Private Public Partnership Project (PPP)

Large-scale Integrated Project (IP)

FIWARE-Metaware

Project full title: Future Internet Core
Project acronym: FI-Core
Contract No.: 632893
description: Metadata Store Management Platform Documentation

Contents

Metaware

Introduction

Welcome to the Metaware Installation and Administration Guide! Metaware is a Java-based web service that deals with metadata information, and it can run on Apache Tomcat.

It is an implementation of the FIWARE Metadata Store Management Platform Generic Enabler, from Telecom Italia and Consoft Sistemi S.p.A.

Any feedback on this document is highly welcome, including bug reports, typos or stuff you think should be included but is not. Please send the feedback through GitHub. Thanks in advance.

System requirements

This section describes the basic requirements of Metaware and how to install them in your system.

Hardware requirements

The hardware requirements depend on the application to be deployed, and for what concerns Metaware, any hardware running Java JRE/JDK 7 or later is supported. Please refer to the main Oracle Java web-page for minimal hardware requirements and further details.

Software requirements

Runtime System:

- Any Java SE JRE 7 or later distribution;
- Apache Tomcat 7 or later;
- MongoDB 2.6 or later.

Development system:

- Any Java SE JDK 7 or later distribution;
- Apache Tomcat 7 or later (if you are working on Windows, you can also use XAMPP);
- MongoDB 2.6 or later.

Apache Maven, or other building tools are necessary to create the final WAR package that will run on Tomcat. We can also suggest to build the WAR package by using your favorite IDE (which should include Maven).

Operating system supported

Any OS running Java JRE/JDK 7 or later is supported.

Metaware is tested on Windows (7) through XAMPP.

Installation

Installation of the Java JDK

Please follow the installation instructions of the respective Java distribution:

- OpenJDK

- Oracle Java SE JDK

To verify that the installation is correct, please open a terminal, a shell, or a command line interface (cmd.exe), and check that the `java` command is executable:

```
$ java -version
java version "1.8.0_31"
Java(TM) SE Runtime Environment (build 1.8.0_31-b13)
Java HotSpot(TM) 64-Bit Server VM (build 25.31-b07, mixed mode)
```

If the `java` command is not found, please make sure that the `<java_home>/bin` directory is in your `PATH` environment variable and the `JAVA_HOME` environment variable is set (see troubleshooting instructions on the Oracle website).

Installation of build tools

Please follow the instructions about how to download and install the Apache Maven tool.

As mentioned before, you can also use the Maven functionality embedded in your favorite IDE.

Integrated development environments

If you prefer working with IDEs to develop or just build the final package, you are free to do it.

The following are just two of the most common IDEs, and of course you can choose the one you prefer.

- Eclipse
- NetBeans

Please, follow the related instructions to download and install your favorite IDE.

Setting up the development environment

To have a functional working environment for Metaware, an Apache Tomcat and MongoDB instances are needed.

Apache Tomcat is a well-known open-source web server and servlet container, which provides a Java HTTP Web server environment. We strongly suggest to follow the instructions from the Tomcat documentation to download and install the Web server; please note that we are referring to the version 7 of Tomcat, but there should be no problem in dealing with version 8.

Another solution consists in installing XAMPP, an open-source cross-platform Web server available for Windows. XAMPP will also install Apache Tomcat on your system.

MongoDB is a well-known document-oriented NoSQL database, and the instructions to download it and install it are available at this web-page.

Build Metaware

Once you have a working environment, you can download the source code of Metaware.

It is very important to mention that Metaware is actually composed by three pieces of software, in order to distinguish the various levels of the application. The first part is `Metaware`, and it is the tier that actually exposes the RESTful APIs web-services. Then, there is the `Metaware-DAO`, which basically defines the various entities inside the Metaware, together with the custom exceptions. Finally, the `Metaware-MongoDBDAOImpl` defines the implementation of the entities specified in `Metaware-DAO`; actually, at this level, a new definition of the entities is done, but this one is more related to MongoDB implementation. If one decides to change MongoDB in favor of other databases, only this last piece of Metaware has to be re-implemented (plus some other minor changes in `Metaware`, but nothing crucial).

The source code for the Metaware's sections are collected in the following GitHub repository: https://github.com/FiwareTIConsoft/fiware-metaware.git/

Using a command line interface, these are some simple instructions to get all the code you need to start working with Metaware:

```
cd ~
mkdir metaware
cd metaware
git clone https://github.com/FiwareTIConsoft/fiware-metaware.git
```

To check that everything is fine, you can use the `tree` command; you should see something like this:

```
$ tree
.
 docker
      Dockerfile
      README.md
 docs
      installation_and_administration_manual.md
      javadocs
          ...
      open_specs
          open_specs_api.apib
      user_and_programming_manual.md
 LICENSE
 Metaware
      pom.xml
      README.md
      src
          main
              java
                  com
                      tilab
                          fiware
                              metaware
                                  ...
              resources
                  ...
              webapp
                  index.html
                  META-INF
                      context.xml
                  WEB-INF
                      web.xml
          test
              ...
 Metaware-DAO
      LICENSE
      pom.xml
      README.md
      src
          main
              java
                  com
                      tilab
                          fiware
                              metaware
                                  ...
          test
              ...
 Metaware-MongoDBDAOImpl
      pom.xml
      README.md
      src
          main
              java
                  com
```

```
                    tilab
                        fiware
                            metaware
                                dao
                                    impls
                                        mongodb
                                            ...
             resources
                 log4j.properties
          test
              ...
pom.xml
README.md

112 directories, 411 files
```

(The number of directories and files may be different due to new features inclusion).

Please, refer to the User and Programming manual for details about the internal architecture of Metaware.

Deploy

In order to deploy the Metaware, you first need to create a WAR package of the application. The final package shall contain all the three parts of Metaware, due to some dependencies between the various parts. To produce the WAR package we suggest to use your favorite IDE, which should include build tools (like Apache Maven), but you are free to use the tool you prefer.

An important note is about testing. For some IDE, while building the WAR package, also the test suite is executed, in order to keep the code working properly; this means that while building, your MongoDB instance has to run properly (some tests require an active connection to the MongoDB instance). A solution consists in removing the tests execution, but we strongly suggest to keep this operation alive.

In Metaware, we use the default port for MongoDB, which we assume it is running on localhost. If this is not your case, you can modify the “Database access information” in the file `fiware-metaware/src/main/resources/metaware.properties` before the creation of the WAR package. This file is extremely important since it specifies all the parameters to connect with MongoDB, so please configure it properly.

Once you get the WAR package from `metaware`, you can deploy it on Tomcat. If the deploy goes fine, the root of Metaware will be available at `http://localhost:8080/metaware/api/v1/`.

Add admin user

Metaware comes with a empty User collection, this means that at the very beginning (almost) none of the APIs will allow the selected operation (`401 Unauthorized` error will return).

In order to really start using Metaware, a first admin user must be manually inserted in the MongoDB. The steps to do so are the following:

1) open the `mongo` shell (or whatever mongo client you prefer)
2) `use MetadataRepo`
3) `db.users.insert({"name" : "admin", surname" : "admin", email" : "", "phone" : "", "company_id" : null, "department_id" : null, "username" : "admin", "password" : "8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918", "role" : "admin"})`

The value `8c6976e5b5410415bde908bd4dee15dfb167a9c873fc4bb8a81f6f2ab448a918` is the Sha256hex of the string “admin”, this means to interact for the first time with Metaware APIs, the credentials to use are “admin” - “admin” (username - password). Of course you can insert whatever username and password you prefer, but keep in mind that the password string is the Sha256Hex of the plain password, and the role field contains the string “admin”.

Sanity check procedures

You can verify that your environment is ready by checking the "responsiveness" of all the needed tools.

Java is automatically detected by Apache Tomcat, so if you are able to retrieve the main page of Tomcat (http://localhost:8080//) this means that Java and Tomcat are fine.

Independently if you are in a Windows or Linux environment, you can check the MongoDB availability by starting the mongod process and connect to it through the MongoDB shell.

The following is the output after launching the `mongod` process:

```
PS C:\Program Files\MongoDB 2.6 Standard\bin> .\mongod.exe
C:\Program Files\MongoDB 2.6 Standard\bin\mongod.exe --help for help and startup options
2015-09-10T12:31:01.632+0200 [initandlisten] MongoDB starting : pid=7916 port=27017 dbpath=\data\db\ 64-bit host=buchs
2015-09-10T12:31:01.643+0200 [initandlisten] targetMinOS: Windows 7/Windows Server 2008 R2
2015-09-10T12:31:01.643+0200 [initandlisten] db version v2.6.7
2015-09-10T12:31:01.643+0200 [initandlisten] git version: a7d57ad27c382de82e9cb93bf983a80fd9ac9899
2015-09-10T12:31:01.643+0200 [initandlisten] build info: windows sys.getwindowsversion(major=6, minor=1, build=7601, platform=2, service_pack='Service
 Pack 1') BOOST_LIB_VERSION=1_49
2015-09-10T12:31:01.643+0200 [initandlisten] allocator: system
2015-09-10T12:31:01.643+0200 [initandlisten] options: {}
2015-09-10T12:31:01.732+0200 [initandlisten] journal dir=\data\db\journal
2015-09-10T12:31:01.732+0200 [initandlisten] recover : no journal files present, no recovery needed
2015-09-10T12:31:02.862+0200 [initandlisten] waiting for connections on port 27017
```

The message `waiting for connections on port 27017` confirms the running status of the process. Please note that the port number showed in this message may be different if you specify another port number while launching the `mongod` process.

The MongoDB Shell returns the following output message if the connection successes:

```
PS C:\Program Files\MongoDB 2.6 Standard\bin> .\mongo.exe
MongoDB shell version: 2.6.7
connecting to: test
>
```

List of running processes

1) java.exe (for Tomcat - Metaware)
2) mongod.exe (for MongoDB)

Network interfaces up and down

If everything is set in a default way, the only opened ports are:

- 8080 for Tomcat;
- 27017 for MongoDB.

Diagnosis procedures

Metaware includes Swagger-UI framework to check and discover the various available RESTful APIs. You can go to `http://localhost:8080/swagger-ui/` and use the APIs that are available for the various entities and actions (i.e., `datasources`, `discoverObjects`, `companies`, `departments`, `users`, `templates`, `datasets`).

Remote service access

No remote service access is planned for Metaware (excluding Tomcat management page, but this is not up to Metaware and only depends on your system administrator).

Docker

Metaware comes with a Dockerfile that can be user to “Dockerize” a Metaware instance. The Dockerfile (locate in the `docker` folder) represents an extension of the official Tomcat Docker image, and indeed the only extra operation that is executed consists in adding the WAR package of Metaware to be deployed. It is important to mention that the Dockerfile (and consequently the Dockerization of Metaware) will not contain the MongoDB instance, which must be instantiated in another Docker container.

The steps to get a functional environment for Metaware through Docker are the following:

1) install Docker (the Get Started can drive you through the first moves with Docker);
2) retrieve the MongoDB official Docker image, the following command downloads the latest MongoDB image directly from the DockerHub:

```
docker pull mongo
```

3) create a container for the MongoDB image and run it by specifying the port matching `27017:27017` and the name `mongodb` of the container (the former to allow MongoDB to be visibile outside of the container, the latter to allow the creation of the link between the MongoDB container and the Metaware Web application):

```
docker run -d -p 27017:27017 --name mongodb mongo
```

4) once you have a running instance of MongoDB, you can proceed with Metaware by using the proper Docker file, located in the `docker` folder (`fiware-metaware/docker/Dockerfile`);

```
cd fiware-metaware/docker
```

5) build the Metaware image by using the Dockerfile

```
docker build -t metaware/tomcat .
```

6) check that the Metaware Docker image is available (`docker images` command, you should see an image called `metaware/tomcat`), then instantiate a Docker container for the Metaware; the following command is extremely important since the port matching and the link with MongoDB must be executed properly (the link with MongoDB is done by using the option `-link` of Docker `run` command, followed by the name of the container to be linked, `mongodb` in this case):

```
docker run --name tomcat --link mongodb:mongodb -p 8080:8080 metaware/tomcat
```

At this point you can start interacting with Metaware (remember to add an initial admin user). In case you are running the Docker environment in Windows, keep in mind that most probably the IP address will be something similar to 192.168.99.100 (not `localhost`).