

Présentation Du Projet WeMos UI (UPS Interface) :

Le projet WeMos UI a pour objectif de mettre en œuvre une interface permettant de collecter les informations en provenance d'un onduleur solaire hybride WKS, de contrôler à l'aide de deux sondes thermiques la température haute (côté onduleur), la température basse (côté batteries) et de démarrer un système de ventilation si nécessaire. Il maintient également la date et l'heure à jour. La carte ARCELI WeMos D1 R2 embarque une architecture ESP8266 et une carte WIFI intégrée.

Le logiciel embarqué se connecte comme un client au réseau WIFI dont le SSID et le mot de passe sont codés en dur à la compilation. Le système met automatiquement à jour son horloge interne à l'aide d'un client NTP, en prenant en charge l'heure local et les changements d'heure. La date et l'heure sont transmises à l'onduleur au démarrage du module puis tous les jours à 6H du matin via son port série. Un affichage LCD indique la date et l'heure ainsi que la température et le taux d'humidité en partie haute (côté onduleur) et en partie basse (côté batteries). Le logiciel déclenche un relais en fonction de ces températures et / ou en fonction de l'heure pour refroidir l'ensemble. En cas d'anomalie des sondes thermiques ou si la carte venait à être physiquement éteinte, la ventilation se déclenche immédiatement. Le fonctionnement du relais et de la ventilation a été réalisé de tel sorte que ce soit le logiciel qui coupe le relais et donc la ventilation. En l'absence d'un ordre explicite du logiciel sur la carte pour couper le relais, la ventilation est donc toujours en service. Enfin, le logiciel embarque un serveur web permettant présenter les informations collectées sur le port série de l'onduleur, la température et l'humidité au niveau des sondes de température haute et basse et l'état du relais pour la ventilation. Ces informations sont ensuite collectées sur une plateforme Jeedom à l'aide du plugin « Script » toutes les minutes.

Ce document et ce projet sont libres de droit sous licence GPLv3.

Merci simplement de respecter son auteur.

Vous pouvez retrouver l'ensemble de ce projet sur GitHub à l'adresse :

<https://github.com/santeroc/WeMos-D1-WKS-EVO-Jeedom.git>

ainsi que toutes mes documentations et tous mes projets sur mon référentiel

GitHub à l'adresse : <https://santeroc.github.io>.

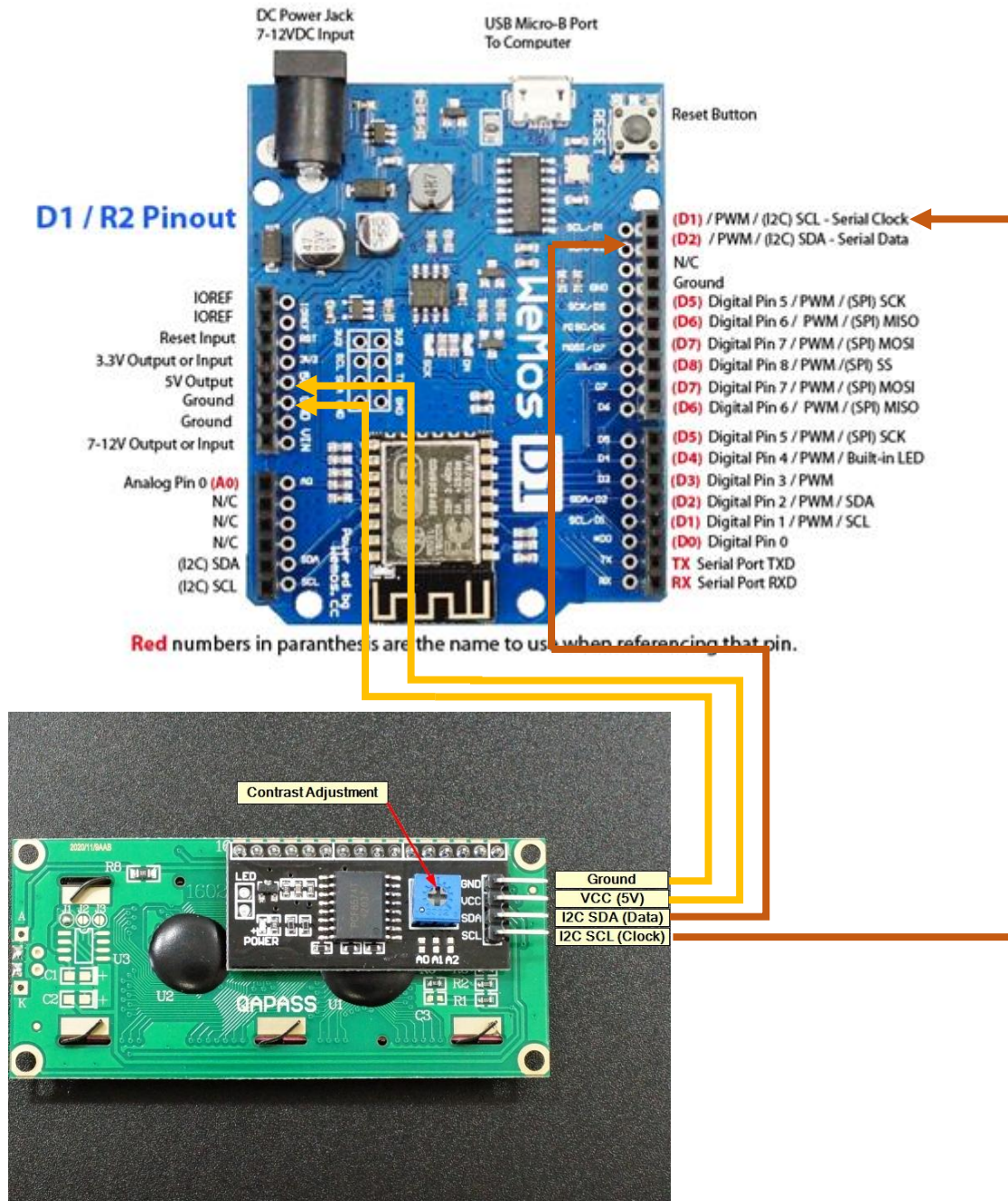


Liste des articles utilisés dans cette documentation :

- **La carte ARCELI :**
ARCELI Programme Arduino UNO Compatible avec la Carte de développement WeMos D1 R2 WiFi ESP8266 de Arduino IDE
https://www.amazon.fr/gp/product/B07J2QKNHB/ref=ppx_yo_dt_b_asin_title_o02_s00?ie=UTF8&psc=1
- **Câbles de liaisons entre les différents modules :**
WheateFull 120 pièces Multicolore Dupont Wire 40pin mâle à femelle + 40pin mâle à mâle + 40pin femelle à femelle Panneau de câbles de cavalier Kit de câbles de ruban pour Arduino
https://www.amazon.fr/gp/product/B06XD8NBTZ/ref=ppx_yo_dt_b_asin_title_o06_s00?ie=UTF8&psc=1
- **Le LCD (étape 1) :**
SunFounder IIC I2C TWI Serial 2004 20x4 LCD Module Shield Compatible with Arduino R3 MEGA 2560
https://www.amazon.fr/gp/product/B01GPUMP9C/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1
- **Les sondes thermiques DHT22 AM2302 (étape 2) :**
AZDelivery DHT22 AM2302 Capteur de Température et d'Humidité avec Câble Compatible avec Arduino et Raspberry Pi incluant Un E-Book!
https://www.amazon.fr/dp/B078SVZB1X/?coliid=I3SM4RYJJS3798&colid=3CZB4MB46B54F&psc=1&ref=lv_ov_lig_dp_it
- **Le convertisseur RS232 (étape 3) :**
DollaTek MAX3232 RS232 Serial Port vers TTL Convertisseur Module DB9 Connecteur W / 4 Jump Cables
https://www.amazon.fr/gp/product/B07DK3874B/ref=ppx_yo_dt_b_asin_title_o07_s02?ie=UTF8&psc=1
- **Connecteur de raccordement (étape 4) :**
YIOVVOM Connecteur de dérivation DB9 vers borne de câblage RS232 D-SUB adaptateurs série mâles Carte de dérivation de Port Module sans Soudure avec boîtier (Adaptateur série mâle + écrou)
https://www.amazon.fr/gp/product/B071DS5GTW/ref=ppx_yo_dt_b_asin_title_o07_s01?ie=UTF8&th=1
- **Le relais (étape 5) :**
ARCELI 5pcs 5V Un Blindage de Carte de Module de Relais à 1 Canal pour Arduino (déclencheur de Bas Niveau)
https://www.amazon.fr/gp/product/B07RJFJJZM/ref=ppx_yo_dt_b_asin_title_o00_s00?ie=UTF8&psc=1
- **Le convertisseur 220v – 12v DC (étape 5) :**
HuaTec Eaglerise Transformateur LED 12V 15W Tension Constant Ultra Fin pour bande de LED Bloc d'Alimentation Driver LED
https://www.amazon.fr/dp/B082VD3QM7/?coliid=I1AE1KZV4JGJ14&colid=TFC9A61478GS&psc=1&ref=lv_ov_lig_dp_it
- **Ventilateur (étape 5) :**
Noctua NF-F12 iPPC-2000, Ventilateur Haut Rendement, 3 Broches, 2000 tr./min (120 mm, Noir)
https://www.amazon.fr/dp/B00KEST8PQ/?coliid=I2B3BR5PYC3TEZ&colid=TFC9A61478GS&psc=1&ref=lv_ov_lig_dp_it
- **Boîte et sa façade (étape 9) :**
JOYKK Boîte de Projet pour boîtier électronique en Plastique étanche
https://www.amazon.fr/gp/product/B07PK4BPM4/ref=ppx_yo_dt_b_asin_title_o03_s00?ie=UTF8&psc=1

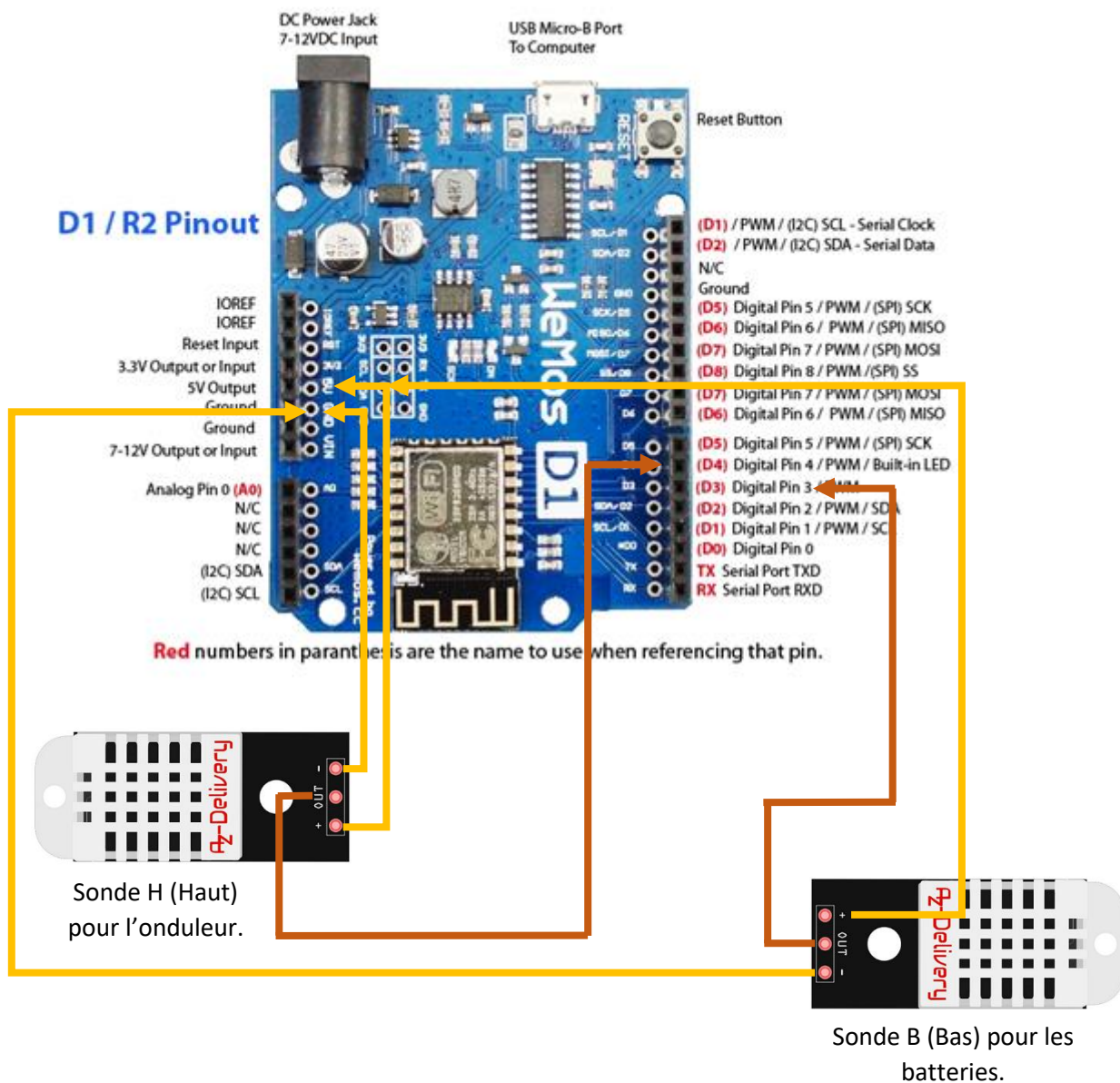
ETAPE 1 - Raccordement du LCD à la carte ARCELI :

Relier la broche Ground du LCD à une broche **Ground** de la carte ARCELI,
Relier la broche **VCC (5v)** à la broche **5V output** de la carte ARCELI,
Relier la broche **I2C SDA (Data)** à la broche **(D2)** de la carte ARCELI,
Enfin relier la broche **I2C SCL (Clock)** à la broche **(D1)** de la carte ARCELI.



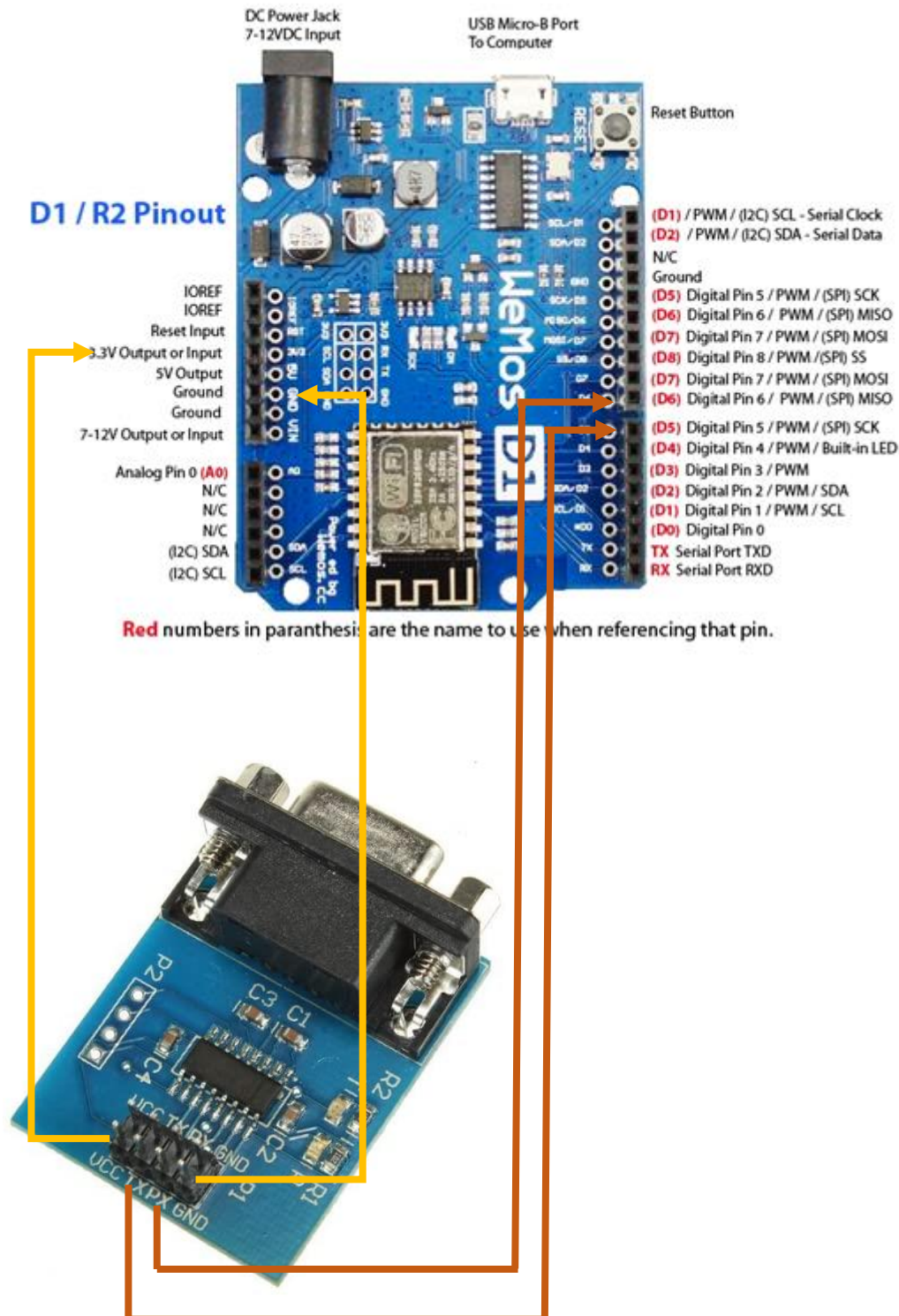
ETAPE 2 - Raccordement des sondes DHT22 AM2302 à la carte ARCELL :

Relier les broches « - » des DHT22 à une broche **Ground** de la carte ARCELL,
Relier les broches « + » des DHT22 à la broche **5V output** de la carte ARCELL,
Relier la broche **OUT** du DHT Bas (B) à la broche **(D3)** de la carte ARCELL,
Enfin relier la broche **OUT** du DHT Haut (H) à la broche **(D4)** de la carte ARCELL.



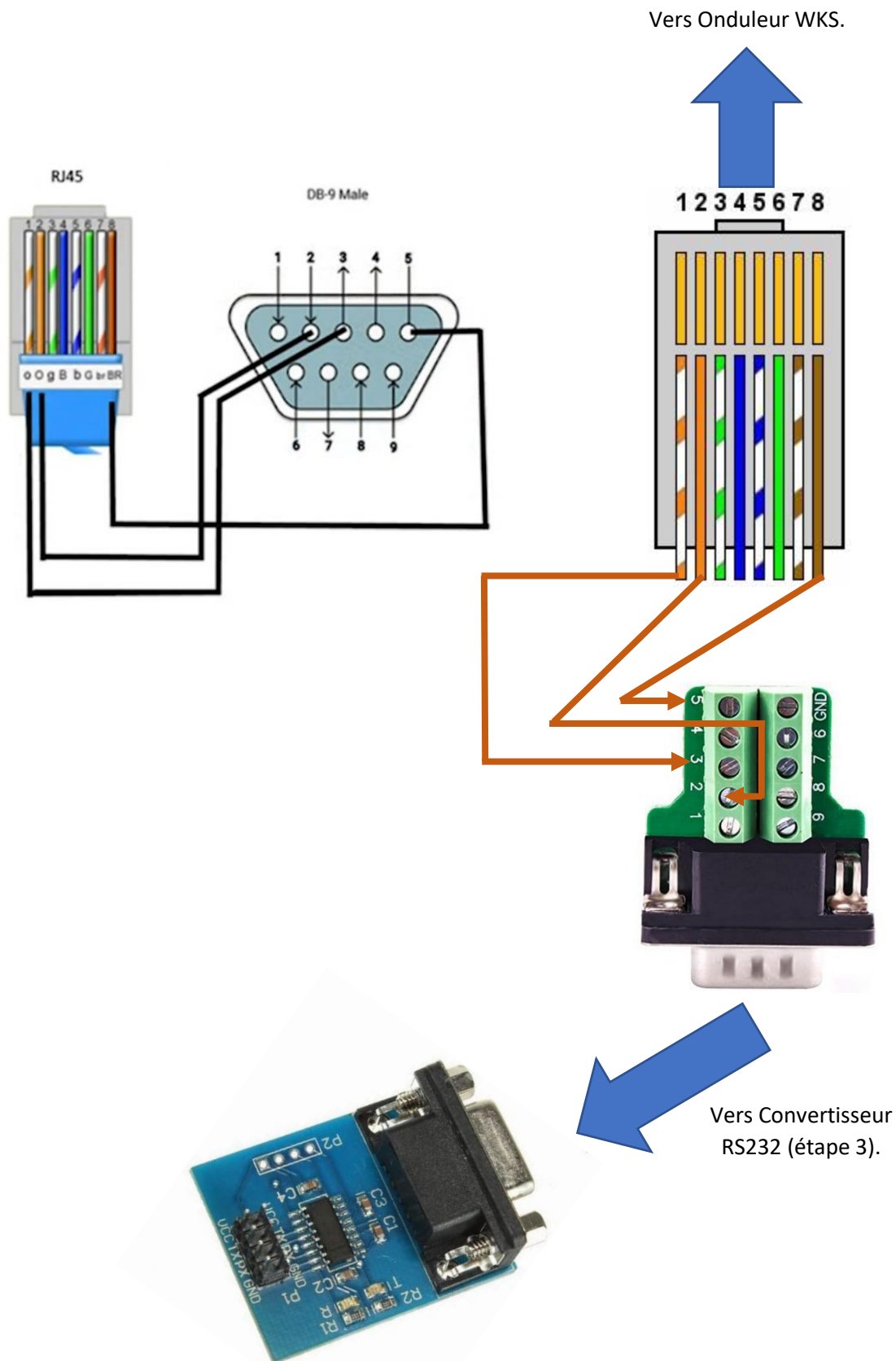
ETAPE 3 - Raccordement du convertisseur RS232 à la carte ARCELI :

Relier la broche **GND** du convertisseur RS232 à la broche **Ground** de la carte ARCELI,
Relier la broche **VCC** du convertisseur RS232 à la broche **3.3V output** de la carte ARCELI,
Relier la broche **TX** du convertisseur RS232 à la broche **(D5)** de la carte ARCELI,
Enfin relier la broche **RX** du convertisseur RS232 à la broche **(D6)** de la carte ARCELI.



ETAPE 4 - Raccordement du convertisseur RS232 à l'onduleur :

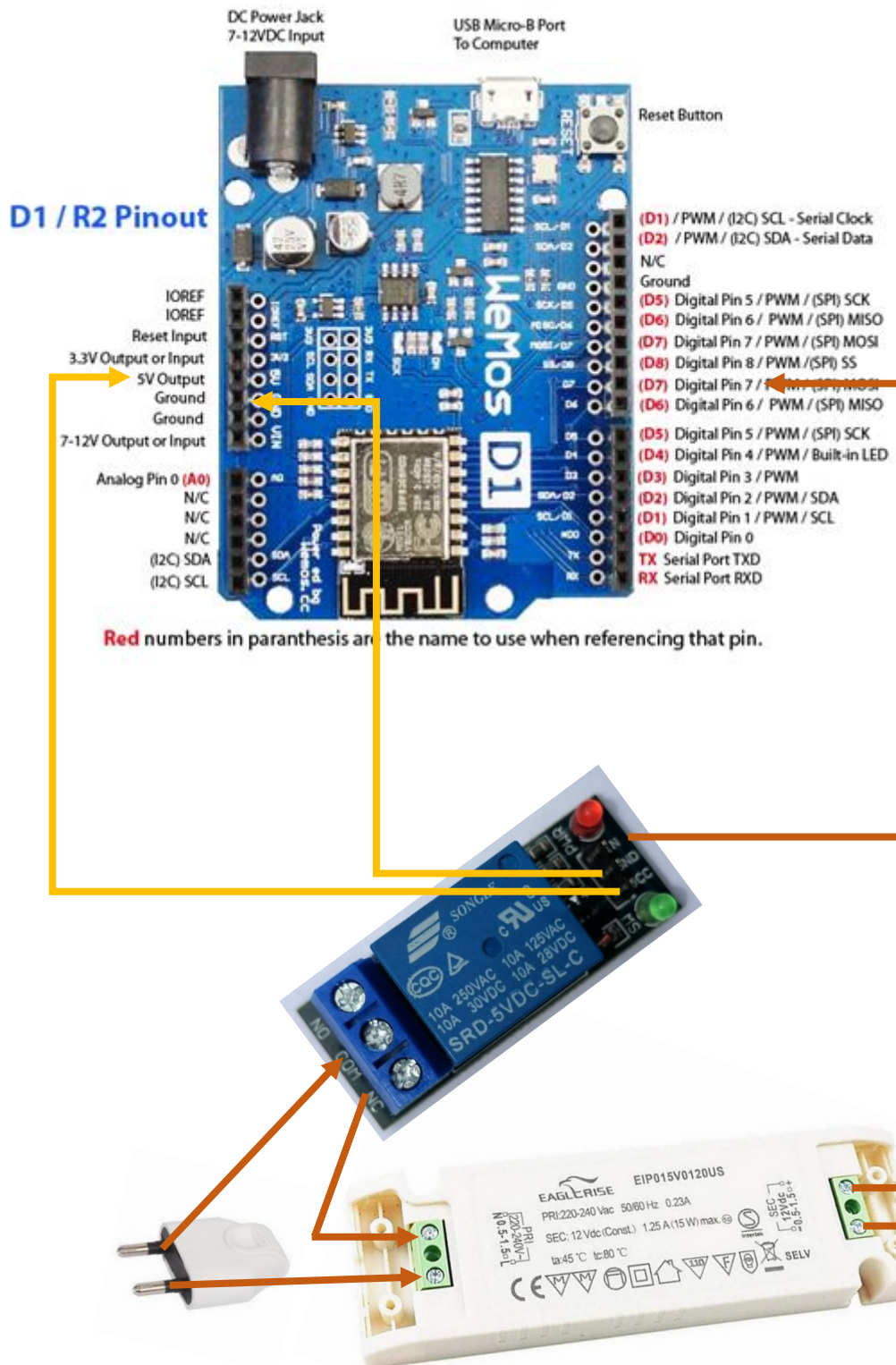
Le connecteur RJ45 se connecte au port COM (série ou RS232) de l'onduleur. Le connecteur RS232 (DB9) vient se connecter sur le convertisseur RS232 préparé en étape 3.



ETAPE 5 - Raccordement du relais à la carte ARCELI :

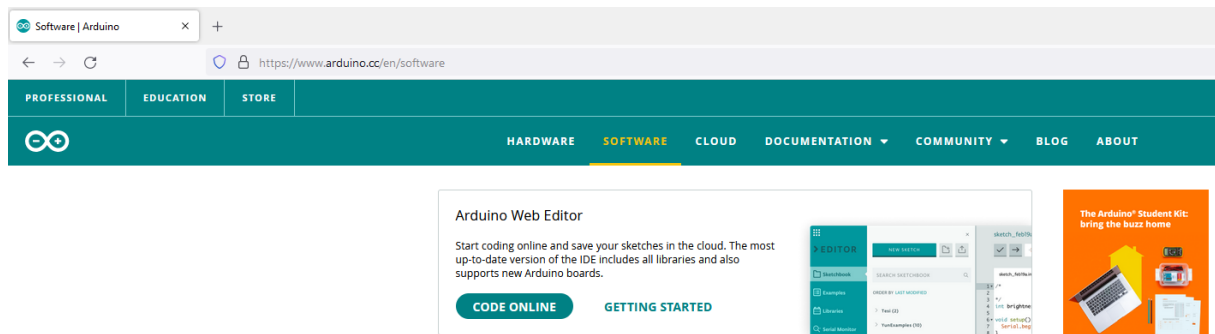
Relier la broche **GND** du relais à la broche **Ground** de la carte ARCELI,
Relier la broche **VCC** du relais à la broche **5V output** de la carte ARCELI,
Enfin relier la broche **IN** du relais à la broche **(D7)** de la carte ARCELI.

Sur le bornier de sortie du relais, couper la phase de l'alimentation secteur des ventilateurs avec les connecteurs **COM** (Common) au centre et le connecteur **NC** (Normally Closed).



ETAPE 6 – Installation du SDK et des bibliothèques logiciels :

Télécharger et installer le logiciel « Arduino IDE » disponible sur le site web <https://www.arduino.cc/> sous l'encart « Software » :



Downloads



Arduino IDE 1.8.19

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. This software can be used with any Arduino board.

Refer to the [Getting Started](#) page for Installation instructions.

SOURCE CODE

Active development of the Arduino software is [hosted by GitHub](#). See the instructions for [building the code](#). Latest release source code archives are available [here](#). The archives are PGP-signed so they can be verified using [this](#) gpg key.


DOWNLOAD OPTIONS

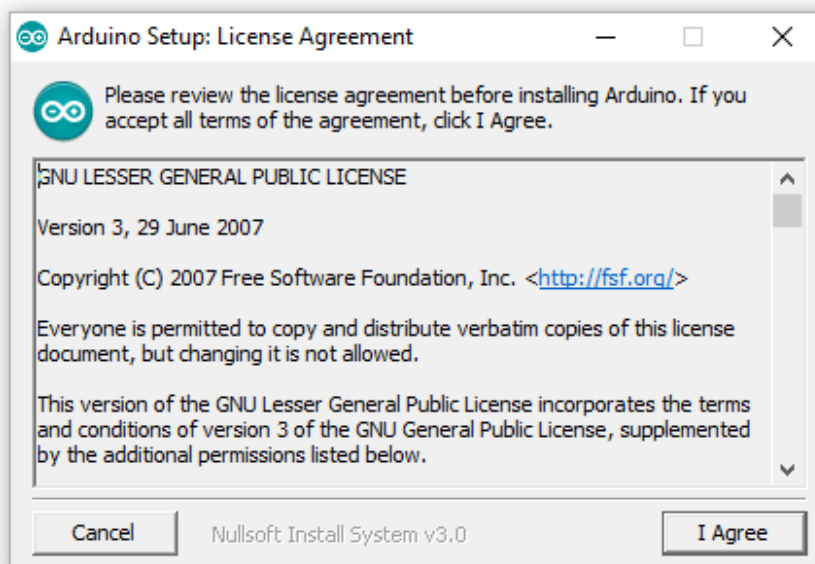
Windows Win 7 and newer
Windows ZIP file
Windows app Win 8.1 or 10 [Get](#)

Linux 32 bits
Linux 64 bits
Linux ARM 32 bits
Linux ARM 64 bits
Mac OS X 10.10 or newer


[Release Notes](#)
[Checksums \(sha512\)](#)

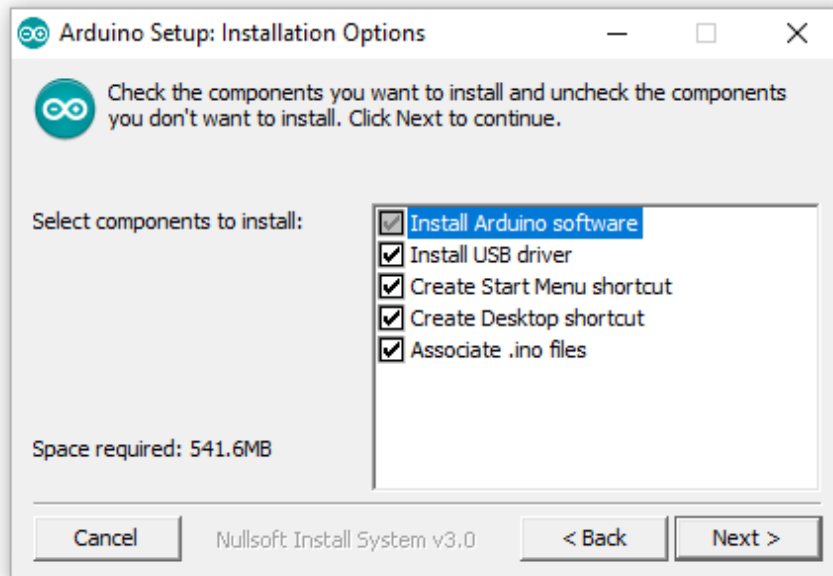
Lancer l'exécutable et accepter l'ensemble des messages...

Nom	Modifié le	Type	Taille
 arduino-1.8.19-windows.exe	13/06/2022 09:26	Application	114 557 Ko




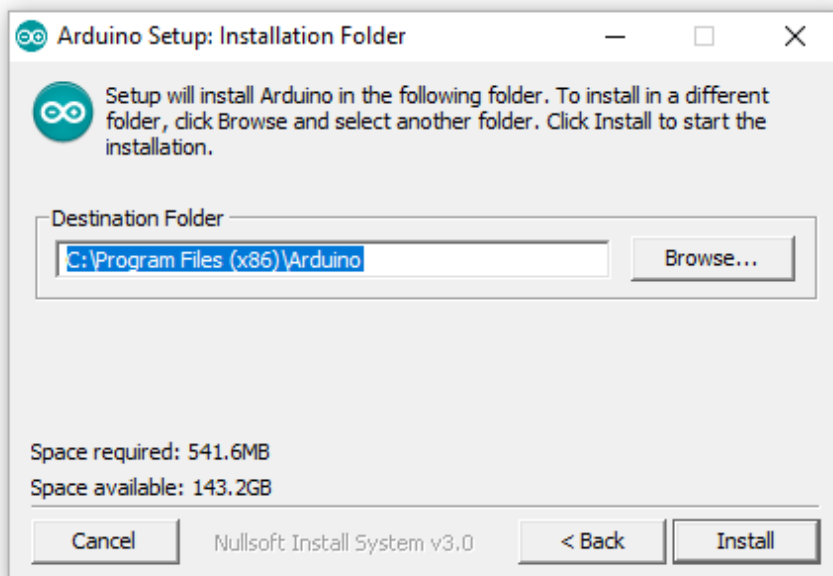
Faire « Next > »...

Nom	Modifié le	Type	Taille
 arduino-1.8.19-windows.exe	13/06/2022 09:26	Application	114 557 Ko




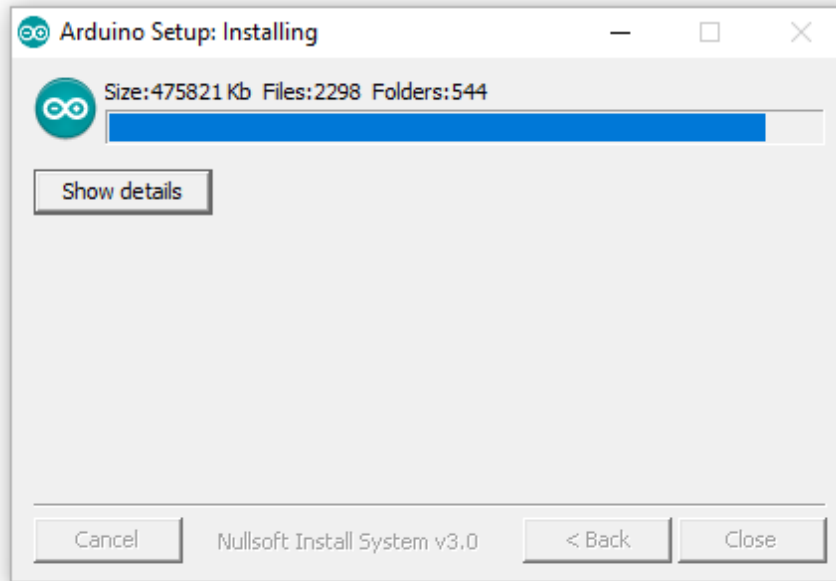
Faire « Install »...

Nom	Modifié le	Type	Taille
 arduino-1.8.19-windows.exe	13/06/2022 09:26	Application	114 557 Ko




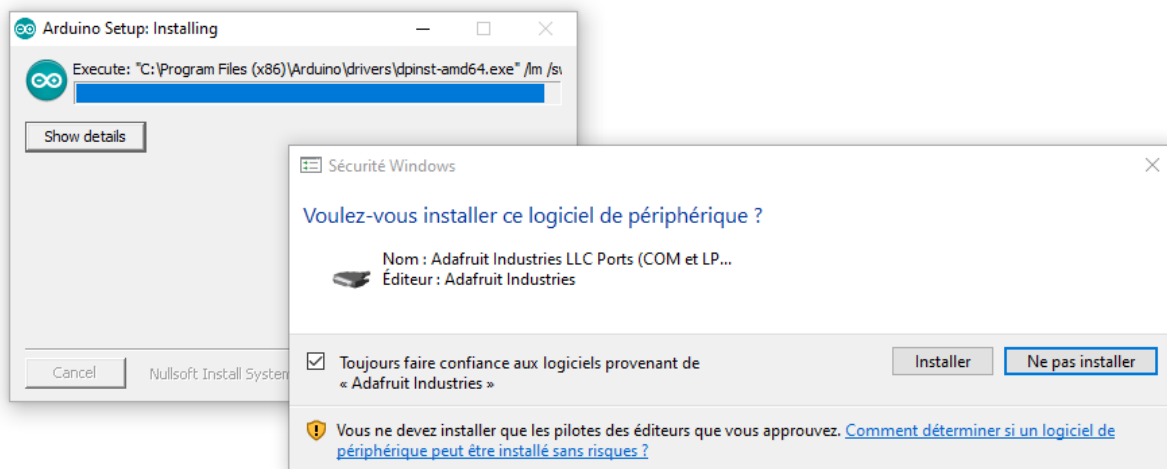
L'installation se déroule...

Nom	Modifié le	Type	Taille
 arduino-1.8.19-windows.exe	13/06/2022 09:26	Application	114 557 Ko

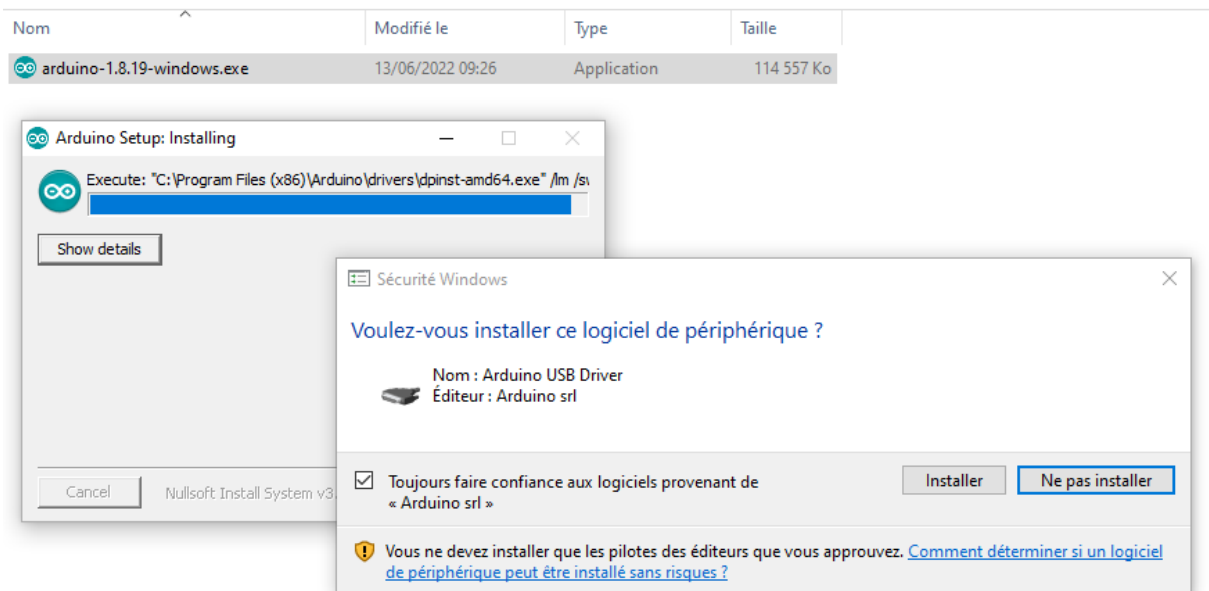


Faire « Installer » pour installer les pilotes périphériques non signés pour pouvoir accéder aux cartes par la suite...

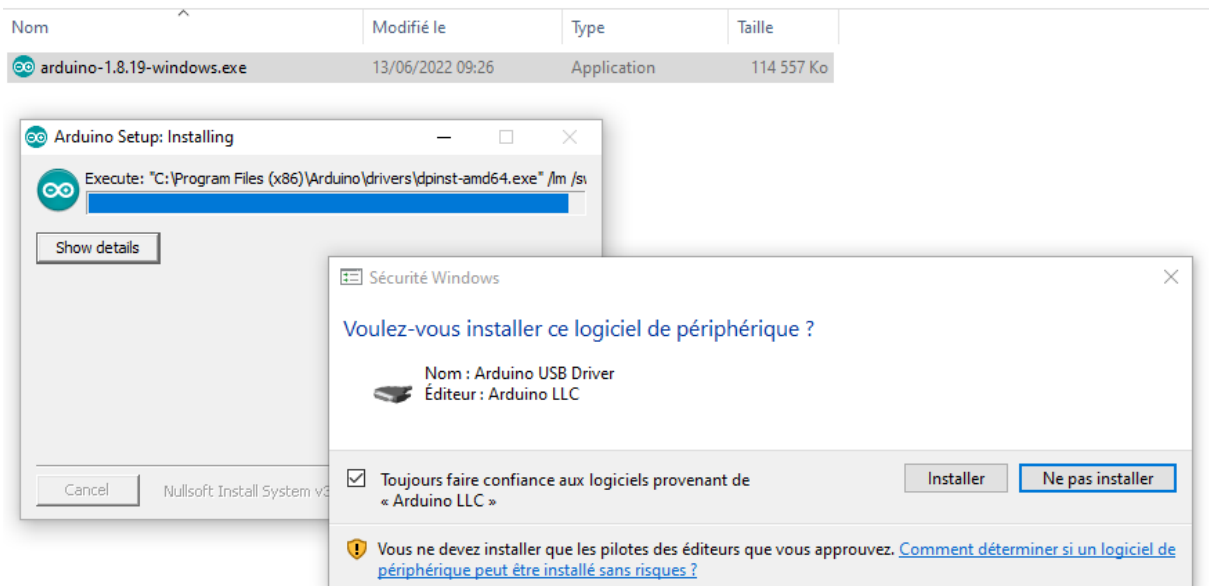
Nom	Modifié le	Type	Taille
 arduino-1.8.19-windows.exe	13/06/2022 09:26	Application	114 557 Ko



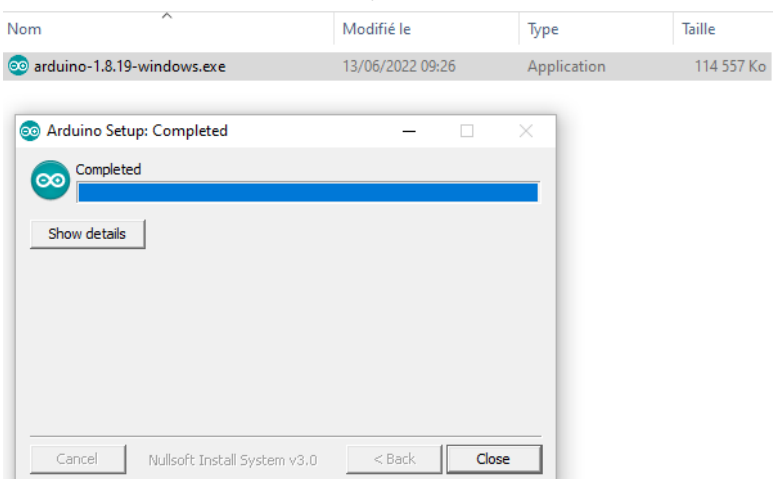
Faire « Installer » pour installer les pilotes périphériques non signés pour pouvoir accéder aux cartes par la suite...



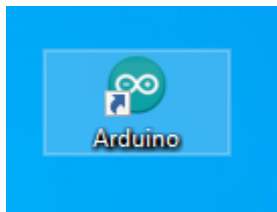
Faire « Installer » pour installer les pilotes périphériques non signés pour pouvoir accéder aux cartes par la suite...



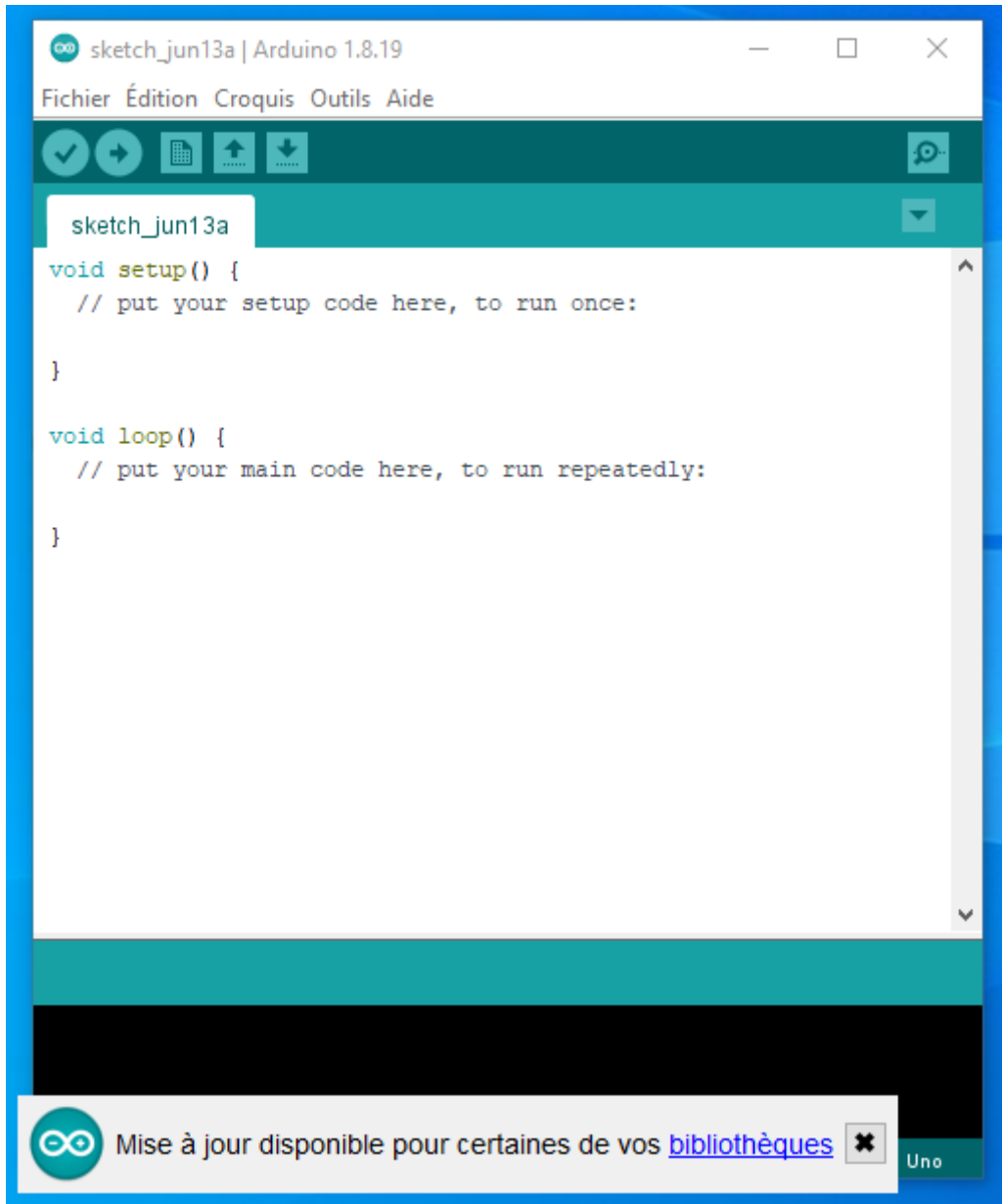
Une fois l'installation terminée, faire « Close »...



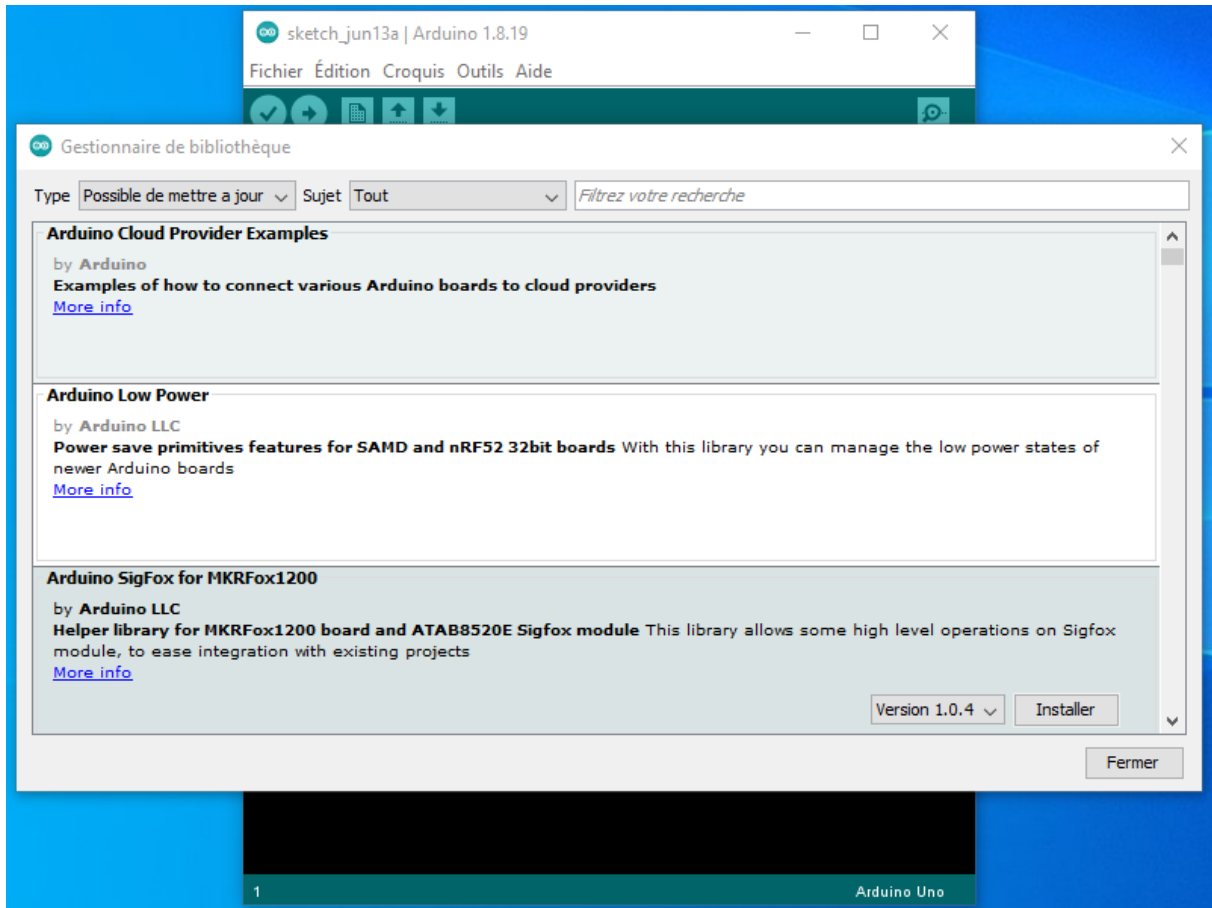
Lancer le logiciel « Arduino » en double cliquant sur l'icône correspondant sur le bureau...



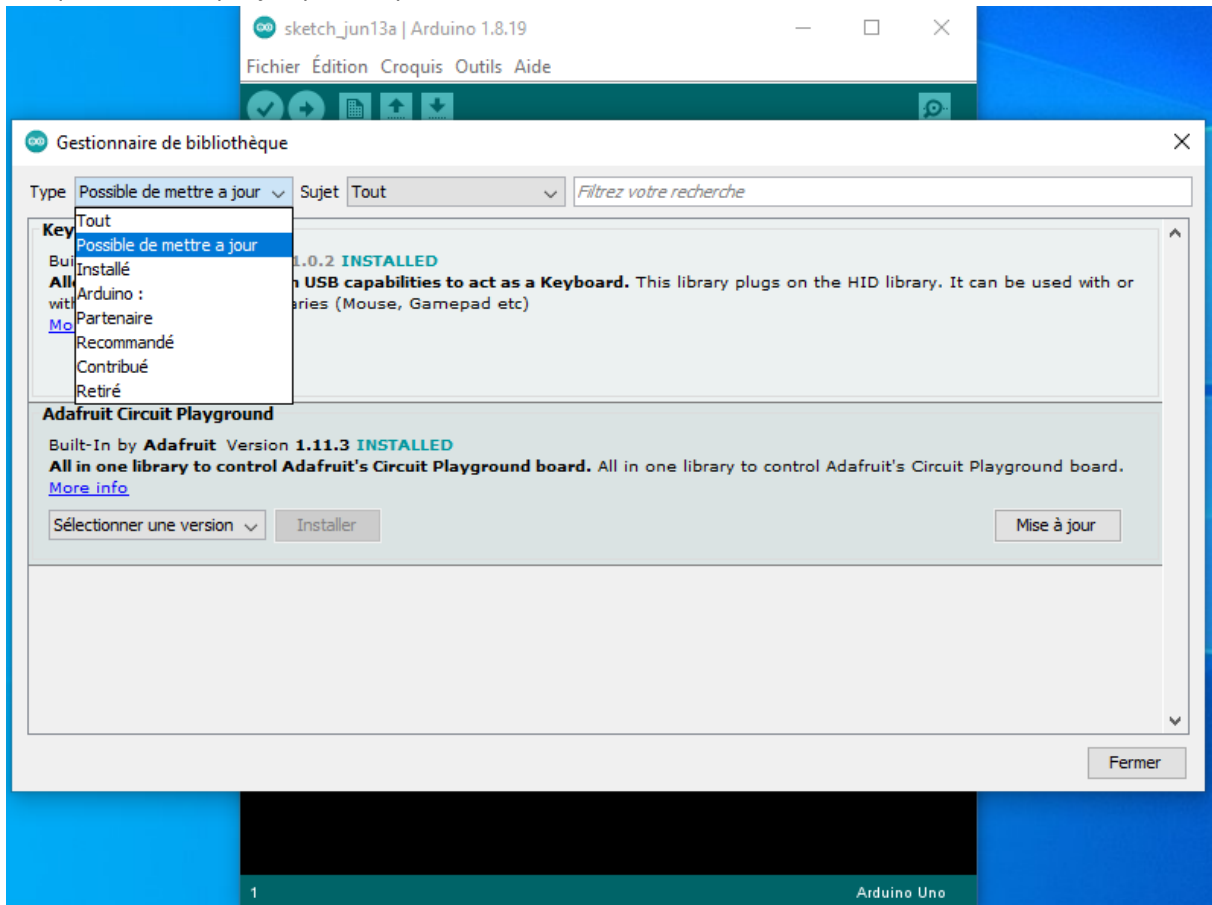
Attendre le message en bas de l'écran pour mettre à jour les bibliothèques et cliquer sur le lien « Bibliothèques »...



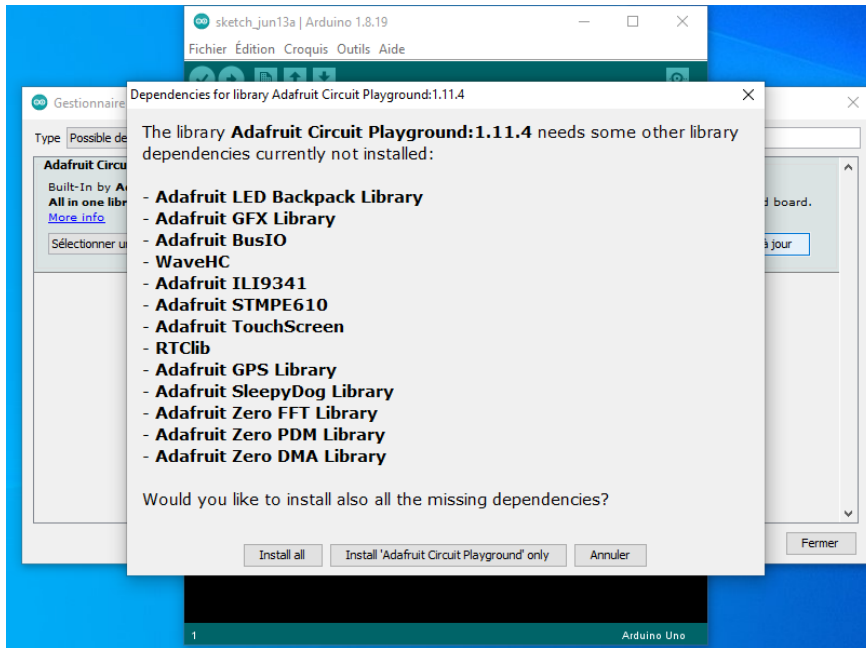
La liste des bibliothèques à mettre à jour apparaît...



Sélectionner le Type « Possible de mettre à jour » puis cliquer sur les boutons « Mise à jour » de chaque bibliothèque jusqu'à ce que la liste soit vide...

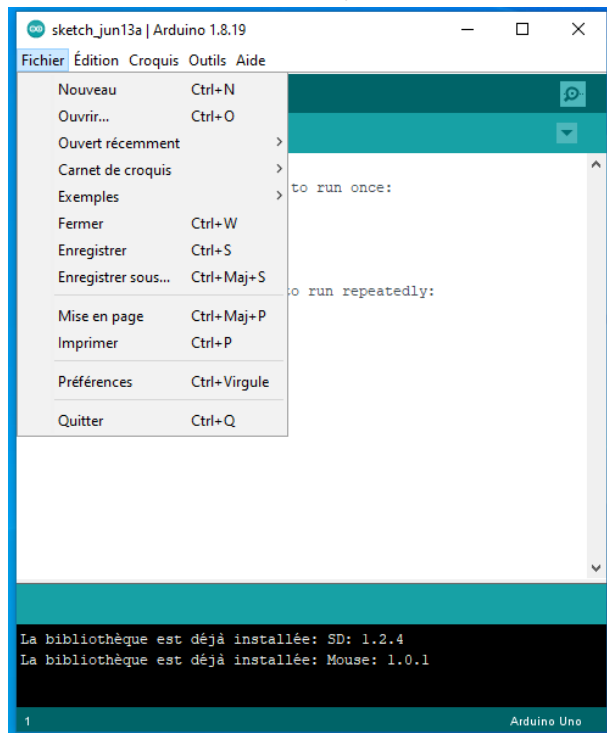


Lorsque des dépendances sont nécessaires faire « Install all »...



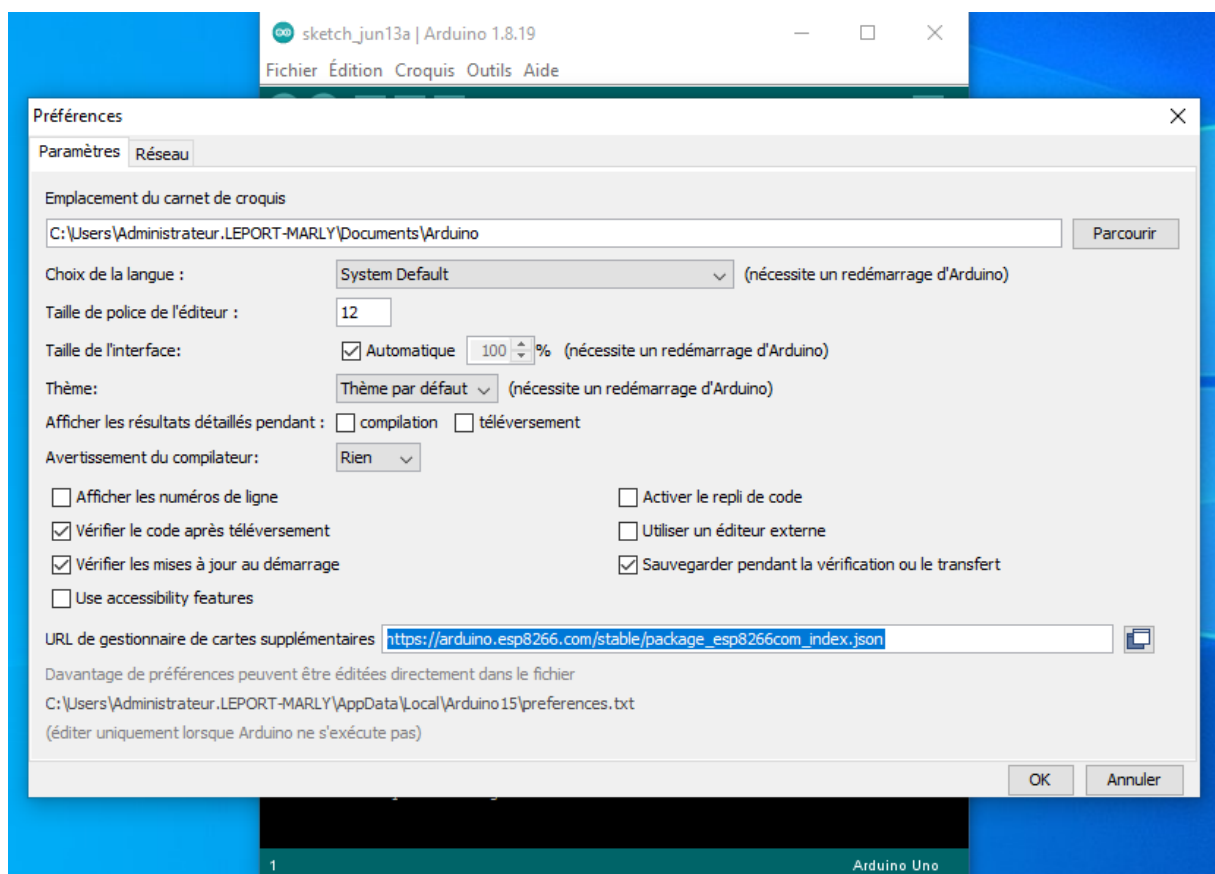
Lorsque les bibliothèques sont à jour, il faut installer les bibliothèques matérielles qui permettent de prendre en charge la carte ARCELI WEMOS D1 R2.

Aller dans le menu « Fichier / Préférences »...

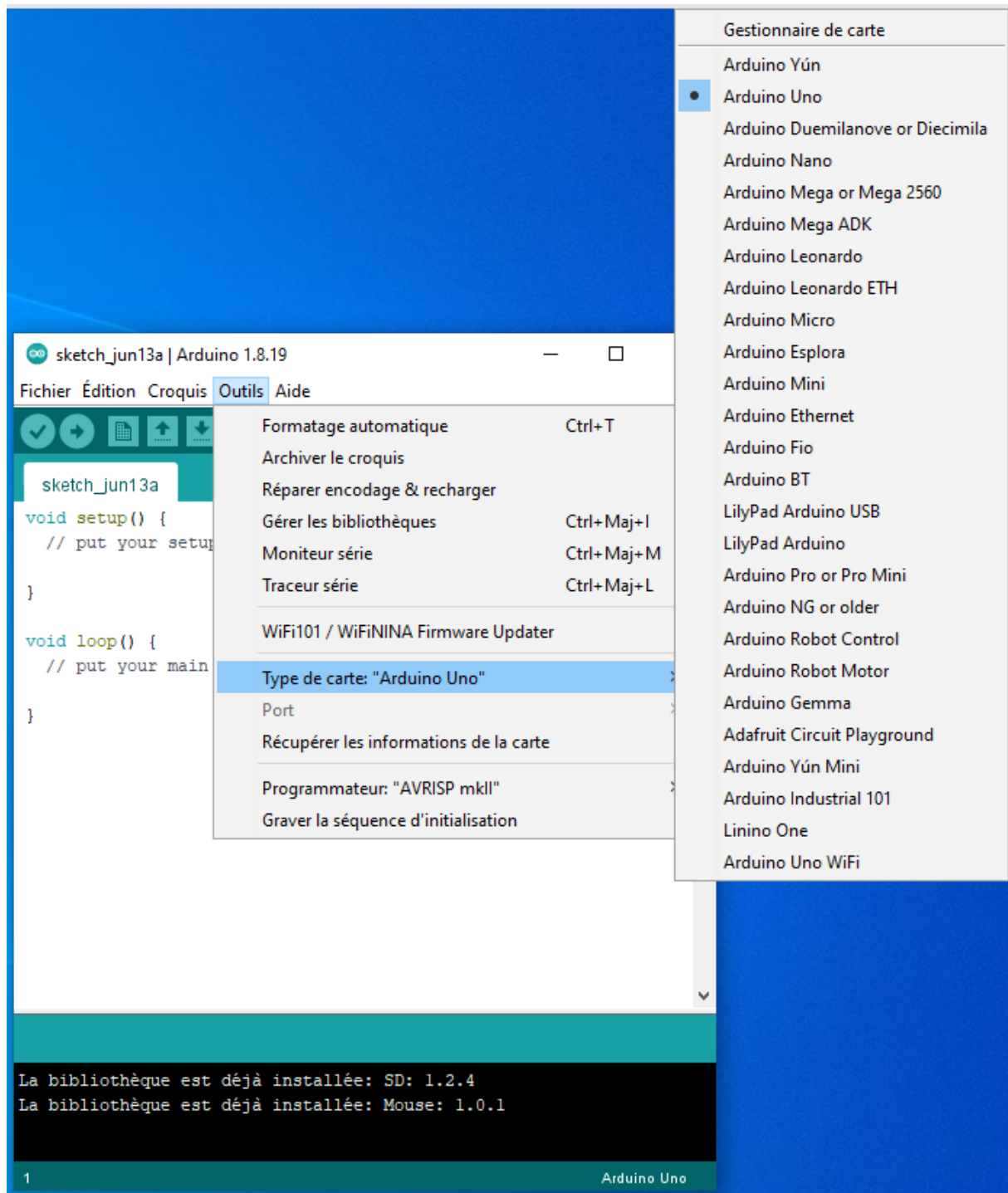


Ajouter sur la ligne « URL de gestionnaire de cartes supplémentaires » l'URL pour prendre en charge les ESP8266 qui est la suivante :

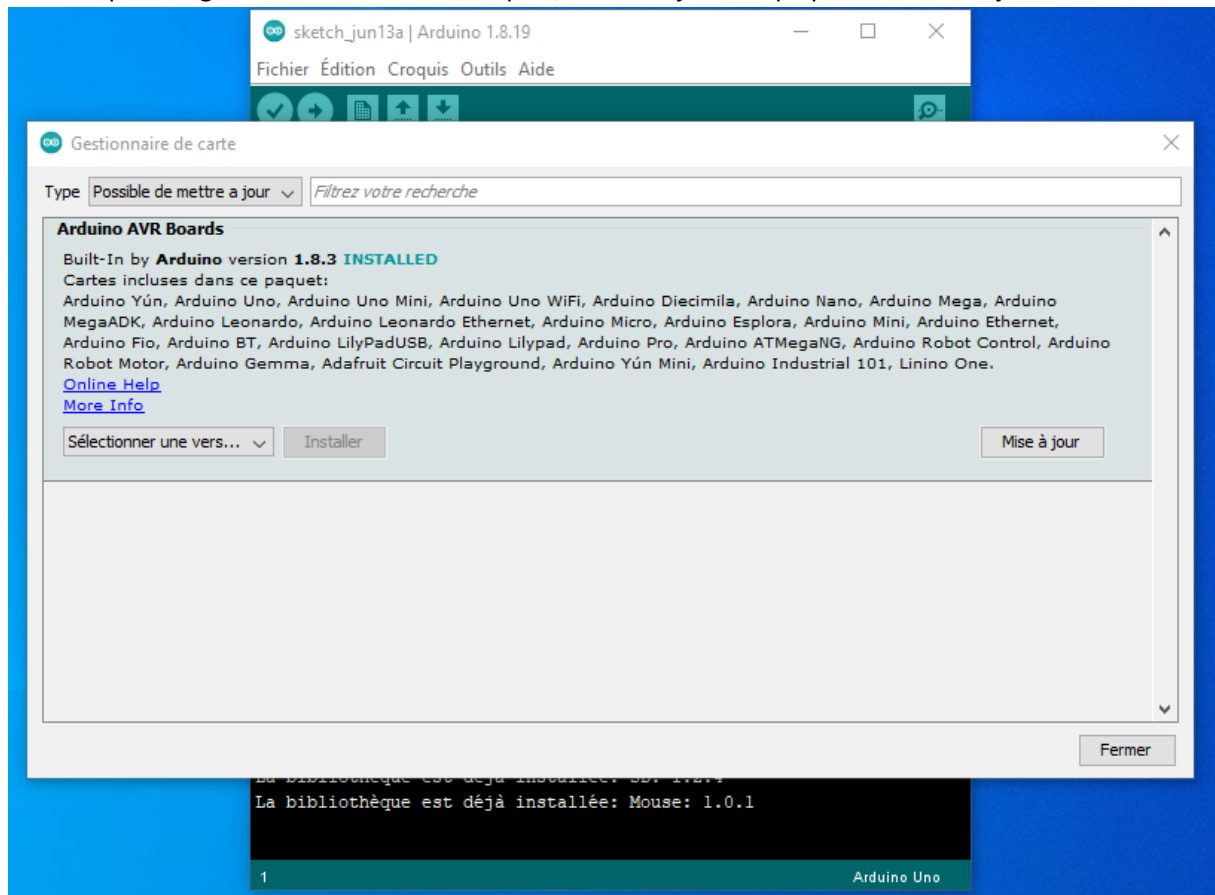
https://arduino.esp8266.com/stable/package_esp8266com_index.json



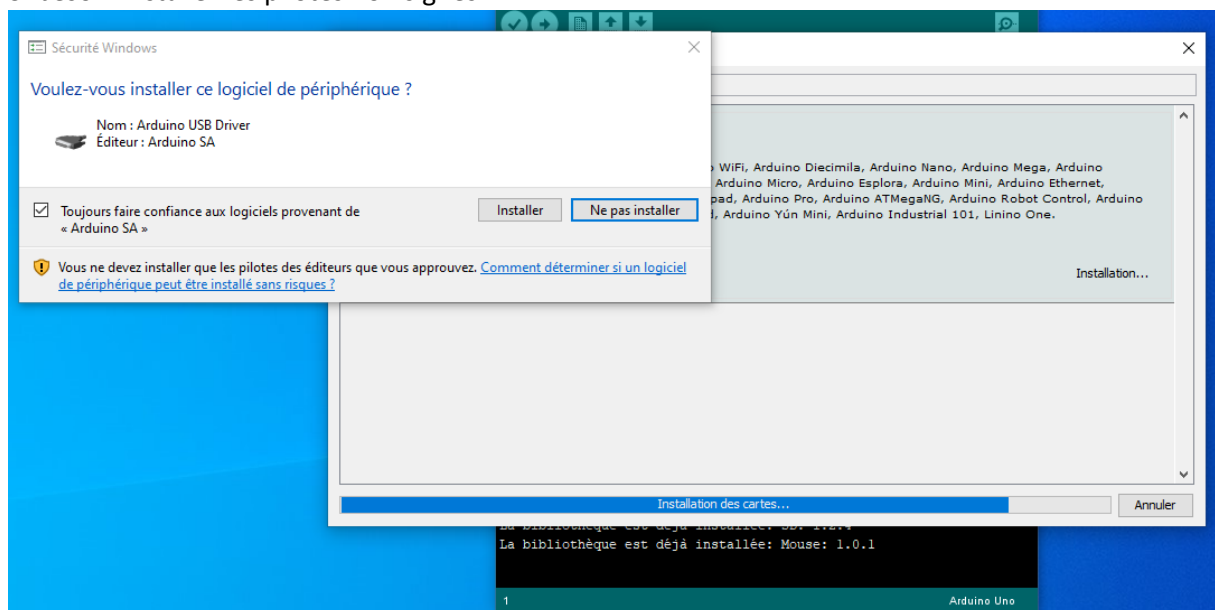
Aller dans le menu « Outils / Type de carte : "Arduino Uno" / Gestionnaire de carte »...



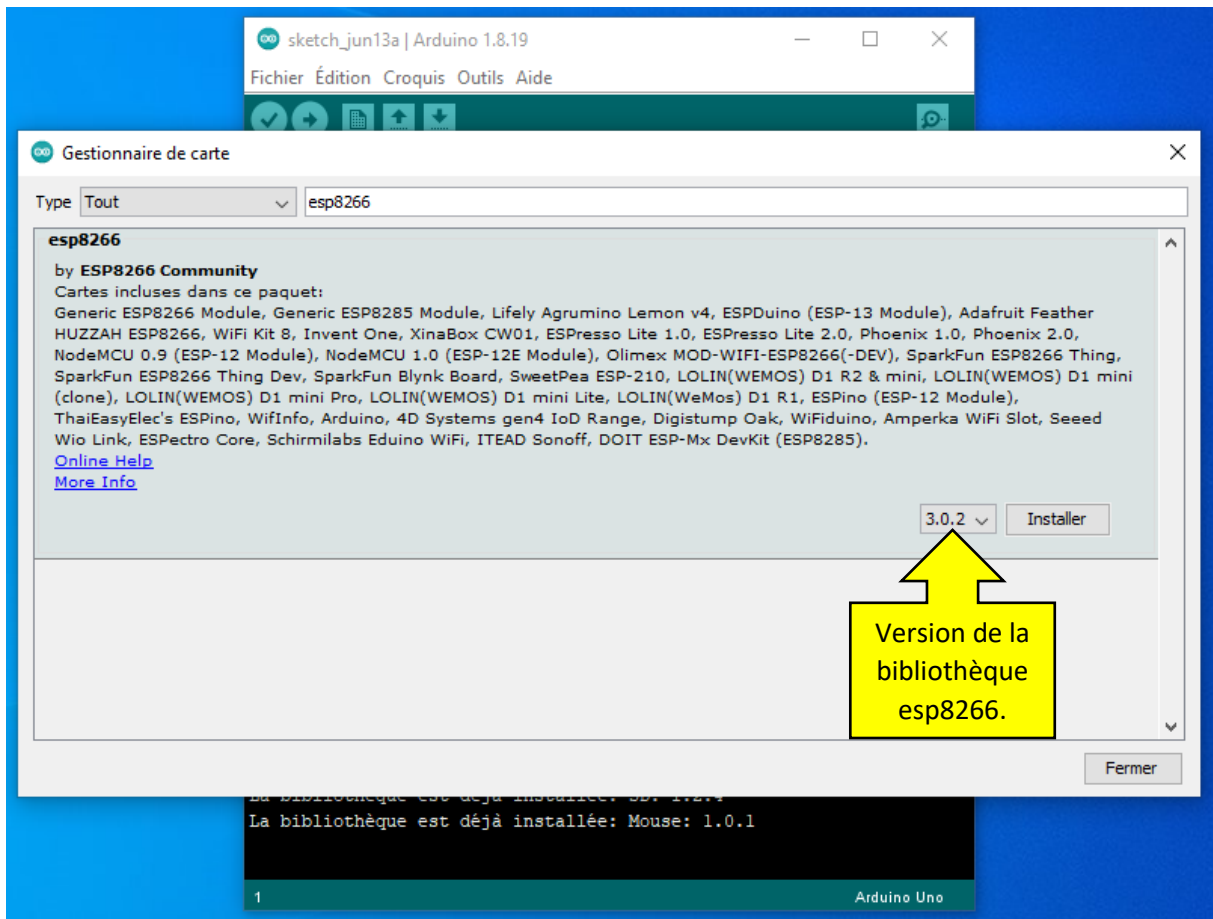
Comme pour le gestionnaire de bibliothèques, mettre à jour ce qui peut être mis à jour...



Si besoin installer les pilotes non signés...



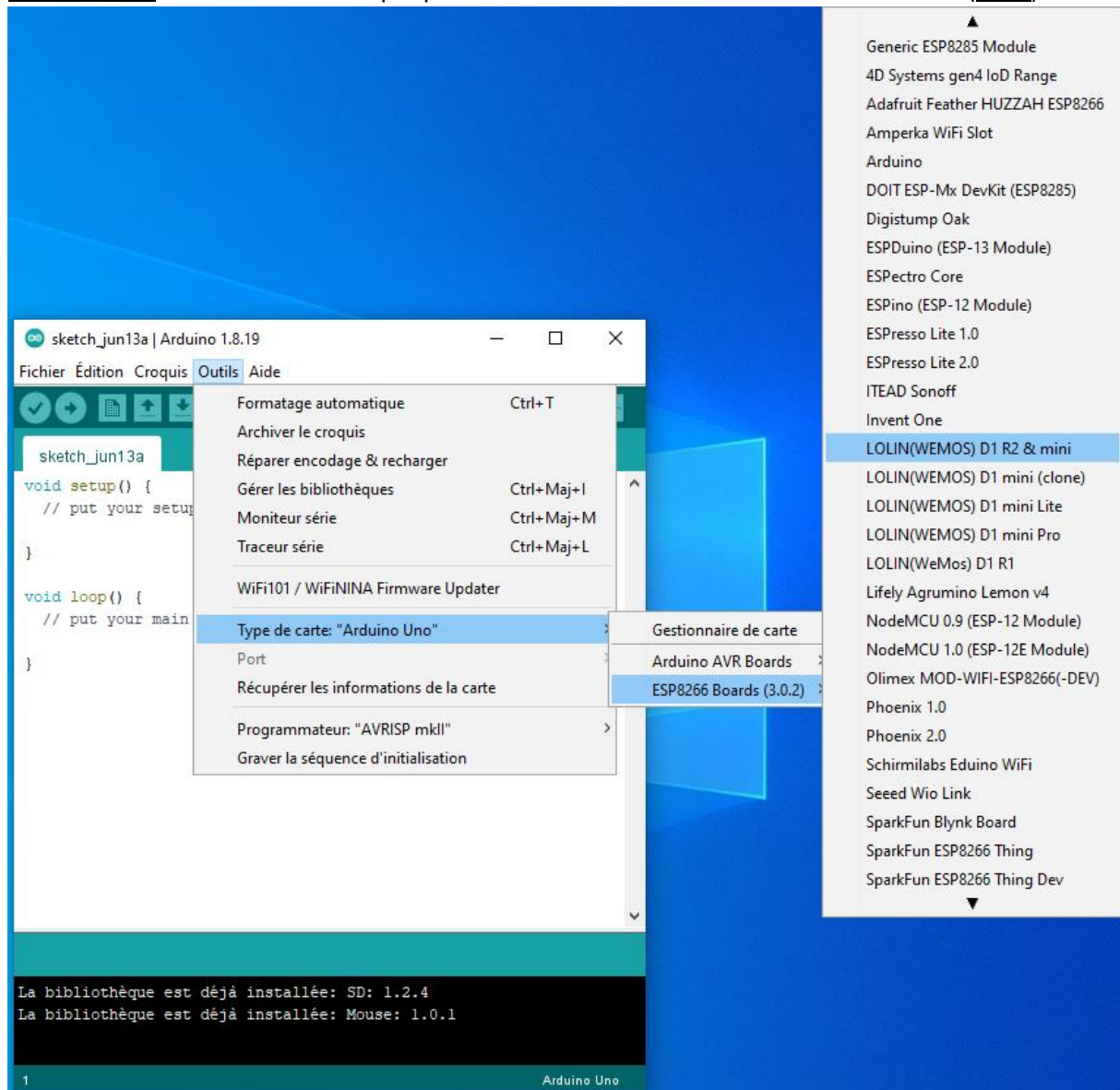
Sélectionner le Type : « Tout » puis rechercher « **esp8266** » pour installer la prise en charge de notre carte ARCELI WEMOS D1 R2...



Le code source a été compilé avec succès avec la bibliothèque esp8266 version 3.0.2 (six mois de fonctionnement sans anomalie) et a dû subir quelques modifications avec la version 3.1.0.

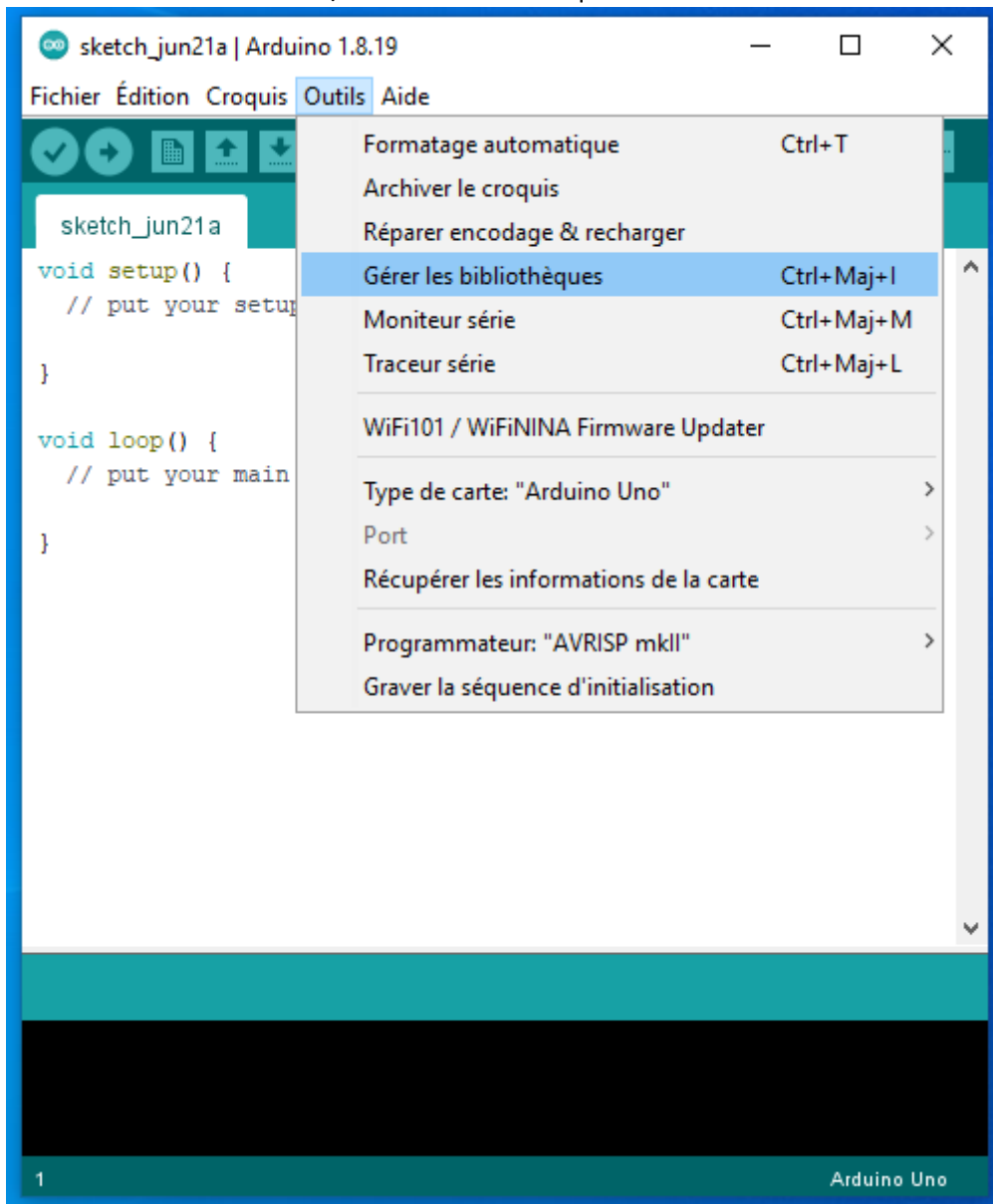
Une fois la bibliothèque matérielle installée, l'utilisation du nouveau matériel sera possible dans l'environnement de développement.

Sélectionner la carte dans le menu « Outils / type de carte / ESP8266 Boards (3.0.2) / **LOLIN(WEMOS) D1 R2 & mini** ». Le code a été adapté pour fonctionner avec la version ESP8266 Boards (3.1.0).



Maintenant que le matériel est reconnu par le logiciel « Arduino IDE », nous devons ajouter les bibliothèques nécessaires à l'exploitation des différents composants du projet (WIFI, Relais, sondes DHT22, LCD, etc...).

Aller dans le menu « Outils / Gérer les bibliothèques »...



Installer les bibliothèques suivantes pour pouvoir compiler le projet (les versions des bibliothèques peuvent être plus récentes) :

- **NTPClient :**

Gestionnaire de bibliothèque

Type Sujet

NTPClient
by **Fabrice Weinberg**
An NTPClient to connect to a time server Get time from a NTP server and keep it in sync.
[More info](#)

Version 3.2.1

- **Adafruit Unified Sensor :**

Adafruit Unified Sensor
by **Adafruit**
Required for all Adafruit Unified Sensor based libraries. A unified sensor abstraction layer used by many Adafruit sensor libraries.
[More info](#)

Version 1.1.5

- **DHT sensor library :**

Type Sujet

DHT sensor library
by **Adafruit**
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
[More info](#)

Version 1.4.3

- **DHT sensor library for ESPx :**

Type Sujet

DHT sensor library for ESPx
by **beegee_tokyo**
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors Optimized library to match ESP32 requirements. Last changes: Fix negative temperature problem (credits @helijunky)
[More info](#)

Version 1.18.0

- **LiquidCrystal I2C :**

Type Sujet

LiquidCrystal I2C
by **Marco Schwartz**
A library for I2C LCD displays. The library allows to control I2C displays with functions extremely similar to LiquidCrystal library. THIS LIBRARY MIGHT NOT BE COMPATIBLE WITH EXISTING SKETCHES.
[More info](#)

Version 1.1.2

- **Time :**

Type	Tout	Sujet	Tout	time
<p>Time</p> <p>by Paul Stoffregen</p> <p>Timekeeping functionality for Arduino Date and Time functions, with provisions to synchronize to external time sources like GPS and NTP (Internet). This library is often used together with TimeAlarms and DS1307RTC.</p> <p>More info</p> <p>Version 1.6.1 <input type="button" value="Installer"/></p>				

- **Timezone :**

Type	Tout	Sujet	Tout	timezone
<p>Timezone</p> <p>by Jack Christensen</p> <p>Arduino library to facilitate time zone conversions and automatic daylight saving (summer) time adjustments. The primary aim of the Timezone library is to convert Universal Coordinated Time (UTC) to the correct local time, whether it is daylight saving time (a.k.a. summer time) or standard time.</p> <p>More info</p> <p>Version 1.2.4 <input type="button" value="Installer"/></p>				

ETAPE 7 – Inspection du code, Compilation puis Chargement dans la carte ARCELI :

Ouverture du projet et première compilation de vérification :

Ouvrir le fichier « WKS_ip_Claude_LCD2004_Et_Temp_H_B_R_T.ino » en double cliquant dessus dans l'explorateur Windows (attention : Le fichier doit absolument être seul dans son dossier).



Cliquer sur le bouton « V » pour compiler le projet et vérifier qu'il n'y a pas d'erreur :



Si la compilation se déroule bien (en version 3.0.2 de la bibliothèque esp8266), seul deux avertissements sans importance apparaissent dans la fenêtre de compilation :

ATTENTION : la bibliothèque Timezone prétend être exécutable sur la (ou les) architecture(s) avr et peut être incompatible avec votre carte actuelle qui s'exécute sur esp8266.

ATTENTION : la bibliothèque LiquidCrystal_I2C prétend être exécutable sur la (ou les) architecture(s) avr et peut être incompatible avec votre carte actuelle qui s'exécute sur esp8266.

Executable segment sizes:

```
ICACHE : 32768          - flash instruction cache
IROM   : 299116         - code in flash (default or ICACHE_FLASH_ATTR)
IRAM    : 30561 / 32768 - code in IRAM (IRAM_ATTR, ISRs...)
DATA    : 1652 )        - initialized variables (global, static) in RAM/HEAP
RODATA  : 3992 ) / 81920 - constants (global, static) in RAM/HEAP
BSS     : 27448 )        - zeroed variables (global, static) in RAM/HEAP
```

Le croquis utilise 335321 octets (32%) de l'espace de stockage de programmes. Le maximum est de 1044464 octets.

Les variables globales utilisent 33092 octets (40%) de mémoire dynamique, ce qui laisse 48828 octets pour les variables locales. Le maximum est de 81920 octets.

En version 3.1.0 de la bibliothèque esp8266 :

ATTENTION : la bibliothèque Timezone-master prétend être exécutable sur la (ou les) architecture(s) avr et peut être incompatible avec votre carte actuelle qui s'exécute sur esp8266.

ATTENTION : la bibliothèque LiquidCrystal_I2C prétend être exécutable sur la (ou les) architecture(s) avr et peut être incompatible avec votre carte actuelle qui s'exécute sur esp8266.

. Variables and constants in RAM (global, static), used 33208 / 80192 bytes (41%)

SEGMENT	BYTES	DESCRIPTION
DATA	1648	initialized variables
RODATA	4008	constants
BSS	27552	zeroed variables

. Instruction RAM (IRAM_ATTR, ICACHE_RAM_ATTR), used 63019 / 65536 bytes (96%)

SEGMENT	BYTES	DESCRIPTION
ICACHE	32768	reserved space for flash instruction cache
IRAM	30251	code in IRAM

. Code in flash (default, ICACHE_FLASH_ATTR), used 299500 / 1048576 bytes (28%)

SEGMENT	BYTES	DESCRIPTION
IROM	299500	code in flash

A ce stade, il est nécessaire de configurer certains paramètres directement dans le code avant de l'envoyer dans la carte ARCELI :

```
// Paramètres WIFI...
static const String monSSID = "Insérer ICI le SSID de votre WIFI";
static const String monPass = "Insérer ICI le mot de passe de votre WIFI";
static const String HostName = "Insérer ICI le nom d'hôte de votre périphérique";

static const String WeMos_UPS_Version = "6.9"; // Vous pouvez changer le No de version si
                                                nécessaire.

// Paramètre Du Relais Ventilos...
static const float TempStartWKS = 32.50; // Température de déclenchement des ventilos pour
                                          le WKS.

static const float TempStopWKS = 29.50; // Température d'arrêt des ventilos pour le WKS si
                                          ventilos démarrés.

static const float TempStartBat = 25.50; // Température de déclenchement des ventilos pour
                                          les batteries.

static const float TempStopBat = 22.50; // Température d'arrêt des ventilos pour les
                                          batteries si ventilos démarrés.

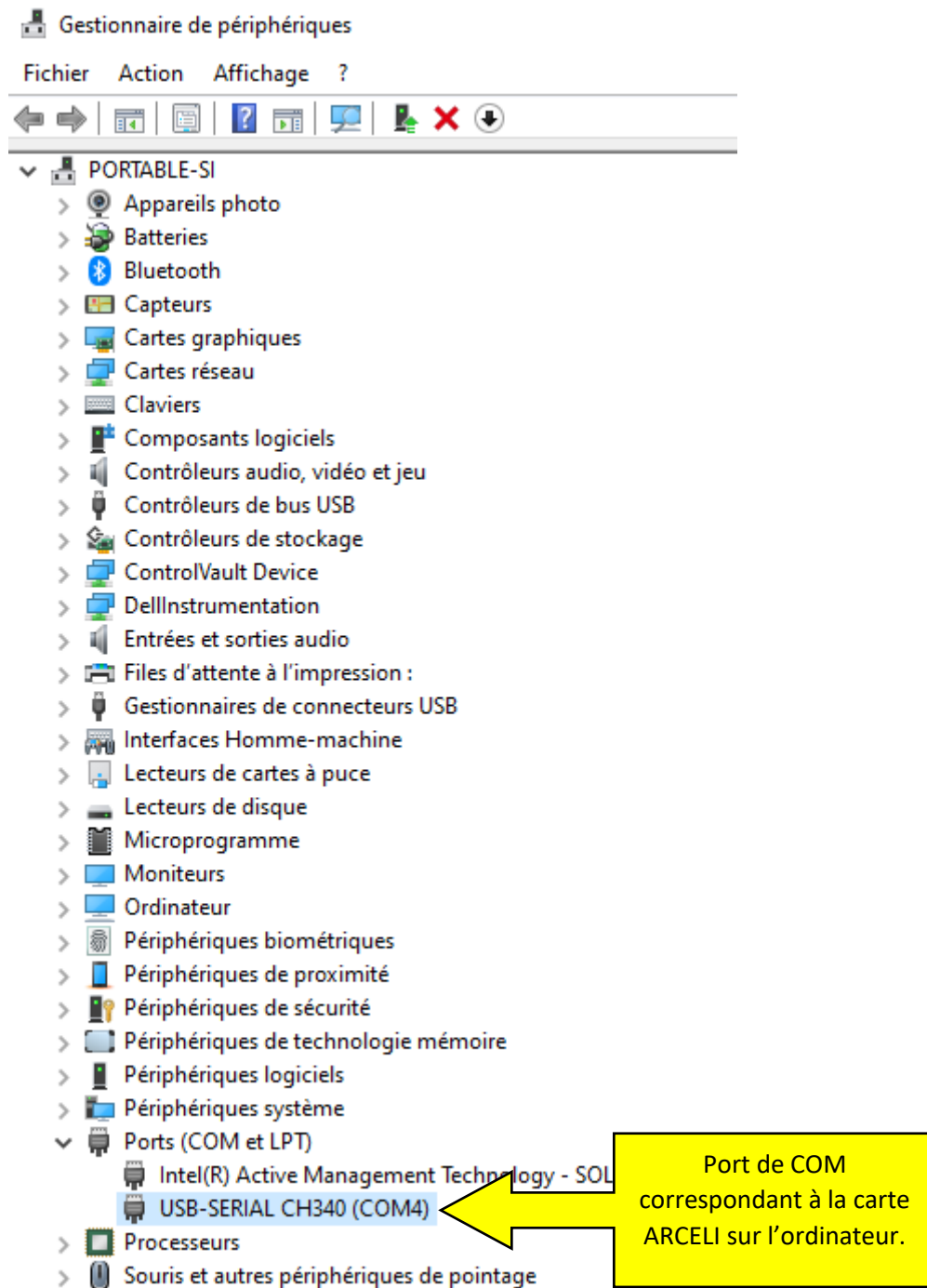
static const int HStart = 2; // Heure de lancement des ventilos forcée.

static const int HStop = 7; // Heure d'arrêt des ventilos forcée (heure comprise
                             donc 7 signifie arrêt à 8H).
```

Localiser la fonction **void setup()** pour y modifier les derniers paramètres réseaux suivants :
ATTENTION : Les paramètres suivants sont séparés par des virgules et non des par points.

```
IPAddress LAN_ADDR(192,168,0,6); // Adresse IP du périphérique.
IPAddress LAN_GW(192,168,0,1); // Adresse IP de la passerelle.
IPAddress LAN_SUBNET(255,255,255,0); // Masque de sous réseau.
IPAddress LAN_DNS(192,168,0,1); // Adresse IP du serveur DNS.
```

Avant d'envoyer le code dans la carte ARCELI, il faut trouver le port série sur lequel est connecté la carte ARCELI sur l'ordinateur. Reliez le connecteur micro-USB de la carte ARCELI à un port USB du PC puis ouvrez le gestionnaire de périphériques. Repérez sous la section **Ports (COM et LPT)** un périphérique qui porterait un nom comme **CH340**. En face figure le No du port de COM à configurer dans le logiciel « Arduino IDE ». Dans l'exemple ci-dessous il s'agit du port **COM4**.



Dans le logiciel « Arduino IDE », ouvrir le menu « Outils / Port: » puis sélectionner le port de COM correspondant à celui que vous avez trouvé ci-dessus. Dans notre exemple il s'agit de **COM4**.

sketch_jun24a | Arduino 1.8.19

Fichier Édition Croquis Outils Aide

The screenshot shows the Arduino IDE interface. The 'Tools' menu is open, and the 'Port' option is highlighted. A yellow callout box with an arrow points to the 'Port' option, containing the text: 'Ouvrir le menu « Outils / Port: » et sélectionner le port adéquat.' The 'Port' sub-menu is also visible, showing 'COM4' selected with a checkmark. The background shows the Arduino IDE code editor with a sketch named 'sketch_jun24a'.

```
void setup() {  
  // put your setup code here  
}  
  
void loop() {  
  // put your main code here, to be executed  
  // over and over again  
}
```

Tools menu options:

- Formatage automatique (Ctrl+T)
- Archiver le croquis
- Réparer encodage & recharger
- Gérer les bibliothèques (Ctrl+Maj+I)
- Moniteur série (Ctrl+Maj+M)
- Traceur série (Ctrl+Maj+L)
- WiFi101 / WiFinINA Firmware Updater
- Type de carte: "LOLIN(WEMOS) D1 R2 & mini" >
- Upload Speed: "115200" >
- CPU Frequency: "80 MHz" >
- Flash Size: "4MB (FS:2MB OTA:~1019KB)" >
- Debug port: "Disabled" >
- Debug Level: "Rien" >
- IwLP Variant: "v2 Lower Memory" >
- VTables: "Flash" >
- C++ Exceptions: "Disabled (new aborts on oom)" >
- Stack Protection: "Disabled" >
- Erase Flash: "Only Sketch" >
- SSL Support: "All SSL ciphers (most compatible)" >
- MMU: "32KB cache + 32KB IRAM (balanced)" >
- Non-32-Bit Access: "Use pgm_read macros for IRAM/PROGMEM" >
- Port: "COM4" >
- Récupérer les informations de la carte
- Programmeur >
- Graver la séquence d'initialisation

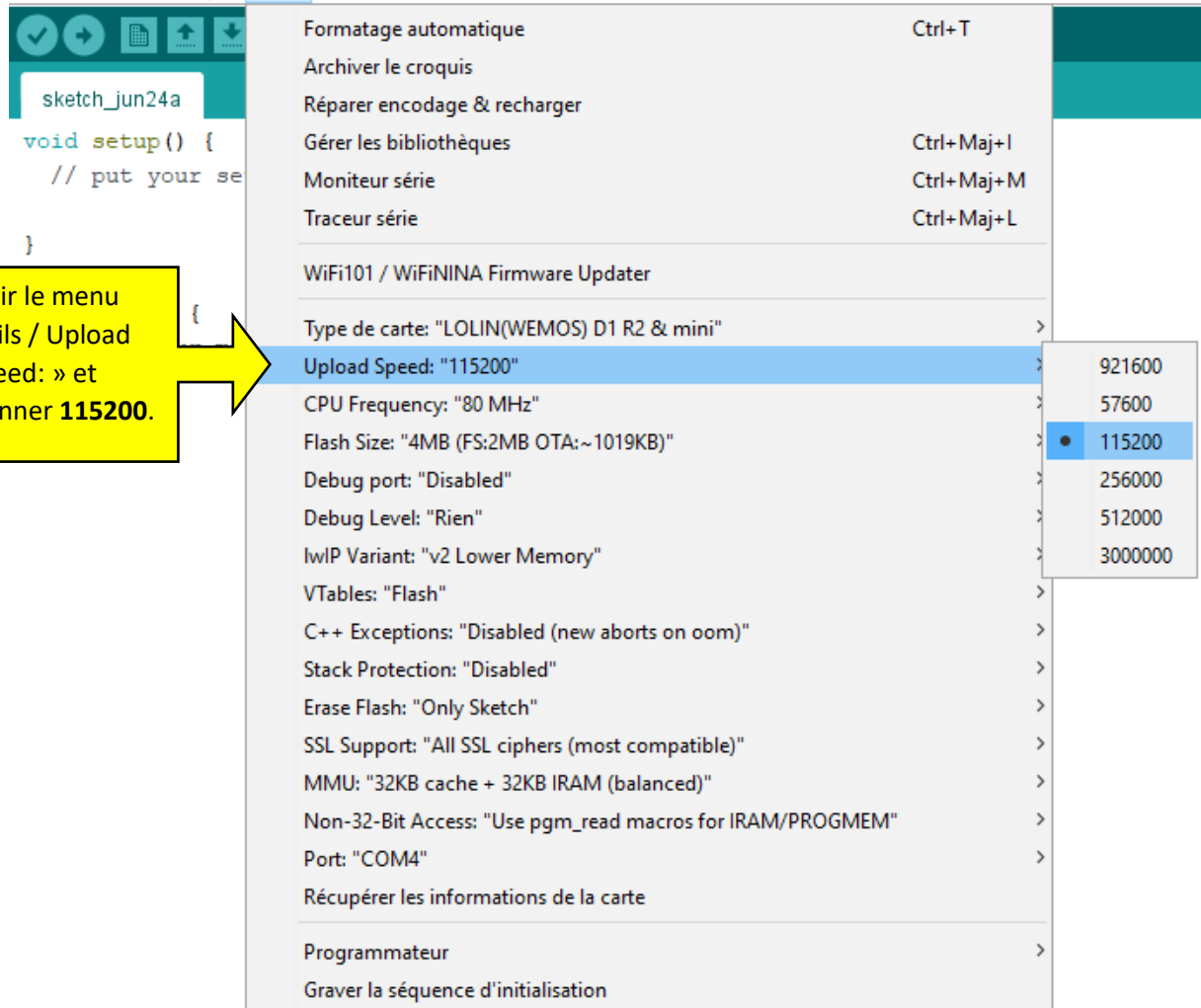
Port: "COM4" sub-menu:

- Ports série
- ✓ COM4
- COM7

Enfin, il ne reste plus qu'à sélectionner « **115200** » dans le menu « Outils / Upload Speed » :

sketch_jun24a | Arduino 1.8.19

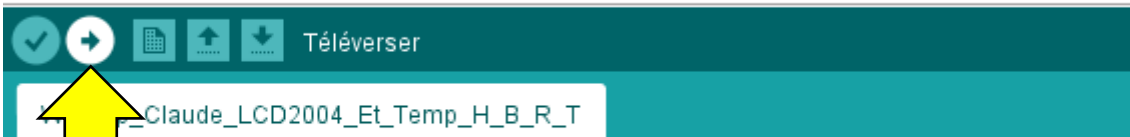
Fichier Édition Croquis Outils Aide



A ce stade, tous les composants de l'étape 1 à l'étape 5 sont câblés sur la carte ARCELI, la compilation se déroule sans erreur et les paramètres d'échange entre le PC et la carte ARCELI sont configurés. Cliquer sur le bouton en forme de flèche en face du « V » pour compiler et transférer le code dans la carte ARCELI.

WKS_ip_Claude_LCD2004_Et_Temp_H_B_R_T | Arduino 1.8.19

Fichier Édition Croquis Outils Aide



Cliquer sur le bouton en forme de flèche.

```
#include <ESP8266WiFi.h>
#include <WiFiClient.h>

#include <TimeLib.h>
#include <time.h>           // https://github.com/PaulStoffregen/Time
#include <Timezone.h>       // https://github.com/JChristensen/Timezone
#include <WiFiUdp.h>
#include <NTPClient.h>

#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <SoftwareSerial.h>
```

Une fois le code transféré, la carte ARCELI redémarre.

```
Téléversement terminé
Writing at 0x00034000... (93 %)
Writing at 0x00038000... (100 %)
Wrote 339472 bytes (244692 compressed) at 0x00000000 in 21.8 seconds (effective 124.3 kbit/s)...
Hash of data verified.

Leaving...
Hard resetting via RTS pin...
```

ETAPE 8 – Interrogation de la carte ARCELI dans JEEDOM :

Lorsque le module ARCELI est connecté au réseau WIFI et qu'il accède à internet, il se met immédiatement à jour et affiche l'heure. La ligne suivante sur le LCD présente un indicateur qui évolue à chaque requête qui sont faites à la carte. Ensuite s'affiche TH et TB pour indiquer la température de la sonde Haute (WKS) et la température Basse (celle des batteries).

Exemple d'affichage :

21/06/2022 - 16:30
<- - ->
TH : 30.50°C/41.10%.
TB : 22.10°C/54.20%.

Les caractères mis en rouges dans l'exemple ci-dessus fournissent des informations complémentaires :

En face de la date et de l'heure 3 cas possibles :

Rien : Ventilos arrêtés normalement.

* : Température au-dessus de TempStopWKS ou TempStopBat dans le code.

+ : Régulation, les ventilos fonctionnent dans la plage horaire comprise entre HStart et HStop dans le code.

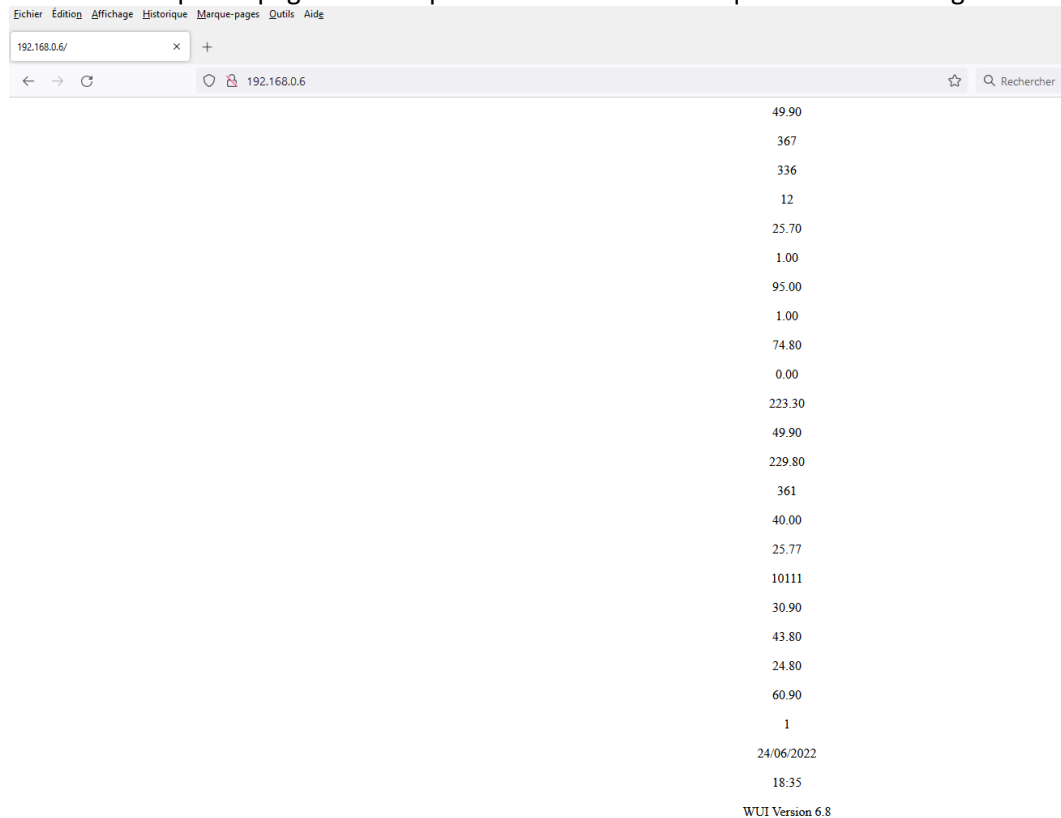
En face de chaque ligne TH et TB le dernier caractère en rouge peut-être :

. : (Un point) La température est inférieure TempStartWKS pour TH et TempStartBat pour TB dans le code.

* : La température TH est supérieure à TempStartWKS pour TH et TempStartBat pour TB dans le code.

<- - -> : Barre d'avance qui progresse à chaque fois qu'on interroge le module en http.

Une fois configuré et mis en service, le projet est accessible via un simple navigateur web en accédant en http à l'adresse IP configurée dans le code ci-dessus du module ARCELI. Dans notre exemple l'adresse IP de notre carte ARCELI est 192.168.0.6, il suffit de saisir dans la barre d'adresse de son navigateur web l'adresse http://192.168.0.6 puis [ENTREE] pour obtenir les informations du module. A chaque requête http traitée par le module ARCELI, le barre graphe sur le LCD progresse (< – – >). Voici un exemple de page obtenue après l'exécution d'une requête dans un navigateur web :

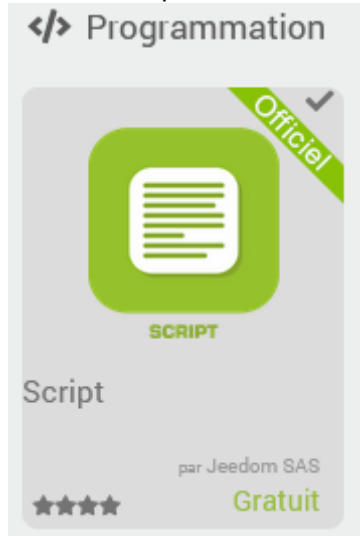


Voici à quoi ressemble le code source de la page :

```
<!DOCTYPE HTML>
<html>
<head>
</head>
<body style='text-align:center;'>
<div class='infos_output_frequence'>49.90</div><br />
<div class='infos_output_apparent_power'>368</div><br />
<div class='infos_output_active_power'>339</div><br />
<div class='infos_output_load_percent'>12</div><br />
<div class='infos_batterie'>25.70</div><br />
<div class='infos_batterie_charge_current'>1.00</div><br />
<div class='infos_batterie_capacity'>95.00</div><br />
<div class='infos_PV_current'>1.00</div><br />
<div class='infos_PV_voltage'>78.00</div><br />
<div class='infos_batterie_discharge_current'>0.00</div><br />
<div class='infos_Grid_Voltage'>227.60</div><br />
<div class='infos_Grid_frequence'>49.90</div><br />
<div class='infos_output_Voltage'>230.10</div><br />
<div class='infos_BUS_voltage'>362</div><br />
<div class='infos_Inverter_heat_sink_temperature'>40.00</div><br />
<div class='infos_Battery_voltage_from_SCC'>25.64</div><br />
<div class='infos_PIP_status'>10111</div><br />
<div class='infos_dht_H_temp'>30.90</div><br />
<div class='infos_dht_H_humidity'>44.50</div><br />
<div class='infos_dht_B_temp'>24.60</div><br />
<div class='infos_dht_B_humidity'>62.30</div><br />
<div class='infos_RelayFanStatus'>1</div><br />
<div class='infos_date_stamp'>24/06/2022</div><br />
<div class='infos_time_stamp'>18:43</div><br />
<div class='infos_système'>WUI Version 6.8</div><br />
</body>
</html>
```

Les informations en face de chaque **<div class** de la page web produite par le module ARCELI peuvent être traduites dans Jeedom à l'aide du plugin « Script ».

Commencer par aller dans le Market pour installer le plugin suivant :



Ajouter un nouveau script, nommer le « WKS » par exemple, puis configurer les commandes comme dans l'exemple ci-dessous :

Nom	Type script	Type	Requête	Paramètres
output_load_percent	HTML	Info Numérique	.infos_output_load_percent	http://192.168.0.6 Vérifier SSL 20 maximum

Nom de l'information dans Jeedom.

Choisir « HTML ».

Choisir « Info » et « Numérique » pour les chiffres, « Autre » pour la date.

Indiquer ici le nom de la classe div sur la page web précédé d'un point.

Indiquer l'adresse du module et 20 en « Timeout (s) ».

En complément, il est possible d'historiser les valeurs et de donner une unité de mesure :

Donner une unité de mesure et cocher « Historiser » si nécessaire.

Dans la version 7.2, j'ai ajouté la possibilité de redémarrer le module à l'aide d'un bouton « Reboot ». Pour cela il suffit d'ajouter une commande Action en indiquant l'adresse du module en http suivit de /Command?Reboot, ce qui donne : http://192.168.0.6/Command?Reboot.

Nom de la commande dans Jeedom.

Choisir « HTML ».

Choisir « Action »




Saisir « .Reboot ».

Indiquer l'adresse du module pour redémarrer : « http://192.168.0.6/Command?Reboot » et 40 en « Timeout (s) ».

Information Importante : Si le script est exécuté toutes les 1 minute, le « Timeout (s) » des commandes d'information du script doit être positionné à 20. De même, le « Timeout (s) » des commandes d'action du script doit être positionné à 40.

Actualisation du script toutes les 1 minute.

Finalement, avec une présentation de type « tableau » il est possible d'afficher sur son Dashboard Jeedom la présentation suivante (sur ma dernière version du logiciel 7.6) :

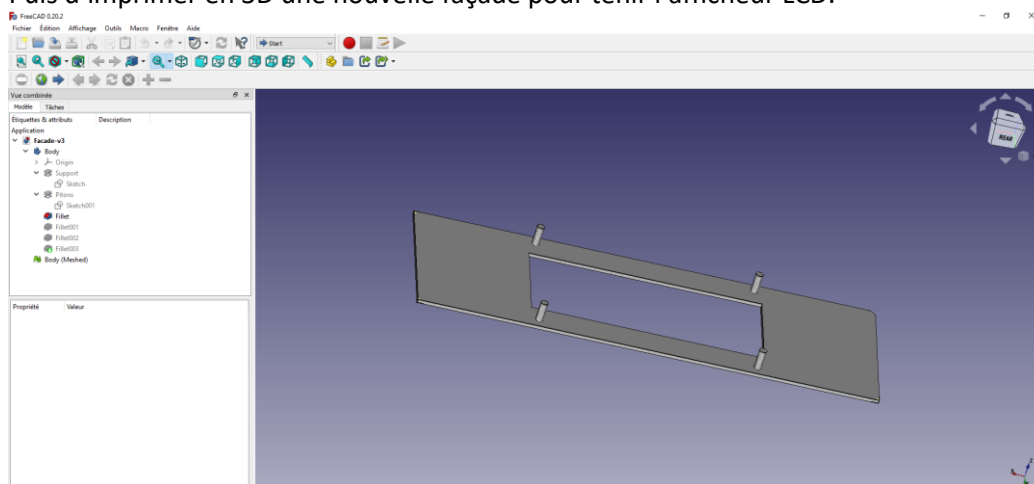
WKS			
Charge Totale :	4 %		
Puissance Apparente / Active :	138 W	107 W	
Arrivé Onduleur :	223.8 V	49.9 Hz	
Départ Onduleur :	230 V	49.9 Hz	
Batterie Capacité :	84 Ah		
Batterie Tension / Tension SCC :	24.9 V	24.93 V	
Batterie Courant De Charge :	0 A		
Batterie Courant De Décharge :	0 A		
PV Tension :	52.4 V		
PV Intensité / Puissance :	1 A	52.4 W	
Ventilation (32.5-29.5 / 25.5-22.5) :			
Onduleur Temp Externe / Interne :	27 °C	35 °C	
Onduleur Hum :	43.1 %		
Batterie Temp :	17.5 °C		
Batterie Hum :	72.6 %		
Mise à l'heure de l'onduleur :		DAT221128060000	
Horodatage :	28/11/2022	10:34	
WUI Version 7.6	10111	Reboot	

ETAPE 9 – Mise en boîte de la carte ARCELI :

A ce stade, il ne reste plus qu'à placer le projet dans une boîte...



Puis d'imprimer en 3D une nouvelle façade pour tenir l'afficheur LCD.

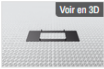


Pour imprimer la façade, il suffit de transmettre le projet « Facade-v3-Body (Meshed).stl » disponible sur mon GitHub sur le site web <https://www.easy3d.io/>.

easy3D

[L'atelier](#) [F.A.Q](#) [Mon compte](#) [Panier 1](#)

Votre panier

	Facade-v3-Body (Meshed).stl Taille : 194,00 × 65,00 × 10,00 mm Options supplémentaires	Matériau : PLA	Couleur : Noir (RAL9017)	Qté : 1	Prix :
Remplissage : 100%		Finition : 0.1mm			