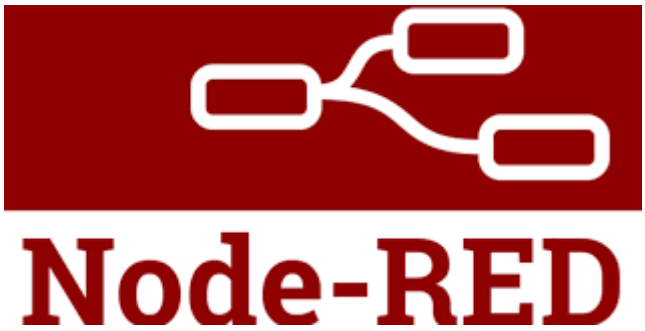
Node-RED



Introduction

Node-RED est une plateforme de développement open-source basée sur un système de programmation visuelle. Elle permet de créer des applications de traitement de données et d'automatisation en reliant des composants via une interface graphique intuitive. Développée par IBM, Node-RED est particulièrement utilisée pour l'Internet des Objets (IoT), l'automatisation des processus industriels, et l'intégration de systèmes avec des services web.

Fonctionnalités principales

- Programmation visuelle : Créez des flux de travail en connectant des "nœuds" pour manipuler et traiter des données.
- Nœuds : Nombreux nœuds prêts à l'emploi pour intégrer des API, des services cloud, des bases de données, des protocoles IoT.
- Facilité d'intégration : Compatible avec divers protocoles et technologies, y compris les systèmes en temps réel, les bases de données, les appareils IoT et les services cloud comme AWS, Google Cloud, etc.
- Interface Web intuitive : L'interface utilisateur est accessible via un navigateur web pour une gestion facile des flux.
- Extensible et personnalisable : Des milliers de nœuds disponibles via la bibliothèque Node-RED, et il est possible d'en ajouter de nouveaux pour plus de fonctionnalités.
- Exécution en continu : Permet l'exécution continue des flux de données et l'intégration des capteurs et des événements en temps réel.

Installation de Node-RED

Installation sur Linux

Node-RED s'appuie sur le gestionnaire de module NPM et Node.JS pour fonctionner

1. J'ouvre un terminal et tape la commande : npm install -g –unsafe-perm node-red
2. J'accède à l'interface sur le port local 1880 : http://127.0.0.1:1880
3. Les fichiers s'enregistrent dans le répertoire /home/user/.Node-red sur Linux

Utilisation de Node-RED

L'interface

- Palette de nœuds : Sur la gauche, les nœuds préinstallés ou ajoutés pour vos projets.
- Zone de travail : Espace central où vous allez assembler les nœuds pour créer vos flux.
- Propriétés du nœud : À droite, vous pouvez configurer les propriétés de chaque nœud sélectionné.
- Menu : En haut à droite, il y a des options pour gérer les flux, démarrer/arrêter le serveur, importer/exporter des flux, etc.

Création d'un flux simple

1. Ajouter un nœud : Glissez-déposez un nœud depuis la palette dans la zone de travail
2. Relier les nœuds : Reliez les nœuds entre eux pour créer un flux de données (par exemple, reliez un nœud HTTP à un nœud de traitement des données).
3. Configurer les nœuds : Cliquez sur un nœud pour configurer ses paramètres.
4. Déployer le flux : Cliquez sur le bouton "Déployer" en haut à droite pour lancer l'exécution de votre flux.

Gestion des nœuds

Installation de nouveaux nœuds : La palette Node-RED permet de rechercher et d'installer des nœuds supplémentaires pour étendre les capacités de votre projet.
Supprimer des nœuds : Supprimer un nœud en le sélectionnant et en appuyant sur la touche "Delete".

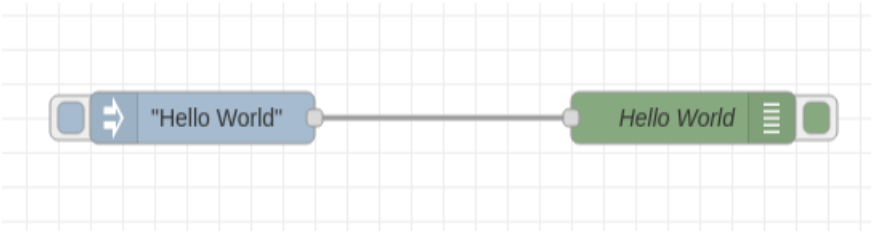
Sauvegarde et exportation des flux

1. Exporter un flux : Vous pouvez exporter un flux en cliquant sur le menu en haut à droite, puis sur "Exporter".
2. Importer un flux : Vous pouvez importer un flux en cliquant sur "Importer" dans le même menu pour partager des configurations entre différents projets.

Exercices

La première application est l'affichage d'un « Hello World ».

1. Glisser et déposer le symbole « common -> inject » au centre
2. Double-clic sur le symbole pour créer la chaîne de caractères « Hello World »
3. Glisser et déposer le symbole « common debug » à droite du symbole inject « Hello World »
4. Double-clic pour renommer le débogueur en « Hello World »
5. Créer un lien entre les 2 points de ces symboles.
6. Activer le programme en cliquant sur le bouton rouge « deploy » en haut à droite et cliquer sur l'onglet « debug ».
7. Cliquer sur la zone bleue à gauche du symbole injecteur « Hello World » (à chaque clic, la fenêtre de débogage affiche un message)



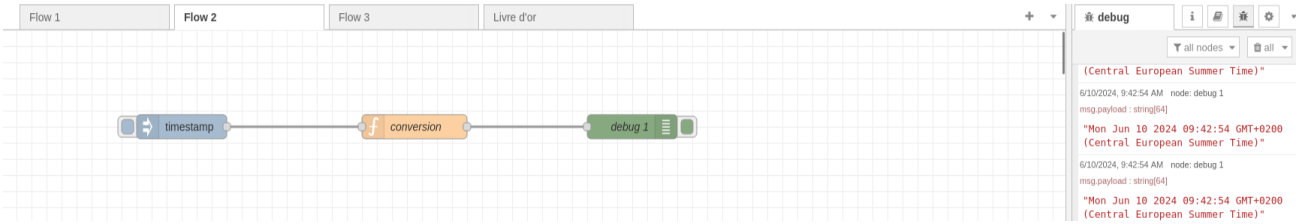
La seconde application est un passage de valeur.

1. Relier un « common -> inject » et un « common -> debug ».
2. A chaque clic, un estampillage horaire s'affiche (nombre en secondes)
3. Il faut insérer une fonction pour convertir le timestamp en format date
4. Glisser un objet « function -> function » entre les deux objets précédents
5. Double cliquer sur l'objet fonction en le renommant « conversion » et avec un onglet [onMessage]

Onglet [onMessage]

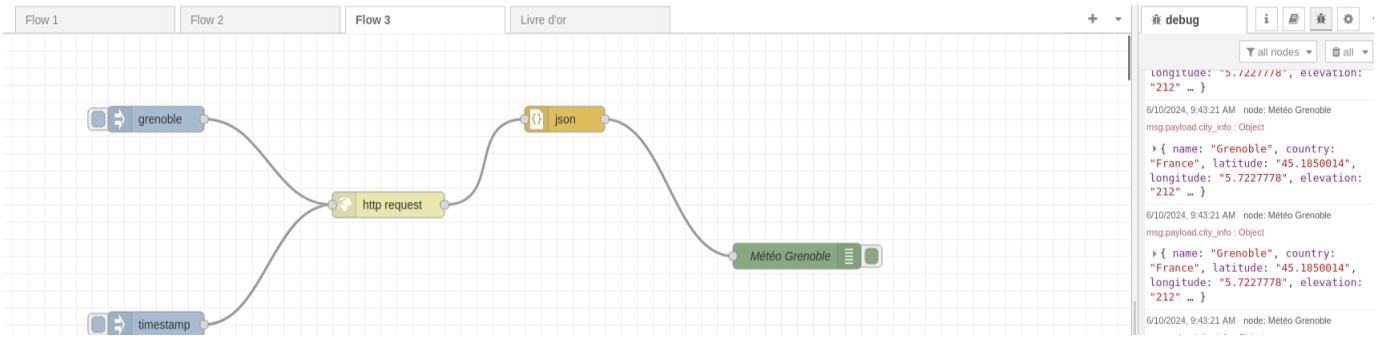
```
let date = new Date(msg.payload);
msg.payload = date.toString();
return msg;
```

6. Valider et lancer le déploiement de la solution avec « deploy ».
7. Un message s'affiche à chaque clic sur l'objet « timestamp »



La troisième application est l'interrogation d'un API REST.

1. Dans l'objet « common -> inject », il faut modifier la valeur du msg.payload par « grenoble »
2. Dans l'objet « network -> http request », il faut ajouter {{payload}} après l'url du service API (URL = https://www.prevision-meteo.ch/services/json/{{payload}})
3. Dans l'objet « parser -> join », choisir l'action « Convert between JSON String and Object »
4. Dans l'objet « common -> debug », modifier l'affichage pour une partie de l'objet : msg.payload.city_info
5. Relier les éléments et déployer



La quatrième application est une interaction MySQL.

1. Importer la bibliothèque MySQL et aller dans le menu Manage Palettes.
2. Dans l'onglet [Install], rechercher 'mysql' puis cliquer sur le bouton install.
3. Dans le menu des objets, il y a un objet « Storage -> MySQL »
4. Créer un nouvel onglet « Livre d'Or »
5. Configurer la base

CREATE DATABASE livredor CHARACTER SET utf8mb4 COLLATE utf8mb4_unicode_ci; USE livredor; CREATE TABLE `tlivre` (`id` int(11) NOT NULL, `name` varchar(50) NOT NULL, `message` varchar(300) NOT NULL, `evaluation` int(1)) ENGINE=InnoDB DEFAULT CHARSET=utf8;---- Contenu de la table `matable`-INSERT INTO `tlivre` (`id`, `name`, `message`, `evaluation`) VALUES (1, 'NORRIS Chuck', 'Best web site David ;)', 5), (2, 'KENT Clark', 'Need informations about Batman please', 4), (3, 'PARR Hélène', 'Hey, nothing about girl in computers science!!!', 0), (4, 'DE LA VEGA Don Diego', 'sometime Zorro? Your website is zero', 0), (5, 'M. BEANS', 'Are you serious? Do not play with security', 2); CREATE USER 'admtlivre'@'localhost' IDENTIFIED BY 'adm123'; GRANT ALL PRIVILEGES ON * . * TO 'admtlivre'@'localhost';

6. Glisser et déposer le symbole « storage -> mysql » dans l'espace principal.
7. Double clic sur celui-ci pour afficher les propriétés et cliquer sur l'icône d'édition (crayon) pour saisir les informations sur la base de données.

User : admtlivre
Password : adm123
Database : livredor
Name : Exemple-Node-Red

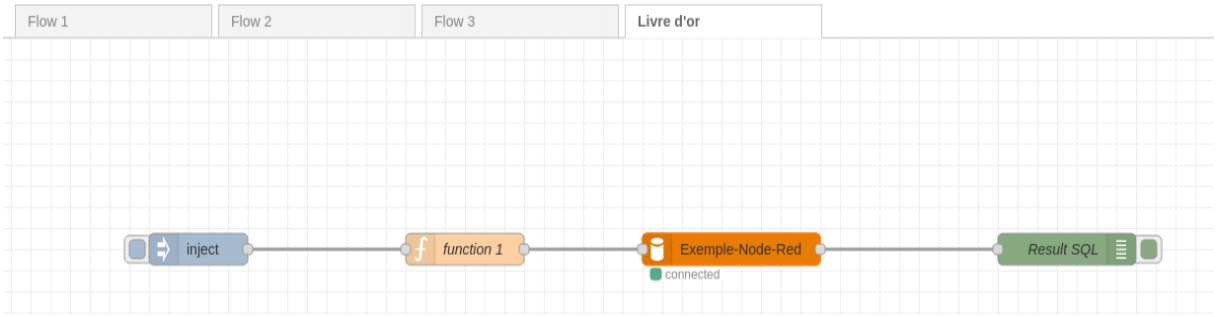
8. Cliquer sur le bouton « add » et valider la création en cliquant sur le bouton « done ».
9. La configuration est active si le symbole mysql n'a plus de triangle rouge.
10. Insérer un objet « function -> function » et double clic dessus pour insérer une requête SQL

msg.topic : contient la requête SQL
msg.payload : contient les données à insérer dans la table ou la base
return.msg : contient le résultat de la requête

Le contenu de la fonction est :

```
msg.topic = "SELECT * FROM tlivre";
return msg;
```

11. Ajouter un bouton « common -> inject » et un « common -> debug » puis le renommer « Result SQL »
12. Cliquer sur le bouton « deploy » de l'éditeur



Ensuite, on va écrire dans la base en utilisant le champ msg.payload.
Pour la requête préparée : voici le contenu de la fonction :

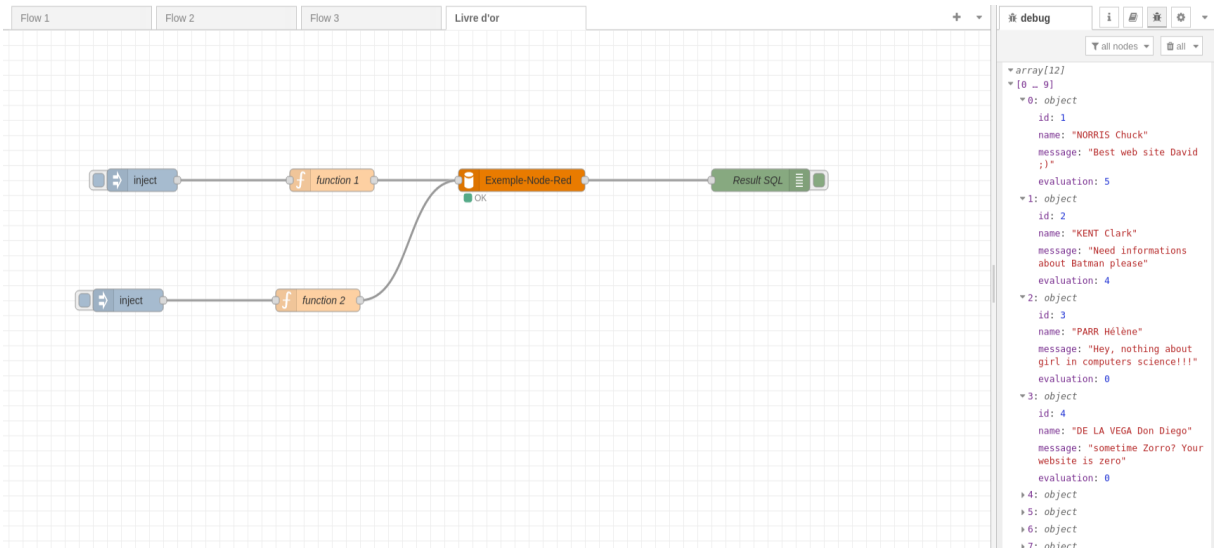
```
msg.payload=[6, 'MUSK Elon', 'Best book ever seen about me, no?', 5]
msg.topic = "INSERT INTO `tlivre` (`id`, `name`, `message`, `evaluation`) VALUES (?, ?, ?, ?)";
return msg;
```

Pour la requête paramétrée :

```
msg.payload={}
msg.payload.uid = 6;
msg.payload.uname = "MUSK Elon";
msg.payload.cmt = "Best book ever seen about me, no?";
msg.payload.mark = 5;

msg.topic = "INSERT INTO `tlivre` (`id`, `name`, `message`, `evaluation`) VALUES
(:uid,:uname,:cmt,:mark) ON DUPLICATE KEY UPDATE `message`=:cmt;";
return msg;
```

1. Ajouter un bouton « common -> inject » pour cette fonction.
2. Cliquer sur le bouton « deploy »



Ensuite, je vais récupérer un formulaire HTML.

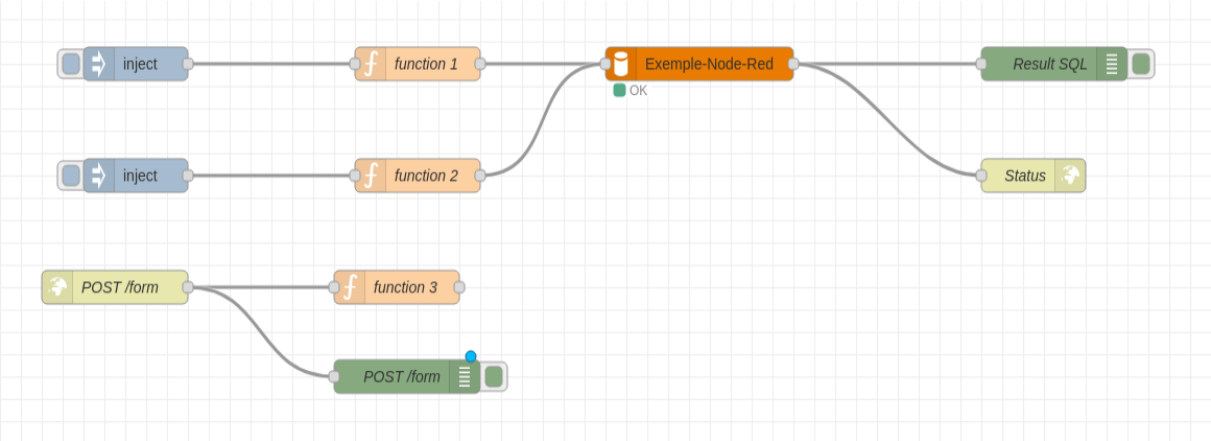
1. Installer un serveur web (XAMPP par exemple) ou Python.
2. Créer le fichier HTML suivant

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Test Node-Red</title>
</head>
<body>
    <form action="http://localhost:1880/form" method="post">
        <input name="uid" value="7">
        <input name="uname" value="DrNozman">
        <input name="cmt" value="Trop stylé, ce Node-Red">
        <input name="mark" value="5">
        <button>Envoyer</button>
      </form>
</body>
</html>
```

3. Insérer un objet « Network -> http » et double clic dessus.
4. Choisir la méthode POST, indiquer l'url /form et le nommer POST /form
5. Copier la fonction SQL écriture et coller la nouvelle en dessous.
6. Supprimer la variable msg.payload.

```
msg.topic = "INSERT INTO `tlivre` (`id`, `name`, `message`, `evaluation`) VALUES
(:uid,:uname,:cmt,:mark) ON DUPLICATE KEY UPDATE `message`=:cmt;";
return msg;
```

7. Ajouter un objet « Network -> http response » nommé 'Status' et relié à la sortie de l'objet mysql.
8. Ajouter un objet « common -> debug » relié à la sortie de l'objet POST /form pour obtenir le résultat brut dans la console de débogage.



9. Lancer la solution avec « deploy » et ouvrir le formulaire HTML dans un navigateur.
10. Cliquer sur le bouton envoyer.

Ces exercices sont réalisés à partir du site du Reseau Certa : découverte de la programmation low-code.