

Projet RevolvAir.org

Configuration et mode d’emploi   
d’une station d’analyse de la qualité de l’air

Table des matières

[Introduction 2](#__RefHeading___Toc1150_3299000868)

[Assemblage d’une station 2](#__RefHeading___Toc1152_3299000868)

[Matériel nécessaire 2](#__RefHeading___Toc1154_3299000868)

[Connecter le matériel 2](#__RefHeading___Toc1156_3299000868)

[Connectivité Wifi 2](#__RefHeading___Toc1158_3299000868)

[Connexion au routeur 2](#__RefHeading___Toc1160_3299000868)

[Point d’accès 2](#__RefHeading___Toc1162_3299000868)

[Codification d’une station 3](#__RefHeading___Toc1164_3299000868)

[Emplacement du code 3](#__RefHeading___Toc1166_3299000868)

[Architecture du projet logiciel 3](#__RefHeading___Toc1168_3299000868)

[Dépendances logicielles 3](#__RefHeading___Toc1170_3299000868)

[Procédure de création de compte pour l’API 3](#__RefHeading___Toc2302_2508117755)

[Configuration du code 3](#__RefHeading___Toc1172_3299000868)

[Compilation et téléversement du code 3](#__RefHeading___Toc1174_3299000868)

[Vérification du bon fonctionnement 3](#__RefHeading___Toc1176_3299000868)

[Bogues connus 3](#__RefHeading___Toc1178_3299000868)

[Contact 3](#__RefHeading___Toc1180_3299000868)

Rédigé par \_Gabriel Bertrand\_ et \_Cédric Bouchard\_

# Introduction

Ce projet permet à partir d’un Arduino (ESP32) d’avoir une station d’analyse de qualité de l’air pour particule fine, qui permet aussi de mesurer l’humidité ainsi que la température. De plus, nous affichons ces données dans un serveur web, et nous les partageons aussi à un API.

# Assemblage d’une station

## Matériel nécessaire

Ajouter les références des spécifications matérielles (*datasheet*) des composants utilisés.

Voici les éléments à considérer

* Esp32Dev  
  (<https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>)
* PMS7003   
  (<https://download.kamami.pl/p564008-PMS7003%20series%20data%20manua_English_V2.5.pdf>)
* LED RGB
* DHT22 (<https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>)
* Câble micro USB

## Connecter le matériel

# Led RGB KS:

# - R -> IO12

# - G -> IO13

# - B -> IO14

# - “-” -> GND

# Senseur DHT22:

# - DAT -> IO18

# - GND -> GND

# - VCC -> 5V

# Senseur PMS2.5:

# - VCC -> 5V

# - GND -> GND

# - TX -> IO16

# RX -> aucune connexion

On peut aussi, au besoin, connecter un second senseur PMS2.5 avec le IO25 et une planche à pain pour les autres connections, soit ceux du GND au GND et du VCC au 5V.

# Connectivité Wifi

Description de la section.

## Connexion au routeur

Que se soit une connexion 2.4Gz ou 5G, sécurisé ou pas, cela ne change rien. Le seul prérequis est que la connexion est possible, car si ce n’est pas le cas, nombreuse fonctionnalité ne seront point disponible comme il sera impossible de faire un point d’accès.

## Point d’accès

À partir de votre propre point d’accès Windows, vous pouvez choisir par vous-même un nom de réseau ainsi qu’un mot de passe. Il vous nécessitera tout simplement d’aller dans « Modifier ».

Dans notre cas (voir la photo ci-dessous), par défaut, nous avons « DESKTOP-N3EGH27 0046 » comme nom de réseau et « }24s129G » comme mot de passe.

Une image contenant texte

Description générée automatiquement

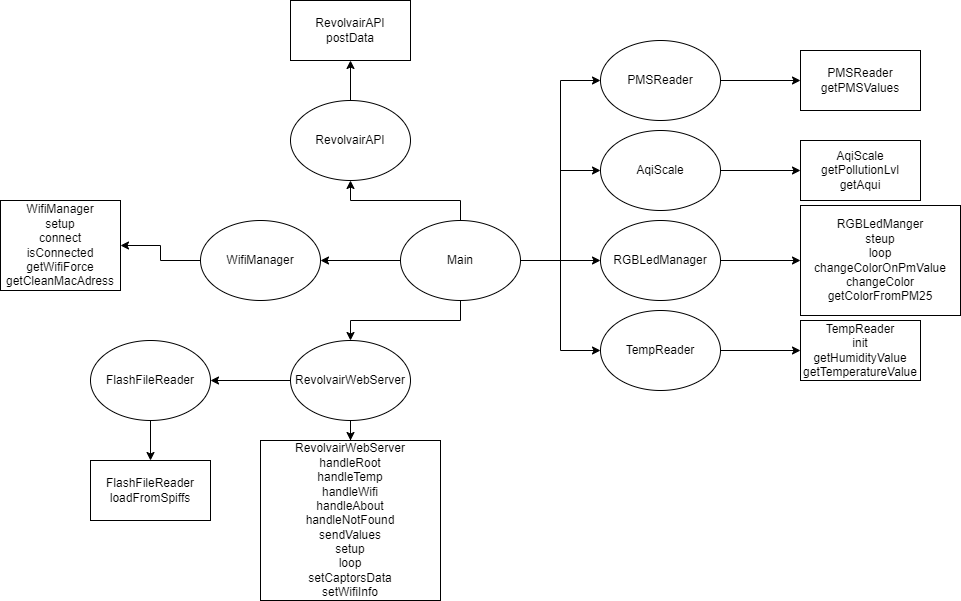
# Codification d’une station

## Emplacement du code

Gitlab: <https://gitlab.com/objConnecte/tp1>

(Ce projet est présentement privé et nécessite des droits pour y avoir accès.)

## Architecture du projet logiciel



## Dépendances logicielles

## fu-hsi/PMS Library@^1.1.0

## adafruit/Adafruit Unified Sensor@^1.1.6

## adafruit/DHT sensor library@^1.4.4

## [bblanchon/ArduinoJson@^6.19.4](mailto:bblanchon/ArduinoJson@%5e6.19.4)

SPIFFS

FS

ESPmDNS

HTTPClient

WiFiClientSecure

WiFi

WebServer

## Procédure de création de compte pour l’API

Veuillez contacter Guillaume Simard par le site Revolvair.org avec le Chip Id et l’adresse mac lié à votre Arduino afin d’avoir les accès.

## Configuration du code

Au besoin, changer dans le « config.h » le SSID, soit le nom du réseau, ainsi que le mot de passe, ou « PASSWORD », permettant donc de se connecter au réseau.

De plus, si vous avez une disposition différente pour vos led, vous pouvez les changer le même fichier « config.h ».

Sinon, rien n’est nécessaire à changer. Cependant, si vous voulez des données plus rapides pour le web serveur et faire plus d’appelle API, vous pouvez changer la valeur de « CONNECTION\_INTERVAL\_WIFI » qui est l’intervalle du temps pour les appelles API et pour donner de nouvelles données au web serveur, et ce, en millième de seconde.

## Compilation et téléversement du code

À l’aide de l’outil PlateformIO (<https://platformio.org/>) avec le « board » du nom de « Espressif ESP32 Dev Module », ouvrer le projet et vous pouvez build le projet avec le Arduino connecté à votre ordinateur. Veuiller bien évidemment vérifier si le « plateformio.ini » est configuré les bonnes valeurs dans un environnement qui sert au build et au déploiement.

Une image contenant texte

Description générée automatiquement

Si jamais vous désirer votre le moniteur série en tant que console, veuillez, à l’aide de l’extension « Plateform.io », faire « Upload and Monitor ».

Si jamais c’est votre premier « Upload », veuillez faire « Upload Filesystem Image » afin de rendre les fichiers dans le data disponible pour le Arduino. Si jamais un problème survient avec ces derniers, vous pouvez faire « Clean All » et refaire le « Upload Filesystem Image ».

## Vérification du bon fonctionnement

Pour vérifier que le wifi fonctionne : Observer le moniteur série pour voir les sorties du SSID et du l’adresse IP de la connexion.

Pour vérifier que le serveur web fonctionne : Une fois l’Arduino connecté, en utilisant un second appareil connecté au même réseau que celui défini dans le « config.h », à l’aide du même IP obtenu précédemment, allez sur un moteur de recherche internet et chercher la route « « / » de l’adresse IP. Si jamais aucun fichier n’est trouvé, vérifier bien la procédure dans le dernier paragraphe de « Compilation et téléversement du code » avec le « Upload Filesystem Image »

Pour vérifier que l’API fonctionne : Observer le moniteur série selon l’intervalle défini dans le « Main.cpp ». Si jamais un problème survient, veuillez vérifier si vos accès on bel et bien accorder.

Pour vérifier les senseurs : Observer le moniteur série, les données sont supposées apparaitre dans celui à faire boucle.

# Bogues connus

Aucun bogue connu.

# Contact et sites web

<https://revolvair.org/>

<https://platformio.org/>

<https://gitlab.com/objConnecte/tp1>