Eze EMS

Release Date: Aug 18, 2023

Eze EMS xAPI Java Sample Application Guide

This document contains information about getting started with Eze EMS xAPI using Java.

Table of Contents

Legal Information

Copyright

Content

Information in this document is subject to change without notice. In the event that you are using a version of SS&C Eze products other than the most recent version, there may be discrepancies between the content of this documentation and the operation or visual presentation of your older version of the product. SS&C Eze does not warrant that this documentation is error free.

Trademarks

SS&C Eze

http://www.ezesoft.com/

Revision History

The table below provides a snap-shot of the updates in each revision of this document. A bar is displayed on the right side of the page to help you identify updates in the current release.

Version No.	Date	Summary of Update
v2023.5.0.947, v2023.4.0.809, v2023.3.0.638, v2023.2.0.478, v2023.1.0.2, v2022.8.0.4, v2022.5.0.0	NA	There are no documentation updates in these xAPI releases.
v2022.3.0.0	Sep 02, 2022	Initial release.

Introduction

The purpose of this document is to help clients get started with the EMS xAPI application using Java. This document provides a step-by-step process of generating the scripts in Java language and running them using Eclipse IDE for Java Developers, and start using the APIs.

Eze EMS xAPI is robust and easy-to-use application that allows programmers and trading businesses to complete various trading workflows, and also access key information, including:

- Automating order routing - to smart order routers, algorithms and other trading systems.
- Routing orders to multiple brokers, dark pools, ATS, and MTFs via the Eze EMS Global Routing Network - across asset classes.
- Staging or routing single or pairs orders.
- Accessing balances, positions, executions, and other order details.
- Accessing comprehensive list and basket capabilities.

Although EMS xAPI can operate with all gRPC compatible languages, only Java language references are provided in this document as an example. Refer to this link for more information on gRPC.

Eze EMS xAPI Basics

The Eze EMS xAPI operates in conjunction with your existing Eze EMS account permissioning and entitlements. **The Eze EMS xAPI is not a standalone data feed application that is provided to you independent of the Eze EMS.** Please contact Eze Client Service if you need to request or make changes to appropriate permissions for your account.

Eze EMS xAPI Use Restrictions

As an Eze EMS xAPI user, you are prohibited from retransmitting any Eze Market Data using the Eze EMS xAPI, without the express prior written consent of Eze EMS and the exchanges or other third-party data providers (referred to as ***"Sources"*** in your end user agreement). Any unauthorized retransmission of Eze Market Data is a breach of your end user agreement and will cause immediate termination of your use of the Eze EMS, Eze Market Data, and the Eze EMS xAPI.

Any non-display usage of Eze Market Data, such as use of real- time data in algorithmic trading or program trading, is subject to the rules, regulations, and policies of the applicable exchanges and additional exchange fees may apply. In addition, you may have a non-display usage of Eze Market Data even if a display of real-time data occurs. Please review your Eze EMS end user agreement, and the exchanges' and third-party data providers' rules, regulations, and policies that apply to your use of the Eze EMS API (which apply to Eze EMS xAPI) and/or Eze Market Data. It is the sole responsibility of the Eze EMS xAPI user and each user receiving, directly or indirectly accessing or otherwise using Eze Market Data to determine whether your receipt, access or use is reportable and/or fee liable.

Eze EMS xAPI Version

This document covers all the APIs and updates to the Eze EMS xAPI that are part of 2023.5.0.947 release.

Download EMS xAPI

Contact your SS&C Eze client service representative for downloading Eze EMS xAPI.

Developer Support

- If you are an existing Eze EMS user, log in to access developer support documentation and sample code.
- You can contact us or request a demo if you want to explore more about EMS xAPI.
- You can send us an e-mail apisupport@ezesoft.com or call +1 312-442-8122.

Downloading and Installing Java for Eze EMS xAPI

Prerequisites

Java

Eze EMS xAPI sample application can run on Java version 1.8 or above. To download and install the latest Java version, refer https://www.java.com/en/download.

Protoc Compiler

Protoc compiler is needed to compile the proto files and generate .java (protobuf) files (e.g., MarketData.java). To download and install the latest protoc compiler refer https://github.com/protocolbuffers/protobuf/releases/tag/v21.1. For more details, refer Generating .java Files.

For step-by-step procedure to download and install protobuf files, refer https://www.geeksforgeeks.org/how-to-install-protocol-buffers-on-windows/.

Proto-gen-grpc-java

Proto-gen-grpc-java is needed to compile the proto files and generate the stub files for Java language (e.g., MarketDataServiceGrpc.java). To download and install the latest protoc-gen-grpc-java version refer https://repo1.maven.org/maven2/io/grpc/protoc-gen-grpc-java/. For more details, refer Generating Service Stub Files.

Note: Make sure the protoc-gen-grpc-java.exe file is stored in the protoc folder you created while downloading and installing the Protoc compiler above (e.g., C:\EzeEMSxAPI\protoc-21.1-win64\bin\).

Eclipse IDE

To download and install the latest Eclipse IDE for Java Developers version refer https://www.eclipse.org/downloads/. You can also refer the step-by-step procedure to download and install Eclipse IDE https://www.eclipse.org/downloads/packages/installer.

Compiling Protobuf Files

Generating .java Files

Protoc compiler is used to compile the market_data.proto, order.proto, and utilities.proto files and generate the MarketData.java, Order.java, and Utilities.java files accordingly.

To compile proto files and generate .java (protobuf) files:

1. Create a folder on your local machine (e.g., C:\EzeEMSxAPI) to copy and paste the files for compilation and store the generated files.
2. Copy and paste the downloaded Protoc compiler files in the folder you have created in step 1 (e.g., C:\EzeEMSxAPI\protoc-21.1-win64).
3. Create a sub-folder to store the proto files (e.g., C:\EzeEMSxAPI\Protos).

Note: Contact your SS&C Eze client service representative for latest proto files or download them from GitHub.

4. Run the following command in command prompt to generate ***.java** files:

```
>protoc -I=$SRC_DIR --java_out=$DST_DIR $SRC_DIR\market_data.proto
```

- $SRC_DIR — The source path to fetch the proto files
- $DST_DIR — The destination path for storing the generated files

For example, run the below command to generate the **market_data.java** file using **market_data.proto**.

```
>protoc -I=C:\EzeEMSxAPI\Protos --java_out=C:\EzeEMSxAPI\Protos
C:\EzeEMSxAPI\Protos\market_data.proto
```

You can generate the .java files for Order and Utilities by replacing **market_data.proto** with **order.proto** and then with **utilities.proto** in the above command. The **Order.java** and **Utilities.java** files are generated on running the command.

Generating Service Stub Files

Proto-gen-grpc-java is used to compile the market_data.proto, Order.proto, and Utilities.proto files and generate the MarketDataServiceGrpc.java, SubmitOrderServiceGrpc.java, and UtilityServicesGrpc.java files accordingly.

To compile proto files and generate service stub files:

Run the following command in command prompt to generate the service stub files:

```
>protoc --plugin=protoc-gen-grpc-java=%DIR_OF_PROTOC_FILE%\%FILENAME% --grpc-java_
out=lite:%OUTPUT_FILE% --proto_path=%DIR_OF_PROTO_FILE% %PROTO_FILE%
```

- %DIR_OF_PROTOC_FILE% — The source path to fetch the protoc file
- %FILENAME% — Name of the protoc file
- %OUTPUT_FILE% — The destination path for storing the generated files
- %PROTO_FILE% — Name of the proto file

For example, run the below command to generate the **UtilityServicesGrpc.java** file using **utilities.proto**.

```
>protoc --plugin=protoc-gen-grpc-java=C:\EzeEMSxAPI\protoc-21.1-win64\bin\protoc-gen-grpc-
java-1.47.0-windows-x86_64.exe --grpc-java_out=lite:C:\EzeEMSxAPI\Protos --proto_
path=C:\EzeEMSxAPI\Protos Utilities.proto
```

You can generate the service stub files for Order and Market Data by replacing **utilities.proto** with **order.proto** and then with **market_data.proto** in the above command. The **SubmitOrderServiceGrpc.java** and **MarketDataServiceGrpc.java** files are generated on running the command.

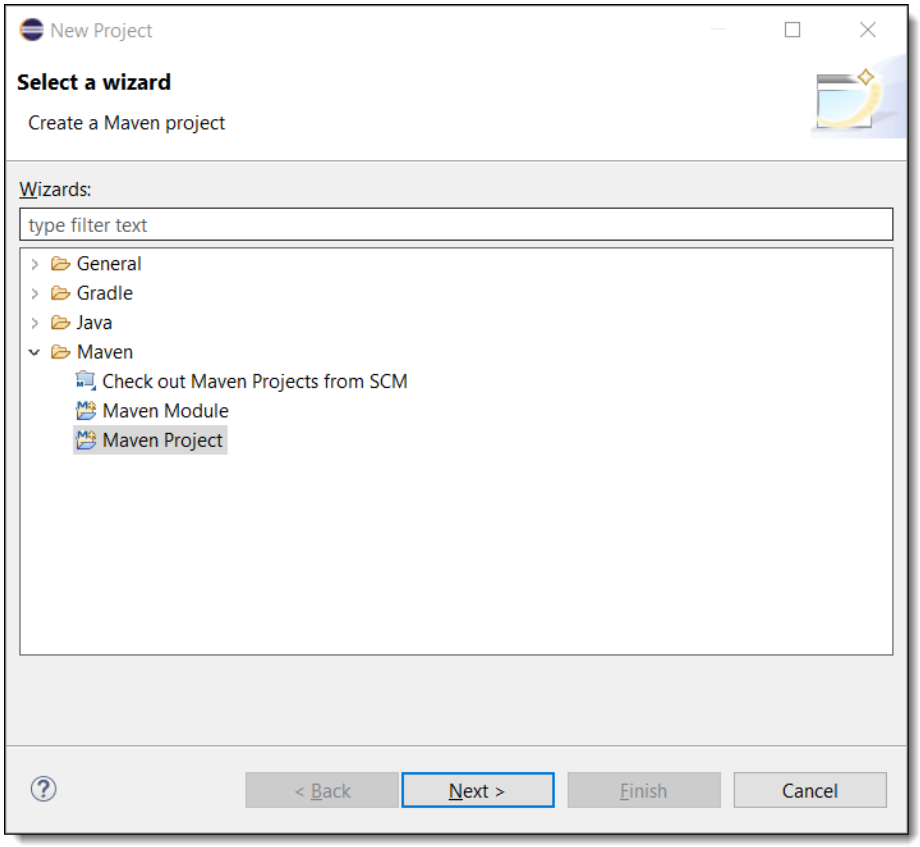
Eclipse IDE

Setting up Eclipse IDE for Java

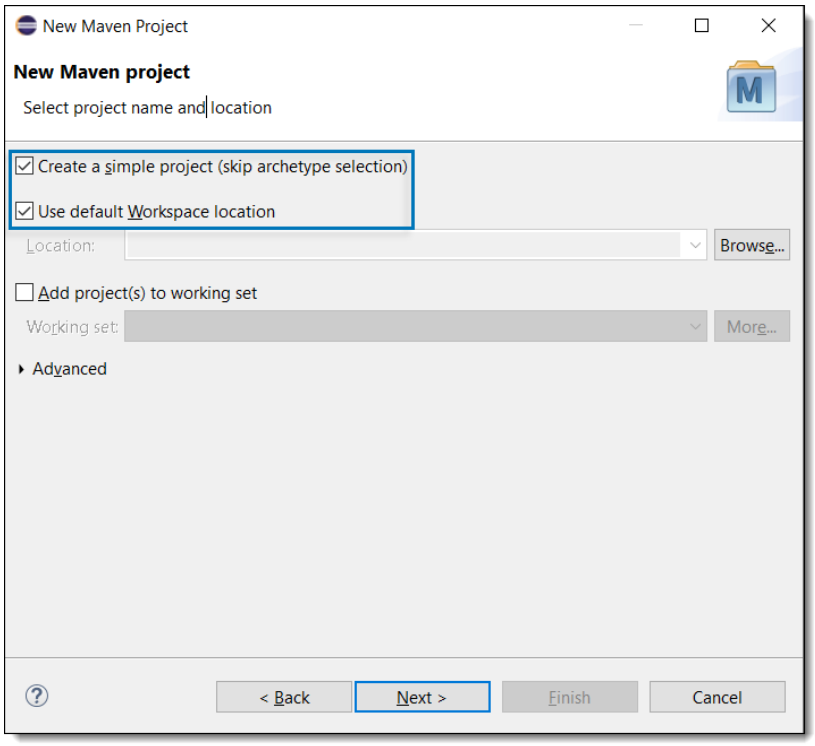
The **Eclipse IDE for Java Developers** application is required to compile the Java and stub files, and run the script files.

To setup Eclipse IDE:

1. Launch Eclipse IDE.
2. Navigate to **File** > **New** > **Project...**.The New Project window opens.
3. Click **Maven** > **MavenProject** to create a new Maven Project. Click **Next >**.



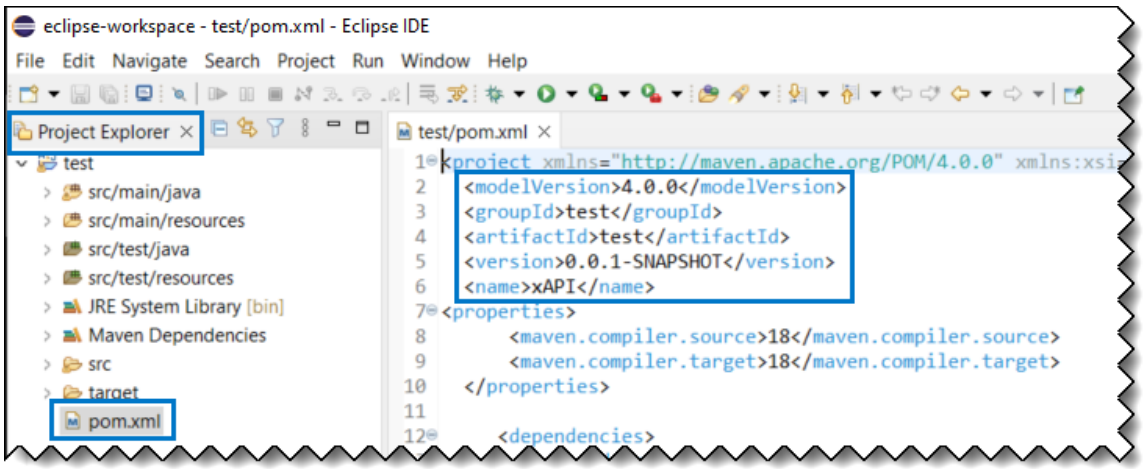
4. Make sure the **Create a simple project (skip archetype selection)** and **Use default Workspace location** checkboxes are enabled, as shown below. Click **Next >**.



5. Enter the details. Click **Finish**.

   Your Maven project (e.g., test) is created successfully.

6. Navigate to **Package Explorer** > **test** > **pom.xml**. Open the **pom.xml** file. Verify if the **modelVersion**, **groupId**, **artifactId**, **version**, and **name** are the same that were used in step 5.



7. To add the gRPC and maven plugin dependencies to your **pom.xml** file, refer the code below. Copy and paste it into your pom.xml file.

```
-<properties>
    <maven.compiler.source>18</maven.compiler.source>
    <maven.compiler.target>18</maven.compiler.target>
</properties>
-<dependencies>
    -<dependency>
        <groupId>io.grpc</groupId>
        <artifactId>grpc-netty-shaded</artifactId>
        <version>1.46.0</version>
        <scope>compile</scope>
    </dependency>
    -<dependency>
        <groupId>io.grpc</groupId>
        <artifactId>grpc-protobuf</artifactId>
        <version>1.46.0</version>
    </dependency>
    -<dependency>
        <groupId>io.grpc</groupId>
        <artifactId>grpc-stub</artifactId>
        <version>1.46.0</version>
    </dependency>
    -<dependency>
            <!-- necessary for Java 9+ -->
        <groupId>org.apache.tomcat</groupId>
        <artifactId>annotations-api</artifactId>
        <version>6.0.53</version>
        <scope>provided</scope>
    </dependency>
    -<dependency>
        <groupId>javax.annotation</groupId>
        <artifactId>javax.annotation-api</artifactId>
        <version>1.3.2</version>
```

```
        </dependency>
        </dependencies>
    -<build>
        -<extensions>
            -<extension>
                <groupId>kr.motd.maven</groupId>
                <artifactId>os-maven-plugin</artifactId>
                <version>1.6.2</version>
            </extension>
        </extensions>
        -<plugins>
            -<plugin>
                <groupId>org.xolstice.maven.plugins</groupId>
                <artifactId>protobuf-maven-plugin</artifactId>
                <version>0.6.1</version>
            </plugin>
        </plugins>
    </build>
</project>
```

8. Right-click **test** (**Package Explorer** > **test**) select **Maven** > **Update Project...**. Click **OK**.
9. Create a new folder to store the Java (protobuf) files by navigating to **Package Explorer** > **test**, then right-click **src\main\java**, click **New** > **Folder**. The New Folder window appears.
    a. Enter a name in the **Folder name** field (e.g., xapi).
    b. Copy and paste the files that are generated after Compiling Protobuf files to this folder.
10. Create a new folder to store the scripts by navigating to **Package Explorer** > **test**, then right-click **src\main\java** > **New** > **Folder**. The New Folder window appears.
    a. Enter a name in the **Folder name** field (e.g., scripts).
    b. Copy and paste the script files to this folder.

Note: Contact your SS&C Eze client service representative for script files.

You have setup the Eclipse IDE project successfully for Java.

Running the Script in Eclipse IDE

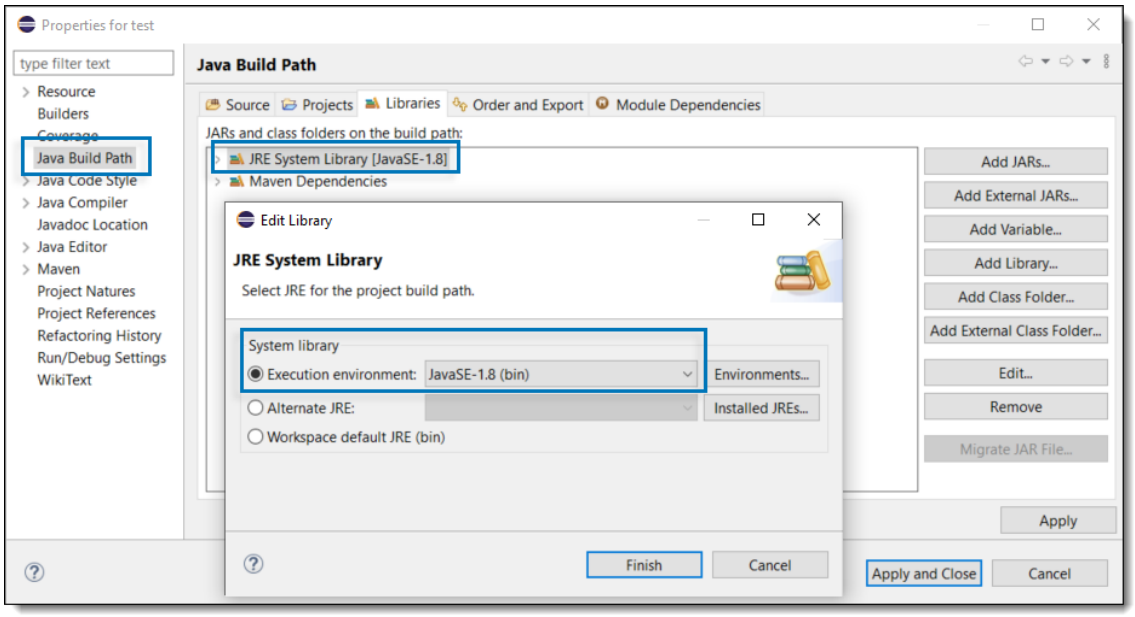
Verify Java Version and Linking

To verify Java link in Eclipse IDE:

1. Launch Eclipse IDE.
2. Navigate to **Package Explorer** > right-click **test** > **Properties**. The Properties for test window opens.
3. Click **Java Build Path** in the left index, then click **Libraries** tab in the right pane. Select **JRE System Library** and click **Edit**. The Edit Library window opens.

Note: If you notice that the JRE is unbound or see build path errors, refer Appendix A: Troubleshooting section to ensure the Java version is linked properly.

4. Select the Java version you have installed in your machine.



5. Click **Finish**.
6. Click **Apply and Close**.

Setup Credentials

After verifying the Java version and its proper linking in Eclipse IDE, open the scripts file and ensure that there are no errors.

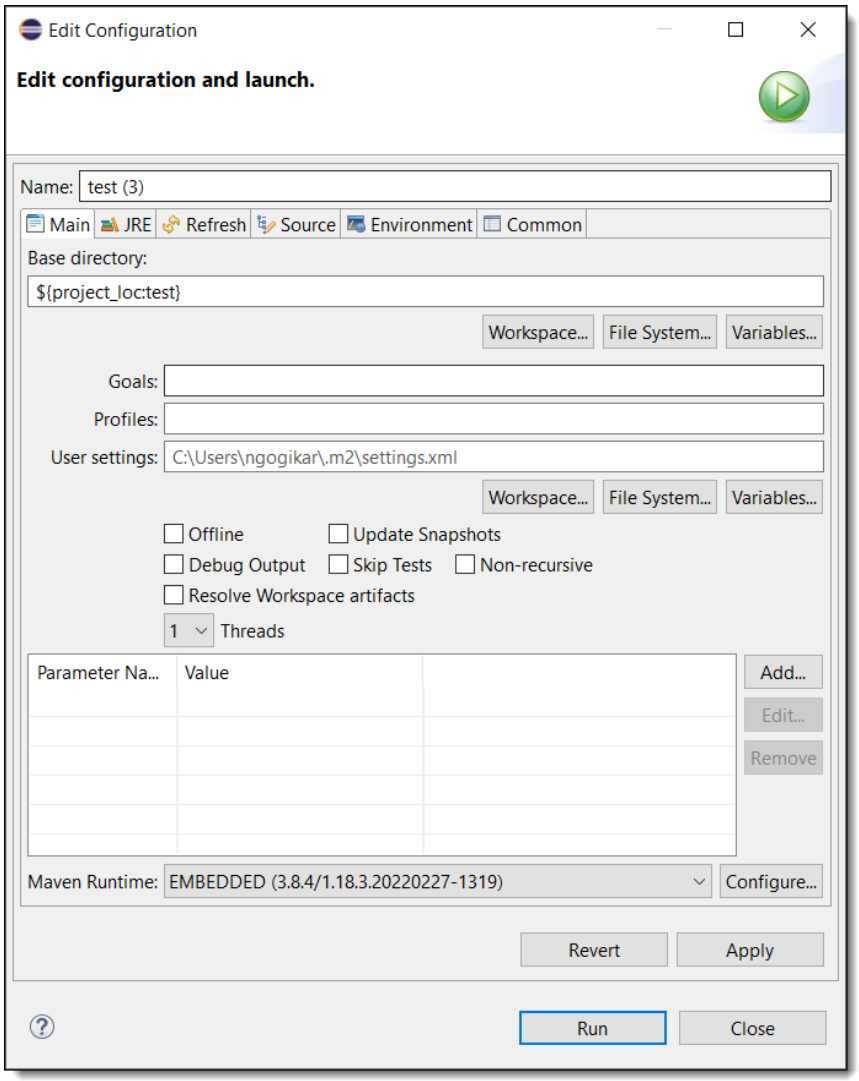
To establish a connection, fill in the **user**, **domain**, **locale**, **password**, and **server** to setup your log in credentials.

Note: Contact your SS&C Eze client service representative for any issues related to login.

Running the Script

To run the script in Eclipse IDE:

1. Navigate to **Package Explorer** > **test** > **scripts** > right-click ***.java** > **Run As** > **3 Maven build...**. The Edit Configuration window opens, as shown below.



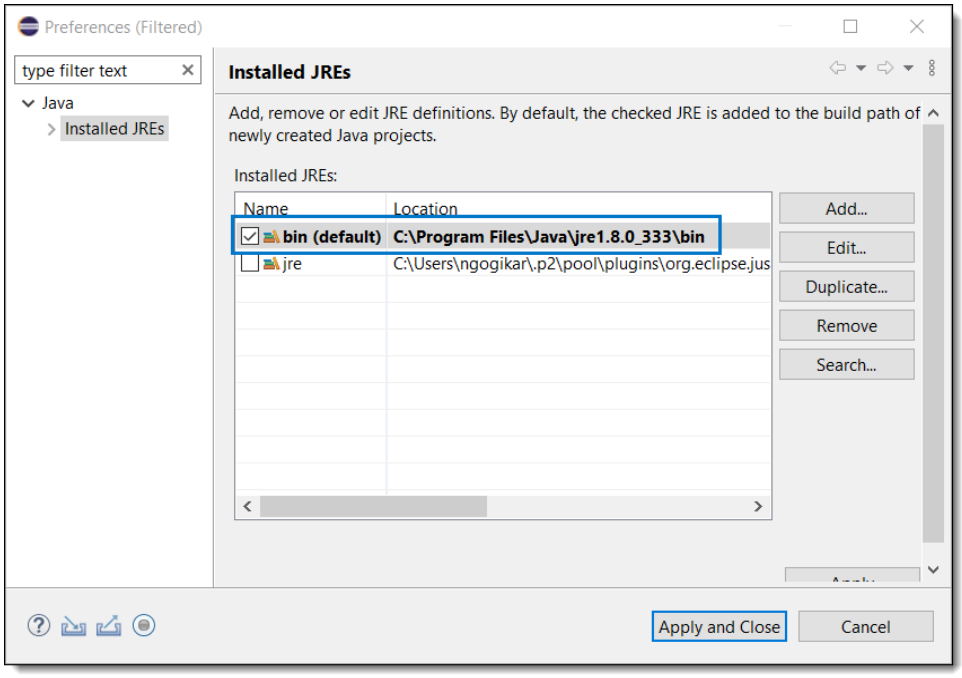
2. Enter **compile** in the **Goals** field.
3. Click **Run**.

Appendix A: Troubleshooting

Follow the steps below if you see build path errors in Eclipse IDE. This error occurs when the local Java version is not linked properly in Eclipse IDE.

To link local Java version in Eclipse IDE:

1. Launch Eclipse IDE.
2. Navigate to **Package Explorer** > right-click FOLDER > **Build Path** > **Configure Build Path...**. The Properties for FOLDER window opens. Here, FOLDER refers to your locally created folder.
3. Click **Java Build Path** in the left index, then click **Libraries** tab in the right pane.
4. Select **JRE System Library [bin] (unbound)**, click **Edit**.
5. Select **Alternate JRE** > **Installed JREs**. The Preferences (Filtered) window opens.
6. Click **Add...**. Select **Standard VM** in the JRE Type window. Click **Next>**. The Add JRE window opens.
7. Click **Directory...** for **JRE home:** field. The Select Folder window opens.
8. Navigate to JRE bin in your local machine. By default, the downloaded Java files are placed in **C:\Program Files\Java\jre1.8.0_333\bin** of your local machine. Click **Select Folder**.
9. Click **Finish**.
10. In Preferences (Filtered) window enable the bin path, as shown below. Click **Apply** > **Apply and Close**.



11. Click **Finish**.
12. Click **Apply** > **Apply and Close**.