

BASES de données

TD3 nodered

<https://www.youtube.com/playlist?list=PLYCQztVKrSmDaZG0bwnbd0PfgdXGe9Q2c>

<https://noderedguide.com/>

<https://nodered.org/docs/tutorials/>

<https://www.youtube.com/watch?v=kHtzghrNzAE>

1 Node-red : premiers pas

1.1 Introduction

Node-RED est un outil de programmation visuelle open-source, basé sur **Node.js**, qui permet de créer rapidement des applications en reliant des blocs fonctionnels appelés **nœuds**. Il est particulièrement utilisé dans les domaines de l'**IoT**, de l'**automatisation**, de la **supervision**, et de l'**intégration de services**.

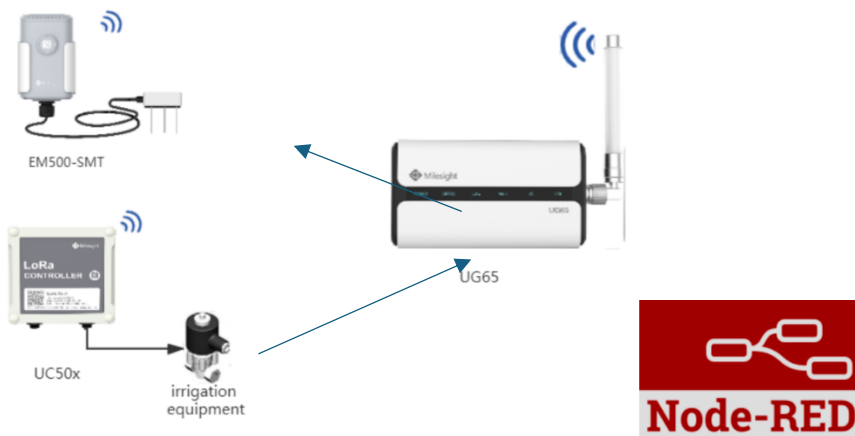
L'idée centrale de Node-RED est simple :

👉 *au lieu d'écrire beaucoup de code, on construit des flux logiques en reliant graphiquement des nœuds entre eux.*

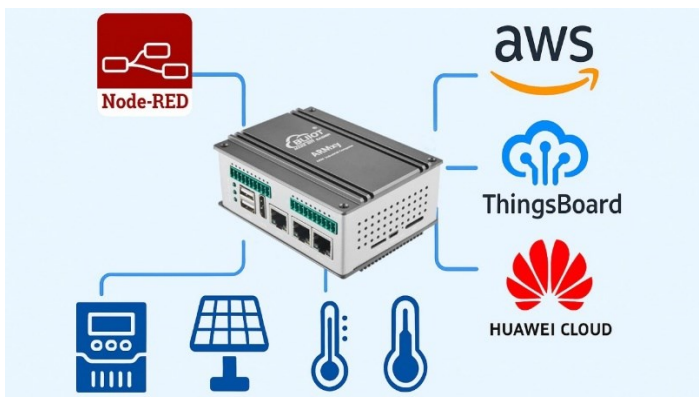
Node-RED est utilisé dans de nombreux contextes (industriels et tertiaires)

- ✓ automatisation industrielle et domotique
- ✓ supervision de systèmes (température, énergie, états machines)
- ✓ communication MQTT, HTTP, Modbus, OPC UA, BACNET
- ✓ prototypage rapide d'applications

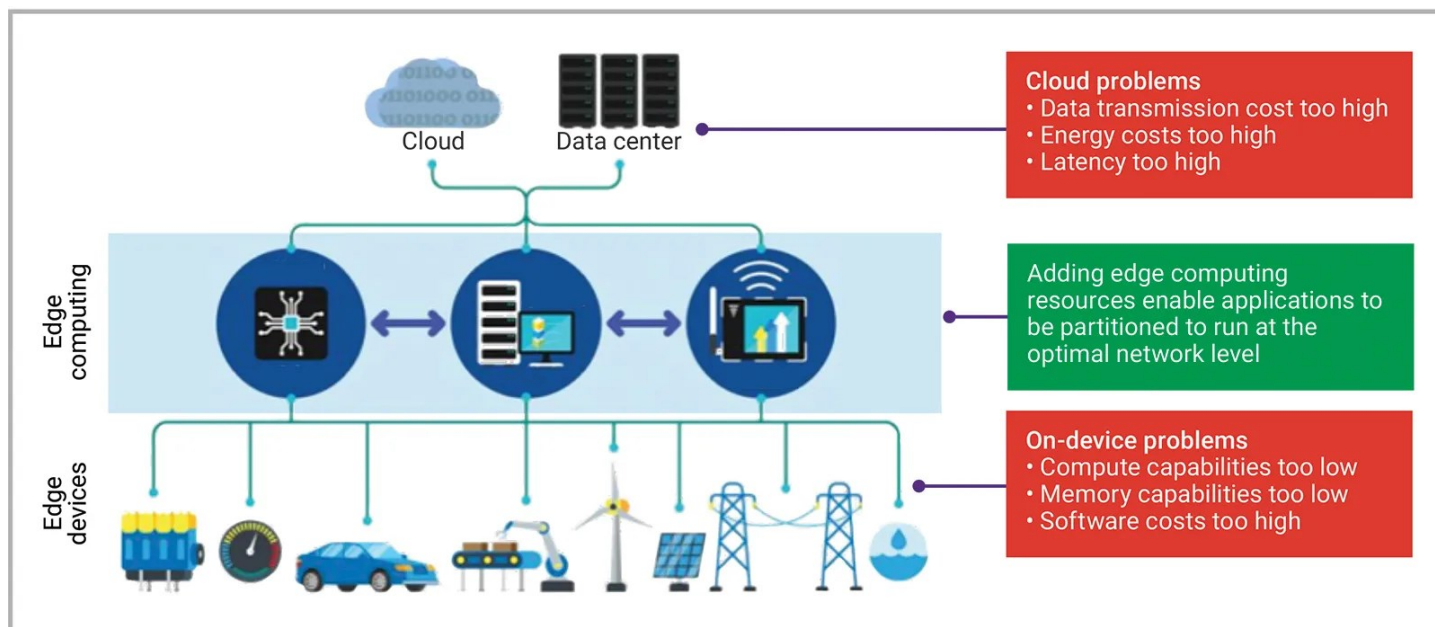
Cas d'usage n1 : traitement d'informations capteurs LoRawan au travers d'une passerelle avec paramétrage et programmation via nodered.



Cas d'usage n°2 : edge computer pour le traitement et/ou la remontée d'informations sur le cloud (PV, capteurs domotiques, industriel...) avec de nombreuses solutions (wago, sick, Siemens, ...)



Cas général industriel : intégrer des edge computer (en général des modules fanless) intégrant souvent nodered et qui permettent de garder l'information importante en local.



Cas numéro 4 : sans oublier les makers et le do it yourself permettant de domotiser à moindre cout. Solution que l'on peut retrouver aussi au sein des entreprises innovantes.



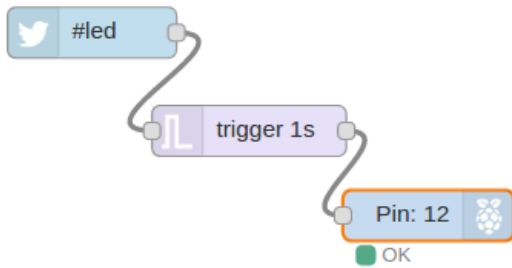
Quelques idées de projets node-red :

<https://monraspberry.com/10-idees-de-projets-a-realiser-avec-node-red-sur-votre-raspberry-pi/>

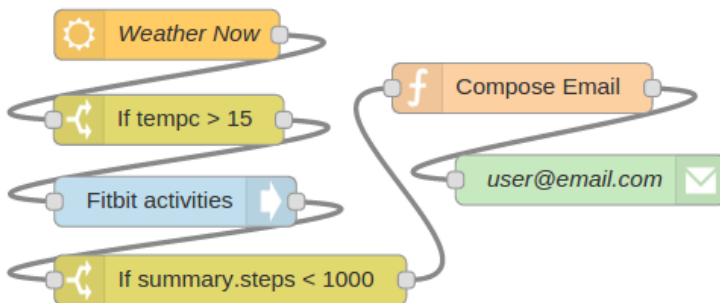
1.2 Des exemples node-red

Avant de commencer, voyons ce qu'est une programmation node-red au travers 2 exemples de programmation :

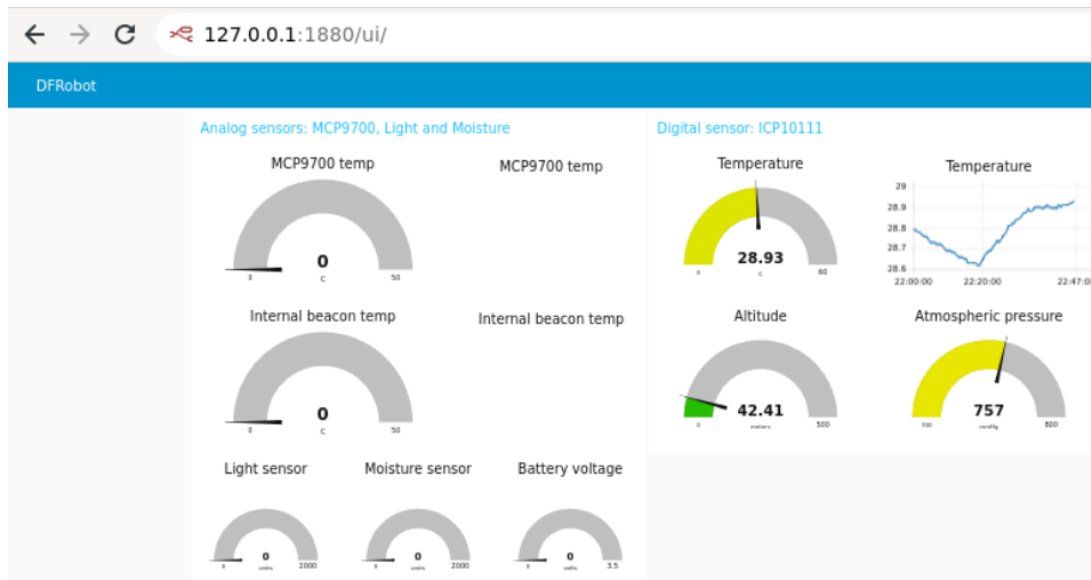
Exemple 1 : A chaque réception d'un tweet sur un sujet précis, allumer une led sur le raspberry pi pendant 1s



Exemple 2 : cette mini-application vient lire les informations de la météo (site web Openweather) sur Nice par exemple et si la température extérieure est supérieure à 15°C et que le nombre de pas sur la montre connectée est inférieur à 1000 m'envoie un mail pour me dire d'aller marcher (pourquoi faire simple quand on peut faire compliqué 😊)

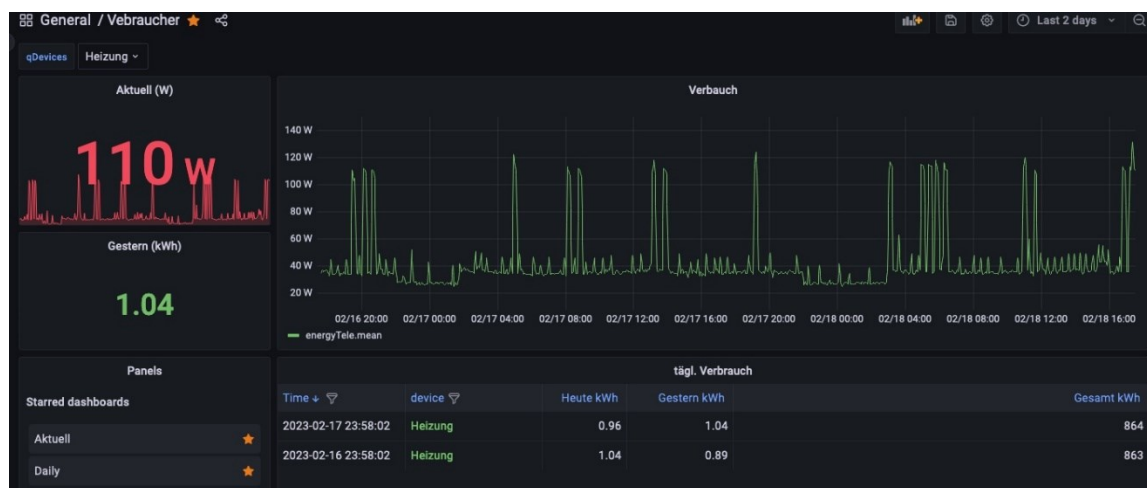
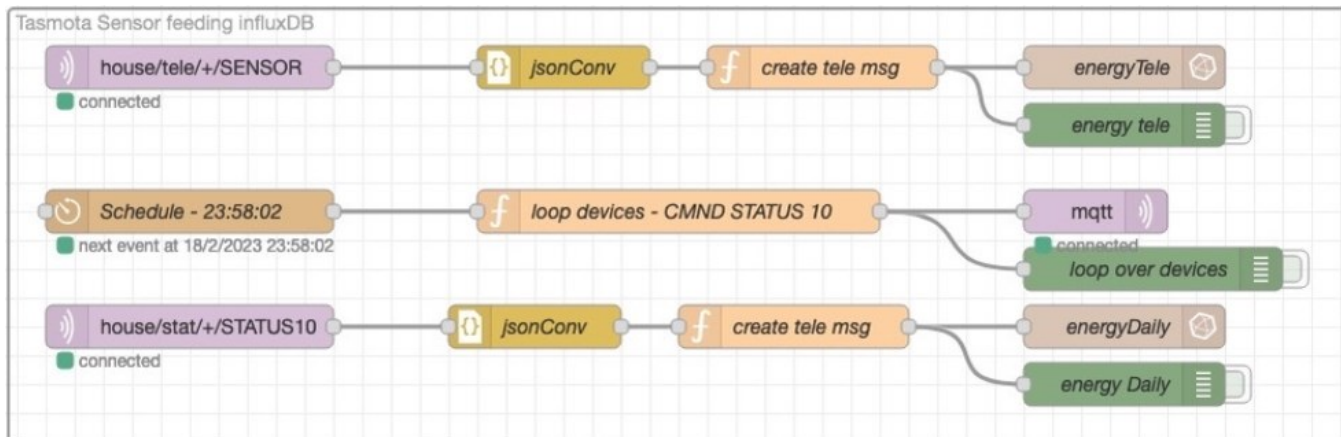


Nodered c'est de la programmation graphique (low code) mais c'est aussi la possibilité de générer une interface graphique, voici un exemple :



La visualisation des données peut aussi être réalisée par grafana avec un transfert de données vers influxdb (base de données temporelle) mais qui pourrait être aussi une base de données mariadb. Voici un exemple de code venant lire des données capteurs et les envoyer vers influxdb (grafana s'occupe alors d'afficher les données enregistrées dans influxdb).

<https://flows.nodered.org/flow/9e223d4f816dd15385687bb6bc05a2f6/in/b4yLAVuxd6if>



1.3 Lancement de node-red (217.154.16.99)

node-red peut s'installer sur n'importe quelle machine. Nous aurions pu utiliser windows : <https://nodered.org/docs/getting-started/windows> .

Ce n'est pas le choix que nous allons faire : nous avons déjà utilisé notre machine virtuelle ionos à l'adresse **217.154.16.99**, pour tester mariadb, il aurait été dommage de s'arrêter la, donc node-red a été installé sur cette machine et vous avez chacun la possibilité de travailler sur votre compte /home/votre-login. C'est ce que nous allons faire .

Pour lancer node-red, vous allez vous connecter sur 217.154.16.99 avec votre login et mot de passe (exemple pour ben_saga2025)

Travail à faire : ouvrir un terminal putty sur votre compte, lancer node-red puis se connecter via un navigateur

Pour lancer nodered, il suffit de taper la commande **node-red**

```
ben_saga2025@my-vps: ~  
ben_saga2025@my-vps:~$ ls  
Animaux.csv  departement.sql  dossier_test  insertion_animaux.sql  test  test1  
ben_saga2025@my-vps:~$ node-red  
23 Jan 18:16:09 - [info]  
  
Welcome to Node-RED  
=====  
  
23 Jan 18:16:09 - [info] Node-RED version: v4.1.2  
23 Jan 18:16:09 - [info] Node.js version: v18.20.4  
23 Jan 18:16:09 - [info] Linux 6.1.0-42-amd64 x64 LE  
23 Jan 18:16:09 - [info] Loading palette nodes  
23 Jan 18:16:10 - [info] Settings file : /home/ben_saga2025/.node-red/settings.  
23 Jan 18:16:10 - [warn] Encrypted credentials not found  
23 Jan 18:16:10 - [info] Server now running at http://127.0.0.1:2882/  
23 Jan 18:16:10 - [info] Starting flows  
23 Jan 18:16:10 - [info] Started flows
```

⚠ Non sécurisé 217.154.16.99:2882

Ouvrir le navigateur et aller sur le port de votre compte :

Login : votre login (ici barrat2025) et mot de passe **1234**



Information sur le mot de passe utilisateur 1234 et sa modification éventuelle :

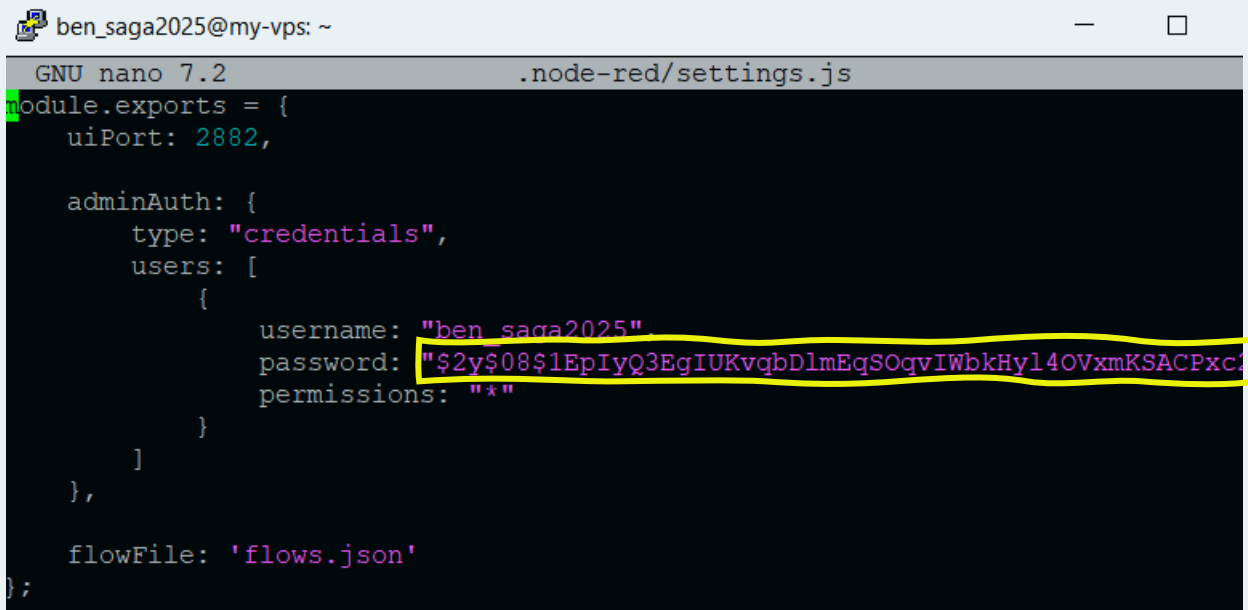
Si vous désirez changer le mot de passe (pour information), il faut lancer la commande `node-red admin hash-pw` (mot de passe géré par node-red) comme l'exemple ci-dessous (un mot de passe vous est demandé, ici XXXX mais le mot de passe ne s'affiche pas à l'écran) le résultat du hashage est ensuite affiché

```
ben_saga2025@my-vps:~$ node-red admin hash-pw
Password: XXXX
$2y$08$zvWj6NAo16oQ0vVDTLcokuJurPWgEHCGUhCxMOUhxZTlrWoEqi4Ey
```

Pour mettre à jour le mot de passe node-red, aller sur `./node-red` et ouvrir `settings.js` et copier le nouveau hash

```
ben_saga2025@my-vps:~$ cd ~/.node-red
ben_saga2025@my-vps:~/node-red$ nano settings.js
```

Vous pouvez alors copier (vous pouvez aussi utiliser winscp) le nouveau mot de passe haché).



```
ben_saga2025@my-vps: ~
GNU nano 7.2 .node-red/settings.js
module.exports = {
  uiPort: 2882,

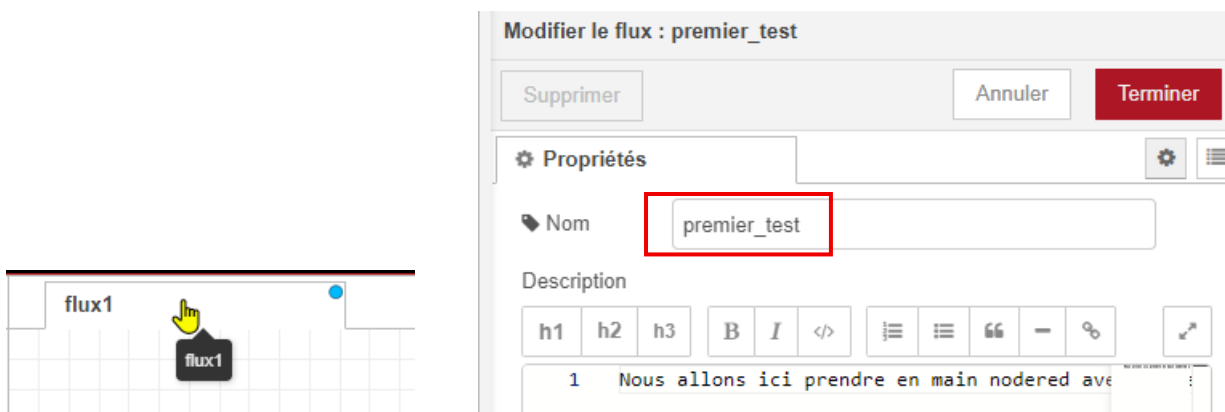
  adminAuth: {
    type: "credentials",
    users: [
      {
        username: "ben_saga2025",
        password: "$2y$08$1EpIyQ3EgIUKVqbDlmEqSOqvIWbkHyl14OVxmKSACPxc1",
        permissions: "*"
      }
    ]
  },

  flowFile: 'flows.json'
};
```

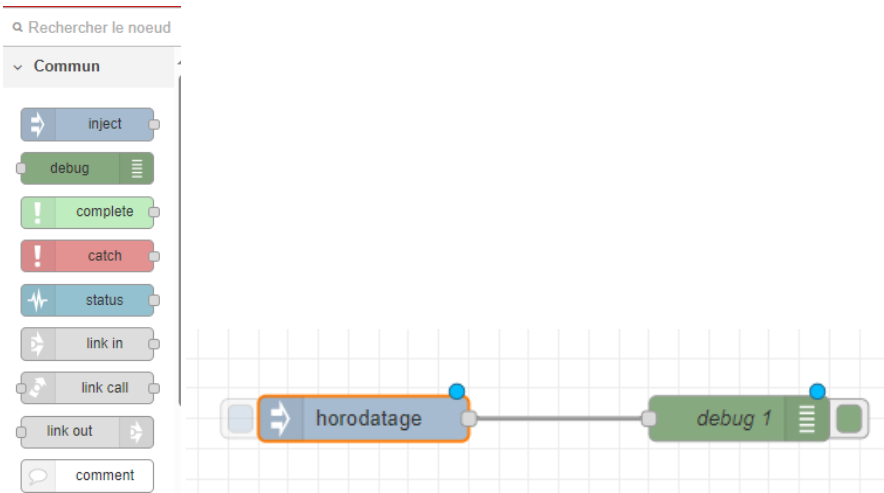
1.4 Premiers tests

Ce premier programme va nous permettre de nous familiariser avec l'environnement node-red.

1. Modifier le nom du flux (double clic sur flux1)

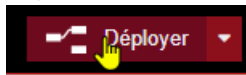


2. Ajouter inject et debug (glisser/déposer) qui sont les 2 premiers nœuds que nous allons utilisés et qui sont très souvent utilisés pour créer le premier programme Hello world ! (un classique)



3. Déployer (il faudra à chaque modification (point bleu) déployer le programme pour le tester (équivalent à

une compilation en C)



4. Cliquer sur Horodatage

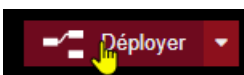
Le temps depuis 1970, un clic pour voir le temps en format lisible

5. Nous pouvons modifier le type de message envoyé par horodatage et choisir chaîne de caractère pour notre premier message Hello world

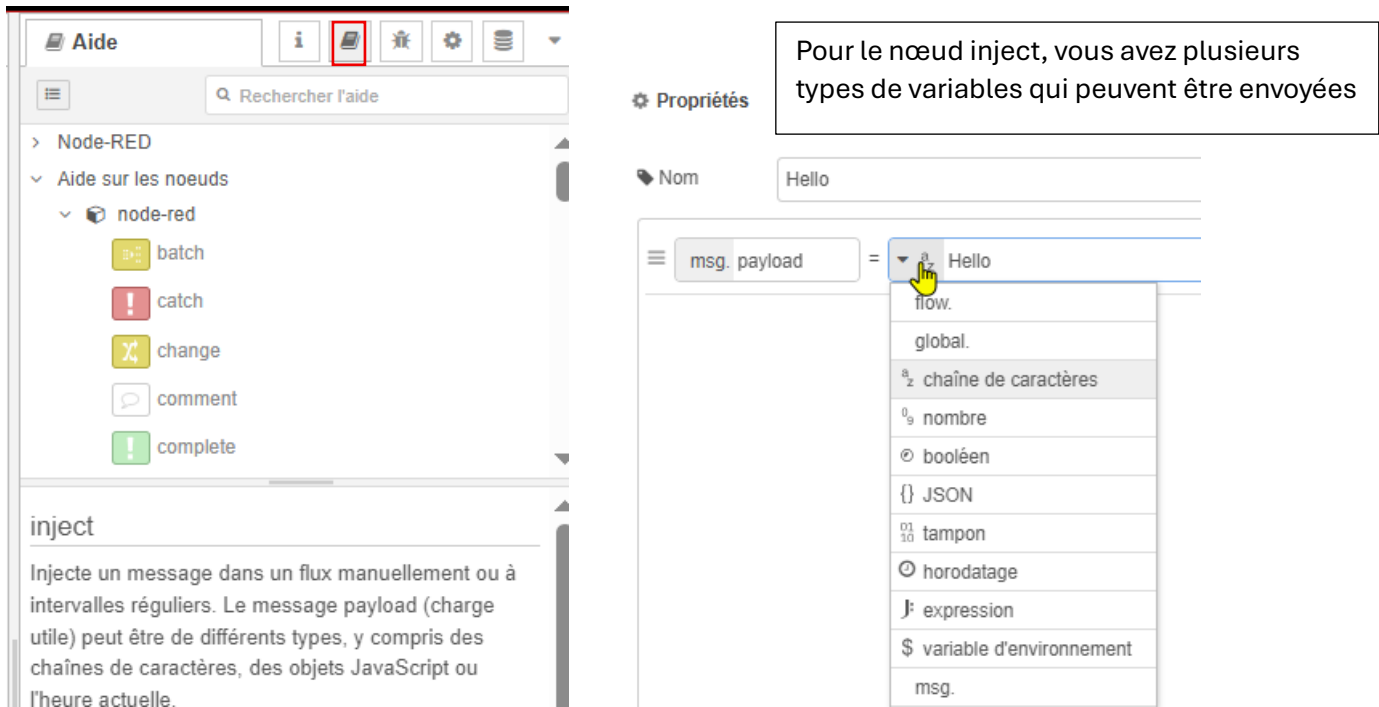
Le nom du noeud

Le message envoyé au noeud debug

6. Déployer

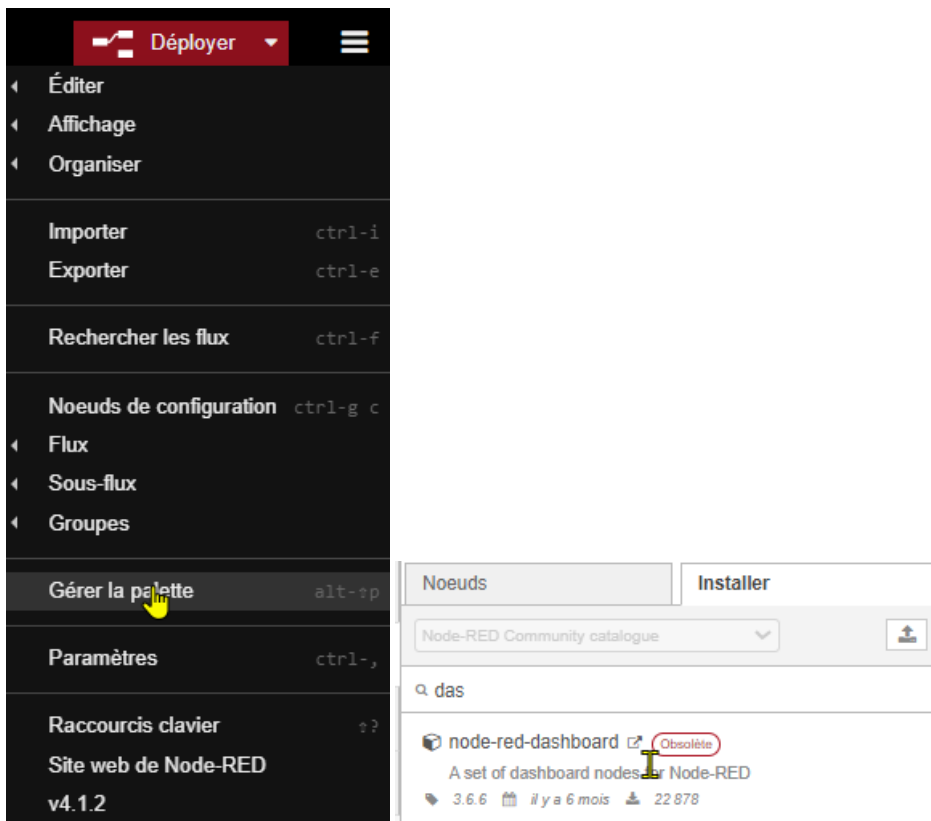


Pour avoir de l'information sur les différents nœuds, on pourra aller sur l'aide



The image shows two parts of the Node-RED interface. On the left is the 'Aide' (Help) panel, which is open to the 'inject' node documentation. The 'inject' section explains that it injects a message into a flow manually or at regular intervals, and that the message payload can be of various types like strings, JavaScript objects, or the current time. On the right is the configuration panel for the 'inject' node. The 'Nom' (Name) field is set to 'Hello'. Below it, the 'msg.payload' field is set to 'Hello', and a dropdown menu is open, showing various data types: 'flow.', 'global.', 'chaîne de caractères' (string), 'nombre' (number), 'booléen' (boolean), 'JSON', 'tampon' (buffer), 'horodatage' (timestamp), 'expression', 'variable d'environnement' (environment variable), and 'msg.'. A text box above the dropdown states: 'Pour le nœud inject, vous avez plusieurs types de variables qui peuvent être envoyées'.

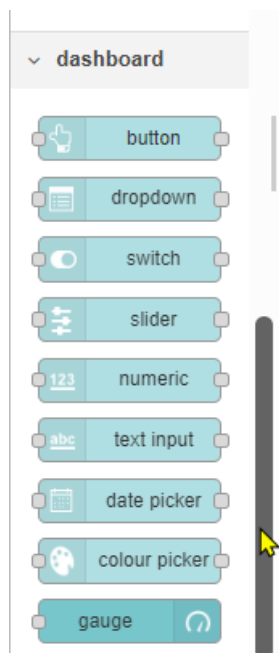
Node-red c'est aussi la possibilité d'ajouter des nœuds, commençons par le nœud dashboard pour pouvoir créer une interface graphique



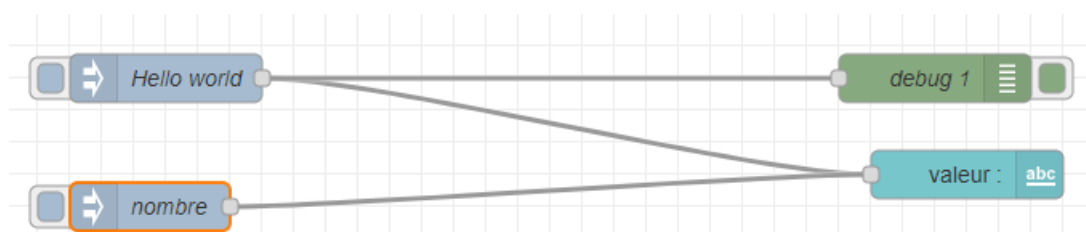
The image shows two parts of the Node-RED interface. On the left is the main menu, which is open to the 'Gérer la palette' (Manage Palette) option. The menu includes options like 'Déployer', 'Éditer', 'Affichage', 'Organiser', 'Importer', 'Exporter', 'Rechercher les flux', 'Noeuds de configuration', 'Flux', 'Sous-flux', 'Groupes', 'Paramètres', 'Raccourcis clavier', and 'Site web de Node-RED'. On the right is the 'Noeuds' (Nodes) panel, which is open to the 'Installer' (Install) tab. The 'Node-RED Community catalogue' is selected, and a search for 'das' has been performed. The search results show 'node-red-dashboard' as the top result, with a note that it is 'Obsolète' (obsolete). The description for 'node-red-dashboard' is 'A set of dashboard nodes for Node-RED', and it shows a version of 3.6.6, last updated 6 months ago, with 22,878 downloads.

Il existe une nouvelle interface flowfuse dashboard (Dashboard 2) qui a l'inconvénient de prendre plus de place mémoire (et comme nous avons des ressources limitées nous restons avec l'ancienne interface).

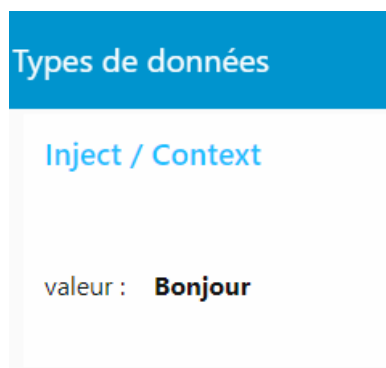
Une nouvelle palette a été installée :



Notre nouveau programme : le nœud inject noté nombre envoie la valeur 10, le nœud Hello World envoie le texte bonjour à tous. Tester le clic sur l'un ou l'autre sur la fenêtre debugs puis sur la page web 217.154.16.99 :1882/ui



Cliquer sur nombre et hello world



1.5 Fonctionnement de node-red

Nous avons affaire à une programmation événementielle. Un évènement déclenche l'exécution d'un flux.

Chaque nœud envoie la donnée msg.payload (mais plus généralement msg) de nœud en nœud. Cette variable est modifiée par chaque nœud.



La fonction peut modifier le message envoyé :

```
msg.payload += " world";  
return msg;
```



ou bien créer un nouveau message

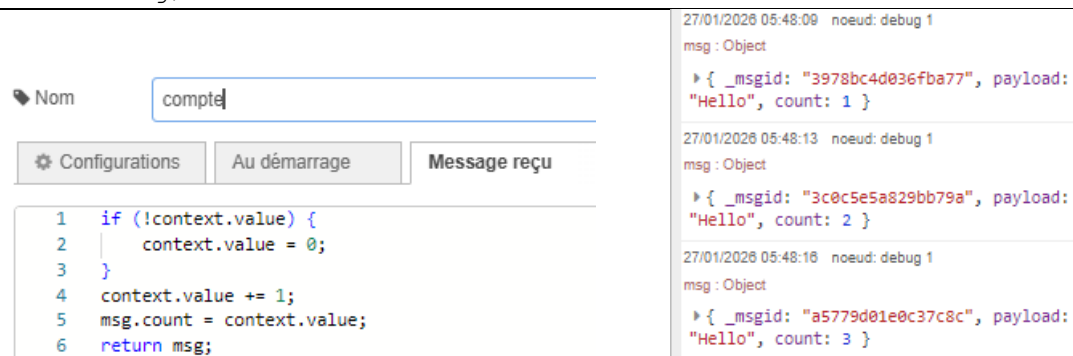
```
let newMsg = { payload: "Hello World" };  
return newMsg;
```

(<https://noderedguide.com/node-red-lecture-5-the-node-red-programming-model/>)

Dans le cas où l'on veut par exemple avoir un compteur, ce type de programmation ne peut pas fonctionner (un nœud se reboucle sur lui-même -> impossible). Il existe donc une solution pour contourner ce problème : les variables de contexte

Contexte : c'est l'équivalent d'une variable globale pour le flux. Ce qui permet de créer par exemple un compteur

```
if (!context.value) {  
  context.value = 0;  
}  
context.value += 1;  
msg.count = context.value;  
return msg;
```



Dans cet exemple le payload n'a pas été changé (Hello), on a par contre ajouté un champs count au message.

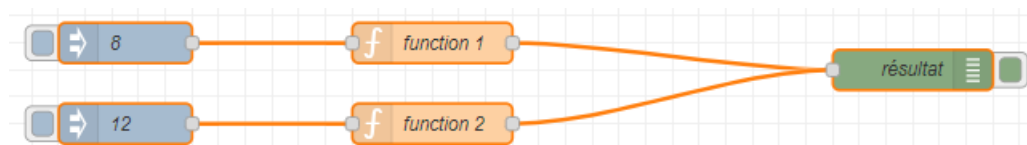
Contexte.global, c'est la même chose mais cette fois, c'est une variable globale partagées par tous les flux

Function 1 :

```
context.global.startTime = new Date().getTime();  
return msg;
```

Function 2 :

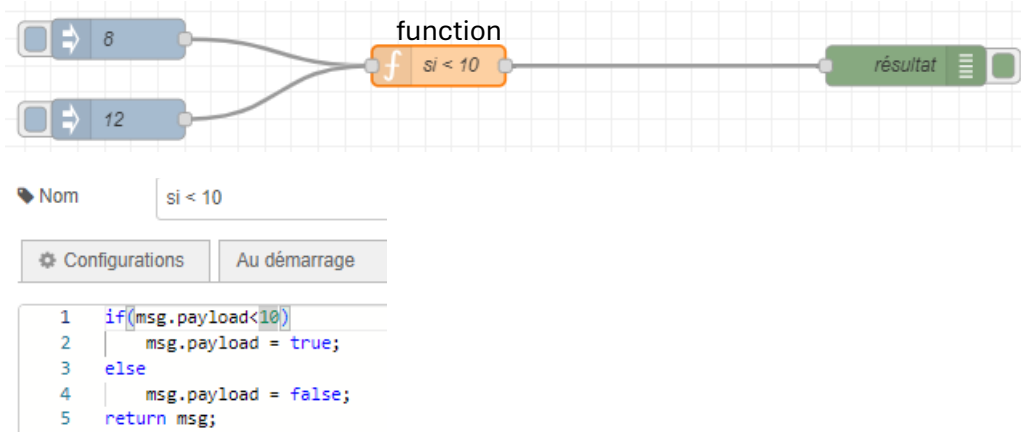
```
let currentTime = new Date().getTime();  
let timeElapsed = (currentTime - context.global.startTime) / 1000;  
msg.payload = "Time elapsed is: " + timeElapsed + " s";  
return msg;
```



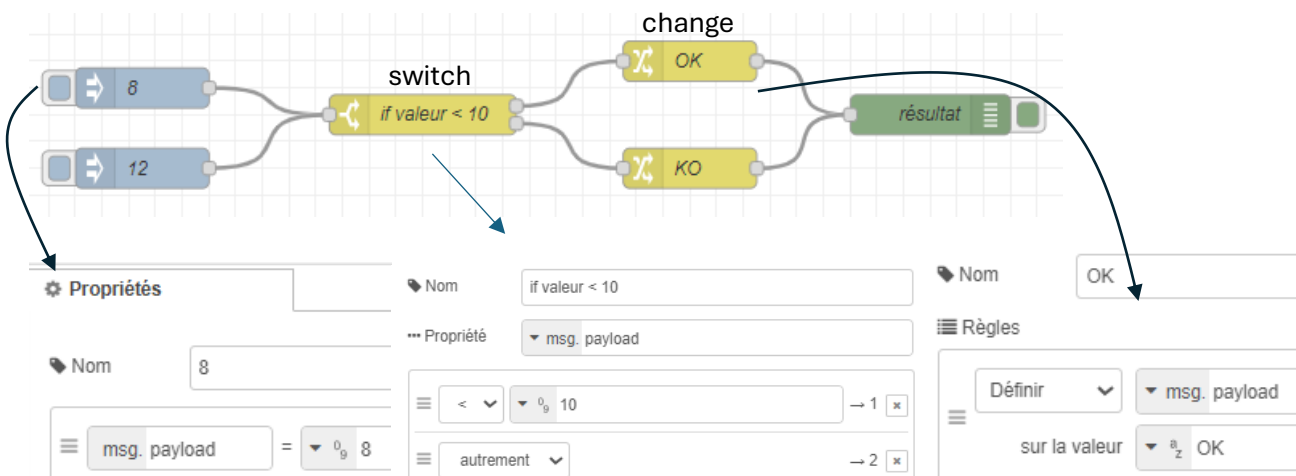
La mesure de temps se fait entre l'appui sur 8 et l'appui sur 12.

1.6 Exemple 1 : faire un if else (test de seuil)

solution 1 : avec fonction

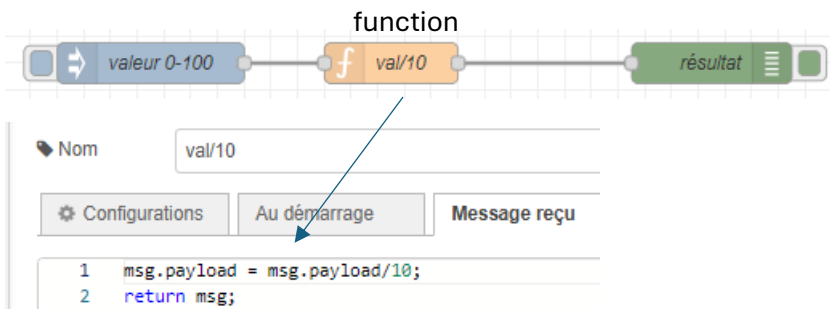


Solution 2 : programmation tout graphique

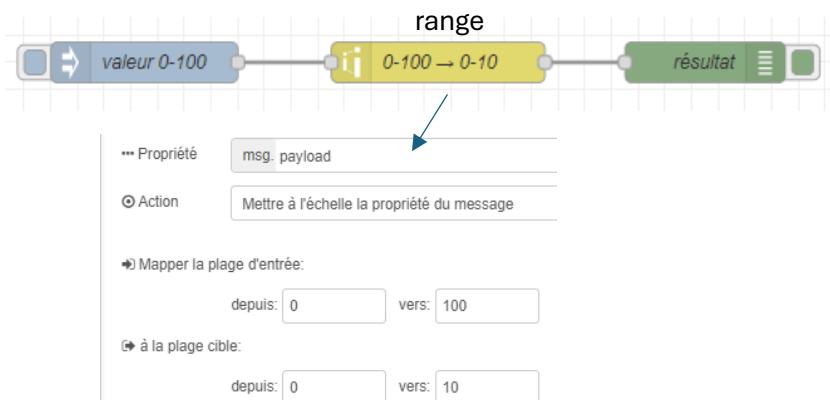


1.7 Faire une mise à l'échelle

solution 1 : avec fonction

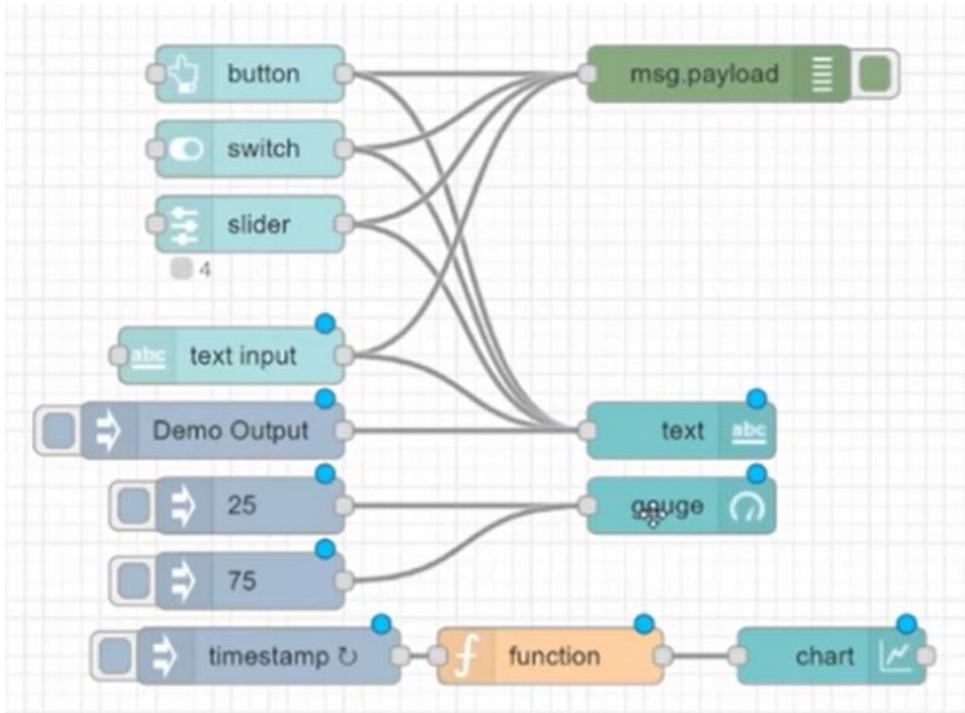


Solution 2 : tout graphique



1.8 Auto-formation dashboard

- Suivre la formation au dashboard sur <https://www.youtube.com/watch?v=C0hCQuj1BM4>



- Tester l'ajout d'un icone sur le button: <https://klarsys.github.io/angular-material-icons/>

Montrer le résultat à l'enseignant

Nous allons maintenant ajouter 2 images dans le répertoire public.

- **Créer le répertoire public/img dans votre répertoire .nodered puis copier les 2 images du répertoire /mnt/data vers votre répertoire .node-red/public/img**

/home/user_test/.node-red/public/img/	
Nom	Taille
light_off.png	25 KB
light_on.png	45 KB

Sur votre compte dans le répertoire .nodered/settings.js ajouter la ligne ci-dessous, ce qui va permettre à node-red de pouvoir afficher les fichiers dans ce répertoire.

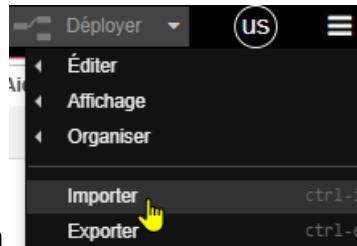
```
httpStatic: '/home/votre-login/.node-red/public',
```

/home/user_test/.node-red/settings.js - ionos user_test - Éditeur - WinSCP

```
module.exports = {
  uiPort: 1880,
  httpStatic: '/home/user_test/.node-red/public',
  adminAuth: {
    type: "credentials",
    users: [
      {
        username: "user_test",
        password: "$2y$08$1EpIyQ3EgIUKvqBDlmEqSOqvIwbk",
        permissions: "*"
      }
    ]
  },
  flowFile: 'flows.json'
};
```

- Relancer node-red (CTR+C puis node-red) dans la console putty.
- Tester que vous pouvez accéder aux 2 images sur le navigateur

http://217.154.16.99:1880/img/light_on.png



Vous allez maintenant importer le fichier json

```
[{"id":"44dd4a206010148c","type":"ui_template","z":"78b49e2e67736c6c","group":"group_lampe","name":"Lampe","order":7,"width":6,"height":6,"format":"<img ng-if=\"msg.payload === true\" src=\"/img/light_on.png\" width=\"120\">\n<img ng-if=\"msg.payload === false\" src=\"/img/light_off.png\" width=\"120\">\n","storeOutMessages":true,"fwdInMessages":true,"resendOnRefresh":true,"templateScope":"local","className":"","x":1590,"y":420,"wires":[[]]},{id":"dbfb2ca4404ealda","type":"ui_switch","z":"78b49e2e67736c6c","name":"","label":"switch","tooltip":"","group":"group_lampe","order":7,"width":2,"height":2,"passthru":true,"decouple":"false","topic":"topic","topicType":"msg","style":"","onvalue":"true","onvalueType":"bool","onicon":"","oncolor":"","offvalue":"false","offvalueType":"bool","officon":"","offcolor":"","animate":false,"className":"btn-center","x":1370,"y":420,"wires":[["44dd4a206010148c"]]},{id":"group_lampe","type":"ui_group","name":"Commande lampe","tab":"tab_demo","order":1,"disp":true,"width":6,"collapse":false},{id":"tab_demo","type":"ui_tab","name":"TP Lampe","icon":"lightbulb_outline","order":1},{id":"262470571c48d887","type":"global-config","env":[],"modules":{"node-red-dashboard":"3.6.6"}]}
```

- Tester le programme

Qu'est-ce que le format json?

JSON = JavaScript Object Notation, c'est un **format de données** utilisé pour **échanger des informations** entre ordinateurs ou applications.

- **Lisible par les humains**
- **Facile à traiter par les machines**

C'est très utilisé pour :

- ✓ API web (ex. récupérer des données météo)
- ✓ Configurations (ex. Node-RED flow)
- ✓ Stockage léger (ex. fichiers .json)

JSON est basé sur **deux structures principales** :

a) **Objet {}** → collection de paires clé/valeur

```
{
  "nom": "Alice",
  "age": 25,
  "ville": "Paris"
}
```

- nom, age, ville → **clés**
- "Alice", 25, "Paris" → **valeurs**

⚠ Les **clés doivent toujours être entre guillemets ""**

Les valeurs peuvent être :

- ✓ chaîne "texte"
- ✓ nombre 123
- ✓ booléen true / false
- ✓ null
- ✓ tableau []
- ✓ objet {}

b) **Tableau []** → liste de valeurs ou objets

```
[
  {"nom": "Alice", "age": 25},
  {"nom": "Bob", "age": 30}
]
```

- Une **liste de deux objets**
- Chaque objet a ses propres clés et valeurs

Pour mieux comprendre ce qu'est un fichier json, aller sur <https://jsonviewer.stack.hu/> et copier le code json du programme node-red que vous venez d'importer.

The left screenshot shows a JSON viewer with the following code:

```
[
  {
    "id": "ui_lampe",
    "type": "ui_template",
    "z": "78b49e2e67736c6c",
    "group": "group_lampe",
    "name": "Lampe",
    "order": 7,
    "width": 6,
    "height": 6,
    "format": "`

**Quel doit être le message à envoyer à l'initialisation pour charger l'image ?**

Injecter une fois après  secondes, puis

Ajouter et configurer un nœud inject pour répondre au problème

**Montrer le résultat à l'enseignant**

## Tester et analyser le programme ci-dessous :

```
[{"id":"457bd919e61792c0","type":"inject","z":"78b49e2e67736c6c","name":"Envoyer Alice","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"onceDelay":"","topic":"","payload":"Alice","payloadType":"str","x":850,"y":160,"wires":[["43f13c57d5ea55e2"]]}, {"id":"10e4c17bb5acfa92","type":"inject","z":"78b49e2e67736c6c","name":"Envoyer Bob","props":[{"p":"payload"}],"repeat":"","crontab":"","once":false,"onceDelay":"","topic":"","payload":"Bob","payloadType":"str","x":850,"y":220,"wires":[["43f13c57d5ea55e2"]]}, {"id":"43f13c57d5ea55e2","type":"switch","z":"78b49e2e67736c6c","name":"Déterminer couleur","property":"payload","propertyType":"msg","rules":[{"t":"eq","v":"Alice","vt":"str"}, {"t":"else"}],"checkall":"true","repair":false,"outputs":2,"x":1080,"y":190,"wires":[["854831517fe5d91e"],["86f7c25b3954023b"]]}, {"id":"854831517fe5d91e","type":"change","z":"78b49e2e67736c6c","name":"Texte vert","rules":[{"t":"set","p":"color","pt":"msg","to":"green","tot":"str"}, {"t":"set","p":"payload","pt":"msg","to":"{{msg.payload}} est OK","tot":"str"}],"x":1300,"y":160,"wires":[["c8927b8134a0d668"]]}, {"id":"86f7c25b3954023b","type":"change","z":"78b49e2e67736c6c","name":"Texte rouge","rules":[{"t":"set","p":"color","pt":"msg","to":"red","tot":"str"}, {"t":"set","p":"payload","pt":"msg","to":"{{msg.payload}} est KO","tot":"str"}],"x":1300,"y":220,"wires":[["c8927b8134a0d668"]]}, {"id":"c8927b8134a0d668","type":"ui_text","z":"78b49e2e67736c6c","group":"grp1","order":1,"width":6,"height":1,"name":"Affichage dynamique","label":"Statut","format":"{{msg.payload}}","layout":"","className":"","style":false,"font":"","fontSize":"","color":"#000000","x":1520,"y":190,"wires":[]}, {"id":"grp1","type":"ui_group","name":"Commande","tab":"tab1","order":1,"disp":true,"width":6}, {"id":"tab1","type":"ui_tab","name":"Démo ON/OFF","icon":"toggle_on"}, {"id":"0189921c619fb143","type":"global-config","env":[],"modules":{"node-red-dashboard":"3.6.6"}}
```

Pour terminer insérer une table (qui prend en paramètre une entrée inject json), mais avant installer dans palette

The image shows a two-step process in Node-RED:

- Installation:** A search for 'table' in the Node-RED palette leads to the 'node-red-node-ui-table' widget. The 'Installer' button is highlighted with a yellow hand cursor.
- Configuration:** The 'table' widget is connected to an 'Inject' node. In the 'Propriétés' (Properties) panel, the 'msg.payload' property is set to a JSON object. A dropdown menu is open, showing 'JSON' selected. To the right, the 'Modifier JSON' (Edit JSON) button is active, displaying the following JSON in the editor:

```
1 [
2 {
3 "age": 15,
4 "taille": 170
5 },
6 {
7 "age": 17,
8 "taille": 180
9 }
10]
```

Montrer le résultat à l'enseignant

## 1.9 Exercices :

1. En utilisant 2 inject et un debug, afficher 1 ou 0 si vous appuyez sur l'un ou l'autre inject
2. En utilisant seulement inject, afficher « ceci est un test » toutes les secondes (case Repeter)
3. En utilisant trigger faire afficher « bonjour » puis au bout de 2s « au revoir »
4. Injecter une température en °C , la transformer en fahrenheit et afficher sur debug
5. Ajouter le slider (valeur entre 0 et 100) et tester maintenant l'affichage debug puis afficher le résultat dans un text ui
6. Utiliser un filter pour n'afficher la température que pour les valeurs comprises entre 20 et 50
7. A chaque clic sur inject le programme incrémente un compteur qui est affiché dans le debug et dans l'ui.
8. Ajouter un bouton pour incrémenter le compteur
9. A chaque clic sur un bouton la couleur et le nom s'inverse
10. Si la valeur entrée est supérieure à 50, le debug affiche trop chaud, sinon il affiche trop froid
11. Créer un nœud qui génère une variable aléatoire comprise entre 0 et 100 et affiche cette valeur sur un camembert et sur un graphe (utiliser `Math.random()*100` et `.toFixed(2)` pour la partie affichage
12. Reprendre le programme de la lampe, mais changer le fond et le texte d'un ui\_text (allumé, éteint) et changer aussi la couleur (red, green).

**Créer un fichier doc dans lequel vous placerez le json et la visu de la réponse à chaque question (à mettre dans la boîte de dépôt).**

## 2 communication avec le monde (API Rest)

### 2.1 Introduction

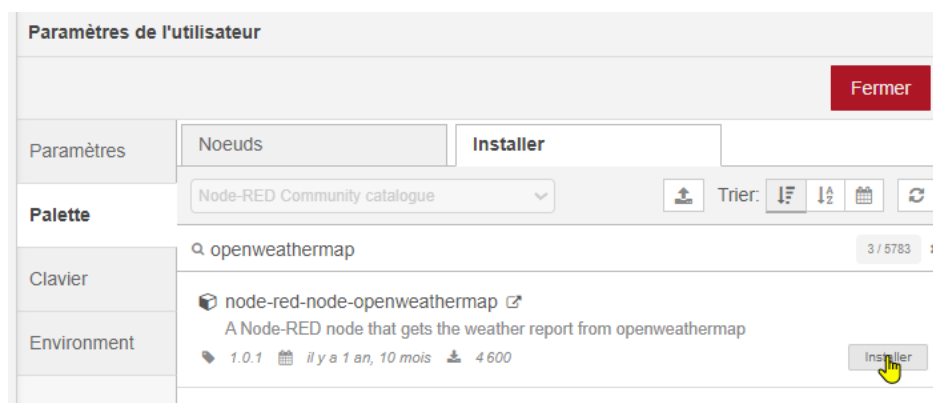
Qui dit bases de données dit données évidemment. Les bases de données ne sont pas seulement utilisées dans les sites web (moodle, google, twitter, whatsapp, ..) mais aussi dans la récolte de données capteurs (PV, énergie, températures, humidité,...) et de leur affichage et traitement.

Afin de bien comprendre l'intérêt de node-red nous allons prendre des exemples de lectures de données. Ces données pourraient venir de capteurs Lorawan, wifi, zigbee, bluetooth, Z-wave, KNX, bacnet, modbus, ... ou bien directement d'un site web qui agrège ces données.

### 2.2 Openweathermap

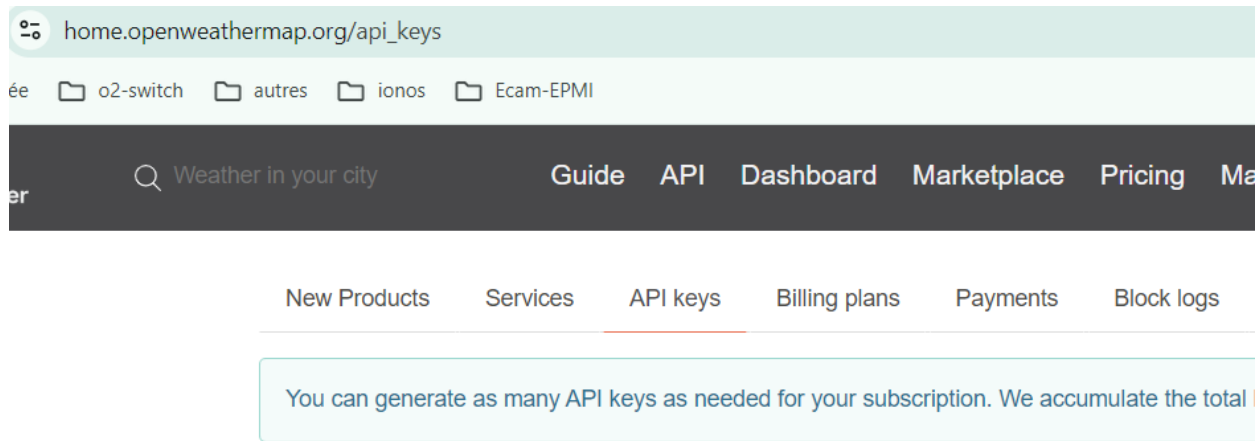
Commençons par un premier exemple classique : openweathermap qui est un site permettant d'obtenir des informations météo dans le monde <https://openweathermap.org/>, ce site propose une API disponible permettant d'interroger la base de données au travers de requêtes web prédéfinies. Nodered a l'avantage d'être open source et donc il est possible de partager les nœuds déjà programmés, ici le node openweathermap.



➤ Installer openweathermap



Avant d'aller plus loin, vous aurez besoin de créer un compte afin d'obtenir un token permettant d'interroger la base de données.

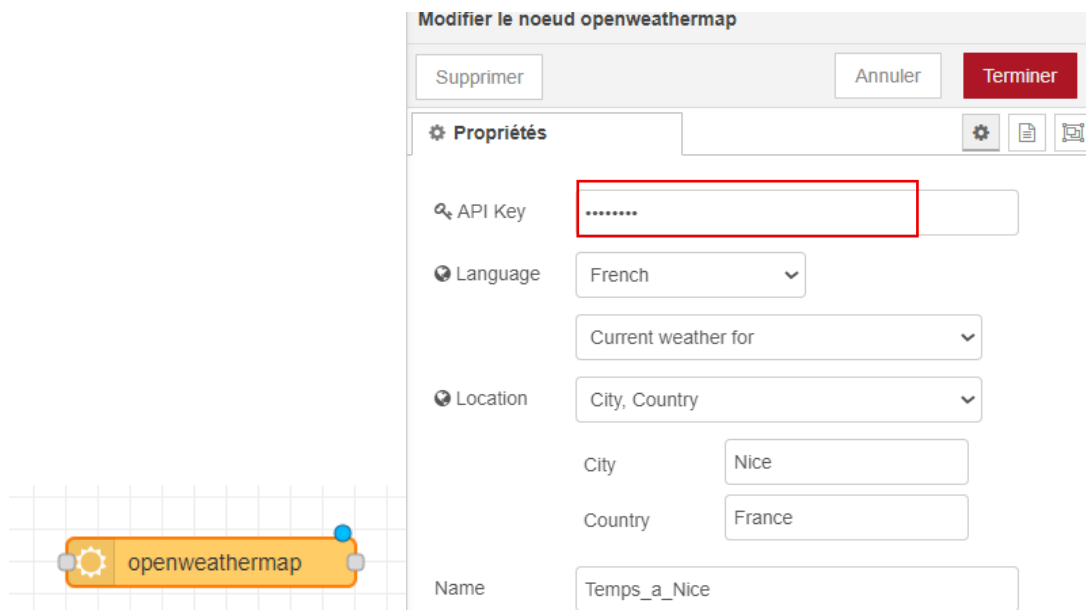
C'est ce que j'ai déjà fait, vous pouvez utiliser cette clé (à tester)...



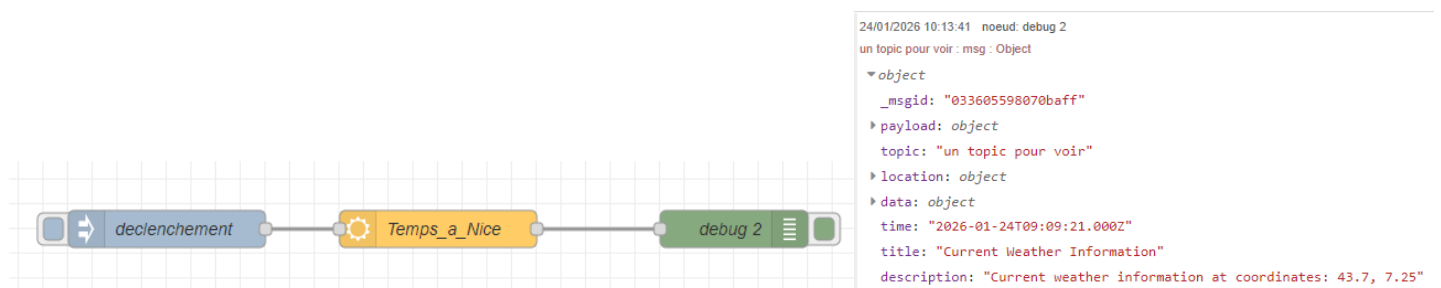
| Key                              | Name         | Status | Actions                                                                                                                                                                 |
|----------------------------------|--------------|--------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 376d506a1ae12d3427d46982ef5dfb93 | test_nodered | Active |   |

Voici ma clé : 376d506a1ae12d3427d46982ef5dfb93

➤ Configurer le nœud openweathermap



Il ne reste plus qu'à tester la connexion.



Pour bien comprendre ce qu'est une API copier/coller la ligne ci-dessous dans un navigateur :

```
https://api.openweathermap.org/data/2.5/weather?q=Paris&appid=376d506a1ae12d3427d46982ef5dfb93&units=metric&lang=fr
```

**Quel est le résultat généré. Expliquer**

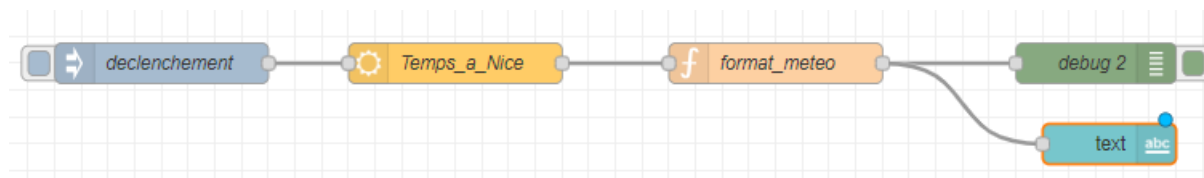
- Tester cette deuxième commande en remplaçant {lat} et {lon} par les coordonnées de Grasse et ajouter la clé à la place de {API key}

```
https://api.openweathermap.org/data/3.0/onecall?lat={lat}&lon={lon}&appid={API key}
```

En utilisant fonction, il est possible de venir extraire dans le fichier json les informations pertinentes.

```
msg.payload = {
 tempc: +(msg.data.main.temp - 273.15).toFixed(1),
};
return msg ;
```

- Ajouter les informations humidité et ville et les afficher dans une boîte de texte



**Montrer à l'enseignant**

## 2.3 Extraction de données openaq.org

Prenons un deuxième exemple qui cette fois récolte les informations de pollution aux particules et ozone. Comme tous les sites publics, il faut créer un compte pour obtenir une clé de cryptage.

<https://explore.openaq.org/account>

### API Key

1167c63aaaba2957263da32cb1064394f3ab00400117da052e19e49c3e49d457

Copy

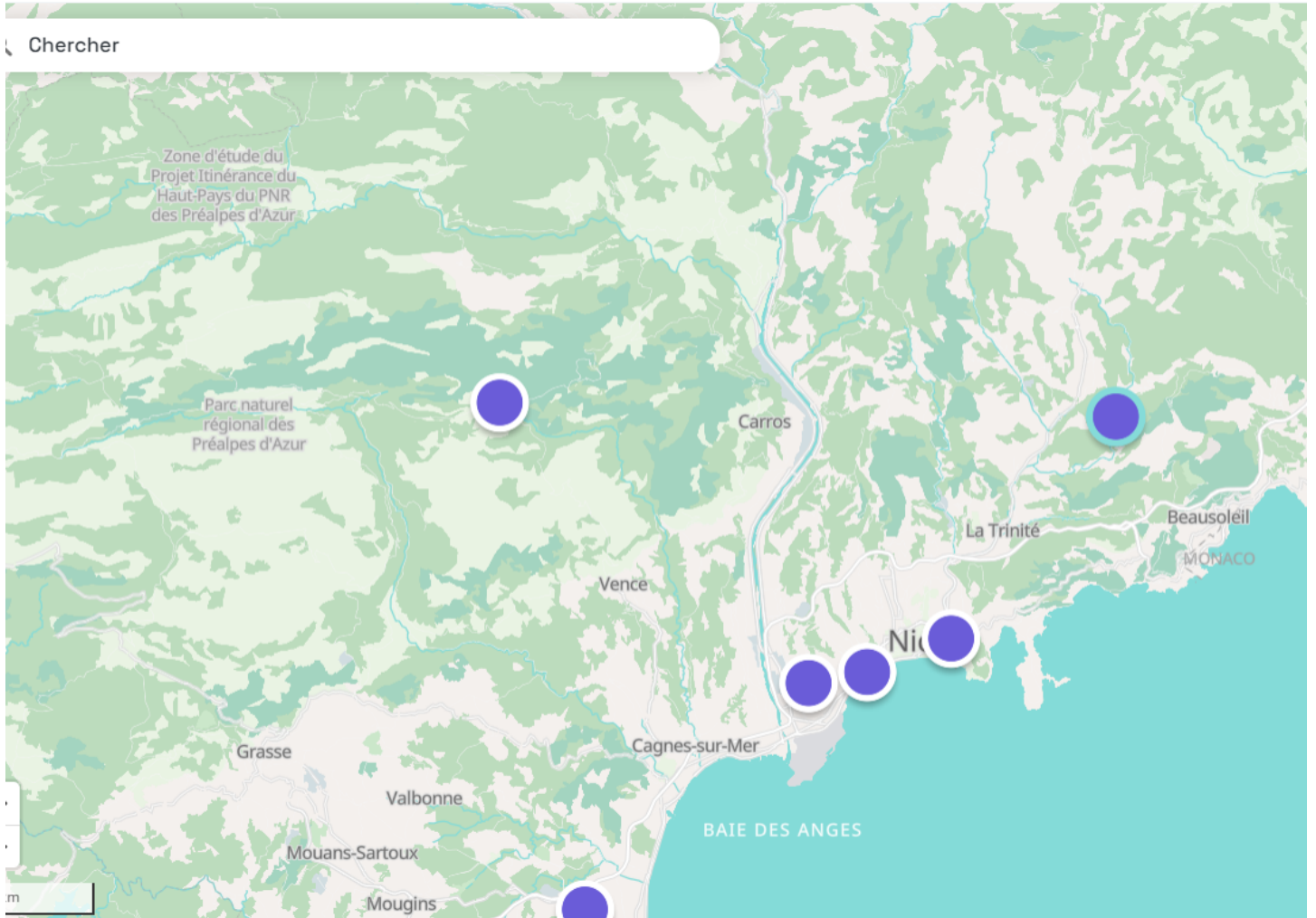
Regenerate

Voici la clé utilisée, vous pouvez créer votre compte pour obtenir votre propre clé .

1167c63aaaba2957263da32cb1064394f3ab00400117da052e19e49c3e49d457

Comme pour le site web openweathermap, il est possible de visualiser les données sur le site ou bien de venir au travers de requêtes et d'une clé récupérer des informations sous forme de format json permettant le traitement de ces données (insertion dans une base de données, affichage, calculs, ...).

<https://explore.openaq.org/>



En cliquant sur une station on peut voir son numéro apparaître dans l'URL, on pourra alors faire une demande de données sur cette station.

- Quel est le numéro de station pour le site du PEILLON et de COURSEGOULE ?

- Tester toutes les stations et notez les stations qui sont en fonctionnement

Exemple de requête pour la station 2162694 (par exemple).

<https://api.openaq.org/v3/locations/2162694/latest>

- Quel est le résultat affiché, Expliquer

L'envoi de la clé pour cette requête ne se fait pas directement dans l'URL, mais au travers du header qui fait partie de la requête envoyée au serveur mais qui n'est pas visible dans l'URL.

Dans sous linux la commande curl qui permet de générer une requête https mais en ajoutant le header (-H) qui intègre la clé de requête.

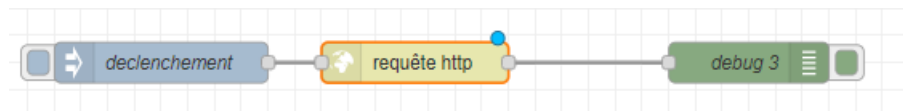
- Tester le résultat renvoyé par :

```
curl -i \
-H "X-API-Key: 92e53d49fbcd93bfce18a64ae99ca1fcab48f1107172ad299ba2fd4c227cf0d8" \
"https://api.openaq.org/v3/locations/2162694/latest"
```

- Tester le résultat renvoyé par cette deuxième requête (on demande l'affichage du type de capteur, chaque capteur a un id unique)

```
curl -i \
-H "X-API-Key: 92e53d49fbcd93bfce18a64ae99ca1fcab48f1107172ad299ba2fd4c227cf0d8" \
"https://api.openaq.org/v3/sensors/1290017"
```

Nous allons faire la même requête vers openaq.org dans nodered, en prenant soin d'ajouter la clé en paramètre d'en-tête.



Modifier le noeud http request

Supprimer Annuler Terminer

Propriétés

Méthode GET

URL https://api.openaq.org/v3/locations/2162694/latest

Charge utile Ignorer

Activer la connexion sécurisée (SSL/TLS)

Utiliser l'authentification

Activer le maintien de la connexion

Utiliser un proxy

N'envoyer que des réponses non-2xx au noeud Catch

Désactiver l'analyse HTTP stricte

Retourne une chaîne UTF-8

En-têtes

X-API-Key 1167c63aaaba2957263da32

- Tester le programme.

**Travail à faire : vous devez utiliser un nœud dropdown dans lequel vous placerez les stations disponibles. En utilisant {{msg.payload}} vous enverrez le choix à httprequest pour aller chercher les informations de la station. Les informations pertinentes seront affichés dans un ui\_text mais aussi dans un graphe et un donut.**

En cas de dépassement d'un seuil d'alerte vous devrez vous envoyer un mail

The screenshot shows the 'Paramètres de l'utilisateur' (User Settings) interface. At the top right is a red 'Fermer' (Close) button. Below are tabs for 'Paramètres', 'Noeuds', and 'Installer'. A search bar contains 'email' and shows 48 / 5784 results. The search results list the 'node-red-node-email' node with a description: 'Node-RED nodes to send and receive simple emails.' It also shows version '5.1.0', 'il y a 1 semaine' (1 week ago), and '14 195' downloads. An 'Installer' button is visible next to the node name.

The screenshot shows the 'Edit email node' configuration form. At the top are 'Delete', 'Cancel', and 'Done' buttons. Below is a 'Properties' section with various fields: 'To' (containing '@gmail.com'), 'Server' (containing 'smtp.gmail.com'), 'Port' (containing '465') with a checked 'Use secure connection.' checkbox, 'Auth type' (set to 'Basic'), 'Userid' (containing '@gmail.com'), 'Password' (masked with dots), 'TLS option' (checked 'Check server certificate is valid'), and 'Name' (containing 'Name').

Ainsi qu'un tweet à condition d'avoir un token sur [developer.twitter.com/en/apps](https://developer.twitter.com/en/apps)

The screenshot shows the 'Keys and tokens' page in the Twitter Developer Portal. The left sidebar contains 'Dashboard', 'Projects & Apps', 'Products' (with a 'NEW' badge), and 'Account'. The main content area has 'Settings' and 'Keys and tokens' tabs. Under 'Consumer Keys', there is a field for 'API Key and Secret' with a 'Reveal API Key hint' link and a 'Regenerate' button. Under 'Authentication Tokens', there is a 'Bearer Token' field with a 'Generate' button. At the bottom, there is an 'Access Token and Secret' field with a 'Generated July 3, 2023' timestamp, a 'Revoke' button, and a 'Regenerate' button.

Vous pourrez alors configurer le tweet

Exemple avec openwheatermap

**Travail à faire : tester l'envoi de mail et de tweet**

## 3 communication avec le monde (mqtt)

### 3.1 introduction

MQTT = Message Queuing Telemetry Transport, c'est un protocole léger de communication conçu pour l'IoT, les capteurs et les objets connectés. Il est simple, rapide et peu gourmand en bande passante. C'est une autre solution simple et rapide pour faire communiquer des objets. Il fonctionne comme un **service de messagerie entre machines**. Les appareils ne parlent pas directement entre eux, mais via un **serveur central appelé Broker**.

#### Les rôles dans MQTT

##### 1. **Broker (serveur) , dans notre cas mosquitto (installé sur la machine 217.154.16.99)**

- ✓ Centre de communication
- ✓ Reçoit les messages et les distribue aux bons clients
- ✓ Exemples : Mosquitto, HiveMQ, EMQX

##### 2. **Client (nodered,mqtt,...)**

- ✓ Tout appareil qui envoie ou reçoit des messages
- ✓ Peut être un capteur, un serveur, un ordinateur, ou Node-RED
- ✓ Chaque client se **connecte au broker**

#### Concept clé : Topic

Les messages sont envoyés sur des **topics** (sujets)

- ✓ Exemple de topic : **maison/salon/temperature**
- ✓ **Topic hiérarchique** (avec des /) pour organiser les données
- ✓ Les clients s'abonnent aux topics qu'ils veulent lire et publient sur ceux qu'ils veulent envoyer

#### Pub/Sub : Publish / Subscribe

MQTT utilise le modèle **Publish/Subscribe** au lieu de client-serveur classique

- ✓ **Publisher** : publie un message sur un topic
- ✓ **Subscriber** : s'abonne à un topic pour recevoir les messages
- ✓ **Broker** : distribue le message à tous les abonnés

### Exemple concret :

- ✓ Capteur température → publie sur maison/salon/temperature
- ✓ Dashboard Node-RED → s'abonne à maison/salon/temperature → affiche la température en temps réel

### QoS (Quality of Service)

MQTT peut garantir la livraison des messages selon 3 niveaux :

#### QoS Signification

|   |                                                            |
|---|------------------------------------------------------------|
| 0 | Au maximum une fois (pas de confirmation)                  |
| 1 | Au moins une fois (confirmation mais duplication possible) |
| 2 | Exactement une fois (plus sûr mais plus lourd)             |

## 3.2 Premiers tests

Le serveur mosquitto est normalement déjà lancé sur la machine **217.154.16.99**, il est présent sur le port **3883**

- Ouvrir putty et vérifier que le service fonctionne

```
ps -ef | grep mosquitto (coté user)
```

- Expliquer le résultat

Information ,Coté root : la commande pour avoir les informations du broket `ss -tulpn | grep mosquitto`

Vous allez tester mosquitto en local dans 2 fenêtres putty différentes et envoyer un message d'une fenêtre à l'autre, pour des raisons de sécurité vous devez ajouter un user (user\_test) et un mot de passe (ici 1234) qui ont été configuré dans le broker

```
mosquitto_sub -h localhost -p 3883 -t "test" -u "user_test" -P "1234" -v
```

Et pour la deuxième fenêtre

```
mosquitto_pub -h localhost -p 3883 -t "test" -u "user_test" -P "1234" -m "Hello MQTT"
```

- Créer un nouveau message avec le topic /maison/temperature et envoyer la température sur la deuxième fenêtre. Placer la copie d'écran ci-dessous pour chaque fenêtre

Fenêtre 1

Fenêtre 2

Un programme a été écrit en python et il envoie des informations issues du CPU et mémoire ainsi qu'une rampe et une température simulée et ceci toutes les 5s.

```
MQTT_TOPIC_CPU = "vps/cpu"
```

```
MQTT_TOPIC_MEM = "vps/mem"
```

```
MQTT_TOPIC_RAMP = "vps/ramp"
```

```
MQTT_TOPIC_TEMP = "vps/temp"
```

➤ Tester l'abonnement à un topic sur putty

```
mosquitto_sub -h localhost -p 3883 -t "vps/ramp" -u "user_test" -P "1234" -v
```

On peut visualiser tous les topics avec #, tester la commande pour voir les 4 topics apparaitre

### 3.3 Tests mqtt avec un client sur le cloud

Aller sur <https://testclient-cloud.mqtt.cool/> et tester la connexion vers notre serveur mqtt 217.154.16.99, n'oubliez pas d'ajouter le username et le password permettant d'accéder au broker.

Connection Settings for 217.154.16.99

| Protocol | Host          | Port |
|----------|---------------|------|
| tcp      | 217.154.16.99 | 3883 |

| Username  | Password |
|-----------|----------|
| user_test | 1234     |

Shared Connection  Dedicated Connection

testclient-cloud.mqtt.cool/?

o2-switch autres ionos Ecam-EPMI

## Test Client

MQTT.Cool

Connection: tcp://217.154.16.99:3883

**Subscriptions**

The topic filter:  QoS 0

Subscribed topics

vps/#

**Publish**

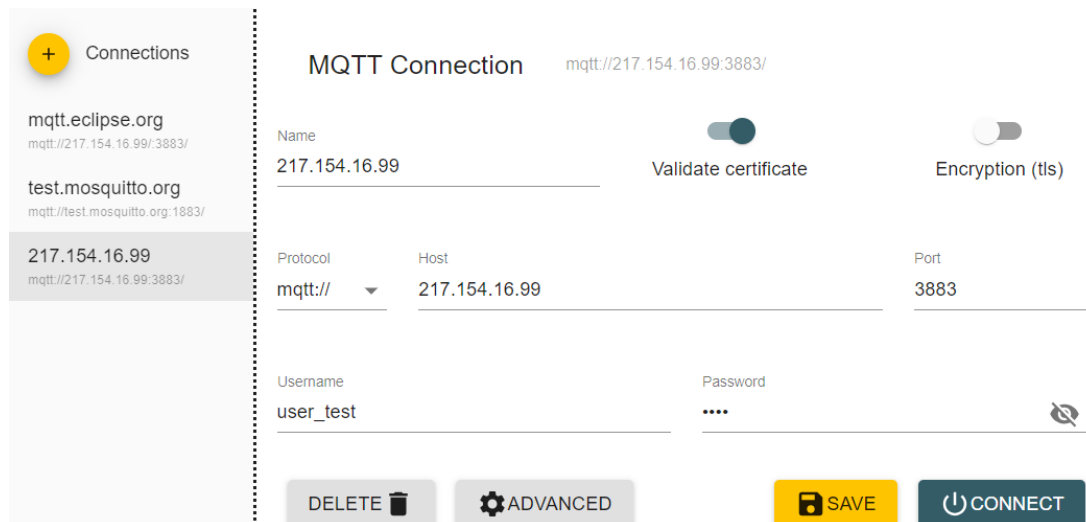
The destination of the message:  QoS 0  Retain

The message text to be sent:

**Messages**

- 2026-0-24 18:30:19.937    
26.6
- 2026-0-24 18:30:19.937    
4
- 2026-0-24 18:30:19.936    
94.4
- 2026-0-24 18:30:19.926    
0.0
- 2026-0-24 18:30:13.935    
2026-0-24 18:30:13.935

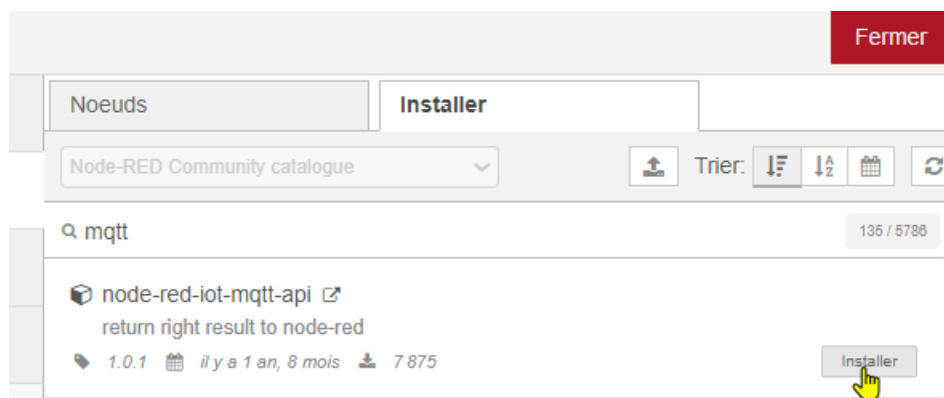
On aurait pu aussi utiliser MQTT explorer (client windows)



The screenshot shows the MQTT Connection configuration window. On the left, there is a 'Connections' sidebar with a list of connections: 'mqtt.eclipse.org', 'test.mosquitto.org', and '217.154.16.99'. The main area is titled 'MQTT Connection' and shows the following fields: Name (217.154.16.99), Protocol (mqtt://), Host (217.154.16.99), Port (3883), Username (user\_test), and Password (masked with dots). There are also toggle switches for 'Validate certificate' and 'Encryption (tls)'. At the bottom, there are buttons for 'DELETE', 'ADVANCED', 'SAVE', and 'CONNECT'.

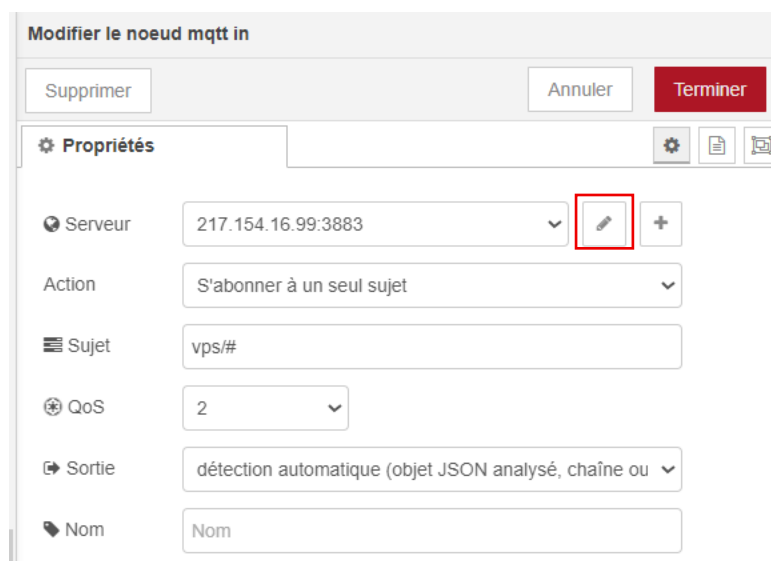
### 3.4 Utilisation de node-red

- Installer le nœud mqtt

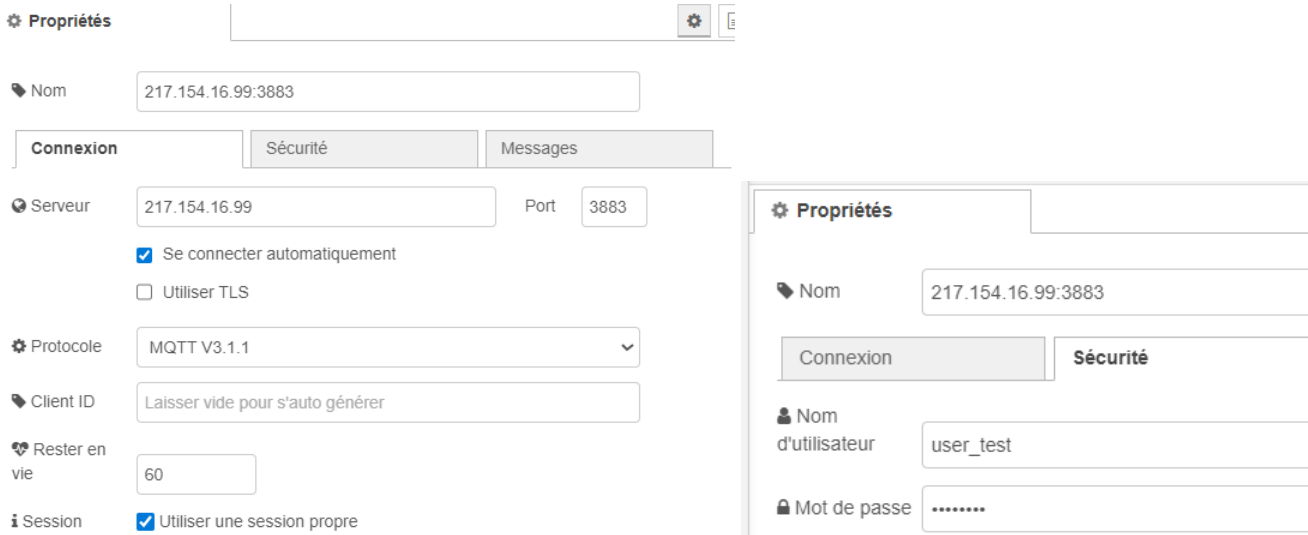


The screenshot shows the Node-RED community catalogue search results for 'mqtt'. The search bar contains 'mqtt' and shows 135 / 5788 results. The first result is 'node-red-iot-mqtt-api' with a description 'return right result to node-red', version '1.0.1', and a download count of 7875. An 'Installer' button is highlighted with a yellow cursor.

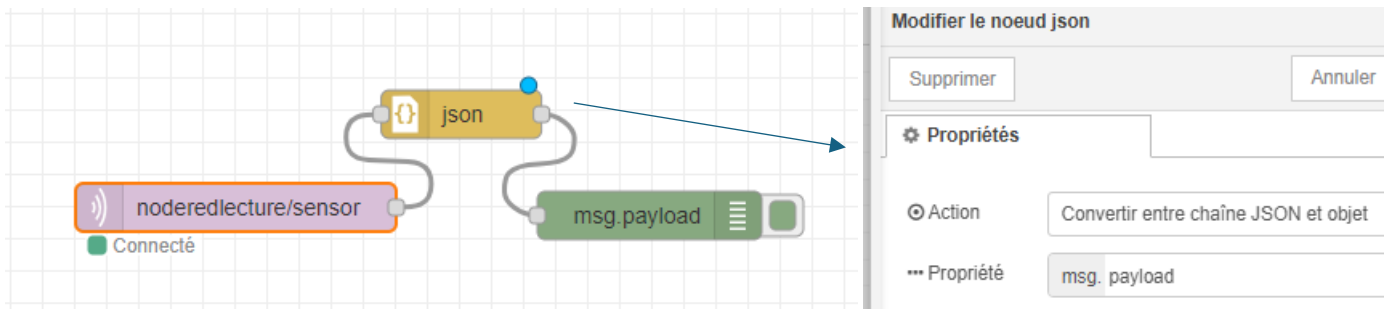
- Insérer le nœud mqtt in et le configurer



The screenshot shows the configuration page for the 'mqtt in' node. The title is 'Modifier le nœud mqtt in'. There are buttons for 'Supprimer', 'Annuler', and 'Terminer'. The configuration is divided into 'Propriétés' (Properties) and 'Options' (Options). The 'Propriétés' section includes: 'Serveur' (217.154.16.99:3883), 'Action' (S'abonner à un seul sujet), 'Sujet' (vps/#), 'QoS' (2), 'Sortie' (détection automatique (objet JSON analysé, chaîne ou)), and 'Nom' (Nom). The 'Serveur' field has a red box around the edit icon.



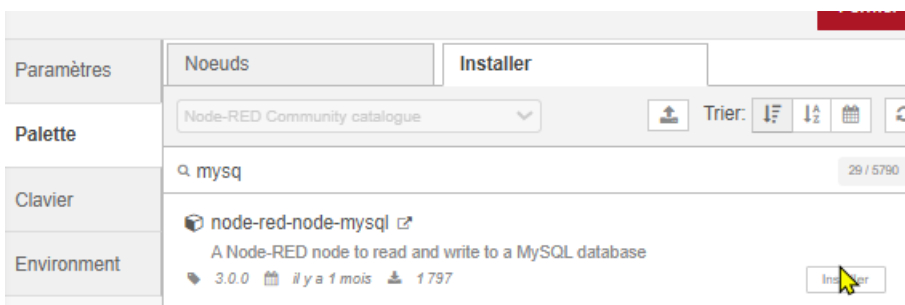
Il faudra de plus ajouter convertisseur string vers json pour remettre en forme la chaîne de caractère dans un objet de type msg.



## 4 Travail avec mysql (mariadb)

Nous allons dans cette dernière partie prendre en main le nœud mysql dans nodered.

- Installer le nœud mysql



Vous avez de nombreux exemples sur le site de node-red, nous allons prendre un flow déjà créé et le modifier

- Allez sur <https://flows.nodered.org/>
- Cherche le flow mariadb

[flows.nodered.org/flow/38ade4a188adb6dccb1d6d3de5ff6a77](https://flows.nodered.org/flow/38ade4a188adb6dccb1d6d3de5ff6a77)

o2-switch autres ionos Ecam-EPMI

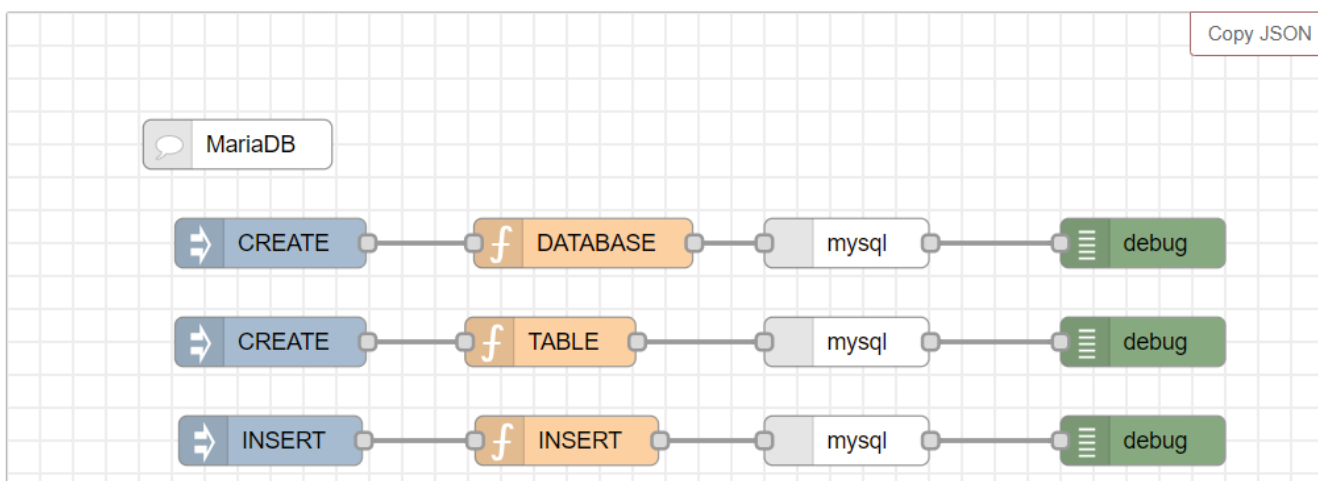
Node-RED

home about blog document

Search library

## MariaDB Flow Sample

MariaDB sample flow. Contains a template node and a mysql node.



```
eat":"","crontab":"","once":
: [{"16d49f4f.2f72e1"}],
,"tosidebar":true,"console":false,"
t":"","crontab":"","once":false,"on
```

Pour l'accès à la base de données : dans le td1, chaque étudiant se connecte à la base de données avec son login et le mot de passe 1234.

Regarder la chaine <https://youtu.be/KPhUmvSShSo> et

**Travail à faire : configurer la base, faite les premiers tests, vérifier avec putty la création de la base l'insertion de données et une requête sql**

**Travail à faire : utiliser un form pour lire des données et les placer dans une table, afficher cette table**

**Insérer les données venant de mqtt et les placer dans la base de données mariadb, afficher ces valeurs sur un graphe**