***Difference between JPA, Hibernate and Spring Data JPA***

***Java Persistence API (JPA)***

* *JPA is a* ***specification*** *(JSR 338) for object-relational mapping (ORM).*
* *It defines the* ***standard interface*** *for persisting, retrieving, and managing Java objects in a relational database.*
* *JPA* ***does not provide an implementation****—instead, vendors like Hibernate implement the JPA standard.*
* *It defines concepts like Entity, EntityManager, and annotations such as @Entity, @Id, etc.*

***Hibernate***

* *Hibernate is a* ***popular ORM framework*** *and a* ***JPA implementation****.*
* *It allows developers to map Java classes to database tables.*
* *Developers need to handle* ***session management*** *and* ***transactions*** *manually (unless integrated with Spring).*
* *It includes both JPA-compliant features and additional non-standard features (like caching and performance optimizations).*

***Example using Hibernate*:**

public Integer addEmployee(Employee employee) {

Session session = factory.openSession();

Transaction tx = null;

Integer employeeID = null;

try {

tx = session.beginTransaction();

employeeID = (Integer) session.save(employee);

tx.commit();

} catch (HibernateException e) {

if (tx != null) tx.rollback();

e.printStackTrace();

} finally {

session.close();

}

return employeeID;

}

***Spring Data JPA***

* *Spring Data JPA is* ***not a JPA implementation****.*
* *Spring Data JPA is a part of the Spring Framework that builds on top of JPA.*
* *It provides a* ***higher abstraction*** *on top of JPA, reducing boilerplate code.*
* *It* ***auto-generates repository methods****, manages transactions, and integrates tightly with Spring Boot.*

***Example using Spring Data JPA:***

***EmployeeRepository.java***

*public interface EmployeeRepository extends JpaRepository<Employee, Integer> {*

*}*

***EmployeeService.java***

@Autowired

private EmployeeRepository employeeRepository;

@Transactional

public void addEmployee(Employee employee) {

employeeRepository.save(employee);

}

**mvnw**

#!/bin/sh

# ----------------------------------------------------------------------------

# Licensed to the Apache Software Foundation (ASF) under one

# or more contributor license agreements. See the NOTICE file

# distributed with this work for additional information

# regarding copyright ownership. The ASF licenses this file

# to you 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.

# ----------------------------------------------------------------------------

# ----------------------------------------------------------------------------

# Apache Maven Wrapper startup batch script, version 3.3.2

#

# Optional ENV vars

# -----------------

# JAVA\_HOME - location of a JDK home dir, required when download maven via java source

# MVNW\_REPOURL - repo url base for downloading maven distribution

# MVNW\_USERNAME/MVNW\_PASSWORD - user and password for downloading maven

# MVNW\_VERBOSE - true: enable verbose log; debug: trace the mvnw script; others: silence the output

# ----------------------------------------------------------------------------

set -euf

[ "${MVNW\_VERBOSE-}" != debug ] || set -x

# OS specific support.

native\_path() { printf %s\\n "$1"; }

case "$(uname)" in

CYGWIN\* | MINGW\*)

[ -z "${JAVA\_HOME-}" ] || JAVA\_HOME="$(cygpath --unix "$JAVA\_HOME")"

native\_path() { cygpath --path --windows "$1"; }

;;

esac

# set JAVACMD and JAVACCMD

set\_java\_home() {

# For Cygwin and MinGW, ensure paths are in Unix format before anything is touched

if [ -n "${JAVA\_HOME-}" ]; then

if [ -x "$JAVA\_HOME/jre/sh/java" ]; then

# IBM's JDK on AIX uses strange locations for the executables

JAVACMD="$JAVA\_HOME/jre/sh/java"

JAVACCMD="$JAVA\_HOME/jre/sh/javac"

else

JAVACMD="$JAVA\_HOME/bin/java"

JAVACCMD="$JAVA\_HOME/bin/javac"

if [ ! -x "$JAVACMD" ] || [ ! -x "$JAVACCMD" ]; then

echo "The JAVA\_HOME environment variable is not defined correctly, so mvnw cannot run." >&2

echo "JAVA\_HOME is set to \"$JAVA\_HOME\", but \"\$JAVA\_HOME/bin/java\" or \"\$JAVA\_HOME/bin/javac\" does not exist." >&2

return 1

fi

fi

else

JAVACMD="$(

'set' +e

'unset' -f command 2>/dev/null

'command' -v java

)" || :

JAVACCMD="$(

'set' +e

'unset' -f command 2>/dev/null

'command' -v javac

)" || :

if [ ! -x "${JAVACMD-}" ] || [ ! -x "${JAVACCMD-}" ]; then

echo "The java/javac command does not exist in PATH nor is JAVA\_HOME set, so mvnw cannot run." >&2

return 1

fi

fi

}

# hash string like Java String::hashCode

hash\_string() {

str="${1:-}" h=0

while [ -n "$str" ]; do

char="${str%"${str#?}"}"

h=$(((h \* 31 + $(LC\_CTYPE=C printf %d "'$char")) % 4294967296))

str="${str#?}"

done

printf %x\\n $h

}

verbose() { :; }

[ "${MVNW\_VERBOSE-}" != true ] || verbose() { printf %s\\n "${1-}"; }

die() {

printf %s\\n "$1" >&2

exit 1

}

trim() {

# MWRAPPER-139:

# Trims trailing and leading whitespace, carriage returns, tabs, and linefeeds.

# Needed for removing poorly interpreted newline sequences when running in more

# exotic environments such as mingw bash on Windows.

printf "%s" "${1}" | tr -d '[:space:]'

}

# parse distributionUrl and optional distributionSha256Sum, requires .mvn/wrapper/maven-wrapper.properties

while IFS="=" read -r key value; do

case "${key-}" in

distributionUrl) distributionUrl=$(trim "${value-}") ;;

distributionSha256Sum) distributionSha256Sum=$(trim "${value-}") ;;

esac

done <"${0%/\*}/.mvn/wrapper/maven-wrapper.properties"

[ -n "${distributionUrl-}" ] || die "cannot read distributionUrl property in ${0%/\*}/.mvn/wrapper/maven-wrapper.properties"

case "${distributionUrl##\*/}" in

maven-mvnd-\*bin.\*)

MVN\_CMD=mvnd.sh \_MVNW\_REPO\_PATTERN=/maven/mvnd/

case "${PROCESSOR\_ARCHITECTURE-}${PROCESSOR\_ARCHITEW6432-}:$(uname -a)" in

\*AMD64:CYGWIN\* | \*AMD64:MINGW\*) distributionPlatform=windows-amd64 ;;

:Darwin\*x86\_64) distributionPlatform=darwin-amd64 ;;

:Darwin\*arm64) distributionPlatform=darwin-aarch64 ;;

:Linux\*x86\_64\*) distributionPlatform=linux-amd64 ;;

\*)

echo "Cannot detect native platform for mvnd on $(uname)-$(uname -m), use pure java version" >&2

distributionPlatform=linux-amd64

;;

esac

distributionUrl="${distributionUrl%-bin.\*}-$distributionPlatform.zip"

;;

maven-mvnd-\*) MVN\_CMD=mvnd.sh \_MVNW\_REPO\_PATTERN=/maven/mvnd/ ;;

\*) MVN\_CMD="mvn${0##\*/mvnw}" \_MVNW\_REPO\_PATTERN=/org/apache/maven/ ;;

esac

# apply MVNW\_REPOURL and calculate MAVEN\_HOME

# maven home pattern: ~/.m2/wrapper/dists/{apache-maven-<version>,maven-mvnd-<version>-<platform>}/<hash>

[ -z "${MVNW\_REPOURL-}" ] || distributionUrl="$MVNW\_REPOURL$\_MVNW\_REPO\_PATTERN${distributionUrl#\*"$\_MVNW\_REPO\_PATTERN"}"

distributionUrlName="${distributionUrl##\*/}"

distributionUrlNameMain="${distributionUrlName%.\*}"

distributionUrlNameMain="${distributionUrlNameMain%-bin}"

MAVEN\_USER\_HOME="${MAVEN\_USER\_HOME:-${HOME}/.m2}"

MAVEN\_HOME="${MAVEN\_USER\_HOME}/wrapper/dists/${distributionUrlNameMain-}/$(hash\_string "$distributionUrl")"

exec\_maven() {

unset MVNW\_VERBOSE MVNW\_USERNAME MVNW\_PASSWORD MVNW\_REPOURL || :

exec "$MAVEN\_HOME/bin/$MVN\_CMD" "$@" || die "cannot exec $MAVEN\_HOME/bin/$MVN\_CMD"

}

if [ -d "$MAVEN\_HOME" ]; then

verbose "found existing MAVEN\_HOME at $MAVEN\_HOME"

exec\_maven "$@"

fi

case "${distributionUrl-}" in

\*?-bin.zip | \*?maven-mvnd-?\*-?\*.zip) ;;

\*) die "distributionUrl is not valid, must match \*-bin.zip or maven-mvnd-\*.zip, but found '${distributionUrl-}'" ;;

esac

# prepare tmp dir

if TMP\_DOWNLOAD\_DIR="$(mktemp -d)" && [ -d "$TMP\_DOWNLOAD\_DIR" ]; then

clean() { rm -rf -- "$TMP\_DOWNLOAD\_DIR"; }

trap clean HUP INT TERM EXIT

else

die "cannot create temp dir"

fi

mkdir -p -- "${MAVEN\_HOME%/\*}"

# Download and Install Apache Maven

verbose "Couldn't find MAVEN\_HOME, downloading and installing it ..."

verbose "Downloading from: $distributionUrl"

verbose "Downloading to: $TMP\_DOWNLOAD\_DIR/$distributionUrlName"

# select .zip or .tar.gz

if ! command -v unzip >/dev/null; then

distributionUrl="${distributionUrl%.zip}.tar.gz"

distributionUrlName="${distributionUrl##\*/}"

fi

# verbose opt

\_\_MVNW\_QUIET\_WGET=--quiet \_\_MVNW\_QUIET\_CURL=--silent \_\_MVNW\_QUIET\_UNZIP=-q \_\_MVNW\_QUIET\_TAR=''

[ "${MVNW\_VERBOSE-}" != true ] || \_\_MVNW\_QUIET\_WGET='' \_\_MVNW\_QUIET\_CURL='' \_\_MVNW\_QUIET\_UNZIP='' \_\_MVNW\_QUIET\_TAR=v

# normalize http auth

case "${MVNW\_PASSWORD:+has-password}" in

'') MVNW\_USERNAME='' MVNW\_PASSWORD='' ;;

has-password) [ -n "${MVNW\_USERNAME-}" ] || MVNW\_USERNAME='' MVNW\_PASSWORD='' ;;

esac

if [ -z "${MVNW\_USERNAME-}" ] && command -v wget >/dev/null; then

verbose "Found wget ... using wget"

wget ${\_\_MVNW\_QUIET\_WGET:+"$\_\_MVNW\_QUIET\_WGET"} "$distributionUrl" -O "$TMP\_DOWNLOAD\_DIR/$distributionUrlName" || die "wget: Failed to fetch $distributionUrl"

elif [ -z "${MVNW\_USERNAME-}" ] && command -v curl >/dev/null; then

verbose "Found curl ... using curl"

curl ${\_\_MVNW\_QUIET\_CURL:+"$\_\_MVNW\_QUIET\_CURL"} -f -L -o "$TMP\_DOWNLOAD\_DIR/$distributionUrlName" "$distributionUrl" || die "curl: Failed to fetch $distributionUrl"

elif set\_java\_home; then

verbose "Falling back to use Java to download"

javaSource="$TMP\_DOWNLOAD\_DIR/Downloader.java"

targetZip="$TMP\_DOWNLOAD\_DIR/$distributionUrlName"

cat >"$javaSource" <<-END

public class Downloader extends java.net.Authenticator

{

protected java.net.PasswordAuthentication getPasswordAuthentication()

{

return new java.net.PasswordAuthentication( System.getenv( "MVNW\_USERNAME" ), System.getenv( "MVNW\_PASSWORD" ).toCharArray() );

}

public static void main( String[] args ) throws Exception

{

setDefault( new Downloader() );

java.nio.file.Files.copy( java.net.URI.create( args[0] ).toURL().openStream(), java.nio.file.Paths.get( args[1] ).toAbsolutePath().normalize() );

}

}

END

# For Cygwin/MinGW, switch paths to Windows format before running javac and java

verbose " - Compiling Downloader.java ..."

"$(native\_path "$JAVACCMD")" "$(native\_path "$javaSource")" || die "Failed to compile Downloader.java"

verbose " - Running Downloader.java ..."

"$(native\_path "$JAVACMD")" -cp "$(native\_path "$TMP\_DOWNLOAD\_DIR")" Downloader "$distributionUrl" "$(native\_path "$targetZip")"

fi

# If specified, validate the SHA-256 sum of the Maven distribution zip file

if [ -n "${distributionSha256Sum-}" ]; then

distributionSha256Result=false

if [ "$MVN\_CMD" = mvnd.sh ]; then

echo "Checksum validation is not supported for maven-mvnd." >&2

echo "Please disable validation by removing 'distributionSha256Sum' from your maven-wrapper.properties." >&2

exit 1

elif command -v sha256sum >/dev/null; then

if echo "$distributionSha256Sum $TMP\_DOWNLOAD\_DIR/$distributionUrlName" | sha256sum -c >/dev/null 2>&1; then

distributionSha256Result=true

fi

elif command -v shasum >/dev/null; then

if echo "$distributionSha256Sum $TMP\_DOWNLOAD\_DIR/$distributionUrlName" | shasum -a 256 -c >/dev/null 2>&1; then

distributionSha256Result=true

fi

else

echo "Checksum validation was requested but neither 'sha256sum' or 'shasum' are available." >&2

echo "Please install either command, or disable validation by removing 'distributionSha256Sum' from your maven-wrapper.properties." >&2

exit 1

fi

if [ $distributionSha256Result = false ]; then

echo "Error: Failed to validate Maven distribution SHA-256, your Maven distribution might be compromised." >&2

echo "If you updated your Maven version, you need to update the specified distributionSha256Sum property." >&2

exit 1

fi

fi

# unzip and move

if command -v unzip >/dev/null; then

unzip ${\_\_MVNW\_QUIET\_UNZIP:+"$\_\_MVNW\_QUIET\_UNZIP"} "$TMP\_DOWNLOAD\_DIR/$distributionUrlName" -d "$TMP\_DOWNLOAD\_DIR" || die "failed to unzip"

else

tar xzf${\_\_MVNW\_QUIET\_TAR:+"$\_\_MVNW\_QUIET\_TAR"} "$TMP\_DOWNLOAD\_DIR/$distributionUrlName" -C "$TMP\_DOWNLOAD\_DIR" || die "failed to untar"

fi

printf %s\\n "$distributionUrl" >"$TMP\_DOWNLOAD\_DIR/$distributionUrlNameMain/mvnw.url"

mv -- "$TMP\_DOWNLOAD\_DIR/$distributionUrlNameMain" "$MAVEN\_HOME" || [ -d "$MAVEN\_HOME" ] || die "fail to move MAVEN\_HOME"

clean || :

exec\_maven "$@"

**mvnw.cmd**

<# : batch portion

@REM ----------------------------------------------------------------------------

@REM Licensed to the Apache Software Foundation (ASF) under one

@REM or more contributor license agreements. See the NOTICE file

@REM distributed with this work for additional information

@REM regarding copyright ownership. The ASF licenses this file

@REM to you under the Apache License, Version 2.0 (the

@REM "License"); you may not use this file except in compliance

@REM with the License. You may obtain a copy of the License at

@REM

@REM http://www.apache.org/licenses/LICENSE-2.0

@REM

@REM Unless required by applicable law or agreed to in writing,

@REM software distributed under the License is distributed on an

@REM "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY

@REM KIND, either express or implied. See the License for the

@REM specific language governing permissions and limitations

@REM under the License.

@REM ----------------------------------------------------------------------------

@REM ----------------------------------------------------------------------------

@REM Apache Maven Wrapper startup batch script, version 3.3.2

@REM

@REM Optional ENV vars

@REM MVNW\_REPOURL - repo url base for downloading maven distribution

@REM MVNW\_USERNAME/MVNW\_PASSWORD - user and password for downloading maven

@REM MVNW\_VERBOSE - true: enable verbose log; others: silence the output

@REM ----------------------------------------------------------------------------

@IF "%\_\_MVNW\_ARG0\_NAME\_\_%"=="" (SET \_\_MVNW\_ARG0\_NAME\_\_=%~nx0)

@SET \_\_MVNW\_CMD\_\_=

@SET \_\_MVNW\_ERROR\_\_=

@SET \_\_MVNW\_PSMODULEP\_SAVE=%PSModulePath%

@SET PSModulePath=

@FOR /F "usebackq tokens=1\* delims==" %%A IN (`powershell -noprofile "& {$scriptDir='%~dp0'; $script='%\_\_MVNW\_ARG0\_NAME\_\_%'; icm -ScriptBlock ([Scriptblock]::Create((Get-Content -Raw '%~f0'))) -NoNewScope}"`) DO @(

IF "%%A"=="MVN\_CMD" (set \_\_MVNW\_CMD\_\_=%%B) ELSE IF "%%B"=="" (echo %%A) ELSE (echo %%A=%%B)

)

@SET PSModulePath=%\_\_MVNW\_PSMODULEP\_SAVE%

@SET \_\_MVNW\_PSMODULEP\_SAVE=

@SET \_\_MVNW\_ARG0\_NAME\_\_=

@SET MVNW\_USERNAME=

@SET MVNW\_PASSWORD=

@IF NOT "%\_\_MVNW\_CMD\_\_%"=="" (%\_\_MVNW\_CMD\_\_% %\*)

@echo Cannot start maven from wrapper >&2 && exit /b 1

@GOTO :EOF

: end batch / begin powershell #>

$ErrorActionPreference = "Stop"

if ($env:MVNW\_VERBOSE -eq "true") {

$VerbosePreference = "Continue"

}

# calculate distributionUrl, requires .mvn/wrapper/maven-wrapper.properties

$distributionUrl = (Get-Content -Raw "$scriptDir/.mvn/wrapper/maven-wrapper.properties" | ConvertFrom-StringData).distributionUrl

if (!$distributionUrl) {

Write-Error "cannot read distributionUrl property in $scriptDir/.mvn/wrapper/maven-wrapper.properties"

}

switch -wildcard -casesensitive ( $($distributionUrl -replace '^.\*/','') ) {

"maven-mvnd-\*" {

$USE\_MVND = $true

$distributionUrl = $distributionUrl -replace '-bin\.[^.]\*$',"-windows-amd64.zip"

$MVN\_CMD = "mvnd.cmd"

break

}

default {

$USE\_MVND = $false

$MVN\_CMD = $script -replace '^mvnw','mvn'

break

}

}

# apply MVNW\_REPOURL and calculate MAVEN\_HOME

# maven home pattern: ~/.m2/wrapper/dists/{apache-maven-<version>,maven-mvnd-<version>-<platform>}/<hash>

if ($env:MVNW\_REPOURL) {

$MVNW\_REPO\_PATTERN = if ($USE\_MVND) { "/org/apache/maven/" } else { "/maven/mvnd/" }

$distributionUrl = "$env:MVNW\_REPOURL$MVNW\_REPO\_PATTERN$($distributionUrl -replace '^.\*'+$MVNW\_REPO\_PATTERN,'')"

}

$distributionUrlName = $distributionUrl -replace '^.\*/',''

$distributionUrlNameMain = $distributionUrlName -replace '\.[^.]\*$','' -replace '-bin$',''

$MAVEN\_HOME\_PARENT = "$HOME/.m2/wrapper/dists/$distributionUrlNameMain"

if ($env:MAVEN\_USER\_HOME) {

$MAVEN\_HOME\_PARENT = "$env:MAVEN\_USER\_HOME/wrapper/dists/$distributionUrlNameMain"

}

$MAVEN\_HOME\_NAME = ([System.Security.Cryptography.MD5]::Create().ComputeHash([byte[]][char[]]$distributionUrl) | ForEach-Object {$\_.ToString("x2")}) -join ''

$MAVEN\_HOME = "$MAVEN\_HOME\_PARENT/$MAVEN\_HOME\_NAME"

if (Test-Path -Path "$MAVEN\_HOME" -PathType Container) {

Write-Verbose "found existing MAVEN\_HOME at $MAVEN\_HOME"

Write-Output "MVN\_CMD=$MAVEN\_HOME/bin/$MVN\_CMD"

exit $?

}

if (! $distributionUrlNameMain -or ($distributionUrlName -eq $distributionUrlNameMain)) {

Write-Error "distributionUrl is not valid, must end with \*-bin.zip, but found $distributionUrl"

}

# prepare tmp dir

$TMP\_DOWNLOAD\_DIR\_HOLDER = New-TemporaryFile

$TMP\_DOWNLOAD\_DIR = New-Item -Itemtype Directory -Path "$TMP\_DOWNLOAD\_DIR\_HOLDER.dir"

$TMP\_DOWNLOAD\_DIR\_HOLDER.Delete() | Out-Null

trap {

if ($TMP\_DOWNLOAD\_DIR.Exists) {

try { Remove-Item $TMP\_DOWNLOAD\_DIR -Recurse -Force | Out-Null }

catch { Write-Warning "Cannot remove $TMP\_DOWNLOAD\_DIR" }

}

}

New-Item -Itemtype Directory -Path "$MAVEN\_HOME\_PARENT" -Force | Out-Null

# Download and Install Apache Maven

Write-Verbose "Couldn't find MAVEN\_HOME, downloading and installing it ..."

Write-Verbose "Downloading from: $distributionUrl"

Write-Verbose "Downloading to: $TMP\_DOWNLOAD\_DIR/$distributionUrlName"

$webclient = New-Object System.Net.WebClient

if ($env:MVNW\_USERNAME -and $env:MVNW\_PASSWORD) {

$webclient.Credentials = New-Object System.Net.NetworkCredential($env:MVNW\_USERNAME, $env:MVNW\_PASSWORD)

}

[Net.ServicePointManager]::SecurityProtocol = [Net.SecurityProtocolType]::Tls12

$webclient.DownloadFile($distributionUrl, "$TMP\_DOWNLOAD\_DIR/$distributionUrlName") | Out-Null

# If specified, validate the SHA-256 sum of the Maven distribution zip file

$distributionSha256Sum = (Get-Content -Raw "$scriptDir/.mvn/wrapper/maven-wrapper.properties" | ConvertFrom-StringData).distributionSha256Sum

if ($distributionSha256Sum) {

if ($USE\_MVND) {

Write-Error "Checksum validation is not supported for maven-mvnd. `nPlease disable validation by removing 'distributionSha256Sum' from your maven-wrapper.properties."

}

Import-Module $PSHOME\Modules\Microsoft.PowerShell.Utility -Function Get-FileHash

if ((Get-FileHash "$TMP\_DOWNLOAD\_DIR/$distributionUrlName" -Algorithm SHA256).Hash.ToLower() -ne $distributionSha256Sum) {

Write-Error "Error: Failed to validate Maven distribution SHA-256, your Maven distribution might be compromised. If you updated your Maven version, you need to update the specified distributionSha256Sum property."

}

}

# unzip and move

Expand-Archive "$TMP\_DOWNLOAD\_DIR/$distributionUrlName" -DestinationPath "$TMP\_DOWNLOAD\_DIR" | Out-Null

Rename-Item -Path "$TMP\_DOWNLOAD\_DIR/$distributionUrlNameMain" -NewName $MAVEN\_HOME\_NAME | Out-Null

try {

Move-Item -Path "$TMP\_DOWNLOAD\_DIR/$MAVEN\_HOME\_NAME" -Destination $MAVEN\_HOME\_PARENT | Out-Null

} catch {

if (! (Test-Path -Path "$MAVEN\_HOME" -PathType Container)) {

Write-Error "fail to move MAVEN\_HOME"

}

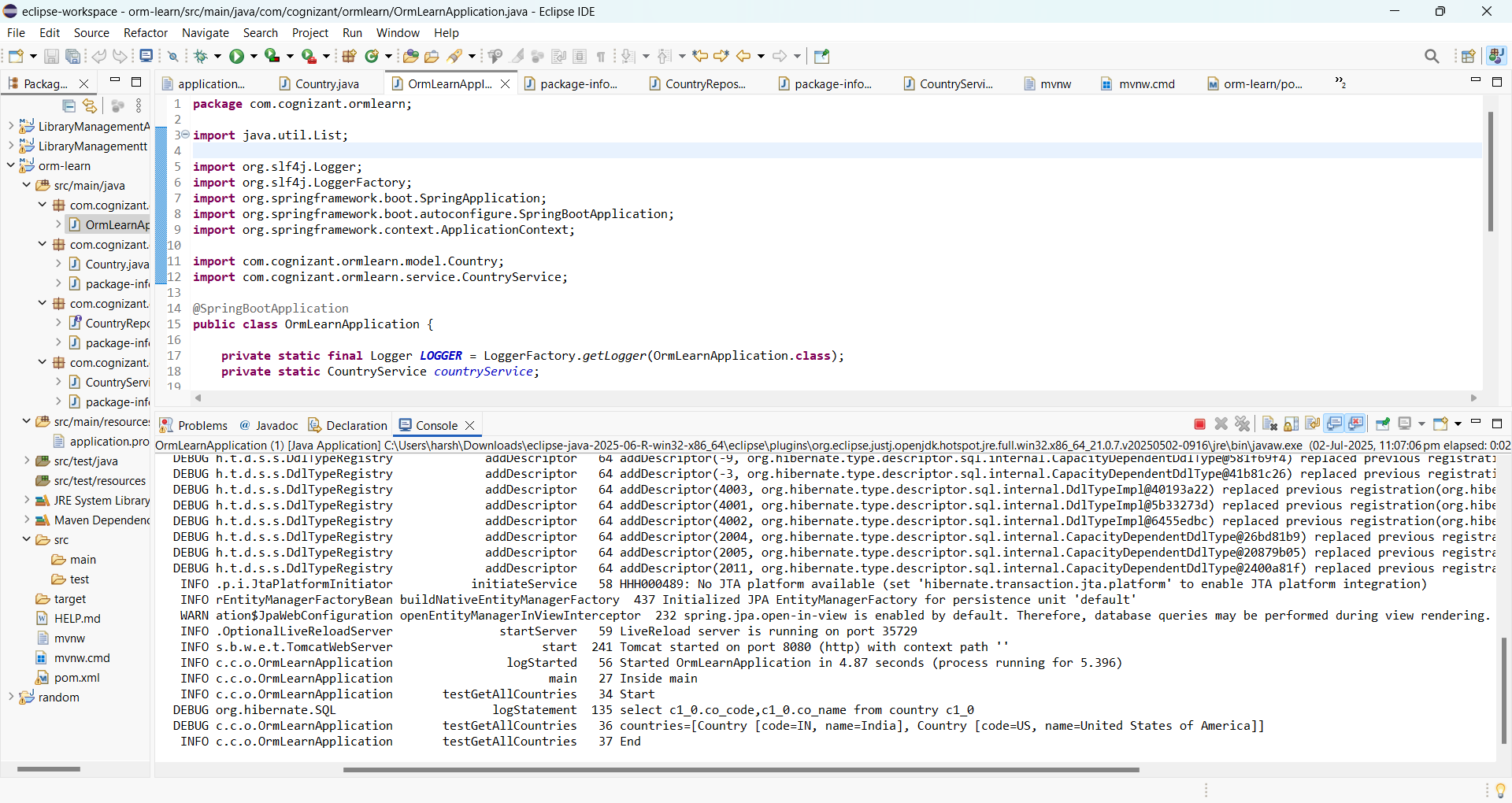
} finally {

try { Remove-Item $TMP\_DOWNLOAD\_DIR -Recurse -Force | Out-Null }

catch { Write-Warning "Cannot remove $TMP\_DOWNLOAD\_DIR" }

}

Write-Output "MVN\_CMD=$MAVEN\_HOME/bin/$MVN\_CMD"

**OUTPUT:**

**pox.xml**

<project xmlns="http://maven.apache.org/POM/4.0.0"

xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

xsi:schemaLocation="http://maven.apache.org/POM/4.0.0

https://maven.apache.org/xsd/maven-4.0.0.xsd">

<modelVersion>4.0.0</modelVersion>

<groupId>com.cognizant</groupId>

<artifactId>orm-learn</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>orm-learn</name>

<description>Demo project for Spring Data JPA and Hibernate</description>

<packaging>jar</packaging>

<parent>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-parent</artifactId>

<version>3.2.5</version>

<relativePath/> <!-- lookup parent from repository -->

</parent>

<properties>

<java.version>17</java.version> <!-- Make sure you're using Java 17 or later -->

</properties>

<dependencies>

<!-- Spring Boot Starter Data JPA -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<!-- MySQL JDBC Driver -->

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<scope>runtime</scope>

</dependency>

<!-- Spring Boot DevTools (optional for hot reload) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-devtools</artifactId>

<scope>runtime</scope>

</dependency>

<!-- Spring Boot Starter Web (optional, useful for REST APIs) -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-web</artifactId>

</dependency>

<!-- Test Dependencies -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-test</artifactId>

<scope>test</scope>

</dependency>

<!-- Spring Boot JPA Dependency -->

<dependency>

<groupId>org.springframework.boot</groupId>

<artifactId>spring-boot-starter-data-jpa</artifactId>

</dependency>

<!-- MySQL Connector -->

<dependency>

<groupId>com.mysql</groupId>

<artifactId>mysql-connector-j</artifactId>

<scope>runtime</scope>

</dependency>

</dependencies>

<build>

<plugins>

<!-- Maven Compiler Plugin -->

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-compiler-plugin</artifactId>

<version>3.10.1</version>

<configuration>

<source>${java.version}</source>

<target>${java.version}</target>

</configuration>

</plugin>

</plugins>

</build>

</project>

**OUTPUT:**

