

Yugandhar Open MDM Hub for Embedded Web Server (EWS)

Development Environment Setup Guide

Yugandhar Open MDM Hub - EWS Release - V1.0.0

Date – 11/06/2018

+++++

Copyright [2017] [Yugandhar Open MDM Hub]

Licensed under the Apache License, Version 2.0 (the "License");

you may not use this file except in compliance with the License.

You may obtain a copy of the License at

<http://www.apache.org/licenses/LICENSE-2.0>

Unless required by applicable law or agreed to in writing, software

distributed under the License is distributed on an "AS IS" BASIS,

WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.

See the License for the specific language governing permissions and

limitations under the License.

Contents

About Yugandhar Open MDM Hub.....	4
About this document	4
System Requirements	5
Software Download links	5
Eclipse	5
Apache maven	5
Database	6
Oracle.....	6
MariaDB	6
Java JDK.....	6
Hibernate Tools.....	6
Oracle SQL Developer	6
Heidi SQL.....	6
Database drivers	7
Setup Database	7
Oracle setup:.....	7
Oracle Install	7
Yugandhar MDM HUB schema setup for oracle	7
MariaDB setup:	8
MariaDB Install.....	8
Yugandhar MDM HUB schema setup for MariaDB.....	8
Setup Workspace	10
Setup up Network Connections	12
Eclipse network.....	12
Maven Settings	12
Set JDK Path	15
Import Yugandhar MDM Hub java projects in the workspace	19
Properties file changes.....	21
application.properties.....	21
yugandhar-mdmhub-app.properties	21
Running the application.....	23

TEST With SOAPUI.....	24
Test JSON message	24
Test XML message.....	26

About Yugandhar Open MDM Hub

Master Data Management came a long way in last decade or so. There are currently more than 20 MDM solutions catering to various specializations of MDM like Customer Data Integration (CDI), Product Information Management (PIM), vendor and supplier management etc. However most of these solutions come with licensing costs amounting to thousands of dollar. To offer a completely free solution which would be made available through Apache 2.0 license, A Project is started in 2017 under the name 'Yugandhar Open MDM Project' to build Open Source MDM solutions catering to CDI, PIM and Data Governance Capabilities. Yugandhar in Sanskrit means Ever Lasting and the strongest of its time. Our vision is to build the strongest, Open Source, Multi Domain, Cross Industry and completely free MDM Solution.

We are happy to announce that the first release of the Yugandhar MDM Hub catering to CDI solution is built with Open source technologies like Spring and Hibernate etc, inbuilt data Model, 400+ ready to use services and having incredible Out of the Box capabilities is currently being distributed. We aim to make the current CDI offering the strongest and Planning to bring Data Stewardship and PIM solutions in upcoming years.

About this document

This document covers the system requirements for Yugandhar Open MDM Hub.

System Requirements

Below are the System Requirements for setting up Development Environment

1. OS – Windows 7 enterprise edition, Service Pack 1 or later
2. 8GB RAM and 100 GB Storage
3. Eclipse Java EE IDE for Web Developers. Oxygen.3a Release (4.7.3a) or later
4. apache-maven-3.5.0 or later
5. Java jdk 1.8 (jdk1.8.0_121) or later
6. Spring Boot 2.0.2.RELEASE
7. Jboss (Hibernate) Tools
8. Databases
 - a. Oracle Database 11g Release 11.2.0.2.0 or later OR
 - b. Oracle 12c OR
 - c. MariaDB v10.3.x
9. Oracle SQL Developer/HeidiSQL
10. SOAPUI /postman or any other tool to test REST services

Note - Internet connectivity is needed to setup workspace. If internet is not available because of any reason then all the software and Maven jars needs to be manually downloaded which would be tedious task.

IMPORTANT NOTICE - Please read the licensing terms of all the above listed software before using for commercial as well as non-commercial purpose. Yugandhar team would not be responsible for licensing violations if any.

Software Download links

Eclipse

Eclipse Java EE IDE for Web Developers.

Version - Oxygen.3a Release (4.7.3a) or later

Download page - <http://www.eclipse.org/downloads/>

Apache maven

Apache Maven comes integrated with Eclipse Neon but if you want to install standalone maven then you may download it from below path

Version - apache-maven-3.5.0 or later

Download Page - <https://maven.apache.org/download.cgi>

Database

Download the database as per your choice

Oracle

Download Oracle from Oracle download site

Version - Oracle Database 11g Enterprise Edition Release 11.2.0.3.0 - 64bit OR Oracle Database 12c

Download link - <https://www.oracle.com/database>

MariaDB

Version: MariaDB v10.3 or later

<https://mariadb.org/>

Java JDK

Download Java jdk 1.8 (jdk1.8.0_121) or later from below link

<http://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

Hibernate Tools

You may choose to download the Hibernate Tools (Now renamed as JBoss Tools) from the link below. To install the plugin from eclipse market place using eclipse Menu → Help > Eclipse Marketplace... option.

<http://tools.jboss.org/downloads/jbosstools/oxygen/4.5.1.Final.html#marketplace>

Oracle SQL Developer

Download SQL developer to connect to database

<http://www.oracle.com/technetwork/developer-tools/sql-developer/downloads/index.html>

Heidi SQL

Heidi SQL comes packaged along with Maria DB installable; you may use the same to explore maria db.

Database drivers

- Oracle JDBC drivers: download ojdbc14.jar from below link
<http://www.oracle.com/technetwork/apps-tech/jdbc-10201-088211.html>
- Oracle JDBC drivers: Oracle 11g and 12c driver for JDK7 and JDK8 ojdbc7.jar
<http://www.oracle.com/technetwork/database/features/jdbc/jdbc-drivers-12c-download-1958347.html>
- **MariaDB drivers:** Download mariadb-java-client-2.2.3.jar from below location
<https://mariadb.com/downloads/connector>
<https://downloads.mariadb.com/Connectors/java/connector-java-2.2.3/mariadb-java-client-2.2.3.jar>

Setup Database

Install either Oracle or mariaDB database as per requirement. Oracle 11g/12c/MariaDB 5.5.3 are supported. The step by step installation instructions for installing the Oracle 11g, 12c database and Oracle SQL Developer is out of scope of this document.

By Default Yugandhar Open MDM Hub uses the schema YUG_OWNER. If different user name (schema name) is needed then modify all the scripts with required schema name.

Oracle setup:

Oracle Install

Install the Oracle database using the instructions mentioned below.

Oracle 11g: https://docs.oracle.com/cd/E11882_01/nav/portal_11.htm

Oracle 12c:

http://www.oracle.com/webfolder/technetwork/tutorials/obe/db/12c/r1/Windows_DB_Install_OBE/Installing_Oracle_Db12c_Windows.html

Yugandhar MDM HUB schema setup for oracle

Github repository link - <https://github.com/yugandharproject/yugandhar-open-mdmhub>

Download the scripts from '<gitrepo>/resources/yugmdm-dbsetupscripts-oracle'

1. CreateTablespaces.sql – create YUG_OWNER user as the tables. Needs DBA access
2. FullSchema.sql – creates the full schema. Can be executed with YUG_OWNER user access
3. CreateSequence.sql – Can be executed with YUG_OWNER user access
4. LoadTableDataWrapper.sql – Can be executed with YUG_OWNER user access

Verify the logs and check that all the steps are executed correctly and REF_xxx as well as CONFIG_xxx tables are loaded with sample data. In Summary, below mentioned objects are created in database

Table spaces –

MDM_DATATS – used for data and reference tables

MDM_INDXTS – used for Indexes

Profile:

MDM_PROFILE - Used to create YUG_OWNER user

User Schema:

YUG_OWNER – Default user Schema used by Yugandhar Open MDM Hub.

MariaDB setup:

MariaDB Install

Install MariaDB as per instructions mentioned in below links

<https://mariadb.com/kb/en/library/getting-installing-and-upgrading-mariadb/>

<https://mariadb.com/products/get-started>

Yugandhar MDM HUB schema setup for MariaDB

Download the scripts from github repository resources\dbsetupscripts location and execute the below scripts in sequence

Github repository link - <https://github.com/yugandharproject/yugandhar-open-mdmhub>

Download the sqls from '<gitrepo>/resources/yugmdm-dbsetupscripts-mariadb'

- "1.yugmdm_mariadb_createuser-and-database.sql" – login through root or DBA

Verify the logs and check that all the steps are executed correctly. Also verify that REF_xxx as well as CONFIG_xxx tables are loaded with sample data. In Summary, below mentioned objects to the TABLES, Sequence and INDEXES are created in database

Table spaces:

MDM HUB_DATATS – used for data tables as well as reference tables

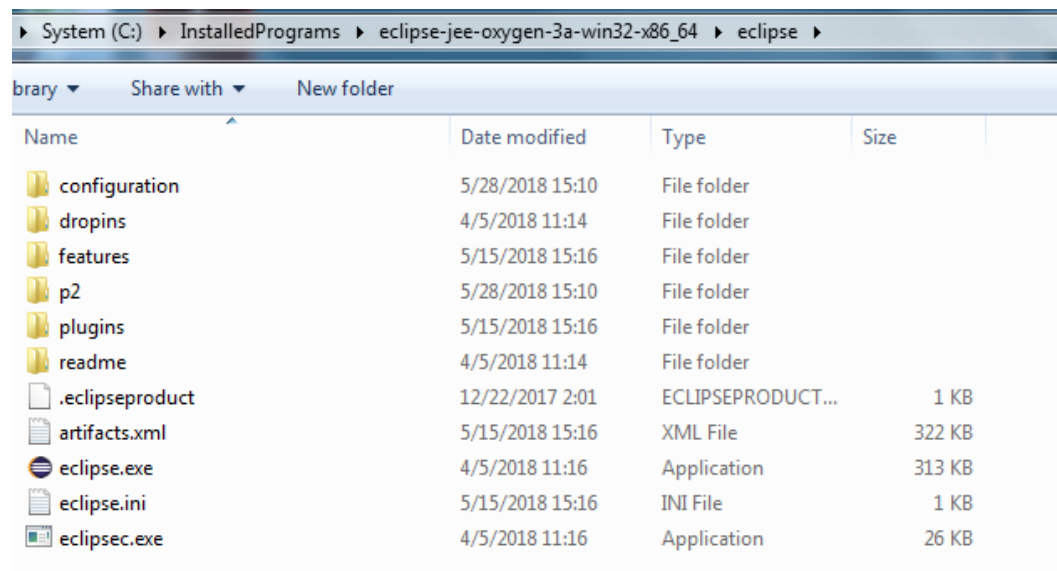
MDM HUB_INDXTS – used for Indexes

DB User YUG_OWNER: Default user used by Yugandhar Open MDM Hub to connect to database.

Database: YUG_OWNER database is created.

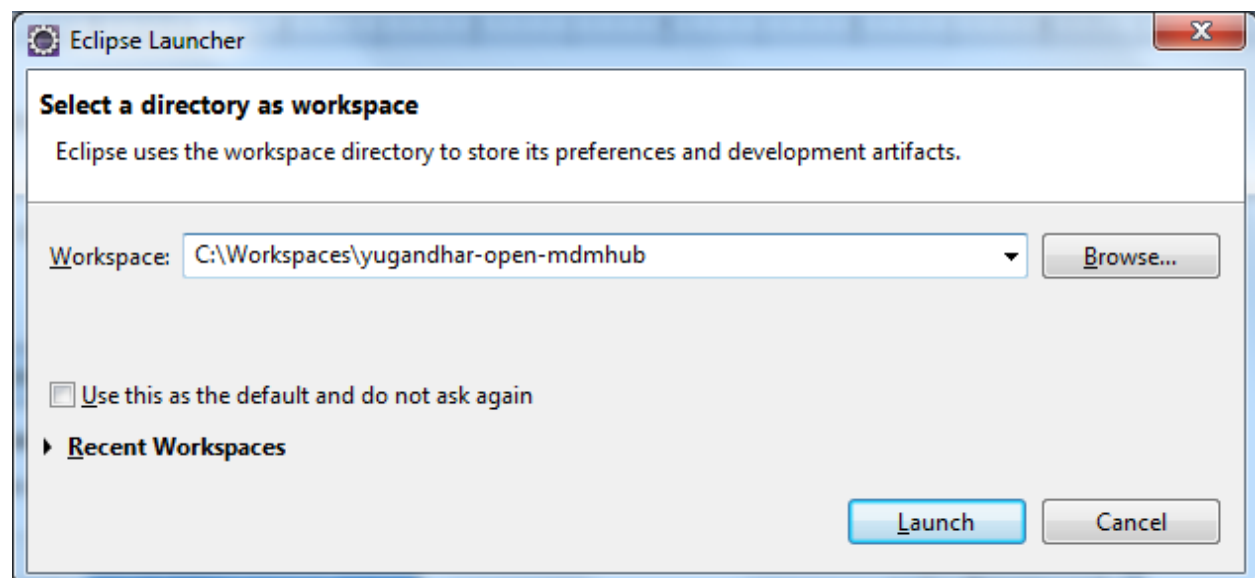
Setup Workspace

Extract the downloaded eclipse archive in any of the folder of your choice.

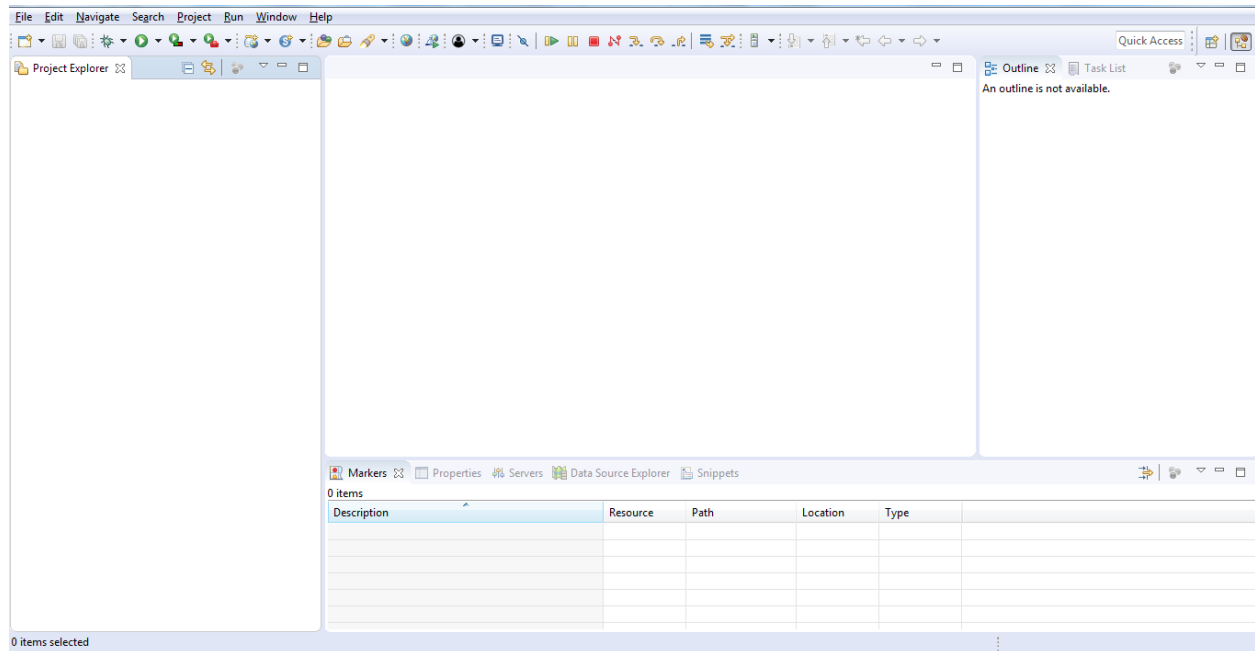


The extracted folder would have above files, click on eclipse.exe to start the eclipse IDE.

The next window would ask you for the workspace location, provide the location as per your choice, for this documentation purpose we are creating workspace in C:\workspaces\yugandhar-MDM Hub folder.



Your workspace is ready to be configured now.



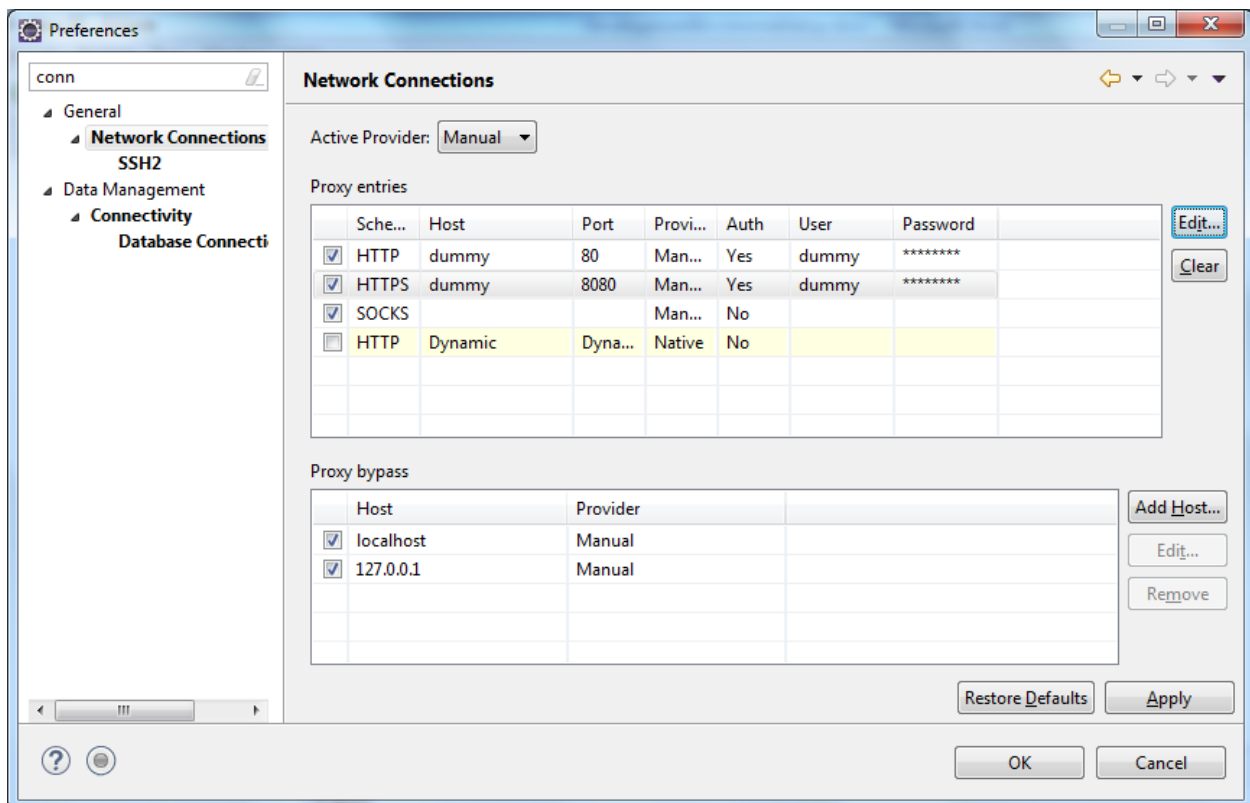
Setup up Network Connections

If you are behind firewall then you need to perform below steps. Those who have direct internet connection (without proxy) can skip the eclipse Network and Maven Settings step.

Eclipse network

Go to Windows -- > preferences → General → Network Connections

Change the Active Provider to manual and set the proxy host, port and user credentials as per your firewall settings. The below settings are dummy so should not be copied as-is. Click on apply and click OK.



Maven Settings

Create a simple text file named MavenSettings.xml in workspace having below content.

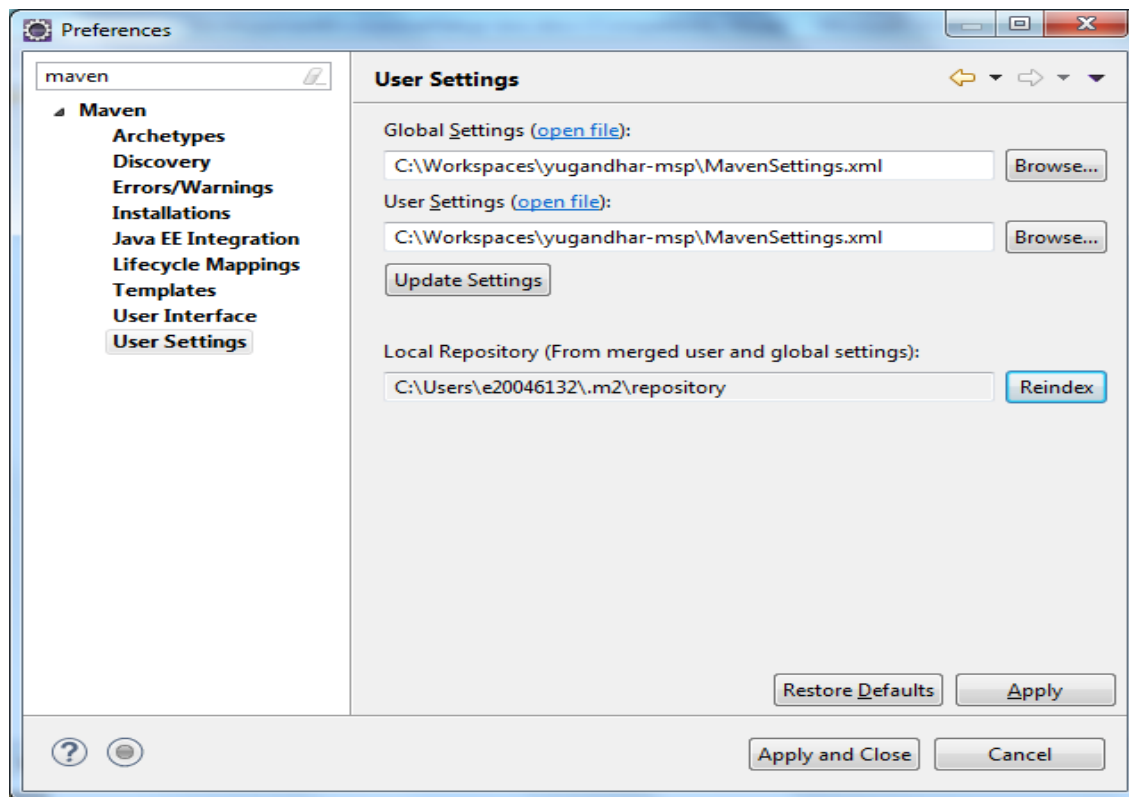
You may also download sample MavenSettings.xml from git hub resources/mavensettings folder.

```
<settings xmlns="http://maven.apache.org/SETTINGS/1.0.0"
```

```
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/SETTINGS/1.0.0
                    http://maven.apache.org/xsd/settings-1.0.0.xsd">
<localRepository>C:\Users\<UserName>\.m2\repository</localRepository>
<interactiveMode/>
<usePluginRegistry/>
<offline/>
<pluginGroups/>
<servers/>
<mirrors/>
<proxies>
  <proxy>
    <id>myproxy</id>
    <active>true</active>
    <protocol>http</protocol>
    <host>dummy</host>
    <port> dummy </port>
    <username>dummy</username>
    <password>dummy</password>
    <nonProxyHosts>localhost,127.0.0.1</nonProxyHosts>
  </proxy>
</proxies>
<profiles/>
<activeProfiles/>
</settings>
```

Note – Make sure to change the host, port, username and password as per your firewall settings. Also change the localRepository to users folder.

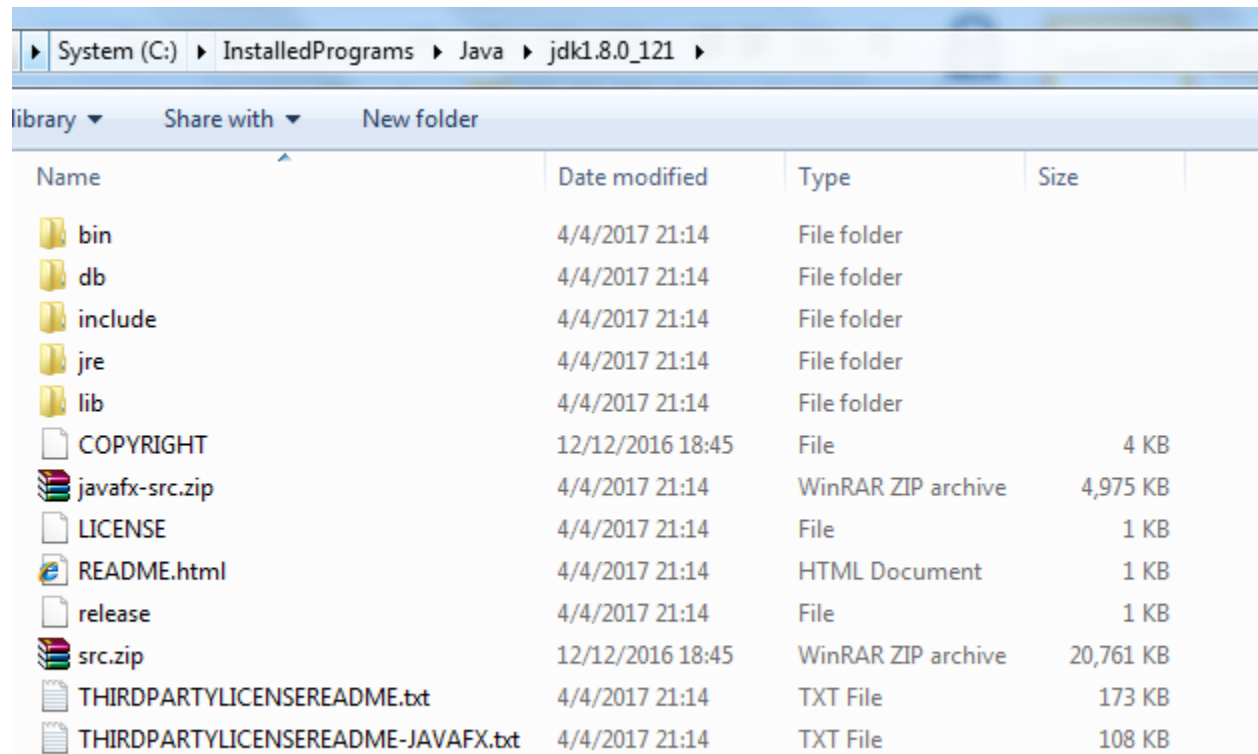
Now go to Eclipse Menu → windows → preferences → maven → User Settings → provide path of the file MavenSettings.xml in the local and global settings as shown in screenshot below



Set JDK Path

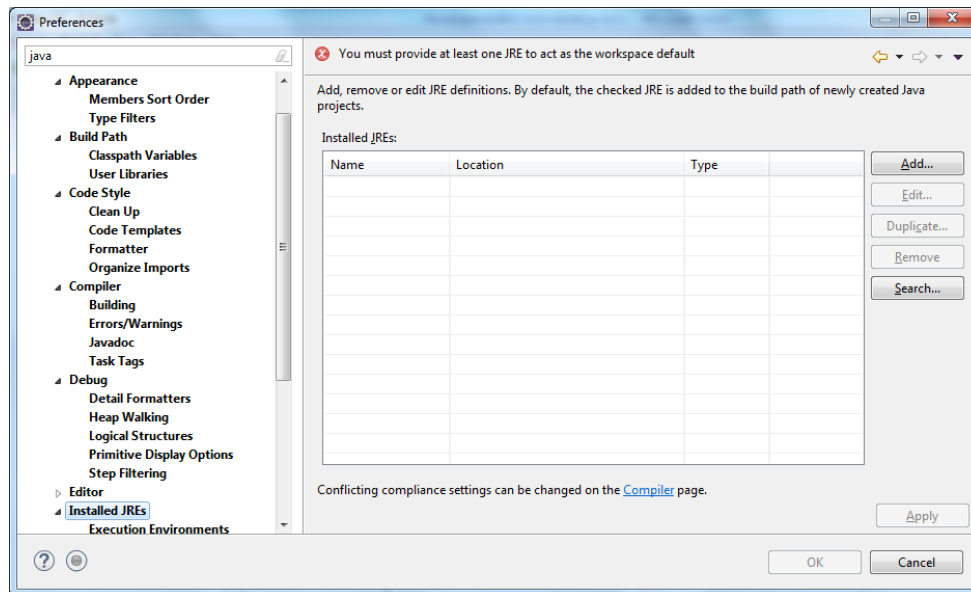
Extract the JDK in a folder of your choice, for the document purpose the jdk directory is

C:\InstalledPrograms\Java\jdk1.8.0_121. The directory should look something like below

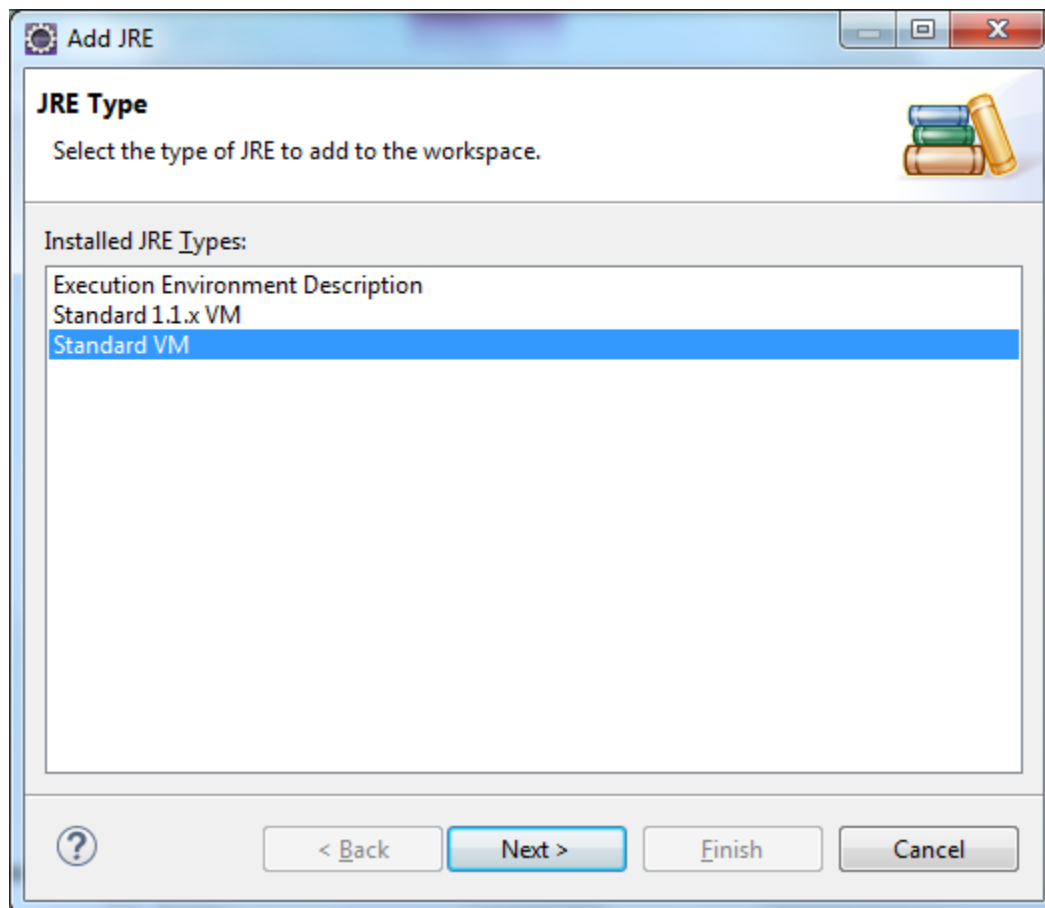


System (C:) > InstalledPrograms > Java > jdk1.8.0_121				
library ▾ Share with ▾ New folder				
Name	Date modified	Type	Size	
bin	4/4/2017 21:14	File folder		
db	4/4/2017 21:14	File folder		
include	4/4/2017 21:14	File folder		
jre	4/4/2017 21:14	File folder		
lib	4/4/2017 21:14	File folder		
COPYRIGHT	12/12/2016 18:45	File	4 KB	
javafx-src.zip	4/4/2017 21:14	WinRAR ZIP archive	4,975 KB	
LICENSE	4/4/2017 21:14	File	1 KB	
README.html	4/4/2017 21:14	HTML Document	1 KB	
release	4/4/2017 21:14	File	1 KB	
src.zip	12/12/2016 18:45	WinRAR ZIP archive	20,761 KB	
THIRDPARTYLICENSEREADME.txt	4/4/2017 21:14	TXT File	173 KB	
THIRDPARTYLICENSEREADME-JAVAFX.txt	4/4/2017 21:14	TXT File	108 KB	

Go to eclipse Menu → Windows → Preferences → Java → Installed JREs

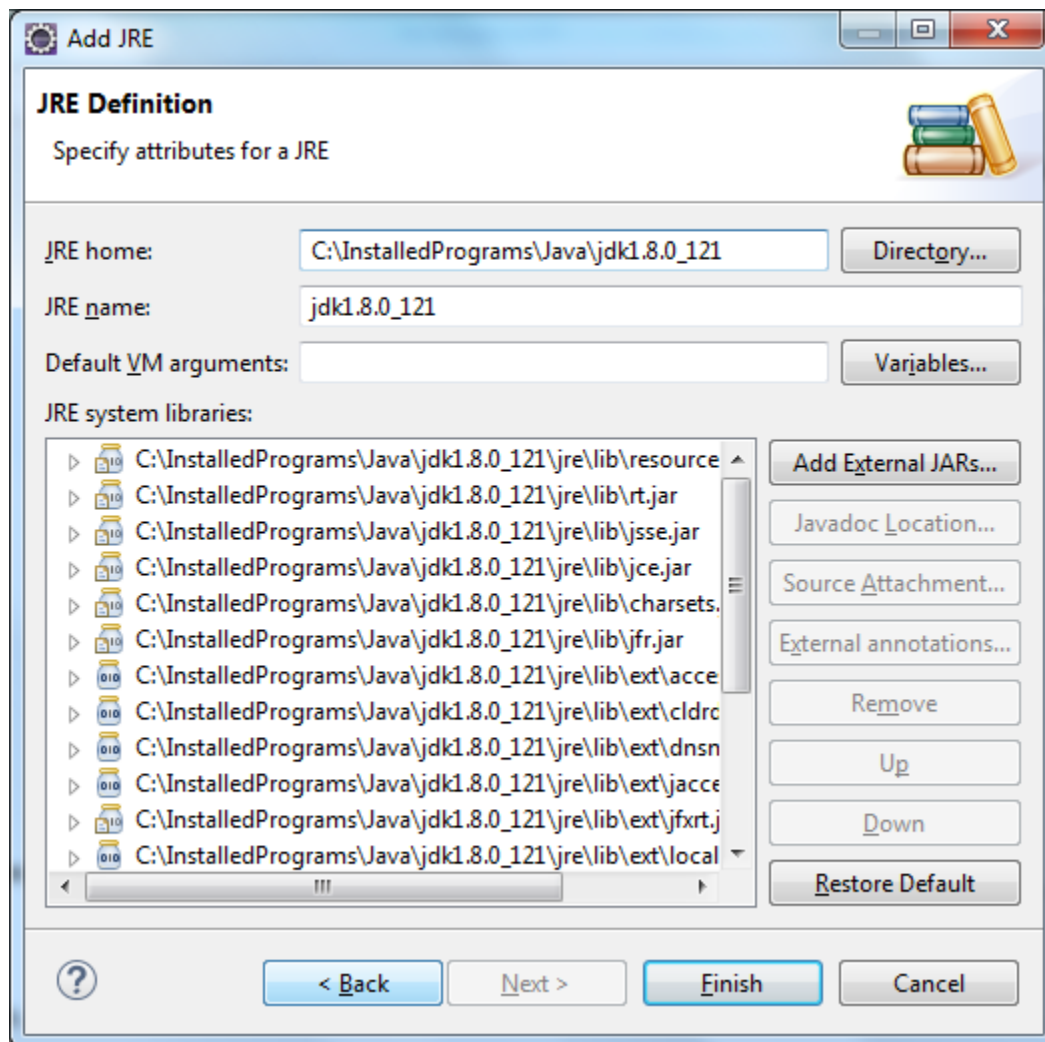


Click Add

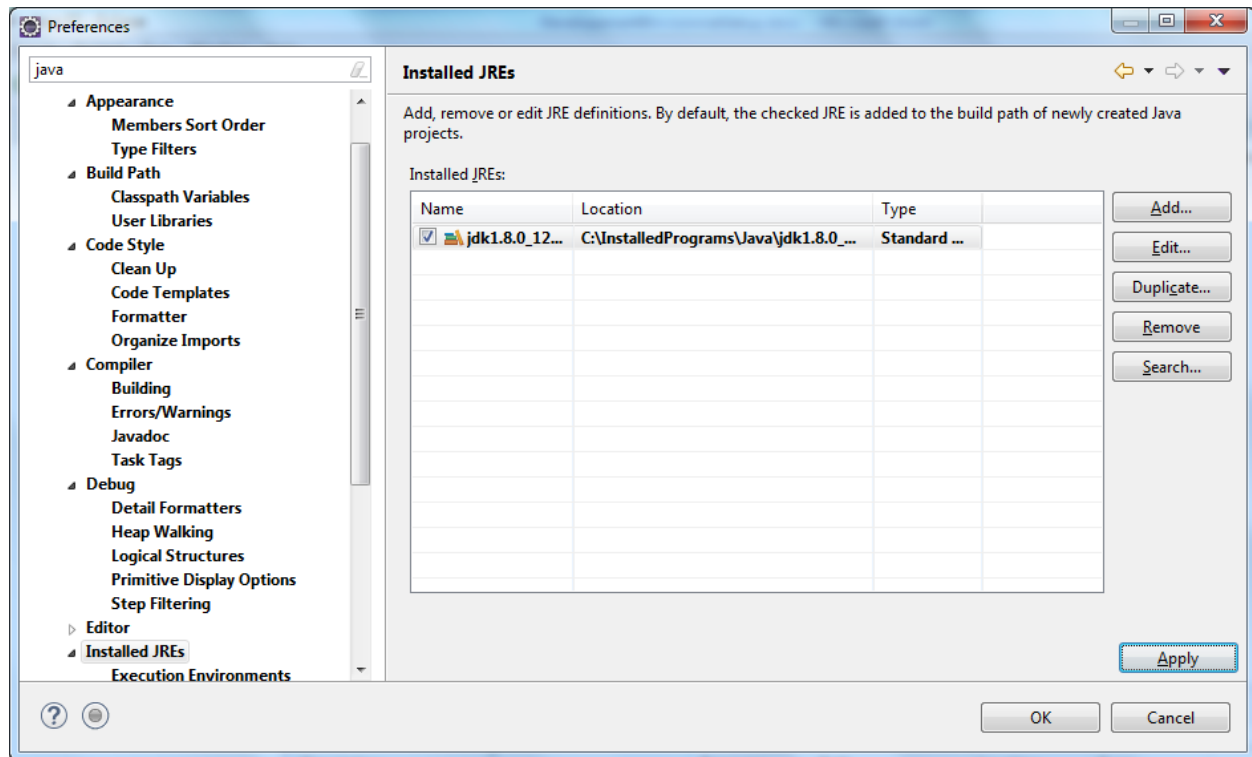


Click Next

Provide the JDK folder name where JDK is extracted



Click finish



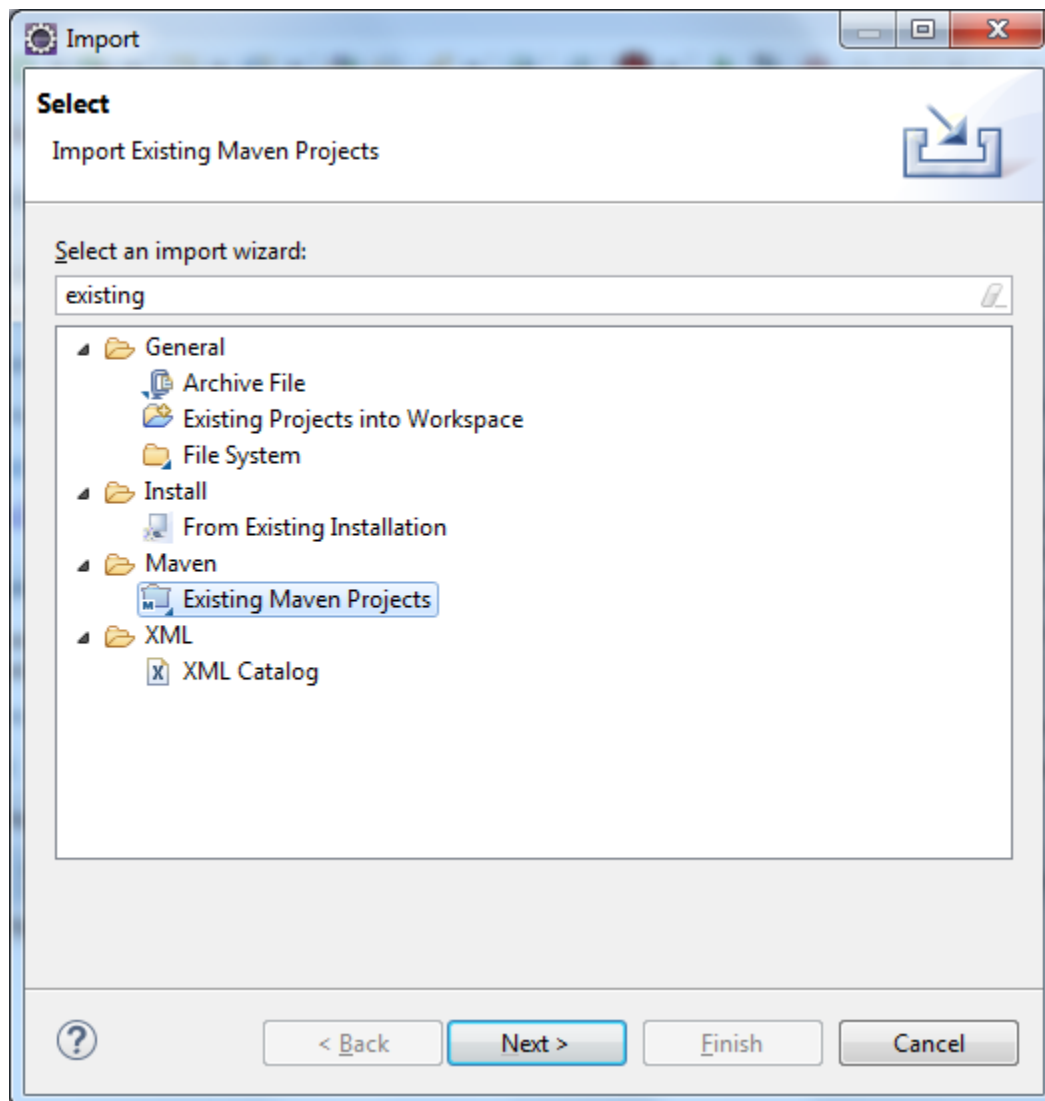
Check the textbox against recently added JDK, click Apply and OK.

Import Yugandhar MDM Hub java projects in the workspace

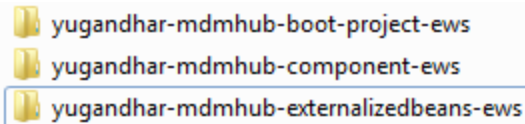
Download the yugandhar-open-mdmhub repository, go to the yugandhar-open-mdmhub-ews directory and copy the below three projects in the workspace directory (e.g. C:\Workspaces\yugandhar-open-mdmhub) and import in Workspace

- yugandhar-mdmhub-boot-project-ews
- yugandhar-mdmhub-component-ews
- yugandhar-mdmhub-externalizedbeans-ews

Go to File → Import Menu → Existing Maven Projects →

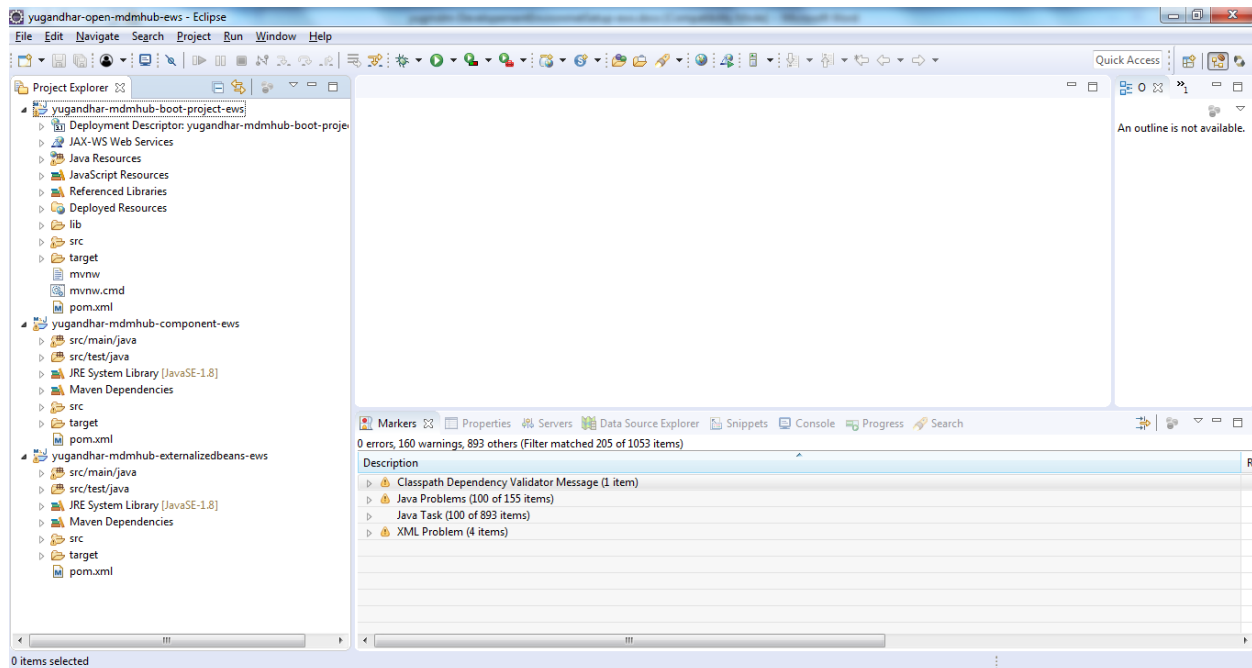


Select the below three projects which must be visible on the next screen and click Finish



It may take some time as eclipse will download the maven jars automatically and build the project. You may track the progress in 'Progress' tab.

Make sure that your workspace is error free



Open MDM Hub does the logging to default folder C:/Yugandhar/logs so create this folder or change the log directory to the directory of your choice in /yugandhar-mdmhub-boot-project-ews/src/main/resources/yugandhar_logback.xml

```
<appender name="TIME_AND_SIZE_BASED_APPENDER" class="ch.qos.logback.core.rolling.RollingFileAppender">
  <file>C:/yugandhar/logs/YugandharCommon.log</file>
  <rollingPolicy class="ch.qos.logback.core.rolling.TimeBasedRollingPolicy">
    <fileNamePattern>C:/yugandhar/logs/YugandharCommon.%d{yyyy-MM-dd}.%i.log</fileNamePattern>
    <timeBasedFileNamingAndTriggeringPolicy class="ch.qos.logback.core.rolling.SizeAndTimeBasedFNATP">
```

Properties file changes

There are below two properties files in the Microservice platform

application.properties

The application properties file covers the below properties

- **Springboot trace:** enable/disable the trace
- **Server port:** set the port number for the tomcat server
- **JPA:** If the generated ddl needs to be logged then enable the property `spring.jpa.show-sql`. By default this is enabled.
- **mariaDB specific settings:**
#Enable both the below properties for mysql/MariaDB, if you are using oracle then comment both the properties.
#spring.jpa.properties.hibernate.globally_quoted_identifiers=true
#spring.jpa.database-platform=org.hibernate.dialect.MariaDB53Dialect
Note-
"spring.jpa.properties.hibernate.globally_quoted_identifiers=true" with Oracle database may result in errors.
- **Oracle specific settings:**
Oracle Specific configuration, use 10g dialect for Oracle 11g database else use 12c
#spring.jpa.database-platform=org.hibernate.dialect.Oracle10gDialect
#spring.jpa.database-platform=org.hibernate.dialect.Oracle12cDialect
- **Logging:** Logback configuration
- **JTA:** Atomikos is the default JTA provider being used by Yugandhar MDM Hub, change the properties as needed.
- **Ehcache:** ehcache properties
- **Json:** json parser related properties
- **Active mq:** active MQ properties
- **Actuator:** spring boot actuator properties
- **Eureka integration:** Eureka integration properties, by default it's disabled.

yugandhar-mdmhub-app.properties

The yugandhar-mdmhub-app.properties file is custom properties file having below properties

- **mariaDB specific settings:**
enable the below properties for mariaDB else comment the same

```
# Datasource for Maria DB
yugandhar.mdm.datasource.url=
```

```
jdbc:mariadb://localhost:3306/YUG_OWNER?pinGlobalTxToPhysicalConnection=true
yugandhar.mdm.datasource.driver-class-name=org.mariadb.jdbc.Driver
yugandhar.mdm.datasource.xa.data-source-class-
name=org.mariadb.jdbc.MariaDbDataSource
```

- **Oracle specific settings:**

Enable the below properties for Oracle else comment the same

```
# Datasource for Oracle
yugandhar.mdm.datasource.url= jdbc:oracle:thin:@localhost:1521/MDMDB
yugandhar.mdm.datasource.driver-class-name=oracle.jdbc.driver.OracleDriver
yugandhar.mdm.datasource.xa.data-source-class-
name=oracle.jdbc.xa.client.OracleXADataSource
```

- **Common Properties:**

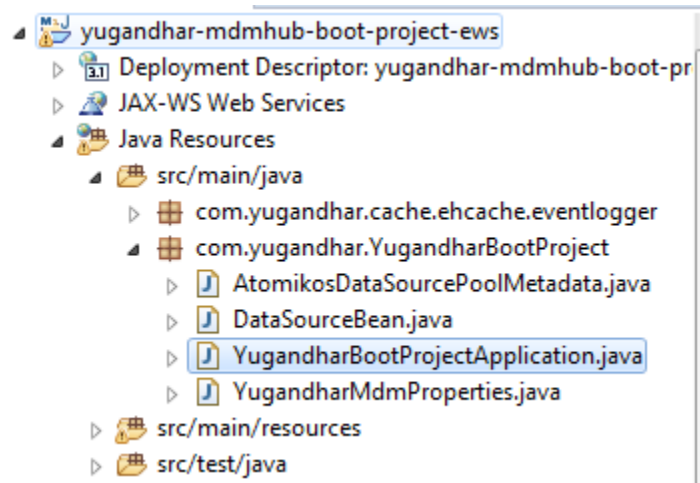
Use either of plain text properties or encrypted properties, if plain text properties provided then encrypted properties will be ignored. Use YugandharEncoderDecoder utility class to encode/decode a given string, refer Encryption Utility section of Development and Customization Guide to understand more.

```
yugandhar.mdm.datasource.username.plaintext=user name e.g. YUG_OWNER
yugandhar.mdm.datasource.password.plaintext=user name e.g. YUG_OWNER
#yugandhar.mdm.datasource.username.encrypted=WVVHX01TUF9PV05FUg
#yugandhar.mdm.datasource.password.encrypted=WVVHX01TUF9PV05FUg
```

Running the application

Once all the configuration is finished then running the application is fairly simple

Go to /yugandhar-mdmhub-boot-project-ews/src/main/java/com/yugandhar/YugandharBootProject/YugandharBootProjectApplication.java



Right click and select 'Run As..' → java application. It will start the Yugandhar MDM Hub application with embedded Tomcat server. Check the logs lines below to verify the application is started.

```
2018-06-11 21:54:51,656 [main] [MBeanExporter.java:895] Bean with name 'refreshScope' has been autodetected for JMX exposure
2018-06-11 21:54:51,658 [main] [MBeanExporter.java:895] Bean with name 'configurationPropertiesRebinder' has been autodetected for JMX exposure
2018-06-11 21:54:51,663 [main] [MBeanExporter.java:675] Located managed bean 'environmentManager': registering with JMX server as MBean [org.springframework.jmx.support.MBeanExporter]
2018-06-11 21:54:51,681 [main] [MBeanExporter.java:675] Located managed bean 'refreshScope': registering with JMX server as MBean [org.springframework.jmx.support.MBeanExporter]
2018-06-11 21:54:51,705 [main] [MBeanExporter.java:675] Located managed bean 'configurationPropertiesRebinder': registering with JMX server as MBean [org.springframework.jmx.support.MBeanExporter]
2018-06-11 21:54:51,731 [main] [DefaultLifecycleProcessor.java:351] Starting beans in phase 2147483647
2018-06-11 21:54:51,756 [main] [DirectJDKLog.java:180] Starting ProtocolHandler ["http-nio-8091"]
2018-06-11 21:54:51,776 [main] [DirectJDKLog.java:180] Using a shared selector for servlet write/read
2018-06-11 21:54:51,822 [main] [TomcatWebServer.java:206] Tomcat started on port(s): 8091 (http) with context path ''
2018-06-11 21:54:51,829 [main] [StartupInfoLogger.java:59] Started YugandharBootProjectApplication in 37.173 seconds (JVM running for 38.102)
```

TEST With SOAPUI

Test JSON message

The rest url is as below.

<http://localhost:8091/rest/YugandharRequestProcessor>

Yugandhar MDM HUB uses the port 8091 as default; you may change it through application.properties file.

Sample json message

```
{
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "refCountryIsoDO": {
      "configLanguageCodeKey": "1"
    }
  }
}
```

For the soap xml message, add the below headers for the request xmls

Header	Value	use
Accept	application/json	This header tells yugandhar rest controller that the response must be sent in json format.
Content-Type	application/json	This header tells yugandhar rest controller that the request message is of type json

Create 'New REST service from URI' in the soapui project, and execute it with attached json message

Request 1

Method: POST Endpoint: http://localhost:8091 Resource: /rest/YugandharRequestProcessor

Name	Value	Style	Level
Required:	<input type="checkbox"/> Sets if parameter is required		
Type:			
Media Type	application/ison	<input type="checkbox"/> Post QueryString	

```

{
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "refCountryIsoDO": {
      "configLanguageCodeKey": "1"
    }
  }
}

```

Header	Value
Content-Type	application/json
Accept	application/json

Auth Headers (2) Attachments (0) Representations (1) JMS Headers JMS Property (0)

Check the response as SUCCESS

Request 1

Method: POST Endpoint: http://localhost:8091 Resource: /rest/YugandharRequestProcessor

Name	Value	Style	Level
Required:	<input type="checkbox"/> Sets if parameter is required		
Type:			
Media Type	application/ison	<input type="checkbox"/> Post QueryString	

```

{
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "refCountryIsoDO": {
      "configLanguageCodeKey": "1"
    }
  }
}

```

Header	Value
Content-Type	application/json
Accept	application/json

```

1 {
2   "responseCode": "SUCCESS",
3   "txnHeader": {
4     "requesterLanguage": "1",
5     "userName": "admin",
6     "userRole": "admin",
7     "txnMessageId": "12312311115999",
8     "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase",
9     "totalExecutionTimeMillies": "254"
10  },
11  "txnPayload": {
12    "paginationIndexOfCurrentSlice": 2,
13    "paginationPageSize": 25,
14    "paginationInfoElementsOnCurrentSlice": 25,
15    "paginationInfoTotalElements": 247,
16    "paginationInfoTotalPages": 10,
17    "refCountryIsoDOList": [
18      {
19        "idPk": "144",
20        "version": 0,
21        "createdTs": "2018-04-16T12:57:17.000+0000",
22        "updatedTs": "2018-04-16T12:57:17.000+0000",
23        "updatedByUser": "admin",
24        "updatedByTxnId": "000000",
25        "configLanguageCodeKey": "1",
26        "key": "500",
27        "value": "USA"
28      }
29    ]
30  }
31 }

```

Test XML message

The rest url is as below.

<http://localhost:8091/rest/YugandharRequestProcessor>

Yugandhar MDM HUB uses the port 8091 as default; you may change it through application.properties file.

For the soap xml message, add the below headers for the request xmls

Header	Value	use
Accept	application/xml	This header tells yugandhar rest controller that the response must be sent in xml format.
Content-Type	application/xml	This header tells yugandhar rest controller that the request message is of type xml

Sample XML message:

```
<TxnTransferObj>
  <txnHeader>
    <requesterLanguage>1</requesterLanguage>
    <userName>admin</userName>
    <userRole>admin</userRole>
    <txnMessageId>12312311115999</txnMessageId>

  <transactionServiceName>findAllRefCountryIsoByLanguageCodeBase</transactionServiceName>
</txnHeader>
<txnPayload>
  <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
  <paginationPageSize>50</paginationPageSize>
  <refCountryIsoDO>
    <configLanguageCodeKey>1</configLanguageCodeKey>
  </refCountryIsoDO>
</txnPayload>
</TxnTransferObj>
```

Request 1

Method

POST

Endpoint

http://localhost:8091

Resource

/rest/YugandharRequestProcessor

Param

Request

Raw

Name

Value

Style

Level

Required:

☐

Sets if parameter is required

Type:

Media Type

application/xml

☐

Post QueryString

```

<TxnTransferObj>
  <txnHeader>
    <requesterLanguage>1</requesterLanguage>
    <userName>admin</userName>
    <userRole>admin</userRole>
    <txnMessageId>12312311115999</txnMessageId>
    <transactionServiceName>findAllRefCountryIsoByLanguageCodeBase</transactionServiceName>
  </txnHeader>
  <txnPayload>
    <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
    <paginationPageSize>50</paginationPageSize>
    <refCountryIsoDO>

```

Header

Value

Content-Type

application/xml

Accept

application/xml

Auth

Headers (2)

Attachments (0)

Representations (2)

JMS Headers

JMS Property (0)

Check the success response.

Request 1

Method: POST, Endpoint: http://localhost:8091, Resource: /rest/YugandharRequestProcessor

Media Type: application/xml

Request Headers:

Header	Value
Content-Type	application/xml
Accept	application/xml

Response (XML):

```

<TxnTransferObj>
  <txnHeader>
    <requesterLanguage>1</requesterLanguage>
    <userName>admin</userName>
    <userRole>admin</userRole>
    <txnMessageId>12312311115999</txnMessageId>
    <transactionServiceName>findAllRefCountryIsoByLanguageCo
  </txnHeader>
  <txnPayload>
    <paginationIndexOfCurrentSlice>1</paginationIndexOfCurre
    <paginationPageSize>50</paginationPageSize>
    <paginationInfoElementsOnCurrentSlice>50</paginationInfo
    <paginationInfoTotalElements>247</paginationInfoTotalEle
    <paginationInfoTotalPages>5</paginationInfoTotalPages>
    <refCountryIsoDOList>
      <idPk>144</idPk>
      <version>0</version>
      <createdTs>2018-04-16T12.57.17.000+0000</createdTs>
      <updatedTs>2018-04-16T12.57.17.000+0000</updatedTs>
      <updatedByUser>admin</updatedByUser>
      <updatedByTxnId>000000</updatedByTxnId>
      <configLanguageCodeKey>1</configLanguageCodeKey>
      <key>500</key>
      <value>MSR</value>
    </refCountryIsoDOList>
  </txnPayload>
</TxnTransferObj>
  
```

This certifies your workspace.

You may test a few more transactions like createLegalentity, createLeAccount etc for which sample messages are provided in the resources/Testing folder.

Go ahead with Development and customization guide, API Transaction reference guide and Code generation guide to understand more.