TopLink Mavenize

Mavenize EclipseLink

Target

Transform EL build system from Apache Ant and Maven (based on Tycho) into pure Maven build system and use there standard Maven directory structure for source, resource and test files. This leads into
removal of binary files (jar files), Java IDE (IDEA, Eclipse) files and other non-project related files.

Execute typical build command "*mvn clean install*" for a build and SRG tests without any additional configuration (property files, testing DB).

Source location

https://github.com/rfelcman/eclipselink/tree/Mavenize_EclipseLink

Project/source transformation

Dependencies

Synchronization scripts mentioned bellow needs following utilities: **bash, rsync**

Description

Current project source and resource files are copied into new project directories/modules by synchronization scripts. Current source files are not modified by transformation process. There is always new copy
in new maven module/directory. Transformation scripts are in *buildsystem/mavenize* directory.
How transformation works (example based on core module):
* Current CORE module in Ant is divided into two modules:
  a. *foundation/org.eclipse.persistence.core* with main source and resource files
  b. *foundation/eclipselink.core.test* with test test source and resource files
* These modules are transformed/merged into single new Maven module *foundation/org.eclipse.persistence.core.maven*
  Note: every new Maven module directory ends with '.maven' suffix.
    Sometimes there are more, than two input modules merged into Mavenized output (MOXy)
* With every new Maven module are related three scripts. For example CORE has *core_environment.sh, core_prepare.sh, core_cleanup.sh*
  core_environment.sh.....set environment variables like source/origin main project path, source/origin test project path.

core_prepare.sh.....copy files into target project
core_cleanup.sh.....delete files from target project.
Note: Why delete? This is prevention against wrong commit into git. In development phase I want commit into git only new 'pom.xml' files.
* There are some global scripts too. *environment.sh, prepare.sh, cleanup.sh*
environment.sh.....there are file synchronization/move/remove commands. In development phase 'rsync -av' command is used to copy synchronize files from origin into new Maven projects.
Before switch into production it will be changed into 'git mv' command. See line with 'export SYNC_CMD=${RSYNC_CMD}'. '${SYNC_CMD}' is used in all prepare scripts.
prepare.sh.....Calls other project related preparation scripts. It's initialization script. It can be called repeatably (due rsync behind). After git resync/rebase don't forget call
this script to resync sources in new Maven project directories.
cleanup.sh.....Opposite to prepare. Remove/delete source files from new Maven projects. pom.xml files will remain untouched.
* If you want resync all sources call *prepare.sh*. If you want resync source files for one project for example CORE call *core_prepare.sh*.

prepare.sh .... this initial step to populate maven modules/dirs by source and resource files. It must be called after checkout from git and before Maven build commands.

Project output files/bundles

Project output files/bundles (eclipselink.jar, eclipselink.zip, javadoc files....) created as a result of a full build are available in *distribution.maven/target* directory.

In the *buildsystem/mavenize* directory are scripts compare_*.sh which can be used to compare output files produced by Ant (build-distribution target) and files/bundles produced by Maven build.
There scripts needs following command line utilities: **diff, zipinfo, sort**

Maven build/build commands

mvn clean install -DskipTests

Build without Oracle dependencies (JDBC driver, XDK, Oracle Spatial....)

mvn clean install -DskipTests -P oracle

Full build with Oracle specific modules and output bundles (eclipselink.jar, eclipselink.zip, javadoc bundles, source bundles. Oracle dependencies are required.

mvn clean install

There are currently blockers which is prevents to execute this command against Apache Derby database see chapter "mvn clean install - blockers/errors".

mvn test -pl :org.eclipse.persistence.core

Execute SRG tests in *org.eclipse.persistence.core* module against default Apache Derby database started by Maven plugin in InMemory mode. No additional configuration required.

mvn test -pl :org.eclipse.persistence.core -P mysql

Execute SRG tests in *org.eclipse.persistence.core* module against MySQL DB (must be installed and configured). Test property file *el-test.mysql.properties* must be available in ${user.home} directory. Property file templates are located in *buildsystem/mavenize/test.properties* directory.

mvn test -pl :org.eclipse.persistence.core -P oracle

Same as command above, but testing database is Oracle (currently 12c). It is not so easy as MySQL because Oracle dependencies (JDBC driver, XDB, XDK.....) are not freely available on Maven central. For more details about Oracle dependencies see Maven profiles/oracle chapter.

mvn test -pl :org.eclipse.persistence.moxy -Dtest=org.eclipse.persistence.testing.jaxb.externalizedmetadata.ExternalizedMetadataTestSuite

Execute just single test suite from some Maven module (*org.eclipse.persistence.moxy*).

mvn clean verify -pl :org.eclipse.persistence.jpa.test -P server-test-jpa-beanvalidation

JPA server side test. It creates server application (ear) and client application, starts JEE server (default is WildFly), deploys it to JEE server, calls client app under maven-failsafe-plugin and stops JEE server. For more details about WildFly see Maven profiles/wildfly chapter. WildFly binaries must be available on the same machine. Testing DB is Apache Derby.

mvn clean verify -pl :org.eclipse.persistence.jpa.test -P server-test-jpa-beanvalidation,weblogic

Same as above but with WebLogic Server.

mvn clean verify -pl :org.eclipse.persistence.jpa.test -P server-test-jpa-beanvalidation,weblogic,oracle

And again with WebLogic Server and Oracle DB.

mvn clean install -pl :org.eclipse.persistence.performance.test -P test-performance
Execute performance tests against Apache Derby.

mvn clean install - blockers/errors (Apache Derby only)

This is list of errors when typical Maven command "mvn clean install" will be executed. There errors are related only with Apache Derby database used as a default testing DB.

org.eclipse.persistence.dbws (EclipseLink DBWS)

Non standard SQL syntax "CREATE TABLE IF NOT EXISTS <table name>..." used in dbsetup_*.sql scripts. Use just "CREATE TABLE <table name>"

org.eclipse.persistence.corba (EclipseLink CORBA Extension)

A lock could not be obtained within the time requested (in ComplexMultipleUnitOfWorkTest, UnitOfWorkComplexRefreshTest).

org.eclipse.persistence.nosql (EclipseLink NoSQL Extension)

Needs MongoDB no SQL database. Maybe enable tests from this module by mongodb profile only?

org.eclipse.persistence.sdo.server.test (EclipseLink SDO Test Server)

Disable server side tests during "mvn clean install"? Test fails on WildFly.

org.eclipse.persistence.jpa.wdf.test (EclipseLink JPA WDF Test)

Test fails on Apache Derby DB. Trouble with table TMP_PROFILE.

org.eclipse.persistence.jpars.server.test (EclipseLink JPA-RS Server Test)

Disable server side tests during "mvn clean install"? Hardcoded MySQL driver reference.

org.eclipse.persistence.dbws.builder (EclipseLink DBWS Builder)

Hard-coded MySQL driver reference.

Testing environment

Default database for tests is Apache Derby started in InMemory mode. Default JEE server for server side tests is WildFly 15.0.1.Final.
Default settings for a test environment are loaded from *buildsystem/mavenize/test.properties/el-test.derby.properties* and *buildsystem/mavenize/test.properties/el-testjee.wildfly.properties*. This is fallback location.
If You need modify test property file (DB url/username/password) copy it to the ${user.home} directory like */home/oracle* and modify properties there. Files *el-test.derby.properties el-testjee.wildfly.properties* are fully commented.
For example there are different files for MySQl and Oracle DBs (*el-test.mysql.properties*, *el-test.oracle.properties*). *el-test.mysql.properties* file is used as a default for tests. In relation with *test.*.properties* files there are Maven profiles: like mysql (sets as a default test file *el-test.mysql.properties*) and oracle (for *el-test.oracle.properties*). Similar design is used for JEE servers too (*el-testjee.wildfly.properties* -> Maven profile: wildfly).

Note: Why *el-test.***.properties* instead of *test.properties* . I think, that *test.properties* file name is too generic and could leads into collisions with another software.

DBs

Apache Derby [default]

This is default test database fully handled by Maven build process. No any additional installation is required. Database is started in InMemory mode. It's started/stopped by org.carlspring.maven:derby-maven-plugin. Log file is located *${project.build.directory}/derby* directory.
Activated by "derby" profile. Active as a default DB. Property file is *el-test.derby.properties* .
Note: InMemory mode is activated by DB URL (jdbc:derby://localhost/**memory**:ecltests;create=true) and DB property createDatabase=create;databaseName=**memory**:ecltests .

Version: 10.14.2.0

MySQL

Maria DB could be used too. Tests using password authentication (in some cases must configured/enabled). This DB must be installed from distribution packages or DB installer.
Activated by "mysql" profile. Property file is *el-test.mysql.properties* .

Version: 5.xx

Oracle

Some tests needs SYSDBA permissions for more details about Oracle DB settings/permission see "Oracle DB Permissions/settings" chapter.
There are some Oracle DB special Maven modules like:
    org.eclipse.persistence.oracle, org.eclipse.persistence.oracle.nosql, org.eclipse.persistence.oracle.spatial
and testing modules
    org.eclipse.persistence.jpa.oracle.test, org.eclipse.persistence.dbws.oracle.test, org.eclipse.persistence.dbws.builder.oracle.test, org.eclipse.persistence.dbws.builder.oracle.server.test

There are some Oracle specific dependencies which is not freely available for a download from Maven central.
There are two possible ways how to make these dependencies available:

- Install Oracle DB or DB client to the build/test machine and register these dependencies localy by *maven_oracle_dependencies.sh* script from *buildsystem/mavenize* directory. Check/modify there environment variables on the begining of the script (ORACLE_HOME, FMW_HOME).
- Add to the Maven Oracle repository (not tested).

All these dependencies are organized under oracle profile in Maven

Activated by "oracle" profile. Property file is *el-test.oracle.properties* .

Version: 12.2.0.1

Mongodb

Must be installed on the build/testing machine. By default there is no any authentication/authorization required.

Used in org.eclipse.persistence.nosql and org.eclipse.persistence.jpa.nosql.test

Property file is *el-test.mongodb.properties* .

Version: 3.2.0

JEE Servers

Server binaries must be available/installed on build/test machine. Location of server binaries/server home is specified by *cargo.container.installation.home* in the property file. No any domain/configuration is required.

Server life-cycle (start, configure datasources, deploy test application, shutdown) is handled by Maven (org.codehaus.cargo:cargo-maven2-plugin).

Log files are available in *${project.build.directory}/logs* directory. Server domain configuration files/logs are in *${project.build.directory}/cargo/configurations*.

WildFly [default]

This default testing server.

Property file is *el-testjee.wildfly.properties* .

There is additional configuration step to set project EclipseLink available as a WildFly module (overide default implementation). Maven plugin

org.codehaus.mojo:xml-maven-plugin is used with XLST stylesheet *eclipselink-wildfly-module.xsl* and elResources.xml data file. This is direct modification of *$WILDFLY_HOME/modules/system/layers/base/org/eclipse/persistence/main/module.xml* . This step is called before WildFly server start.

JDKs: 8, 11

Version: 15.0.1.Final

WebLogic Server

Property file is *el-test.weblogic.properties* .

JDKs: 8

Version: 12.2.1.3.0

Glassfish

Property file is *el-test.glassfish.properties* .

JDKs: 8

Version: 5.1.0

Issues

Functional

- NoSQL JPA server tests (all) are not functional (in Ant version too)
- Oracle JPA server test (proxy user) is not functional

Performance

- MOXy LRG tests - due some troubles process forking is disabled => (test runs 100% longer than Ant version).
- JPA server side tests (LRG). Each test has following life-cycle (clean, build test, start server, deploy to the server, call tests, stop server). This approach is slower but more safe to ensure, that server environment is not modified by previous test.
  Due Travis timeout (50 min) these tests are divided into two parts.
- Travis: every test specified in matrix starts with new build. One build per each JDK specified in the matrix could be enough (one per JDK 8 and one per JDK 11)

Improvements (nice to have)

- Merge JPA Oracle test module (org.eclipse.persistence.jpa.oracle.test) into Oracle Extension module (org.eclipse.persistence.oracle)
- Merge JPA NoSQL test module (org.eclipse.persistence.jpa.nosql.test) into NoSQL Extension module (org.eclipse.persistence.nosql)
- Move classes related with EL test framework into separated module (migrate to JUnit 5 ?)
- Maven plugin for weaving
- Maven plugin for JPA class generator

Maven global profiles

APIs

jpa22 [active by default]

Location: parent pom, jpa/eclipselink.jpa.test.maven

Dependency version for jakarta.persistence is 2.2.2 .

jpa21

Location: parent pom, jpa/eclipselink.jpa.test.maven

Dependency version for javax.persistence is 2.1.1 .

In module jpa/eclipselink.jpa.test.maven some *.jpa22.* packages are excluded from compilation.

DBs

derby [active by default]

Location: parent pom

Loads *el-test.derby.properties* .

mysql

Location: parent pom

Loads *el-test.mysql.properties* .

oracle

Location: parent pom

Loads *el-test.oracle.properties* .

JEE Servers

wildfly [active by default]

Location: parent pom

Loads *el-testjee.wildfly.properties* .

glassfish

Location: parent pom

Loads *el-testjee.glassfish.properties* .

weblogic

Location: parent pom

Loads *el-testjee.weblogic.properties* .

Performace module

test-performance

Location: parent pom

Makes accessible org.eclipse.persistence.performance.test module.

Maven modules

Parent

Parent pom.xml

Location: project root directory

Global profiles like derby, oracle, wildfly, jpa22 are specified there. Complete list of dependencies (versions are in properties).

EclipseLink Commons (org.eclipse.persistence.commons)

Location: *commons.maven*

Contains some common resources like *about.html, license.html, readme.html* files.

Plugins

EclipseLink ANTLR (org.eclipse.persistence.antlr)

Location: *plugins/org.eclipse.persistence.antlr.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	N/A	N/A

Downloads and repackage source files from ANTLR project.
Maven dependency

```
<dependency>
  <groupId>org.antlr</groupId>
  <artifactId>antlr-runtime</artifactId>
</dependency>
```

For repackaging there are following Maven modules:
org.apache.maven.plugins:maven-dependency-plugin.....to unpack origin sources
com.sun.wts.tools.ant:package-rename-task.....to repackage sources

It change classes package from *org.antlr* to *org.eclipse.persistence.internal.libraries.antlr* .

TODO Use substitution string for release version in *about.html, readme.html.*

EclipseLink ASM (org.eclipse.persistence.asm)

Location: *plugins/org.eclipse.persistence.asm.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	OK	N/A

Downloads and repackage source files from ANTLR project.
Maven dependencies

```
<dependency>
    <groupId>org.ow2.asm</groupId>
    <artifactId>asm</artifactId>
<classifier>sources</classifier>
</dependency>
<dependency>
    <groupId>org.ow2.asm</groupId>
    <artifactId>asm-commons</artifactId>
<classifier>sources</classifier>
</dependency>
<dependency>
   <groupId>org.ow2.asm</groupId>
   <artifactId>asm-tree</artifactId>
<classifier>sources</classifier>
</dependency>
```

For repackaging there are following Maven modules:

org.apache.maven.plugins:maven-dependency-plugin.....to unpack origin sources

com.sun.wts.tools.ant:package-rename-task.....to repackage sources

It change classes package from *org.objectweb.asm* to *org.eclipse.persistence.internal.libraries.asm* .

TODO Use substitution string for release version in *about.html, readme.html.*

Foundation

EclipseLink Core (org.eclipse.persistence.core)

Location: *foundation/org.eclipse.persistence.core.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	OK	MySQL - OK (there is system dependency to the tools.jar) (Oracle12c throws for 3 tests "Unsupported feature: createArrayOf": org.eclipse.persistence.testing.tests.customsqlstoredprocedures.CustomSQLTestModel org.eclipse.persistence.testing.tests.insurance.InsuranceObjectRelationalTestModel org.eclipse.persistence.testing.tests.workbenchintegration.MappingWMIntegrationStoredProcedureTestModel) **Derby - hangs during tests**

Merges: *foundation/org.eclipse.persistence.core, foundation/eclipselink.core.test* Ant projects.

There is Maven filtering option applied to src/main/resources and src/test/resources. Following Maven properties are used there (dbPlatform, dbUser, dbPassword, driverClass, dbURL, loglevel). Values for these properties comes from test.*.properties file selected by profile (mysql[default], oracle) in parent pom.
There is name transformation because properties in test.*.properties files has different name, than substitution strings in test resource files.
As a part of build (initialize phase) this module generates version.properties file with basic build properties (version, build date, time, git commit id).

Profiles: see profiles part

TODO: Solve/remove system dependency to the tools.jar (required for LRG-TESTS). In JDK 8 there are different locations for Linux, MacOS

EclipseLink NoSQL Extension (org.eclipse.persistence.nosql)

Location: *foundation/org.eclipse.persistence.nosql.maven*

Status	
Build	**Tests**
OK	OK

Merges: *foundation/org.eclipse.persistence.nosql, foundation/eclipselink.extension.nosql.test* Ant projects.

Unit tests are executed against MongoDB. DB connection properties comes from *el-test.mongodb.properties*.

In origin Ant test module there are two executions of the same test (different classpath), but currently environment is same. Maven module calls test class only once.

EclipseLink CORBA Extension (org.eclipse.persistence.corba)

Location: *foundation/org.eclipse.persistence.corba.maven*

Status	
Build	**Tests**

OK	NoSQL - OK Oracle - OK **Derby - Error**

Merges: *foundation/org.eclipse.persistence.corba, foundation/eclipselink.extension.corba.test Ant projects.*

EclipseLink Extension (org.eclipse.persistence.extension)

Location: *foundation/org.eclipse.persistence.extension.maven*

Status	
Build	**Tests**
OK	OK

Merges: *foundation/org.eclipse.persistence.extension, foundation/eclipselink.extension.test Ant projects.*

DB isn't required for tests.

EclipseLink Oracle Extension (org.eclipse.persistence.oracle)

Location: *foundation/org.eclipse.persistence.oracle.maven*

Status	
Build	**Tests**
OK	**One error:** **Exception Description: Failed to copy the version number to the remote system** **at org.eclipse.persistence.testing.tests.distributedservers.rcm.broadcast.BroadcastSetupHelper$TestWrapperWithEventLock.verify(BroadcastSetupHelper.java:119)**

Merges *foundation/org.eclipse.persistence.oracle, foundation/eclipselink.extension.oracle.test* Ant projects.

Tests:
org.eclipse.persistence.testing.tests.xdb.XDBTestModel
org.eclipse.persistence.testing.tests.xdb.XDBTestModelMWIntegration
org.eclipse.persistence.testing.tests.unwrappedconnection.UnwrapConnectionXDBTestModel
can be executed by *jdbc:oracle:thin* connection instead of *jdbc:oracle:oci* .

OCI driver requires ORACLE_HOME and LD_LIBRARY_PATH env variables specified.

EclipseLink Oracle NoSQL Extension (org.eclipse.persistence.oracle.nosql)

Location: *foundation/org.eclipse.persistence.oracle.nosql.maven*

Status	
Build	**Tests**
OK	OK

Merges: *foundation/org.eclipse.persistence.oracle.nosql, foundation/eclipselink.extension.oracle.nosql.test* Ant projects.

Unit tests are executed against Oracle NoSQL DB and Oracle DB (Advanced Queue).

EclipseLink Oracle Spatial Extension Test (org.eclipse.persistence.oracle.spatial)

Location: *foundation/eclipselink.extension.oracle.spatial.test.maven*

Status

Build	Tests
OK	OK

Migration from: *foundation/eclipselink.extension.oracle.spatial.test* Ant project.

JPA

EclipseLink Hermes Parser (org.eclipse.persistence.jpa.jpql)

Location: *jpa/org.eclipse.persistence.jpa.jpql.maven*

Status	
Build	**Tests**
OK	OK

Merges *jpa/org.eclipse.persistence.jpa.jpql*, *jpa/org.eclipse.persistence.jpa.jpql.test* Ant projects.

EclipseLink JPA (org.eclipse.persistence.jpa)

Location: *jpa/org.eclipse.persistence.jpa.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	N/A	N/A

Migration from: *jpa/org.eclipse.persistence.jpa* Ant project.

EclipseLink JPA Model Generator (org.eclipse.persistence.jpa.modelgen)

Location: *org.eclipse.persistence.jpa.modelgen.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	OK	**N/A**

Migration from: *jpa/org.eclipse.persistence.jpa.modelgen*

There is test class 'org.eclipse.persistence.jpa.test.modelgen.TestProcessor' moved from EclipseLink JPA JSE TEST (org.eclipse.persistence.jpa.jse.test) module.

EclipseLink JPA TEST (org.eclipse.persistence.jpa.test)

Location: *jpa/eclipselink.jpa.test.maven*

Status			
Build	**Tests (SRG)**	**Tests (LRG)**	**Server Tests (LRG)**
OK	OK	OK	OK

Migration from: *jpa/eclipselink.jpa.test* Ant project.

Some test classes *_.java are generated by EclipseLink Canonical Model Processor via maven-processor-plugin .

TODO some persistence.xml (eclipselink-composite-advanced-model-member_, eclipselink-composite-advanced-model-member_2) contains DB URLs to Otawa.

EclipseLink Oracle JPA Test (org.eclipse.persistence.jpa.oracle.test)

Location: *jpa/eclipselink.jpa.oracle.test.maven*

This is Oracle specific version of "org.eclipse.persistence.jpa.test" module. It's solution of cross dependency collision between "org.eclipse.persistence.jpa.test" and "org.eclipse.persistence.oracle" modules.
This is new module. There are server side tests only.

Environment: Oracle DB, WebLogic Server only

Status	
Build	**Server Tests (LRG)**
OK	**Error in: server-test-jpa-proxy-authentication**

EclipseLink JPA NoSQL Test (org.eclipse.persistence.jpa.nosql.test)

Location: *jpa/eclipselink.jpa.nosql.test.maven*

This is NoSQL specific version of "org.eclipse.persistence.jpa.test" module. It's solution of cross dependency collision between "org.eclipse.persistence.jpa.test" and "org.eclipse.persistence.nosql" modules.

Status	
Build	**Server Tests (LRG)**
OK	**Error: Not functional (like Ant version)** **Bugs in *mongo-dynamic.xml, mongo-orm.xml* files.** **MongoDB authentication must be enabled -> collision with test org.eclipse.persistence.nosql module (without authentication) -> use different DB or modify test.**

EclipseLink JPA JSE TEST (org.eclipse.persistence.jpa.jse.test)

Location: *jpa/eclipselink.jpa.test.jse.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	Derby - OK MySQL - OK Oracle - OK	N/A

Migration from: *jpa/eclipselink.jpa.test.jse* Ant project.
TODO: There is bug in JPA ApplyConverters see JPA Bug:539323 . This bug happens only when weaving is disabled. By default static weaving is enabled.

EclipseLink JPA-RS (org.eclipse.persistence.jpars)

Location: *jpa/org.eclipse.persistence.jpars.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	Derby - OK MySQL - OK Oracle - OK	N/A

Merge *jpa/org.eclipse.persistence.jpars, jpa/eclipselink.jpars.test* Ant projects.
TODO modify substitution strings like @DB_USER@ into @db.user@
Check [WARNING] Corrupted STDOUT by directly writing to native stream in forked JVM 1. See FAQ web page and the dump file /.....

EclipseLink JPA Spring TEST (org.eclipse.persistence.jpa.spring.test)

Location: *jpa/eclipselink.jpa.spring.test.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	Derby - OK **MySQL - TODO (.....Table 'ecltests.SPRING_TLE_ROUTE' doesn't exist....)** Oracle - OK	N/A

Migration from: *jpa/eclipselink.jpa.spring.test* Ant project.

TODO Update project/module files to use standard pattern @*@

EclipseLink JPA JAXRS Test (org.eclipse.persistence.jpa.jaxrs.test)

Location: *jpa/eclipselink.jaxrs.test.maven*

Status	
Build	**Integration Tests**
OK	**TODO Problem with deployment to any JEE server - check against standalone JEE server.** **Wrong location of binding-address.xml and binding-phonenumber.xml files**

Migration from: *jpa/eclipselink.jaxrs.test* Ant project.

In origin project modify:

1.move "binding-*.xml" files to META-INF
2.move "jaxb.properties" to org.eclipse.persistence.testing.jaxrs.model folder
3. modify package name in "binding-*.xml" files

EclipseLink JPA WDF Test (org.eclipse.persistence.jpa.wdf.test)

Location: *jpa/eclipselink.jpa.wdf.test.maven*

Status		
Build	**Integration Tests**	**Tests (LRG)**
OK	**TODO**	**Derby - (SQL CREATE TABLE error for TMP_PROFILE, BYTE_ITEM, TMP_VEHICLE_PROFILE tables) '...Syntax error: Encountered "FOR"....'** MySQL - OK Oracle - OK

Migration from: *jpa/eclipselink.jpa.wdf.test* Ant project.

EclipseLink JPA-RS Server Test (org.eclipse.persistence.jpars.server.test)

Location: *jpa/eclipselink.jpars.test.maven*

Status	
Build	**Integration Tests**
OK	**Errors**

Migration from: *jpa/eclipselink.jpars.test.* Ant project.

TODO: Hard coded reference to MySQL DB driver.

MOXy

EclipseLink MOXy (org.eclipse.persistence.moxy)

Location: *moxy/org.eclipse.persistence.moxy.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	OK	OK

Merges: *moxy/org.eclipse.persistence.moxy, moxy/org.eclipse.persistence.moxy.dynamicxjc, moxy/eclipselink.moxy.test* Ant projects.

There are similar test profiles/executions test-moxy-oxm, test-moxy-oxm-dom, test-moxy-oxm-deploymentxml, test-moxy-oxm-deploymentxml-tl with different platform types (SAX, DOM, DOC_PRES) and metadata types (JAVA, XML_ECLIPSELINK, XML_TOPLINK). There is no DB and JEE server required.

EclipseLink MOXy XJC (org.eclipse.persistence.moxy.utils.xjc)

Location: *moxy/org.eclipse.persistence.moxy.utils.xjc.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	N/A	N/A

Migration from: *utils/eclipselink.utils.jaxb* Ant project.

DBWS

EclipseLink DBWS (org.eclipse.persistence.dbws)

Location: *dbws/org.eclipse.persistence.dbws.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	Derby - **TODO Wrong syntax in SQL initialization scripts** MySQL - OK Oracle - **TODO Wrong syntax in SQL initialization scripts**	N/A

Merges: *dbws/org.eclipse.persistence.dbws, dbws/eclipselink.dbws.test* Ant projects.

DB is required for tests. There is usage of sql-maven-plugin to prepare DB before tests and cleanup DB after tests.

EclipseLink DBWS Test Oracle (org.eclipse.persistence.dbws.oracle.test)

Location: *dbws/eclipselink.dbws.test.oracle*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	OK	N/A

Merges dbws/eclipselink.dbws.test.oracle Ant projects.

Oracle DB is required. It's available only if Maven **oracle** profile is active.

SDO

EclipseLink SDO (org.eclipse.persistence.sdo)

Location: *sdo/org.eclipse.persistence.sdo.maven*

Status		
Build	**Tests (SRG)**	**Tests (LRG)**
OK	OK	OK

Merges *sdo/org.eclipse.persistence.sdo, sdo/eclipselink.sdo.test* Ant projects.

There no DB required for tests.

EclipseLink SDO Test Server (org.eclipse.persistence.sdo.server.test)

Location: *sdo/eclipselink.sdo.test.server.maven*

Status	
Build	**Integration Tests**
OK	GlassFish - OK WebLogic - OK WildFly - **testDepartmentService(org.eclipse.persistence.testing.sdo.server.DeptServiceClientTestCases) javax.ejb.EJBException: java.lang.IllegalArgumentException: argument type mismatch**

Migration from: *sdo/eclipselink.sdo.test.server* Ant project.

There no DB required for tests.

Utils

EclipseLink DBWS Builder (org.eclipse.persistence.dbws.builder)

Location: *utils/org.eclipse.persistence.dbws.builder.maven*

Status	
Build	**Tests**
OK	**OK** **Hard coded MySQL driver -> mysql only**

Merges *utils/org.eclipse.persistence.dbws.builder, utils/eclipselink.dbws.builder.test* Ant projects.

EclipseLink DBWS Builder Test Oracle (org.eclipse.persistence.dbws.builder.oracle.test)

Location: *utils/eclipselink.dbws.builder.test.oracle.maven*

Status	
Build	**Tests**
OK	**Problem with dbsetup_alltests.sql scripts (PR #267)**

Migration from *utils/eclipselink.dbws.builder.test.oracle* Ant projects.

Oracle DB is required. It's active only if Maven **oracle** profile is active.

EclipseLink DBWS Builder Test Oracle Server (org.eclipse.persistence.dbws.builder.oracle.server.test)

Location: *utils/eclipselink.dbws.builder.test.oracle.server.maven*

Status	
Build	**Tests**
OK	OK

Migration from *utils/eclipselink.dbws.builder.test.oracle.server* Ant projects.

Oracle DB is required. It's active only if Maven **oracle** profile is active.

EclipseLink Package Rename Utility (org.eclipse.persistence.utils.rename)

Location: *utils/eclipselink.utils.rename.maven*

Status	
Build	**Tests**
OK	N/A

Migration from utils/eclipselink.utils.rename Ant projects.

EclipseLink Package Signature Compare Utility (org.eclipse.persistence.utils.sigcompare)

Location: *utils/eclipselink.utils.sigcompare.maven*

Status	
Build	**Tests**
OK	N/A

Migration from *utils/eclipselink.utils.sigcompare* Ant projects.

Others

EclipseLink Performance Test Oracle (org.eclipse.persistence.performance.test)

Location: *performance/eclipselink.perf.test.maven*

Status	
Build	**Tests**
OK	Derby - OK MySQL - OK Oracle - OK

Migration from *performance/eclipselink.perf.test* Ant projects.

It's active only if Maven **test-performance** profile is active.

Could be started by: *mvn test -pl :org.eclipse.persistence.performance.test -P test-performance*

EclipseLink Distribution (org.eclipse.persistence.distribution)

Location: *distribution.maven*

This is new module. It is used as a place to generate output files/bundles like *eclipselink.jar, eclipselink.zip, eclipselink-src.jar* and other outputs.

Travis CI

Travis CI is configured in similar way like Ant version with one exception. org.eclipse.persistence.distribution (build-distribution target in Ant) with installer and OSGi tests is not called there due a Oracle dependencies (JDBC driver...).

Oracle DB Permissions/settings

org.eclipse.persistence.core:

```
connect
create any context
create view
resource
query rewrite
execute on dbms_rls
```

org.eclipse.persistence.oracle:

```
aq_administrator_role
execute on dbms_flashback
execute on dbms_aq
```

Creation script:

```
CREATE USER scott IDENTIFIED BY tiger
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE temp
QUOTA UNLIMITED ON users;

GRANT CONNECT , RESOURCE, QUERY REWRITE TO scott;
GRANT CREATE VIEW, CREATE ANY CONTEXT TO scott;
GRANT EXECUTE ON dbms_rls TO scott;
GRANT aq_administrator_role TO scott;
GRANT EXECUTE ON dbms_flashback TO scott;
GRANT EXECUTE ON dbms_aq TO scott;
/
```

DB Settings

Module *org.eclipse.persistence.dbws.builder.oracle.test* needs increase DB initialization parameter to *"open_cursors"* to 800.

See DB command (run as SYSDBA):

```
ALTER SYSTEM SET open_cursors = 800 SCOPE=BOTH;
```