

Yugandhar Microservice Platform for Embedded Web Server (EWS)

Development Environment Setup Guide

Yugandhar Microservice Platform - EWS Release - V1.0.0
Date – 23/05/2018

+++++

Copyright [2017] [Yugandhar Microservice Platform]

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 Project..... | 4 |
| About this document..... | 4 |
| System Requirements | 5 |
| Software Download links..... | 6 |
| Eclipse..... | 6 |
| Apache maven..... | 6 |
| Database..... | 6 |
| Oracle..... | 6 |
| MariaDB | 6 |
| Java JDK | 6 |
| Hibernate Tools | 7 |
| Oracle SQL Developer..... | 7 |
| Heidi SQL..... | 7 |
| Database drivers | 7 |
| Setup Database..... | 7 |
| Oracle setup: | 8 |
| Oracle Install..... | 8 |
| Yugandhar MSP schema setup for oracle..... | 8 |
| MariaDB setup: | 8 |
| MariaDB Install..... | 8 |
| Yugandhar MSP schema setup for MariaDB..... | 9 |
| Setup Workspace..... | 10 |
| Setup up Network Connections..... | 12 |
| Eclipse network | 12 |
| Maven Settings..... | 12 |
| Set JDK Path | 15 |
| Import Yugandhar msp java projects in the workspace | 19 |
| Properties file changes | 22 |
| application.properties | 22 |
| yugandhar-msp.properties | 22 |

| | |
|-------------------------------|----|
| Running the application | 24 |
| TEST With SOAPUI..... | 25 |
| Test JSON message..... | 25 |
| Test XML message | 27 |

About Yugandhar Project

The Yugandhar Project is the umbrella project focused on building open source cloud ready solutions. The current offerings include Yugandhar Microservice Platform (MSP) and Yugandhar Open Master Data Management (MDM) Hub.

About Yugandhar Microservice Platform

Yugandhar Microservice Platform (also referred as Yugandhar MSP) provides framework for rapid development of your Microservice. This is an architecturally proven Springboot based application having all the basic components needed for a Microservice application to work which just needs to be extended as per your requirements.

Yugandhar Microservice platform codes with code generation templates for rapid development. Just design your data model and generate the code using Yugandhar templates, your base table services will be ready. To create the composite services, you may have to write custom code which would take minimal efforts. Your new Microservice would be ready in just few hours.

About this document

This document covers the system requirements for Yugandhar Microservice Platform.

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 Microservice Platform uses the schema YUG_MSP. 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 MSP schema setup for oracle

Github repository link - <https://github.com/yugandharproject/yugandhar-microservices-platform>

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

1. "1.yugmsp_ora_create_tablespace_and_user.sql" – Needs DBA access
2. "2.yugmsp_ora_create_fullschema.sql" – execute through User YUG_MSP

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 to the TABLES, Sequence and INDEXES are created in database

Table spaces –

MSP_DATATS – used for data and reference tables

MSP_INDXTS – used for Indexes

Profile:

YUGMSP_PROFILE - Used to create YUG_MSP user

User Schema:

YUG_MSP – Default user Schema used by Yugandhar Microservice Platform.

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 MSP 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-microservices-platform>

Download the sqls from '<gitrepo>/resources/yugmsp-dbsetupscripts-mariadb/' path of the Yugandhar-Microservice-platform

1. "1.yugmsp_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:

MSP_DATATS – used for data tables as well as reference tables

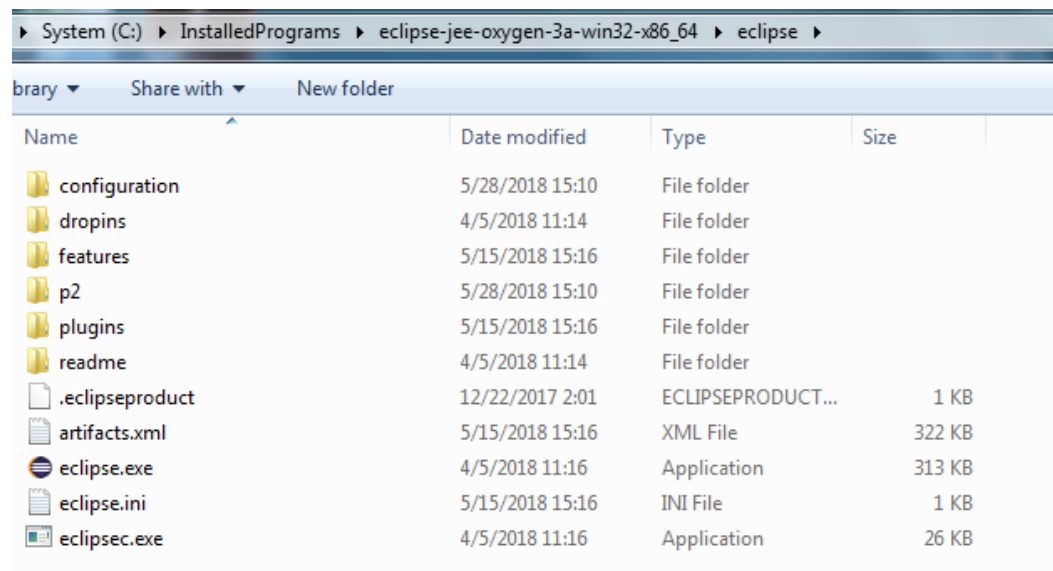
MSP_INDXTS – used for Indexes

DB User – YUG_MSP: Default user used by Yugandhar Microservice Platform to connect to database.

Database: yug_msp database is created.

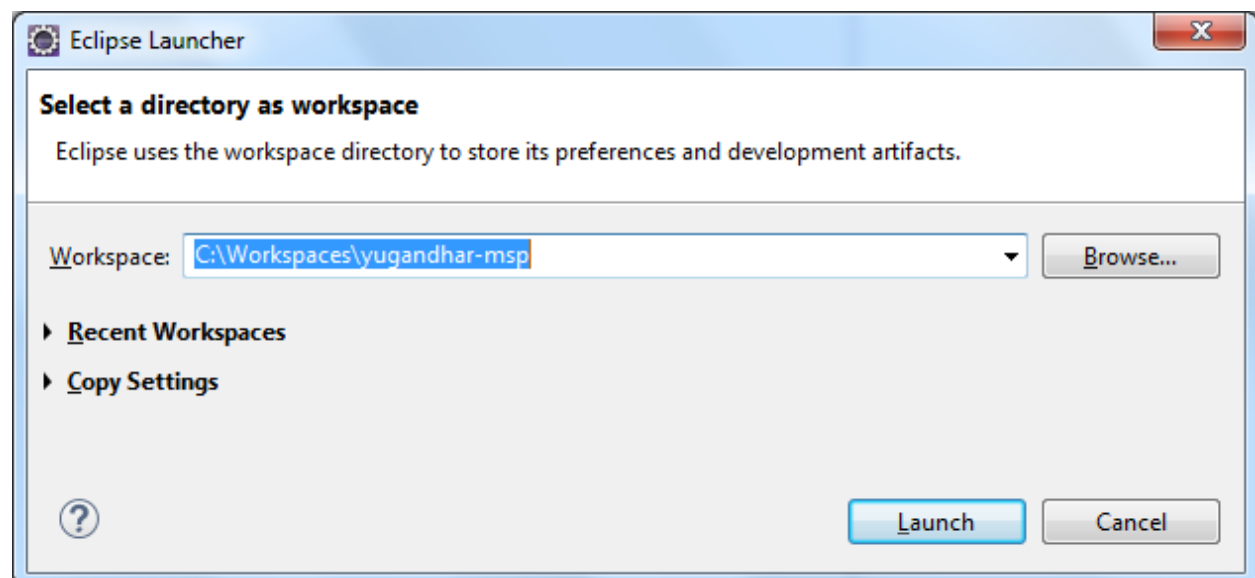
Setup Workspace

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

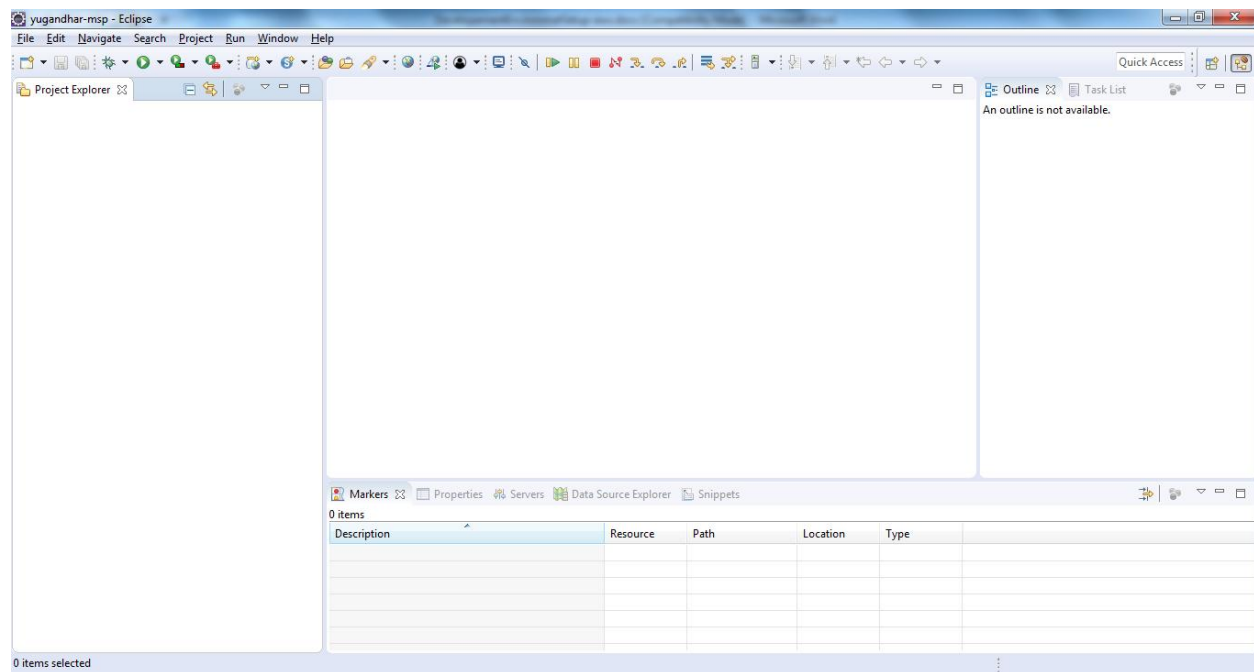


The extracted folder would have above files, click on eclipse.exe to start the eclipse IDE.

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-msp folder.



Your workspace is ready to be configured now.



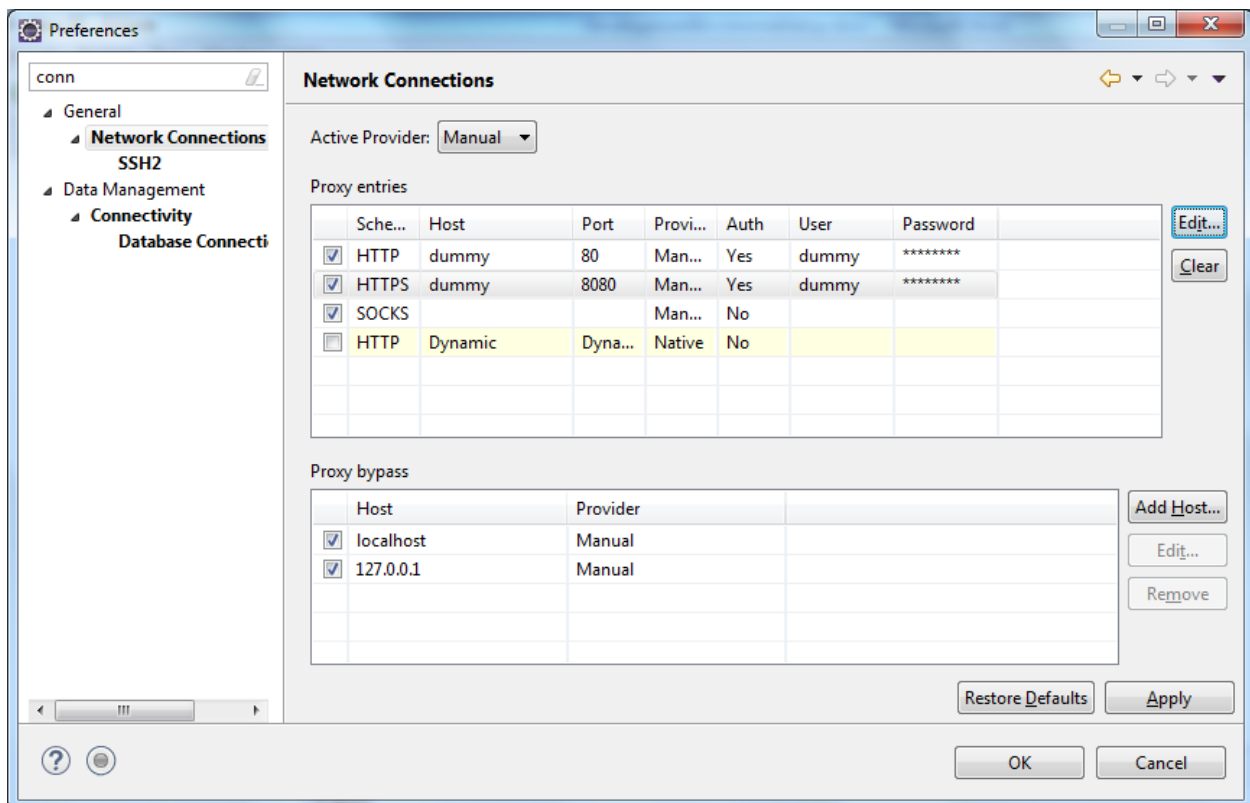
Setup up Network Connections

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.

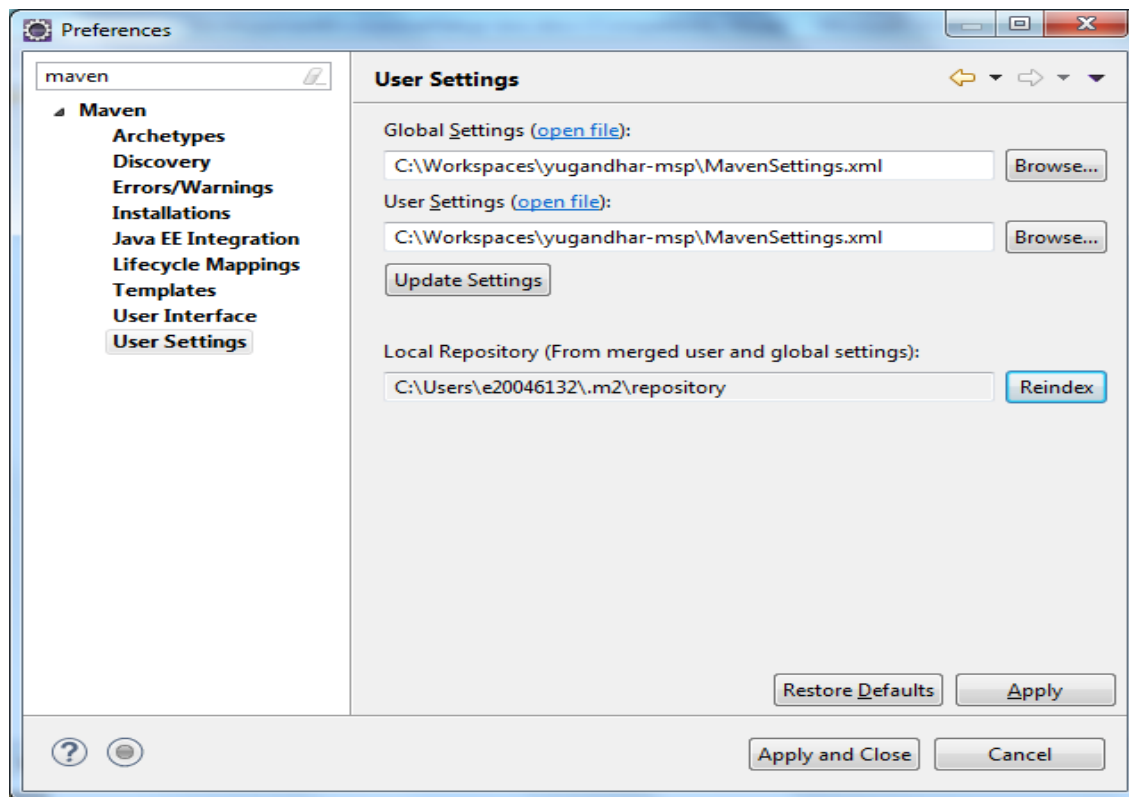
You may also download sample MavenSettings.xml from git hub resources/mavensettings folder.

```
<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.

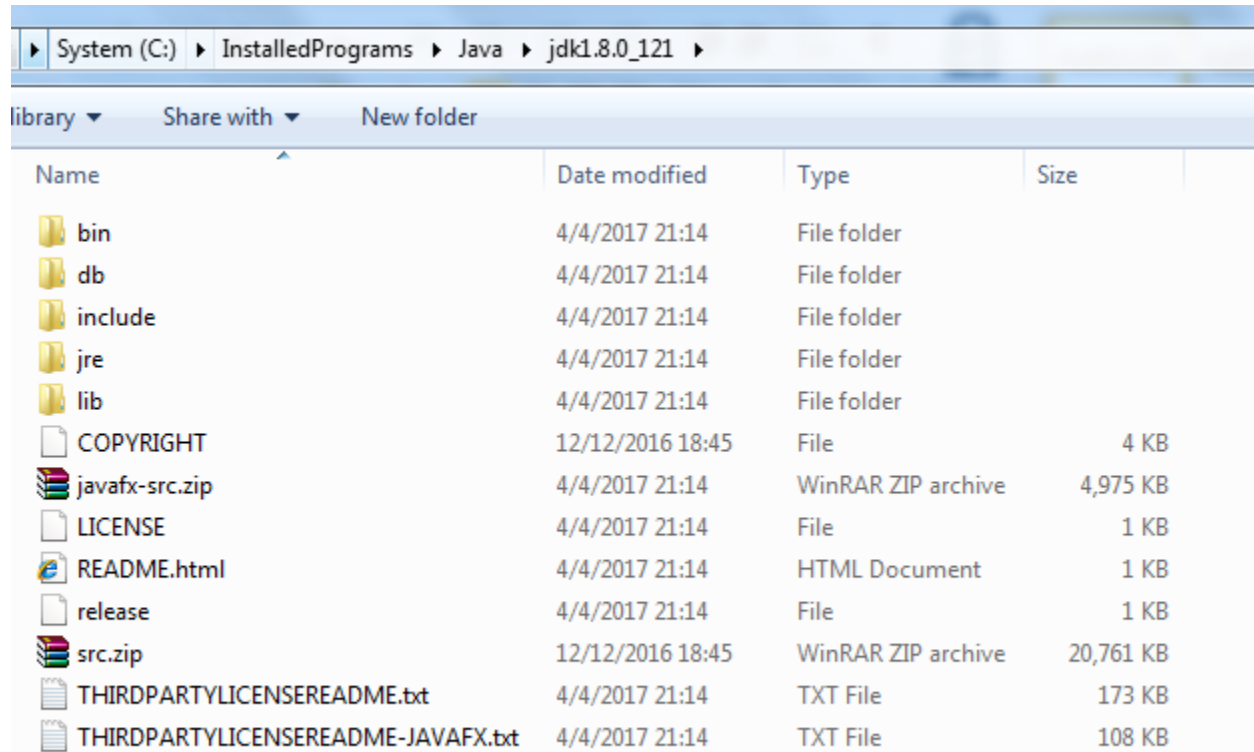
Now go to Eclipse Menu → windows → preferences → maven → User Settings → provide path of the file MavenSettings.xml in the local and global settings as shown in screenshot below



Set JDK Path

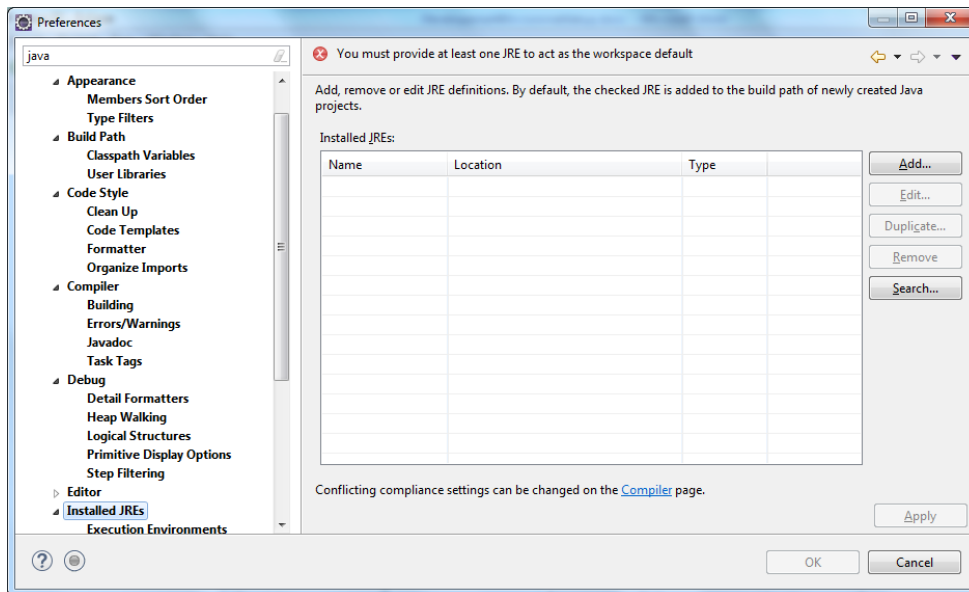
Extract the JDK in a folder of your choice, for the document purpose the jdk directory is

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

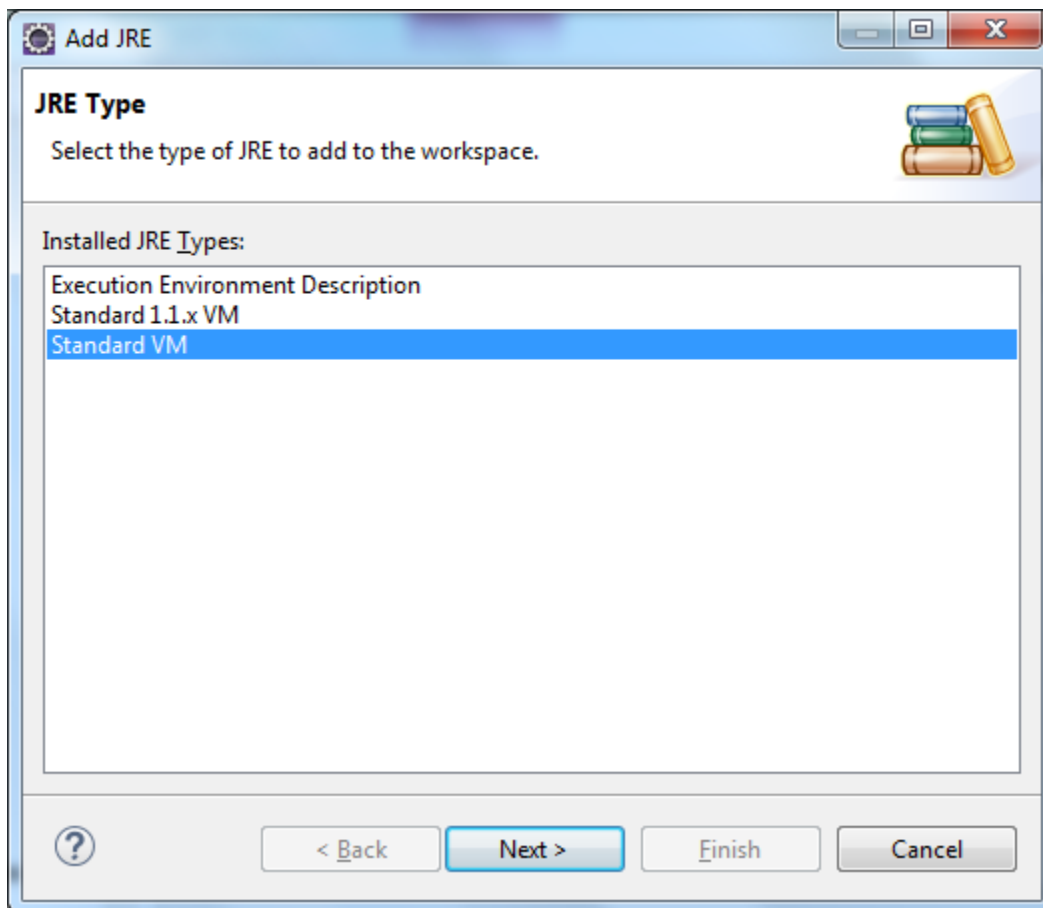


| System (C:) > InstalledPrograms > Java > jdk1.8.0_121 | | | | |
|---|------------------|--------------------|-----------|--|
| library ▾ Share with ▾ New folder | | | | |
| Name | Date modified | Type | Size | |
| bin | 4/4/2017 21:14 | File folder | | |
| db | 4/4/2017 21:14 | File folder | | |
| 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

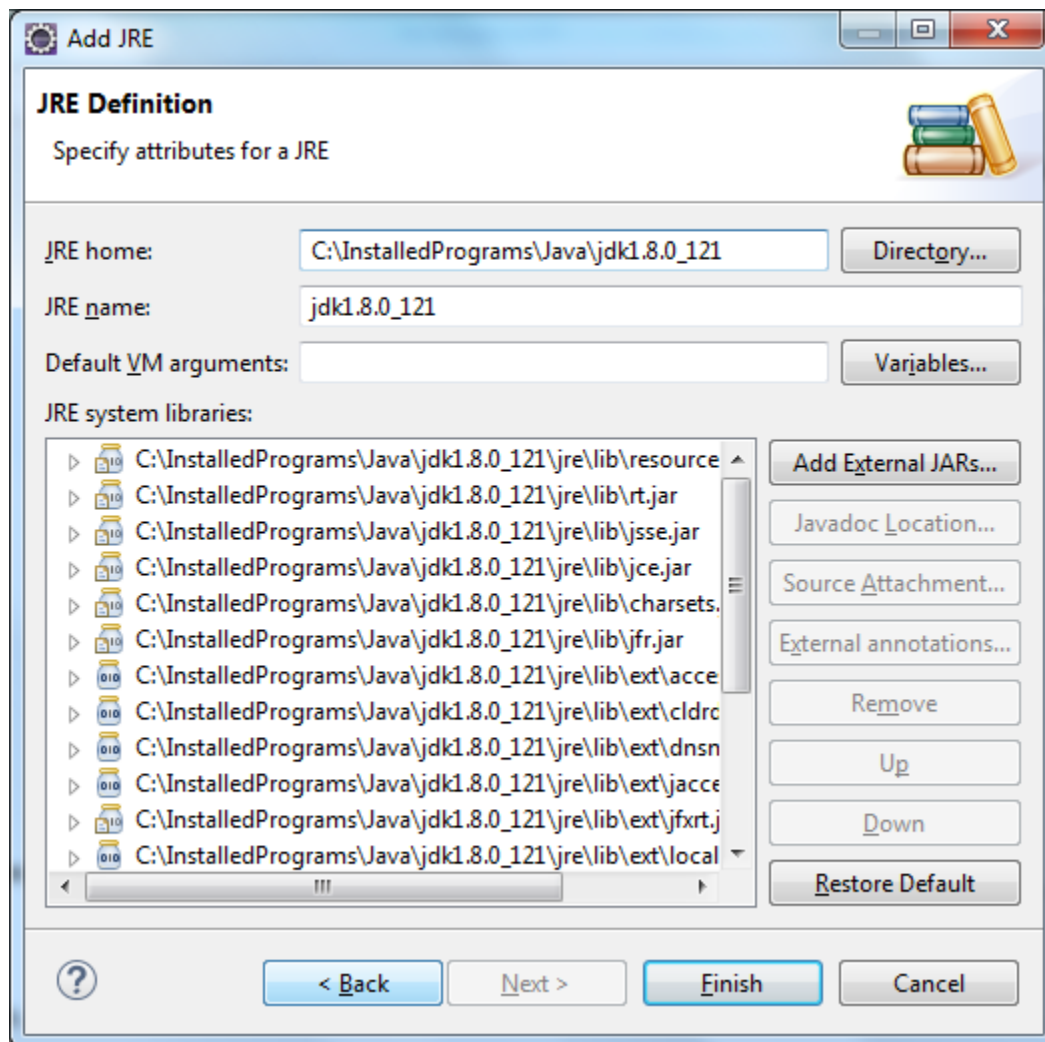


Click Add

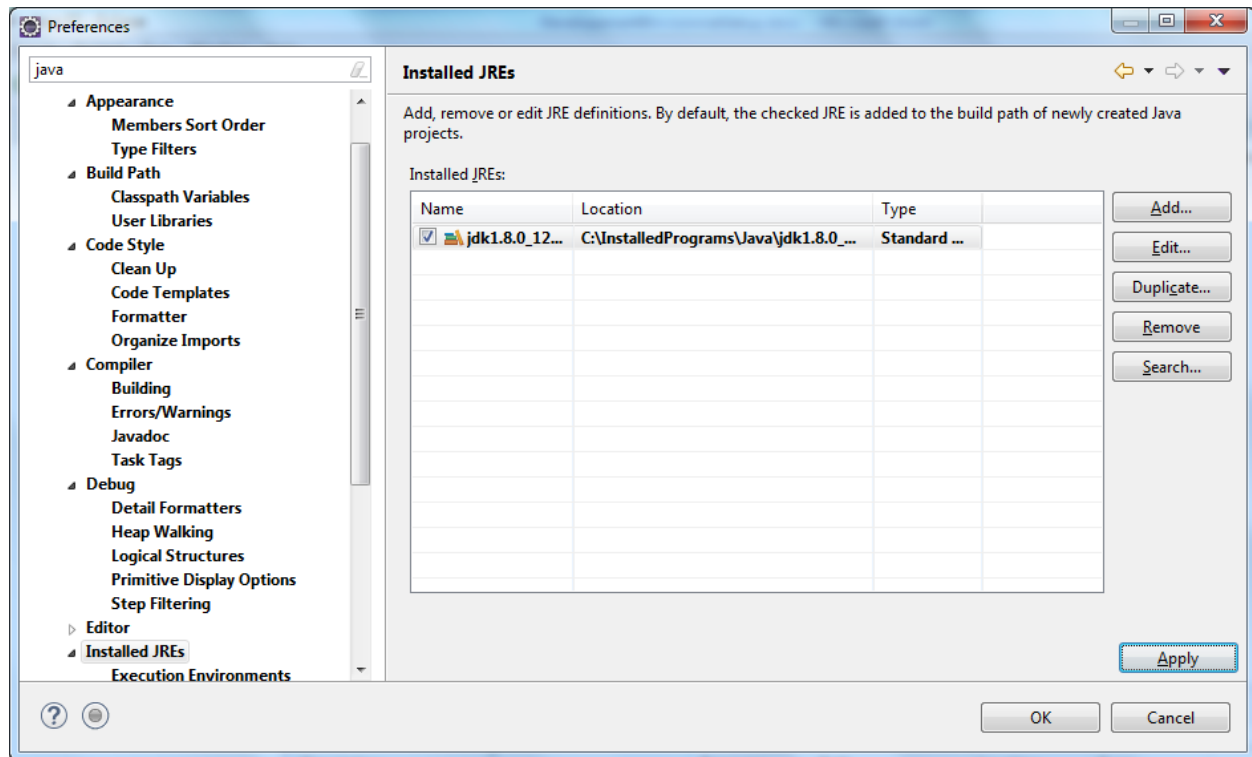


Click Next

Provide the JDK folder name where JDK is extracted



Click finish

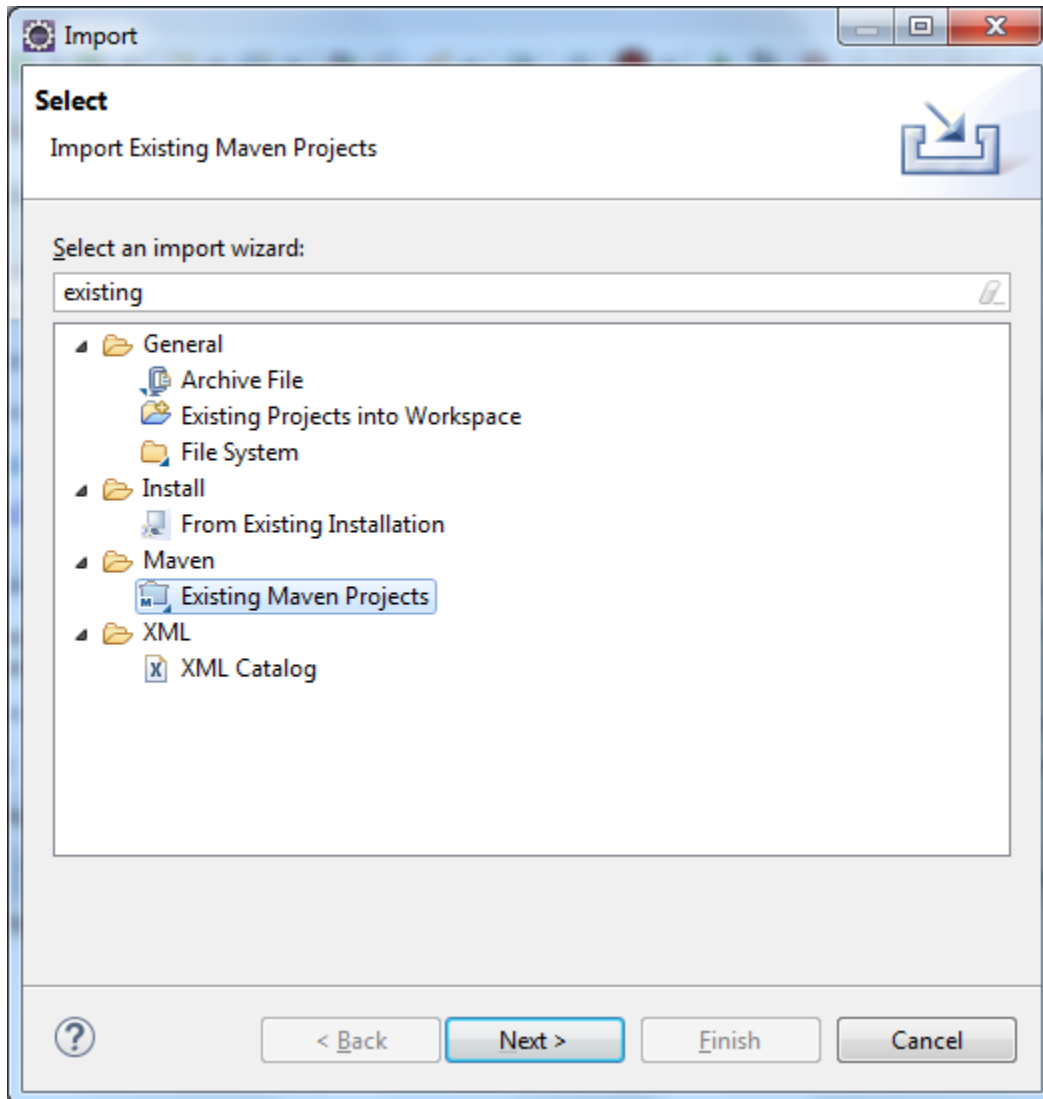


Check the textbox against recently added JDK, click Apply and OK.

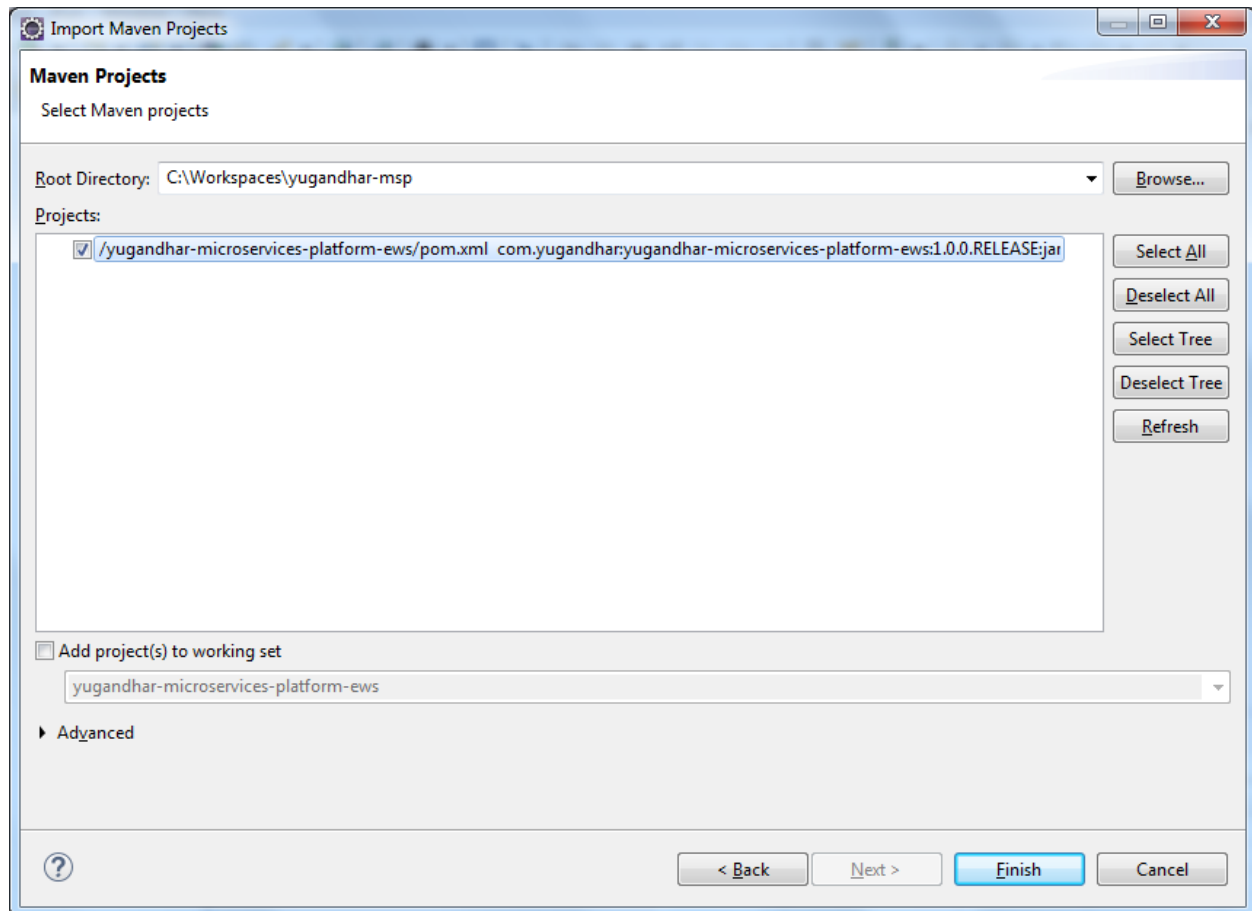
Import Yugandhar msp java projects in the workspace

Download the yugandhar-Microservice-platform-ews from the github repository, extract it in the workspace directory (e.g. C:\Workspaces\yugandhar-msp) and import in Workspace

Go to File → Import Menu → Existing Maven Projects →



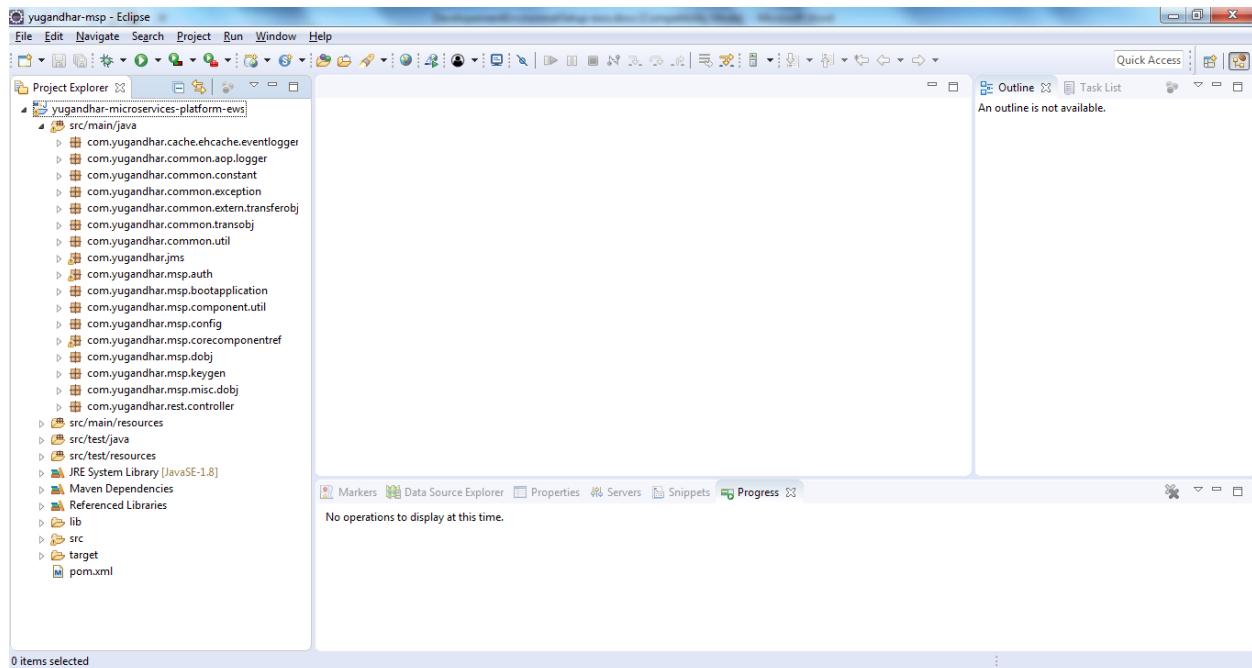
Click on Next Button



Select the project yugandhar-Microservice-platform-ews 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



Microservice platform 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-Microservice-platform-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 msp, 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-msp.properties

The yugandhar-msp.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.msp.datasource.url=  
jdbc:mariadb://localhost:3306/YUG_MSP?pinGlobalTxToPhysicalConnection=true  
#yugandhar.msp.datasource.driver-class-name=org.mariadb.jdbc.Driver  
#yugandhar.msp.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.msp.datasource.url= jdbc:oracle:thin:@localhost:1521/XE  
yugandhar.msp.datasource.driver-class-name=oracle.jdbc.driver.OracleDriver  
yugandhar.msp.datasource.xa.data-source-class-  
name=oracle.jdbc.xa.client.OracleXADataSource
```

- **Common Properties:**

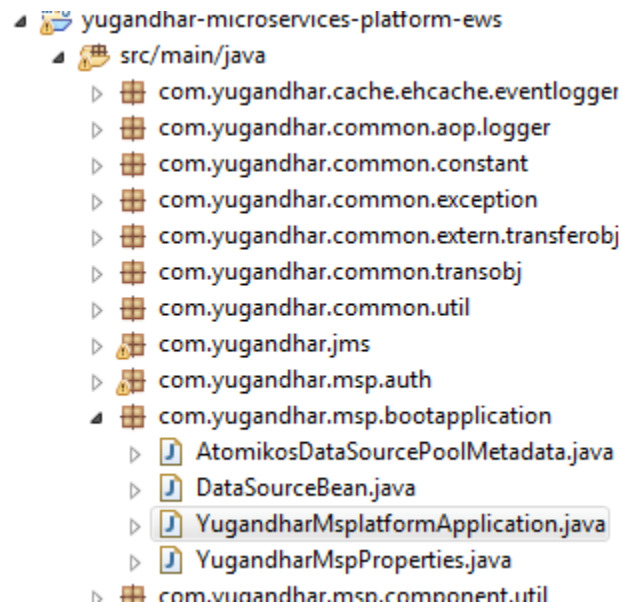
Use either of plain text properties or encrypted properties, if plain text properties provided then encrypted properties will be ignored. Use YugandharEncoderDecoder utility class to encode/decode a given string, refer Encryption Utility section of Development and Customization Guide to understand more.

```
yugandhar.msp.datasource.username.plaintext=user name e.g YUG_MSP  
yugandhar.msp.datasource.password.plaintext=password e.g. YUG_MSP  
#yugandhar.msp.datasource.username.encrypted=encrypted username e.g.  
WVVHX01TUF9PV05FUg==  
#yugandhar.msp.datasource.password.encrypted=encrypted password e.g.  
WVVHX01TUF9PV05FUg==
```

Running the application

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

Go to /yugandhar-Microservice-platform-ews/src/main/java/com/yugandhar/msp/bootapplication/YugandharMsplatformApplication.java.



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

```
2018-05-28 16:51:13,272 [main] [MBeanExporter.java:895] Bean with name 'configurationPropertiesRebinder' has been autodetected for JMX exposure
2018-05-28 16:51:13,276 [main] [MBeanExporter.java:675] Located managed bean 'environmentManager': registering with JMX server as MBean [org.springframework.jmx.support.localreplacemBean
2018-05-28 16:51:13,295 [main] [MBeanExporter.java:675] Located managed bean 'refreshScope': registering with JMX server as MBean [org.springframework.jmx.support.localreplacemBean
2018-05-28 16:51:13,318 [main] [MBeanExporter.java:675] Located managed bean 'configurationPropertiesRebinder': registering with JMX server as MBean [org.springframework.jmx.support.localreplacemBean
2018-05-28 16:51:13,343 [main] [DefaultLifecycleProcessor.java:351] Starting beans in phase 2147483647
2018-05-28 16:51:13,374 [main] [DirectJDKLog.java:180] Starting ProtocolHandler ["http-nio-8091"]
2018-05-28 16:51:13,389 [main] [DirectJDKLog.java:180] Using a shared selector for servlet write/read
2018-05-28 16:51:13,805 [main] [TomcatWebServer.java:206] Tomcat started on port(s): 8091 (http) with context path ''
2018-05-28 16:51:13,814 [main] [StartupInfoLogger.java:59] Started YugandharMsplatformApplication in 24.317 seconds (JVM running for 25.164)
```


TEST With SOAPUI

Test JSON message

The rest url is as below.

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

Yugandhar MSP 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 |

```
{
  "responseCode": "SUCCESS",
  "txnHeader": {
    "requesterLanguage": "1",
    "userName": "admin",
    "userRole": "admin",
    "txnMessageId": "12312311115999",
    "transactionServiceName": "findAllRefCountryIsoByLanguageCodeBase",
    "totalExecutionTimeMillies": "254"
  },
  "txnPayload": {
    "paginationIndexOfCurrentSlice": 2,
    "paginationPageSize": 25,
    "paginationInfoElementsOnCurrentSlice": 25,
    "paginationInfoTotalElements": 247,
    "paginationInfoTotalPages": 10,
    "refCountryIsoDOList": [
      {
        "idPk": "144",
        "version": 0,
        "createdTs": "2018-04-16T12:57:17.000+0000",
        "updatedTs": "2018-04-16T12:57:17.000+0000",
        "updatedByUser": "admin",
        "updatedByTxnId": "000000",
        "configLanguageCodeKey": "1",
        "key": "500"
      }
    ]
  }
}
```

Test XML message

The rest url is as below.

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

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

Request 1

| Method | Endpoint | Resource | Param |
|--------|-----------------------|---------------------------------|-------|
| POST | http://localhost:8091 | /rest/YugandharRequestProcessor | |

Raw

Required: ☐ Sets if parameter is required

Type:

Media Type: ☐ 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.

The screenshot displays a REST client interface for 'Request 1'. The method is POST, the endpoint is http://localhost:8091, and the resource is /rest/YugandharRequestProcessor. The response is shown in XML format, with the raw XML view selected. The XML structure is as follows:

```
<TxnTransferObj>
  <responseCode>SUCCESS</responseCode>
  <txnHeader>
    <requesterLanguage>1</requesterLanguage>
    <userName>admin</userName>
    <userRole>admin</userRole>
    <txnMessageId>12312311115999</txnMessageId>
    <transactionServiceName>findAllRefCountryIsoByLanguageCo
  </txnHeader>
  <txnPayload>
    <paginationIndexOfCurrentSlice>1</paginationIndexOfCurrentSlice>
    <paginationPageSize>50</paginationPageSize>
    <paginationInfoElementsOnCurrentSlice>50</paginationInfoElementsOnCurrentSlice>
    <paginationInfoTotalElements>247</paginationInfoTotalElements>
    <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>
```

The response headers are also visible:

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

This certifies your workspace. Go ahead with Development and customization guide, API Transaction reference guide as well as Code generation guide to understand more.