neoOntologyDocker

neoOntology is a data storage service for ontologies and related files based on nodeJS, Neo4j and neosemantics technologies.

This version is a docker-composed project to use neoOntology using docker images.

For more information, please contact inigofernandezdelamo@outlook.com

Running and exiting neoOntologyDocker

1. To start neoOntology:
    1. In your console, move to neoOntologyDocker directory:

        ```
        cd xxx/{NEOONTOLOGY-DIRECTORY}
        ```

    2. Run the following command:

        ```
        docker-compose up
        ```

    3. The server will be ready for use at http://localhost:3003 when the following appears on the console:

        ```
        store_1 | 2020-07-07 16:26:30.707+0000 INFO Remote interface available at http://localhost:7474/
        ```

    4. To stop neoOntology:
    5. In the same console-window as before, press the following:

        ```
        CTRL+C
        ```

    6. Or in a separate console window:
        1. Move to neoOntolodyDocker directory:

            ```
            cd xxx/{NEOONTOLOGY-DIRECTORY}
            ```

        2. Run the following command:

            ```
            docker-compose up
            ```

    Once neoOntology docker is installed, you can manage the Docker container from the Docker dashboard. Check Docker Docs for more information.

To install neoOntologyDocker, please follow the instructions below.

Accessing neoOntologyDocker from neoOntologyAR

To use neoOntologyAR it is necessary to establish a connection with neoOntology services.

In order to do so, you just need to follow the next instructions when using neoOntologyAR on your HoloLens device:

1. At start, click on the server's address and type yours as follows:

    ```
    http://{YOUR-IP-ADDRESS}:3003
    ```

2. If your machine and associated port are open to the public, neoOntologyAR should be able to connect without further issues.
3. If that is not the case, please ensure your port and IP address are discoverable. Otherwise, contact Iñigo Fernández del Amo.
4. To install neoOntologyAR and neoOntologyCM services, please contact Iñigo Fernández del Amo.

First-time setup

To start this version of neoOntology you must need to do the following:

1. Download and install Docker
2. Download this and unzip this folder from Github
3. On your console, move to the directory where you unzipped this folder and run the following command:

    ```
    docker-compose up
    ```

4. If the neoOntology server has started correctly, you should see something like the following on your terminal:

    ```
    store_1 | 2020-07-07 16:26:30.707+0000 INFO Remote interface available at http://localhost:7474/
    ```

5. At this point, you need to add the default ontologies to your local store for neoOntologyDocker to work correctly. For doing so, follow the next instructions:
    1. Connect to your local data store at http://localhost:7474
    2. You will be requested to authenticate:
        - Username: neo4j
        - Password: opexcellence
    3. Once authenticated, please run the following commands in order to upload the ontologies to the data store:

- `CREATE INDEX ON :Resource(uri)`
- `CREATE INDEX ON :URI(uri)`
- `CREATE INDEX ON :BNode(uri)`
- `CREATE INDEX ON :Class(uri)`
- ``CREATE (:NamespacePrefixDefinition {`http://www.w3.org/XML/1998/namespace`: 'xml', `http://www.w3.org/2001/XMLSchema#`: 'xsd', `http://www.w3.org/1999/02/22-rdf-syntax-ns#`: 'rdf', `http://www.w3.org/2000/01/rdf-schema#`: 'rdfs', `http://www.w3.org/2002/07/owl#`: 'owl', `http://138.250.108.1:3003/api/files/owl/orgont#`: 'orgont', `http://138.250.108.1:3003/api/files/owl/remont#`: 'remont', `http://138.250.108.1:3003/api/files/owl/repont#`: 'repont', `http://138.250.108.1:3003/api/files/owl/geomont#`: 'geomont', `http://138.250.108.1:3003/api/files/owl/diagont#`: 'diagont', `http://138.250.108.1:3003/api/files/owl/damont#`: 'damont'})``
- `CALL semantics.importRDF("http://host.docker.internal:3003/api/files/owl/orgont.owl", "RDF/XML", {shortenUrls: true, typesToLabels: true, commitSize: 15000})`
- `CALL semantics.importRDF("http://host.docker.internal:3003/api/files/owl/remont.owl", "RDF/XML", {shortenUrls: true, typesToLabels: true, commitSize: 15000})`
- `CALL semantics.importRDF("http://host.docker.internal:3003/api/files/owl/repont.owl", "RDF/XML", {shortenUrls: true, typesToLabels: true, commitSize: 15000})`
- `CALL semantics.importRDF("http://host.docker.internal:3003/api/files/owl/geomont.owl", "RDF/XML", {shortenUrls: true, typesToLabels: true, commitSize: 15000})`
- `CALL semantics.importRDF("http://host.docker.internal:3003/api/files/owl/diagont.owl", "RDF/XML", {shortenUrls: true, typesToLabels: true, commitSize: 15000})`
- `CALL semantics.importRDF("http://host.docker.internal:3003/api/files/owl/damont.owl", "RDF/XML", {shortenUrls: true, typesToLabels: true, commitSize: 15000})`

At this point, neoOntologyDocker should work including all services provided by the ontologies above.

To learn more on how to upload other ontologies to the data store (neo4j) please visit neosemantics.

To learn more on how to use neoOntology and its associated services (neoOntologyCM and neoOntologyAR), please contact Iñigo Fernández del Amo.