TP4 Event-Driven Architecture

MQTT

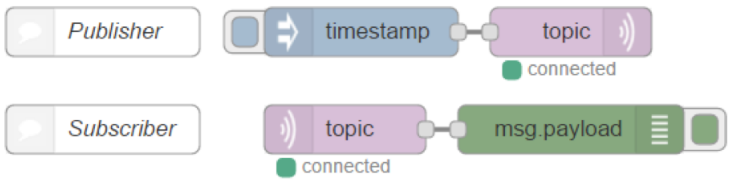
Dans cet exercice nous allons faire un petit rappel sur MQTT à travers quelques manipulations simples. Nous utiliserons Mosquitto, un broker open source (BSD License) qui implémente MQTT. Une instance publique du broker est disponible sur : http://test.mosquitto.org. *Attention ! Ce broker étant public, ne publiez aucunes données sensibles !*

Pour les questions suivantes utilisez un client MQTT de votre choix (vous pouvez par exemple utiliser Node-Red, ou des applications du chrome web store comme : https://chrome.google.com/webstore/detail/mqttbox/kaajoficamnjijhkeomgfljpicifbkaf).

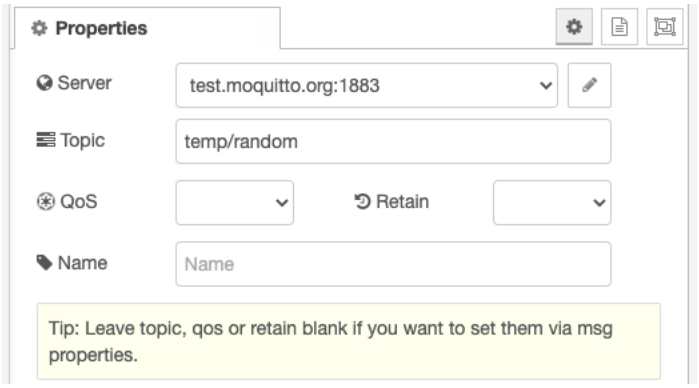
1. Connectez-vous au broker et abonnez-vous aux messages sur le topic « temp/random »
2. Publiez des messages sur ce même topic. Vous devriez recevoir vos messages ainsi que ceux de vos camarades !
3. Savez-vous combien de personnes sont en train de publier des messages sur ce topic ?
4. Savez-vous si tous vos camarades sont toujours en train de publier des messages ?

MQTT et Node-Red

Node-red fourni un ensemble de nœuds pour (i) se connecter à un broker MQTT et (ii) créer un broker MQTT.



Pour se connecter à un broker, il est possible d'utiliser les nœuds « MQTT-in » et « MQTT-out ». Pour souscrire à un topic, il suffit de déposer un nœud MQTT-in dans votre flow Node-red et de configurer (a minima) les propriétés « server », « port », et « Topic » du nœud. Pour publier un message sur un topic, il suffit de déposer un nœud « MQTT-out », et le configurer (même procédure que « MQTT-in »). *Pour rappel, on accède aux propriétés d'un nœud en double cliquant dessus.*



5. Utilisez Node-Red pour souscrire au topic « temp/random » du broker publique `test.mosquitto.org`. Vous pouvez lancer Node-Red via docker en utilisant l'image `nodered/node-red` (n'oubliez pas d'ouvrir les port 1880 !).
6. Utilisez Node-Red pour publier toutes les secondes une valeur de température sur le broker test.mosquitto.org, topic « temp/random ».

Lancer votre propre broker

Vous allez maintenant utiliser docker pour installer sur votre machine votre propre broker MQTT. Nous utiliserons l'image : `eclipse-mosquitto`

Pour éviter les problèmes réseau nos connecterons notre container au réseau de la machine.

7. Lancez Mosquitto en ouvrant au moins le port 1883.
8. Modifiez votre assemblage Node-Red pour qu'il utilise votre propre broker MQTT.

Une application de monitoring

Nous souhaitons maintenant créer une application pour le monitoring de machines se basant sur une architecture event-driven. Dans les faits, pour l'exercice, nous ne collecterons que les données de notre machine.

Préparation

Avant de commencer notre application, nous allons préparer notre environnement de travail. Nous utiliserons le broker `eclipse-mosquitto` et nos event processor seront implémentés avec Node-Red.

- Si vous les avez arrêtés, relancez votre broker MQTT et Node-Red

Notre application

Voici une description textuelle de l'application que nous vous demandons de créer. L'application doit permettre de collecter des informations sur une machine. En particulier nous voulons collecter des informations sur la charge CPU et Mémoire, ainsi

que la Load average. Des alarmes sont émises lorsque (i) le CPU **et** la Load average des 5 dernières minutes dépassent un certain seuil (que vous définirez) et (ii) l'utilisation mémoire dépasse un certain seuil (que vous définirez également). Les alarmes seront affichées sur un tableau de bord. Attention, nous souhaitons faire une application évolutive. Avec le temps, nous souhaitons pouvoir **collecter les données de plusieurs machines** (et toujours visualiser les données de **toutes les machines sur le même tableau de bord sans avoir à reconfigurer/modifier mon tableau de bord pour cela** - i.e., le tableau de bord n'a pas une liste prédéfinie des machines observées). D'autres part, de nouvelles données seront collectées, de nouvelles alarmes seront définies, et **d'autres tableaux de bord pourront être créés** (par exemple juste pour afficher les données CPU et Memoire).

- Proposez une architecture et validez-la avec moi. Quelques-unes des questions que vous devez vous poser :
  - Combien de composants logiciels dois-je avoir dans mon application ?
  - Si j'ai plusieurs composants comment interagissent ils les uns avec les autres ?
  - Est-ce que tous les composants doivent/peuvent être sur la même machine ? Distribués ?
- Votre architecture est de type mediator ou broker ?
- Proposez une liste de topics MQTT et validez-la avec moi.
- Dans vos event-processors, quel type d'architecture mettez-vous en place ?
- Implémentez votre application.
  - Pour collecter les données CPU, mémoire etc. ajoutez à votre palette Node-Red (dans le menu en haut à droite cliquez sur manage palette pour installer des nœuds/modules) les nœuds des modules :
    - `node-red-contrib-os`
    - `node-red-contrib-cpu`
  - Dans le nœud `function`, vous pouvez garder des données entre les messages via des variables globales. Vous pouvez gérer les variables globales avec le code suivant :
    ```
    global.get('name');
    global.set('name','toto');
    ```
  - Pour créer un tableau de bord vous pouvez utiliser les nœuds du module `node-red-dashboard.`

Pour aller plus loin

Écrivez votre propre client MQTT en utilisant le package suivant :
https://www.npmjs.com/package/mqtt