EDGE Node-RED Unified

Node-RED Docker Application for UNIFIED COMFORT PANELS with some of most used Node-RED Nodes preinstalled.

Table of Contents

Install the App

Edge Node-RED Unified comes with pre-builded `edge-node-red-unified_x.x.x.app` package that can be installed specifically on Unified Comfort Panels that runs SIMATIC Edge Runtime.

Requirements

> Before loading this application, check if the requirements above are satisfied by the selected Edge Device for installation.

In order to run this Edge App, the selected Edge System need to satisfy the following requirements:

Description	
Available Load Memory	>= 200 MB

Prerequisites

1. A Unified Comfort Panel with SIMATIC Edge feature enabled.

2. At least one user needs to be signed up

Load App on Unified Comfort Panels

1. Copy the downloaded `edge-node-red-unified_x.x.x.app` file to your Developer PC.
2. Open the Industrial Edge Management Web Page of UCP on `https://<ucp-address>`
3. Connect it both to your Docker engine and to IEM
4. Import the .app file using the *Import Offline* button
5. Wait until App is installed

Run the App

Once the app has been installed the Docker service starts running immediately.

You can access the app web interface by clicking on the *edge-node-red-unified* app icon in Edge Device Web Page or through endpoint. The app has two HTTPS Endpoints configured with SIMATIC Edge Reverse Proxy function:

- `https://<ied-address>/edge-node-red` : this endpoint open the Node-RED editor interface.
- `https://<ied-address>/edge-node-red-ui` : this endpoint open the Node-RED Web Dashboard.

> A user must be logged on Edge Device in order to open Reverse Proxy App endpoints. If no user is logged the endpoint URL will give Error 503 on load.

If you need, is it possible to access the app Endpoints also by using the following direct URLs with mapped ports:

- `https://<ied-address>:41880` : this endpoint open the Node-RED editor interface.
- `https://<ied-address>:41880/ui` : this endpoint open the Node-RED Web Dashboard.

Extra Mapped Port for Additional Services

There is also an additional Port mapped through the docker-compose.yml file that is the port **41890** that could be used for extra services and features that require a dedicated port like e.g. exposure of an OPCUA Server with `node-red-contrib-opcua` node.

WinCC Unified Communication with OpenPipe Nodes

The main reason why this app is specific for Unified Comfort Panels is that, among the various pre-installed nodes, some of them are dedicated to communication with **WinCC Unified** (as default on UCPs) for exchange Tags and Alarms data with the HMI supervision system.

The installed nodes use the **OpenPipe Socket** communication channel, which requires a dedicated volume in the Edge App. These nodes are contained in the `openpipe_nodes` folder copied in the Docker image building phase. For more information on how this feature was integrated, see chapter How the App is built from scratch.

Extra nodes in package.json

Node-RED comes with a core set of useful nodes, but there are many more available from both the Node-RED project as well as the wider community. You can search for available nodes in the Node-RED library.

The following snippets of `package.json` file lists all the extra nodes installed in this Node-RED App:

```
{
  ...
  ...
    "dependencies": {
        "node-red": "1.1.2",
        "@mindconnect/node-red-contrib-mindconnect": "^3.8.1",
        "node-red-contrib-azureiothubnode": "^0.5.3",
        "node-red-contrib-influxdb": "^0.4.1",
        "node-red-contrib-mssql-plus": "^0.4.4",
        "node-red-node-mysql" : "0.1.1",
        "node-red-contrib-postgres-variable": "0.1.4",
        "node-red-contrib-s7": "^2.2.1",
        "node-red-contrib-opcua": "^0.2.72",
        "node-red-contrib-modbus": "^5.13.3",
        "node-red-contrib-cip-ethernet-ip" : "^1.1.2",
        "node-red-contrib-string": "^0.2.2",
        "node-red-contrib-telegrambot": "^8.3.3",
        "node-red-node-ping": "^0.2.1",
        "node-red-contrib-moment": "^3.0.3",
        "node-red-dashboard": "^2.23.0",
        "node-red-hmi-rt-read-alarms": "file:hmi-runtime-read-alarms",
        "node-red-hmi-rt-read-tags": "file:hmi-runtime-read-tags",
        "node-red-hmi-rt-write-tags": "file:hmi-runtime-write-tags"
    }
}
```

You can also install nodes directly within the editor by selecting the `Manage Palette` option from the main menu to open the Palette Manager.

On Node-RED Documentation Website you can find more information on how to Add nodes to the palette.

How the App is built from scratch

This App is based on the official Node-RED Docker Image. On Node-RED Documentation Website you can find more information on How to Run Node-RED Docker and how to generate Image Variation for own customization.

The following sections will describe each file used for generate this App:

`package.json`

To add any extra nodes on Node-RED Docker base image it is necessary to insert them into the original `package.json` file available from node-red-docker Repository and build a new docker image.

See the complete list of available Node-RED nodes for this App version on chapter Extra nodes in package.json

`docker-compose.yml`

The Node-RED Docker base image used in this App is built using docker-compose tool with the command `docker-compose up -d --build` on the following `docker-compose.yml` file:

```
version: "3.4"

services:
  edge-node-red-unified:
    container_name: edge-node-red-unified
    build:
      context: ./edge-node-red-unified
      dockerfile: Dockerfile
      args:
        - NODE_RED_VERSION=1.1.2
    image: edge-node-red-unified:1.1.2
    restart: always
    privileged: true
    environment:
      - TZ=Europe/Rome
    ports:
      - "32880:1880"
      - "32890:30890"
    volumes:
      - edge-node-red-data:/data

volumes:
  edge-node-red-data:
```

The above compose file:

- creates the `edge-node-red-unified` service container
- build our custom Node-RED Docker image using `./edge-node-red-unified/Dockerfile` file and passing to it the argument **1.1.2** as the wanted `NODE_RED_VERSION` to be installed
- sets the timezone to `Europe/Rome`
- Maps the container port 1880 to the the host port **41880** (for Node-RED Web Interface)
- Maps the container port 41890 to the the host port **41890** (for extra features like e.g. exposure of an OPCUA Server with `node-red-contrib-opcua` node.)
- persists the `/data` dir inside the container to the`edge-node-red-data` volume in the Host System.

`Dockerfile`

In order to customize the Docker base image of this App, the following `Dockerfile` was used:

```
ARG NODE_RED_VERSION=latest
################################################################################
######
FROM nodered/node-red:${NODE_RED_VERSION}-minimal as BASE
```

```
# Copy package.json to the WORKDIR so npm builds all
# of your added nodes modules for Node-RED
COPY package.json .

# Copy OpenPipe Commmunication Nodes
COPY openpipe-nodes/ .

################################################################################
######
FROM BASE as BUILD

USER root

# Install devtools for building new nodes on minimal image
RUN apk add --no-cache --virtual buildtools build-base linux-headers udev python
&& \
    npm install --unsafe-perm --no-update-notifier --no-fund --only=production &&\
    chmod -R 777 .

################################################################################
######
FROM BASE AS RELEASE

# copy builded node modules from BUILD
COPY --from=BUILD /usr/src/node-red/node_modules ./node_modules

# Clean up
RUN rm -rf /tmp/*


ENTRYPOINT ["npm", "start", "--cache", "/data/.npm", "--", "--userDir", "/data"]
```

Taking the argument `NODE_RED_VERSION` passed by docker-compose file, docker start building process from node-red base image `nodered/node-red:<NODE_RED_VERSION>-minimal` and the new `package.json` file and `openpipe-nodes` nodes folder are copied to new docker image. The *minimal* version is used to reduce the amount of space of the `RELEASE` Docker Image as it not include all the linux build packages used for install new nodes.

In the `BUILD` phase, the needed Linux packages for nodes installation are installed by `apk add` command, then all nodes are installed by `npm install` command.

For the `RELEASE` phase the `BASE` minimal image is used and the `node_modules` folder builded on `BUILD` phase are copied to the `WORKDIR` folder. The `ENTRYPOINT` command will start Node-RED using `/data` folder as root directory.

Import in Edge App Publisher

By importing the `docker-compose.yml` file in the Edge App Publisher some changes are applied in order to make the app compatible with the SIMATIC Edge environment:

- The `build` parameter is deleted since the image was already builded.

- In the **Storage** Section of the imported App a new volume for USB Media Mounting was added by using the pre-configured option.
- In the **Storage** Section of the imported App a new volume for WinCC Communication with Openpipe Socket was added by using the pre-configured option.
- In the **Network** section of the imported App two Reverse Proxy endpoints was defined with following parameters:

Port	Type	Service Name	Rewrite Target
1880	HTTP	edge-node-red	/
1880	HTTP	edge-node-red-ui	/ui

- The `mem_limit` parameter is added since is a mandatory field for SIMATIC Edge applications.

Below you can find the extracted `docker-compose` file from the Edge App:

```
version: '2.4'
services:
  edge-node-red-unified:
    container_name: edge-node-red-unified
    image: 'edge-node-red-unified:1.1.2'
    restart: always
    environment:
      - TZ=Europe/Rome
    ports:
      - '41880:1880'
      - '41890:41890'
    volumes:
      - 'edge-node-red-data:/data'
      - '/tmp/siemens/automation:/tempcontainer/'
      - '/media/simatic:/media/simatic:ro,slave'
    mem_limit: 200mb

  volumes:
    edge-node-red-data: null
```

References

- Node-RED Library - Official Node-RED Collection of Nodes and Example Flows.
- Node-RED Docker Image - Official Node-RED Docker Image from Docker Hub.
- Node-RED Documentation - Node-RED Documentation Portal for every needs.
- OpenPipe Manual - Siemens Industry Support Manual for OpenPipe Communication functions understanding.

Release History

- 0.0.6
  - The first proper release.

License

Distributed under the MIT License. See `LICENSE` for more information.

Contributing

1. Fork it (https://github.com/yourname/yourproject/fork)
2. Create your feature branch (`git checkout -b feature/fooBar`)
3. Commit your changes (`git commit -am 'Add some fooBar'`)
4. Push to the branch (`git push origin feature/fooBar`)
5. Create a new Pull Request

Contacts

- Davide Maffei - davide.maffei@siemens.com