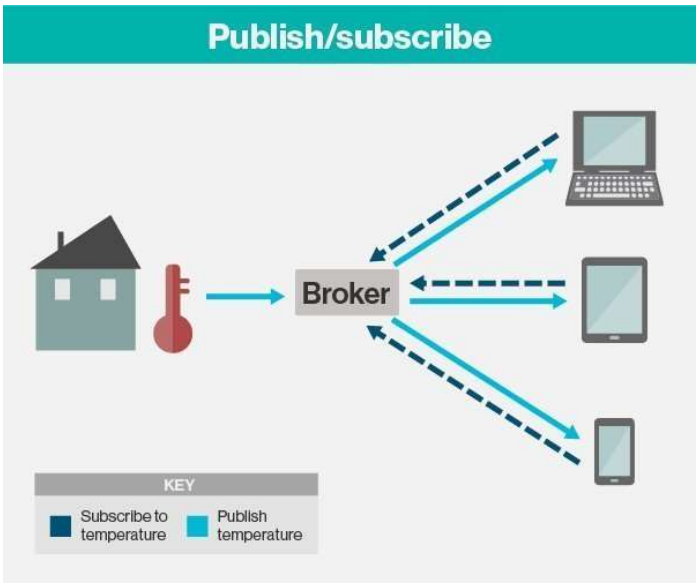
1 – Présentation de MQTT

MQTT est un protocole, basé sur TCP/IP, qui peut remplacer les requêtes http. C'est un protocole léger et les équipements (capteurs) n'ont pas besoin d'être connecté en permanence. C'est un protocole très utilisé dans le monde des objets connectés.

L'ensemble est constitué de :
- Un serveur MQTT appelé Broker.
- D'abonnés "subscribers"
- D'éditeurs "publishers"

Un éditeur publie une information sur un thème, les abonnés à ce thème reçoivent l'information.

Dans l'exemple ci-contre, l'ordinateur, la tablette et le smartphone sont abonnés au thème "la température de la maison". Lorsque le capteur de température publie une nouvelle valeur sur le broker, les abonnés reçoivent l'information.



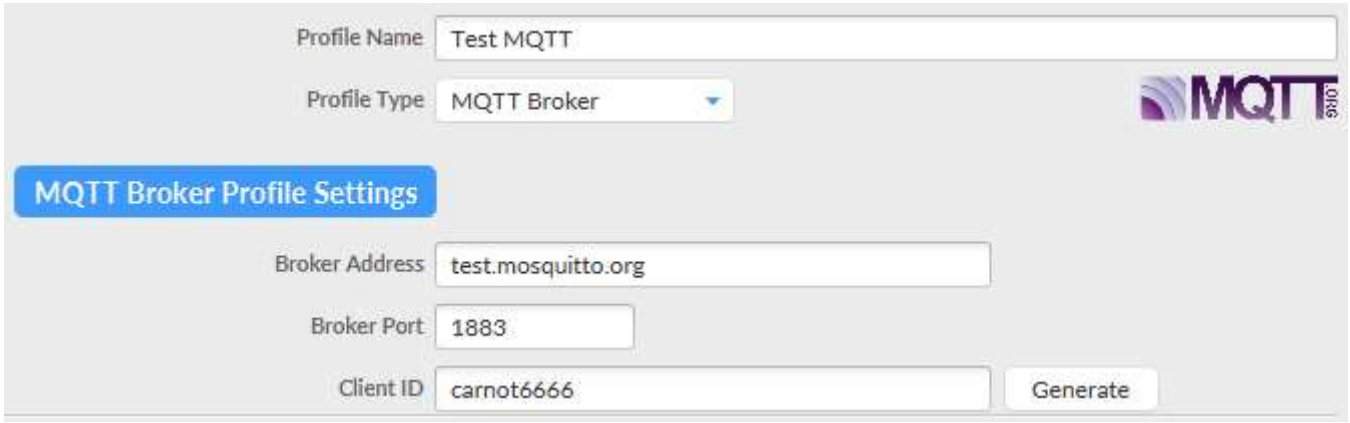
2 – Découverte de MQTT

Le broker est logiquement sur un serveur externe. Dans le cadre des projets on utilisera le serveur de la section, accessible sur internet.

Sur le serveur KWARTZ du lycée, le proxy va nous poser très certainement quelques problèmes.

Dans le cadre de ce TP, on utilisera :

- soit le broker de mosquitto qui se trouve à l'adresse ci-dessous, s'il n'y a pas de soucis avec le proxy :



- soit un broker (Mosquitto par exemple) installé sur une raspberry branchée sur le réseau du lycée.

Il est possible d'installer le broker sur sa propre raspberry, qui servira également de "subscribers" et de "publishers", mais ce fonctionnement n'est pas très logique.
Le protocole MQTT utilise le port 1883 par défaut.

⇨ Demander au prof la solution retenue.

On utilisera comme "subscribers" et "publishers" :

• Sa raspberry branchée sur le réseau du lycée, et le logiciel Node-RED

• Son PC avec le logiciel MQTT.fx (version 1.7.1)

• Eventuellement son Smartphone avec une application (MQTTool sur iOS par exemple) si on utilise un broker externe au réseau du lycée.

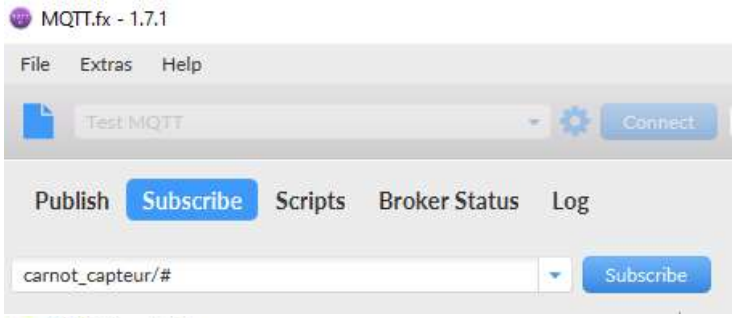
⇨ **Paramétrer le logiciel MQTT.fx :**

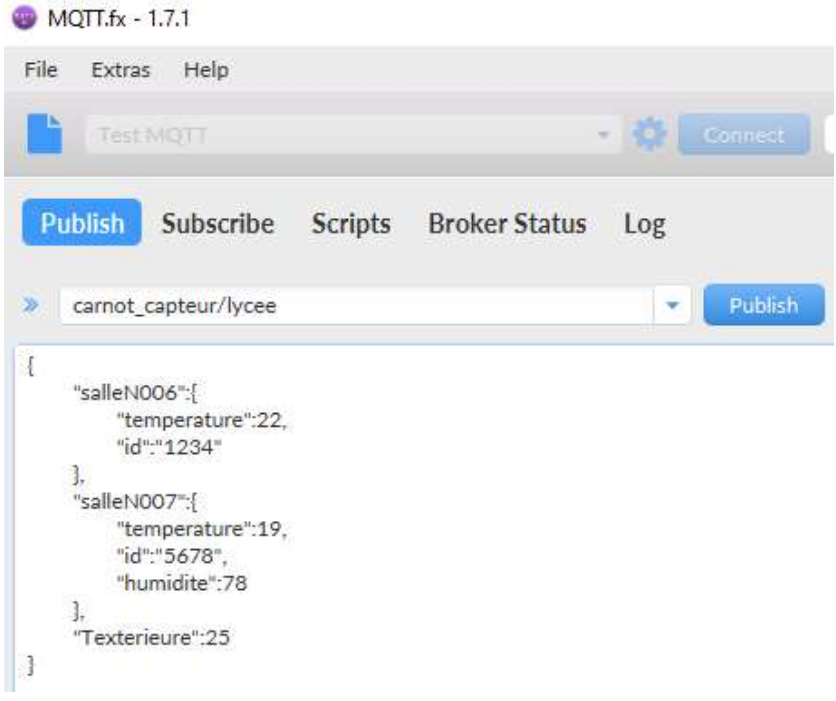
- Pas de mot de passe
- Proxy éventuellement
- Abonnement au sujet : carnot_capteur/# (le # remplace tous les sujets ou topics)
- Editions des messages:

topic : carnot_capteur/lycee

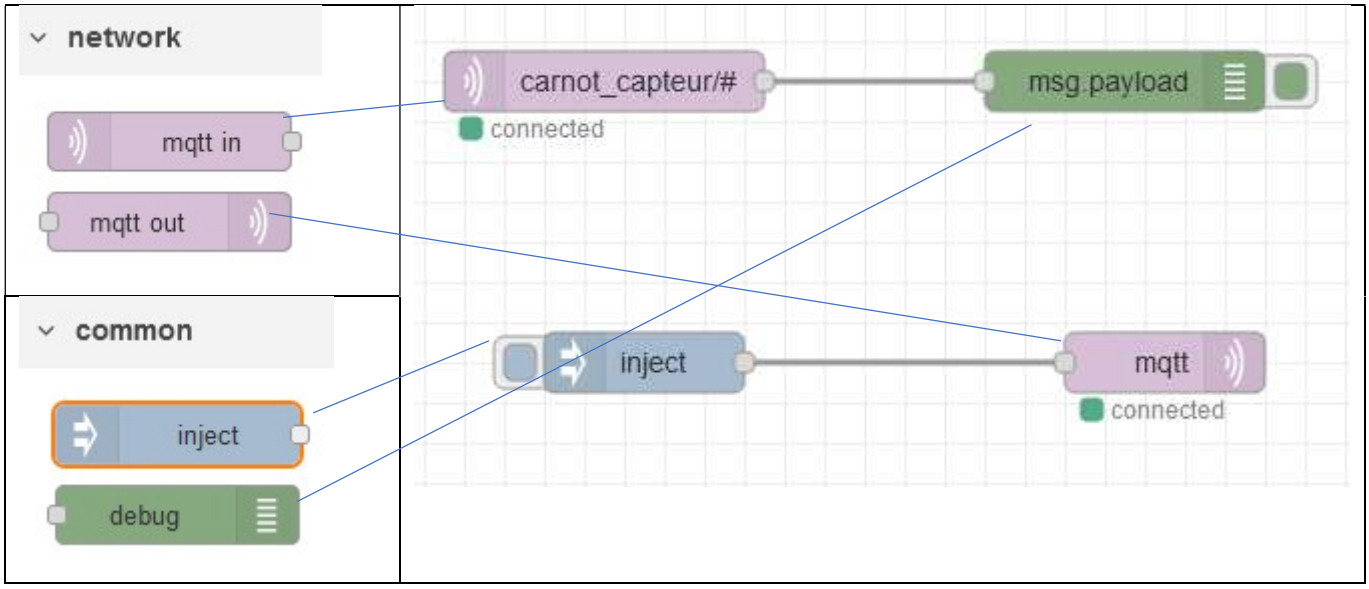
données : le fichier json suivant :

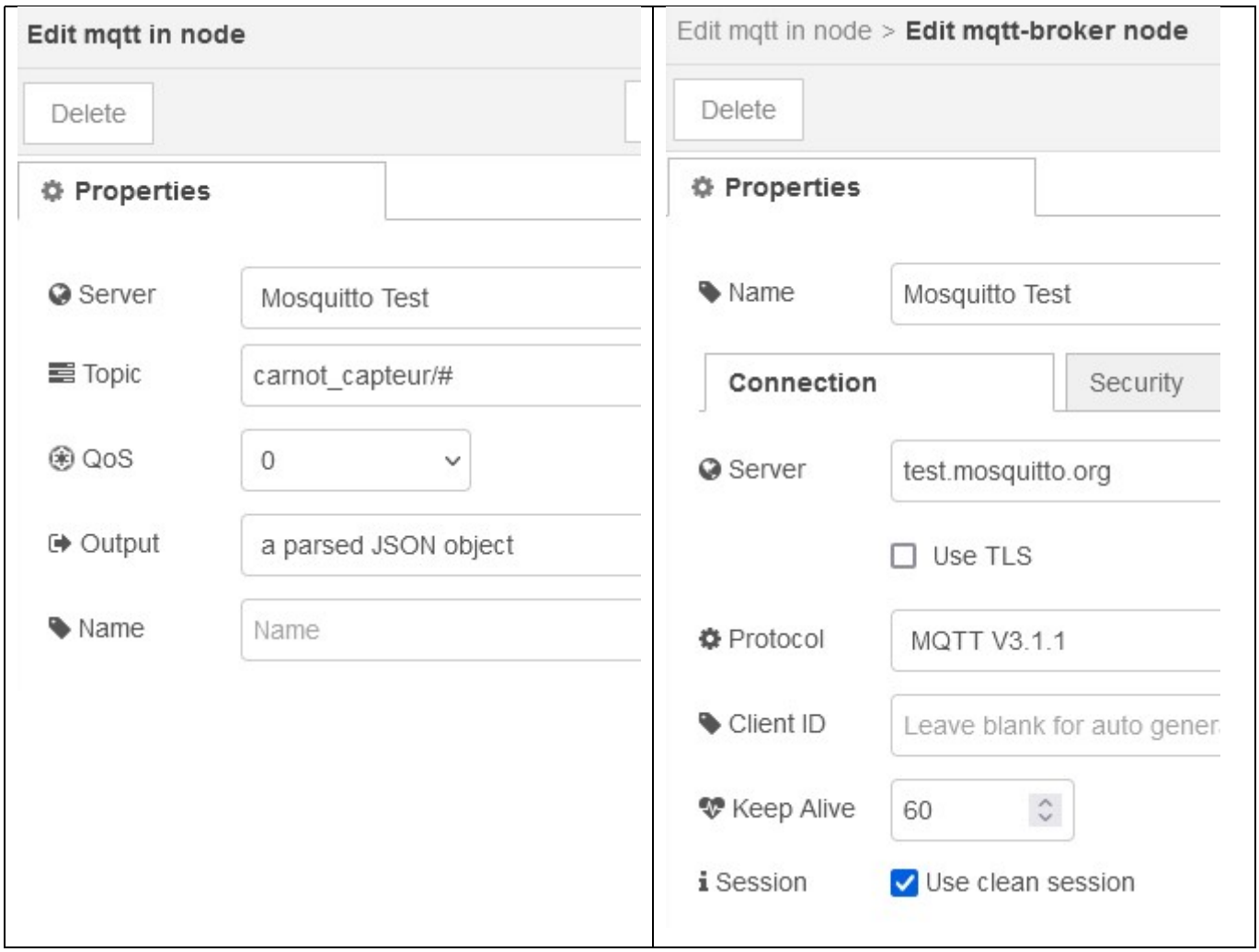
```
{
        "salleN006":{
                "temperature":22,
                "id":"1234"
        },
        "salleN007":{
                "temperature":19,
                "id":"5678",
                "humidite":78
        },
        "Texterieure":25
}
```

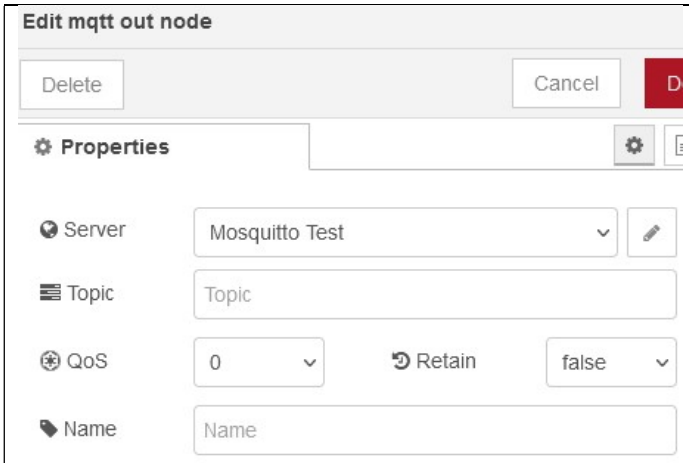


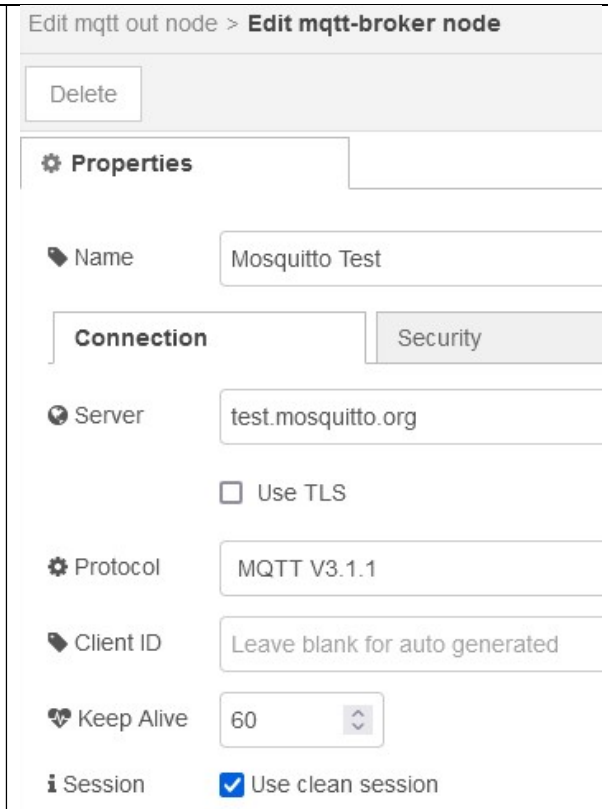


⇨ **Paramétrer sur la Raspberry et sous Node-RED les nodes mqtt in et mqtt out comme ci-dessous.**







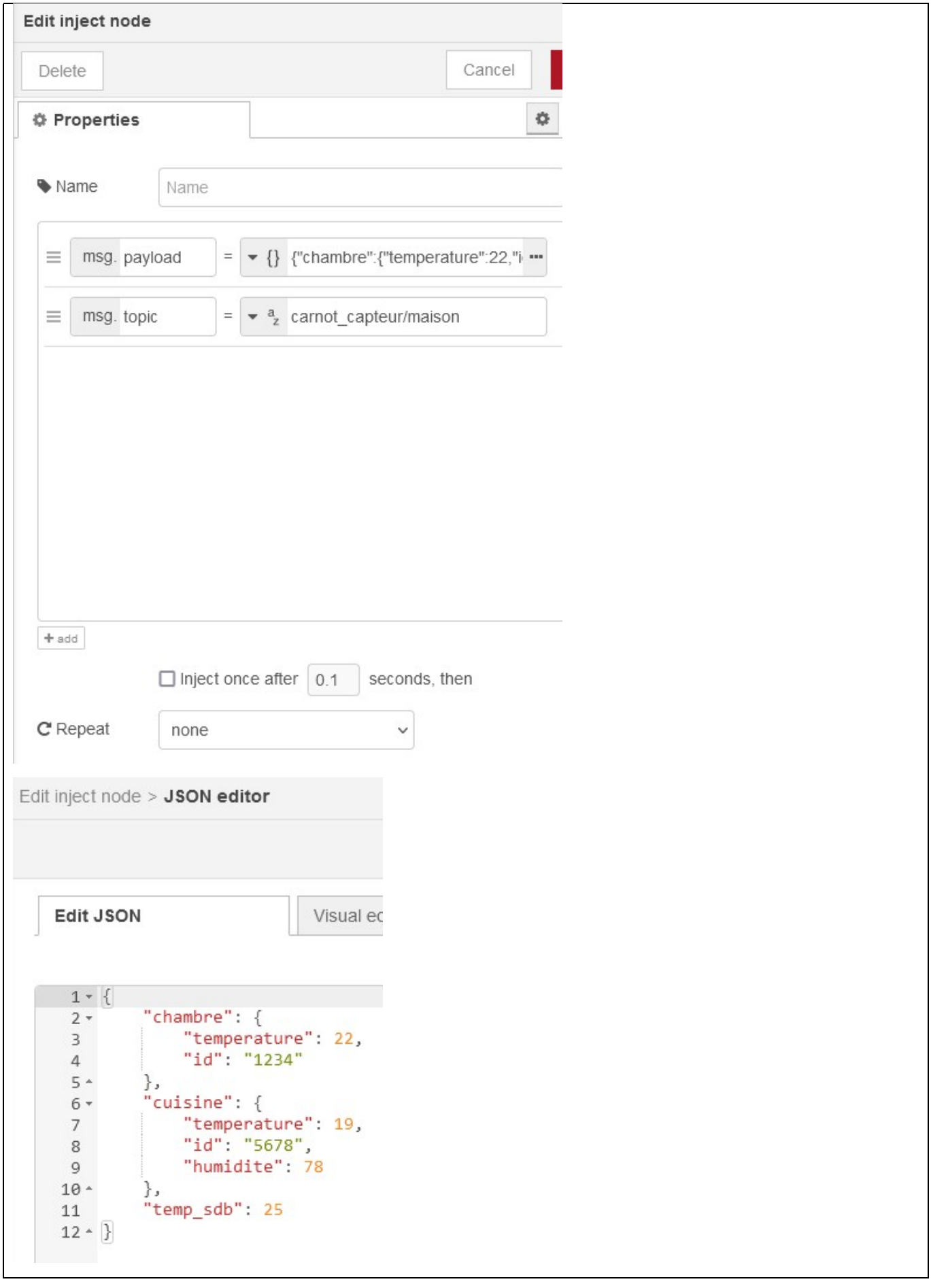


A chaque appui sur le bouton  un message est injecté.

Il est aussi possible d'envoyer un message de façon périodique …

Topic : carnot_capteur/maison – Choisir un format JSON pour le payload. Le fichier JSON a transmettre:

```
{
   "chambre": {
      "temperature": 22,
      "id": "1234"
   },
   "cuisine": {
      "temperature": 19,
      "id": "5678",
      "humidite": 78
   },
   "temp_sdb": 25
}
```



⇨ Envoyer des messages avec le PC (MQTT.fx) et la raspberry (avec node-RED) et vérifier les messages reçus sur les 2 systèmes (fenêtre debug sur node-RED).

⇨ Faire un test éventuellement avec le Smartphone si c'est possible.

Remarque sur la qualité de service :

Avec le protocole MQTT on peut choisir la façon dont le serveur gèrera le message

QoS0 : Le message n'est pas stocké par le serveur, et aucun accusé de réception n'est envoyé. Si l'on redémarre le serveur ou le client, le message sera définitivement perdu

QoS1 : Le message sera envoyé jusqu'à ce que le serveur nous retourne un accusé de réception.

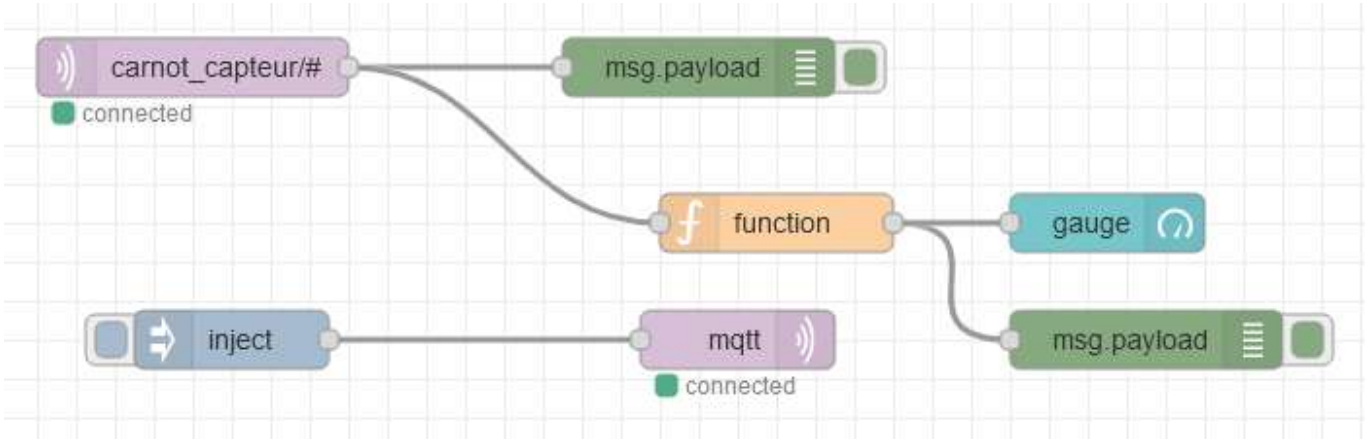
QoS2 : Les messages sont sauvegardés par le serveur, et il les enverra jusqu'à que tous les clients aient reçu le message.

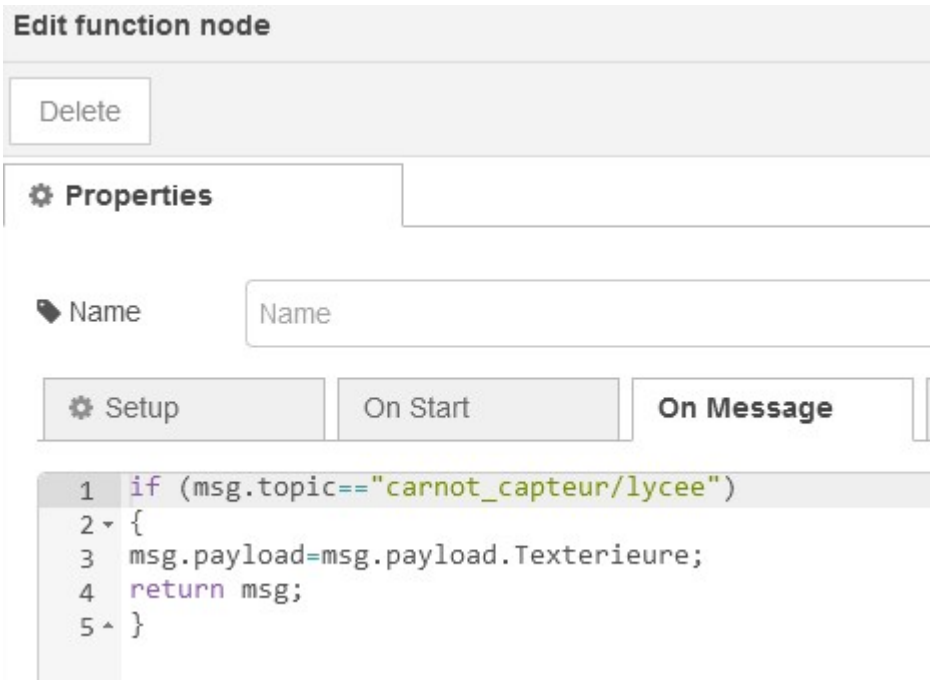
3 – Filtrer les données reçues

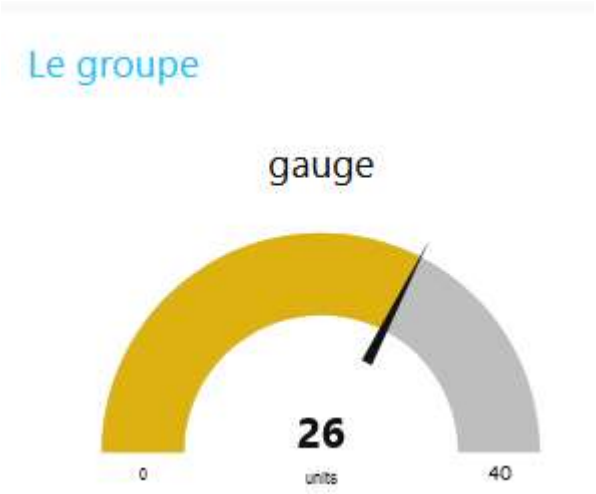
⇨ Ajouter une fonction comme ci-dessous et filtrer l'information Texterieur du topic carnot_capteur/lycee.

Le langage utilisé est le javascript, qui ressemble assez bien au C++.

⇨ Modifier la température Texterieur sur MQTT.fx , publier l'information et visualiser l'interface utilisateur sur la raspberry (ip_raspberry:1880/ui).







⇨ Ajouter une jauge et une fonction pour afficher l'humidité de la salle N007 et tester.