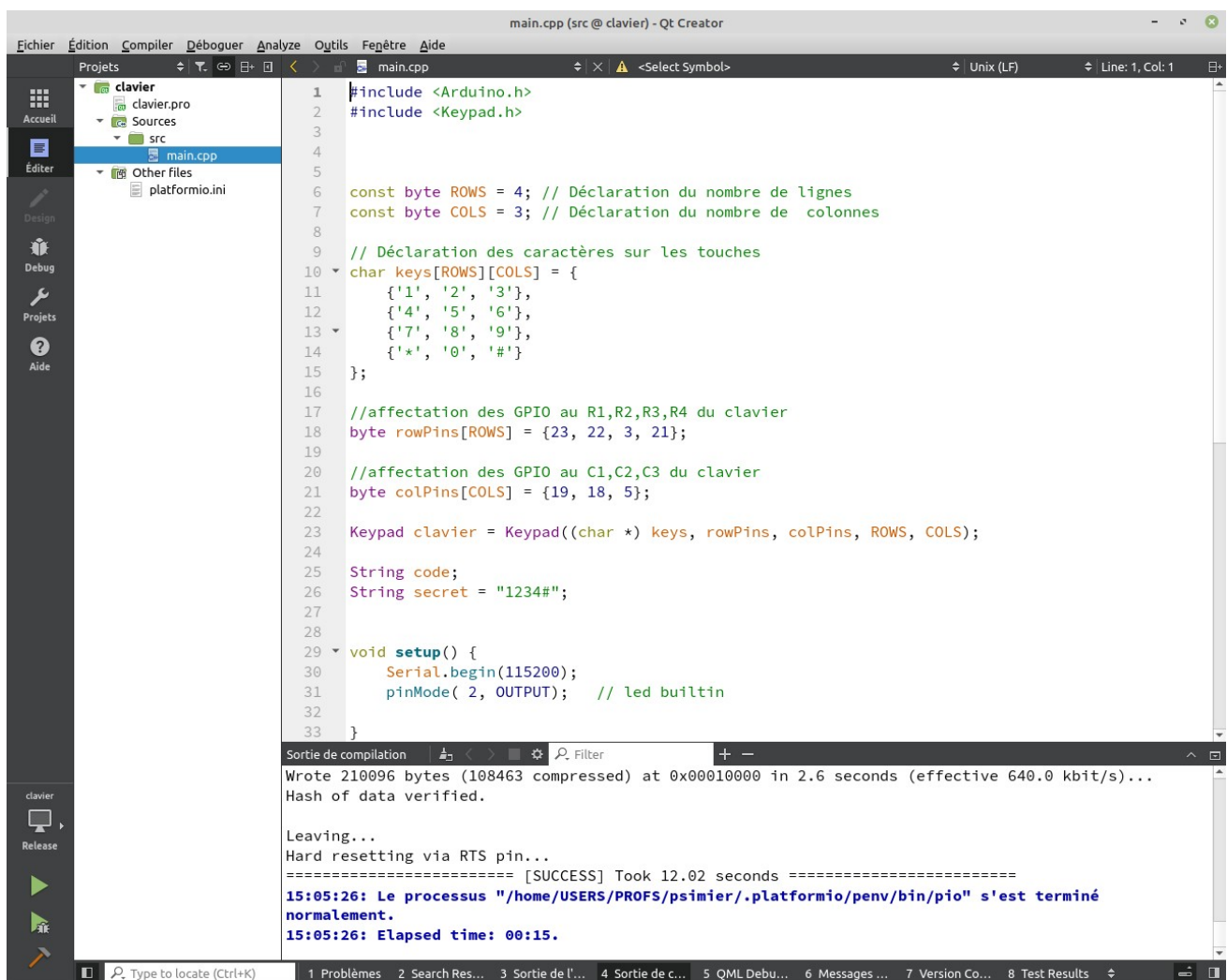
## Table des matières

PlatformIO avec Qt Creator.....	1
1 Introduction.....	1
2 Création d'un projet esp32 pour IDE Qt Creator.....	2
3 Configuration du projet.....	4

## 1 Introduction

Ce document fait suite à l'installation de plateformeIO pour netbeans.

Qt Creator est l'environnement de développement intégré multi-plateforme open source pour Qt. L'éditeur comprend des fonctionnalités telles que la mise en évidence de la syntaxe pour divers langages, un gestionnaire de projet, des systèmes de contrôle de version intégrés, des outils de navigation rapide dans le code et la saisie semi-automatique du code.



The screenshot shows the Qt Creator IDE interface. The top menu bar includes 'Fichier', 'Édition', 'Compiler', 'Débugger', 'Analyse', 'Outils', 'Fenêtre', and 'Aide'. The left sidebar contains icons for 'Accueil', 'Éditer', 'Design', 'Debug', 'Projets', and 'Aide'. The 'Projets' panel shows a project named 'clavier' with subfolders 'clavier.pro', 'Sources', and 'src'. The 'src' folder contains 'main.cpp' and 'platformio.ini'. The main editor window displays the contents of 'main.cpp', which includes headers for 'Arduino.h' and 'Keypad.h', and code for a keypad interface. The bottom panel shows the 'Sortie de compilation' (Compiler Output) window, which displays the successful compilation of the project, including the message '15:05:26: Le processus "/home/USERS/PROFS/psimier/.platformio/penv/bin/pio" s'est terminé normalement.' and the elapsed time '00:15'.

```
1 #include <Arduino.h>
2 #include <Keypad.h>
3
4
5
6 const byte ROWS = 4; // Déclaration du nombre de lignes
7 const byte COLS = 3; // Déclaration du nombre de colonnes
8
9 // Déclaration des caractères sur les touches
10 char keys[ROWS][COLS] = {
11     {'1', '2', '3'},
12     {'4', '5', '6'},
13     {'7', '8', '9'},
14     {'*', '0', '#'}
15 };
16
17 //affectation des GPIO au R1,R2,R3,R4 du clavier
18 byte rowPins[ROWS] = {23, 22, 3, 21};
19
20 //affectation des GPIO au C1,C2,C3 du clavier
21 byte colPins[COLS] = {19, 18, 5};
22
23 Keypad clavier = Keypad((char *) keys, rowPins, colPins, ROWS, COLS);
24
25 String code;
26 String secret = "1234#";
27
28
29 void setup() {
30     Serial.begin(115200);
31     pinMode( 2, OUTPUT); // led builtin
32 }
33
```

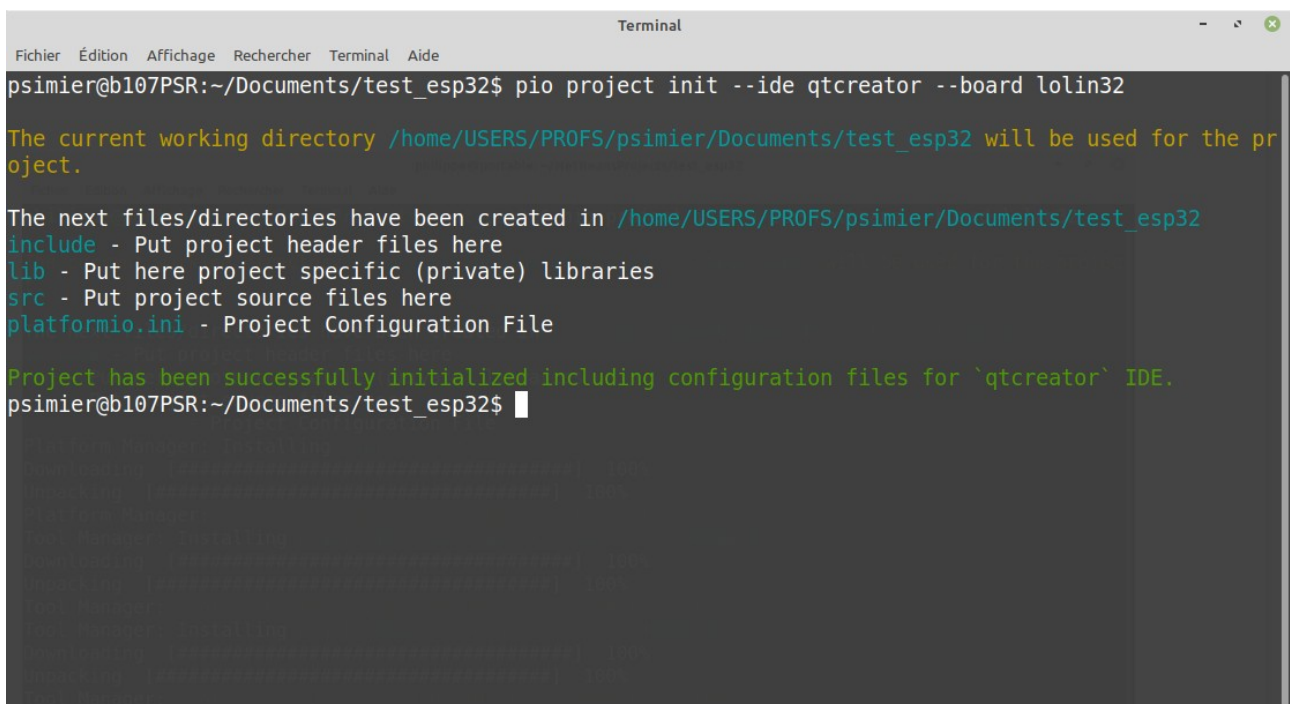
Sortie de compilation  
Wrote 210096 bytes (108463 compressed) at 0x00010000 in 2.6 seconds (effective 640.0 kbit/s)...  
Hash of data verified.  
Leaving...  
Hard resetting via RTS pin...  
===== [SUCCESS] Took 12.02 seconds =====  
15:05:26: Le processus "/home/USERS/PROFS/psimier/.platformio/penv/bin/pio" s'est terminé normalement.  
15:05:26: Elapsed time: 00:15.

## 2 Création d'un projet esp32 pour IDE Qt Creator

Créer un répertoire pour votre projet

puis se déplacer à l'intérieur puis lancer la commande **pio project init** avec comme arguments l'IDE et la carte utilisée.

```
mkdir test_esp32
cd test_esp32
pio project init --ide qtcreator --board lolin32
```



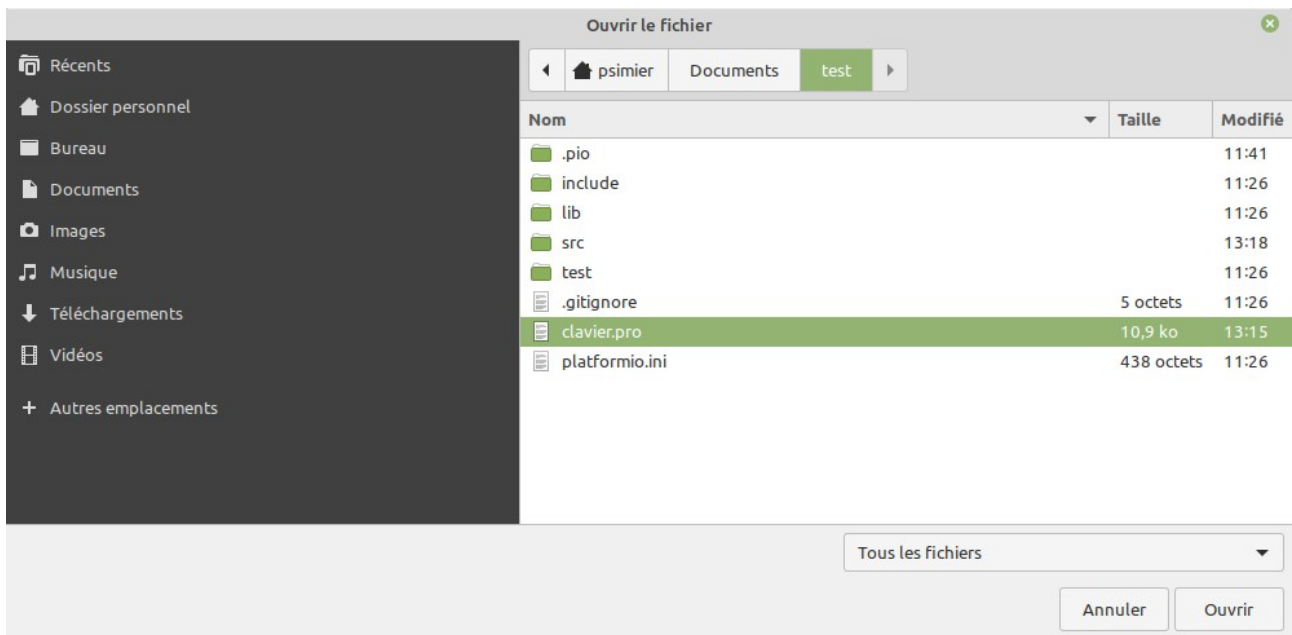
```
Terminal
Fichier  Édition  Affichage  Rechercher  Terminal  Aide
psimier@b107PSR:~/Documents/test_esp32$ pio project init --ide qtcreator --board lolin32
The current working directory /home/USERS/PROFS/psimier/Documents/test_esp32 will be used for the project.
The next files/directories have been created in /home/USERS/PROFS/psimier/Documents/test_esp32
include - Put project header files here
lib - Put here project specific (private) libraries
src - Put project source files here
platformio.ini - Project Configuration File
Project has been successfully initialized including configuration files for `qtcreator` IDE.
psimier@b107PSR:~/Documents/test_esp32$
```

La "tool chaîne" est maintenant installée.

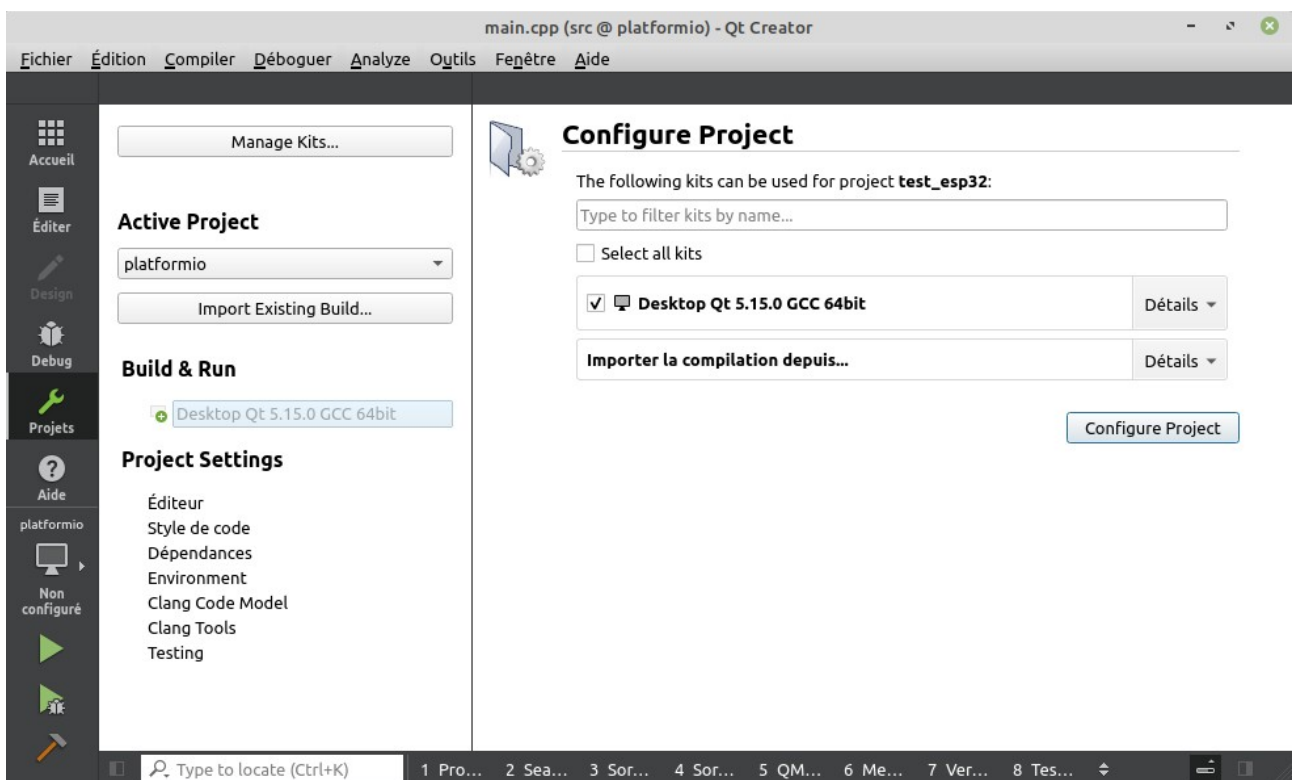
On peut ouvrir ce projet via le Menu: **Fichier > Ouvrir un fichier ou projet...** de Qt creator

Renommer le fichier **platform.io** avec le nom de votre projet par exemple **clavier.pro**

et sélectionnez **clavier.pro** dans le dossier où se trouve « platformio.ini »



Sélectionnez un kit par défaut et cliquez sur le bouton **Configure Project**



### 3 Configuration du projet

Cliquez sur Projects (icon clé verte) puis modifiez l'étape de Build et de Clean

Supprimez tous les éléments des **Étapes Build** et **Étapes Clean** puis cliquez sur Étapes de construction > Ajouter une étape de construction > Étape de processus personnalisé et définissez comme le montre l'image ci dessous.

Command : **pio**

Arguments : **-f -c qtcreator run**

The screenshot shows the 'General' tab of the Qt Creator project configuration dialog. It includes settings for 'Shadow build', 'Build directory' (set to /home/USERS/PROFS/psimier/Documents/test), 'Separate Debug Info' (Leave at Default), 'QML debugging and profiling' (Enable), and 'Qt Quick Compiler' (Leave at Default). Below this is the 'Étapes Build' section with a 'Custom Process Step' of 'pio -f -c qtcreator run'. It shows the 'Command' as 'pio', 'Arguments' as '-f -c qtcreator run', and 'Working directory' as '%{buildDir}'. There is also an 'Ajouter l'étape Build' button. The 'Étapes Clean' section has a 'Custom Process Step' of 'pio -f -c qtcreator run --target clean', with 'Command' as 'pio', 'Arguments' as '-f -c qtcreator run --target clean', and 'Working directory' as '%{buildDir}'. There is also an 'Ajouter l'étape Clean' button. At the bottom is the 'Environnement de compilation' section with a dropdown set to 'Utiliser Environnement système'.

**General**

Shadow build: ☐

Build directory: /home/USERS/PROFS/psimier/Documents/test [Parcourir...](#)

Separate Debug Info: [Leave at Default](#)

QML debugging and profiling: [Enable](#)

⚠ Might make your application vulnerable.  
Only use in a safe environment.

Qt Quick Compiler: [Leave at Default](#)

**Étapes Build**

**Custom Process Step:** pio -f -c qtcreator run [Détails](#)

Command: pio [Parcourir...](#)

Arguments: -f -c qtcreator run

Working directory: %{buildDir} [Parcourir...](#)

[Ajouter l'étape Build](#)

**Étapes Clean**

**Custom Process Step:** pio -f -c qtcreator run --target clean [Détails](#)

Command: pio [Parcourir...](#)

Arguments: -f -c qtcreator run --target clean

Working directory: %{buildDir} [Parcourir...](#)

[Ajouter l'étape Clean](#)

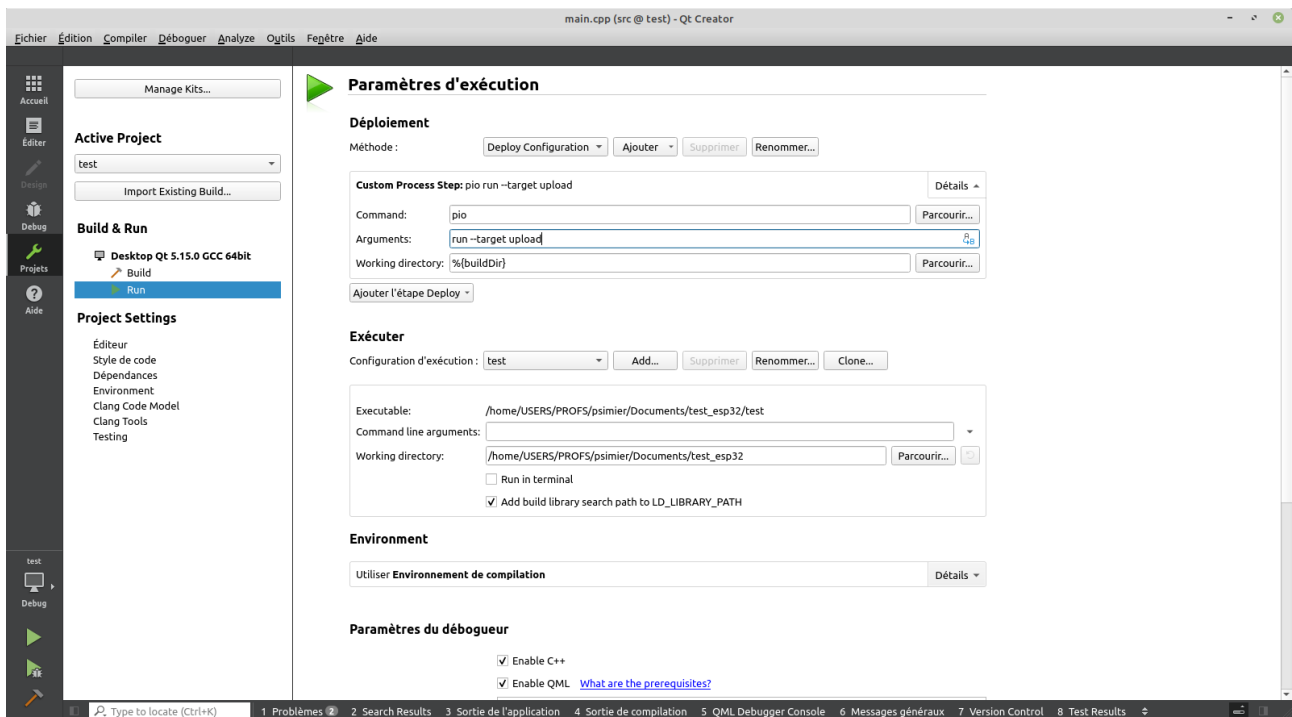
**Environnement de compilation**

Utiliser **Environnement système** [Détails](#)

Dans Paramètre d'exécution cliquez sur **Ajouter l'étape Deploy**

Commande **pio**

Arguments **run --target upload**



Pour supprimer le message d'erreur lors de l'exécution vous pouvez modifier les options de **Exécuter** cliquer sur le bouton **Add** sélectionner **Custom Executable** puis modifier les paramètres comme le montre la copie d'écran

### Exécuter

Configuration d'exécution : Custom Executable Add... Supprimer Renommer... Clone...

Executable:	echo	Parcourir...
Command line arguments:	"Lancé sur la cible"	Parcourir...
Working directory:		Parcourir...
<input type="checkbox"/> Run in terminal		

### Environment

Utiliser Environnement de compilation	Détails
---------------------------------------	---------

