



1)_환경 설정 셋팅

- 1. ojdbc 추가
- 2. maven - dependency 사용 방법
- 3. dependency 추가
 - 1. DB 관련 라이브러리
 - 2. 스프링 관련 라이브러리
 - 3. MyBatis 관련 라이브러리 추가
- 4. 데이터베이스 관련 설정
 - 1. log4jdbc
 - 2. log4jdbc 파일 추가
 - 3. root-context.xml 내용 수정

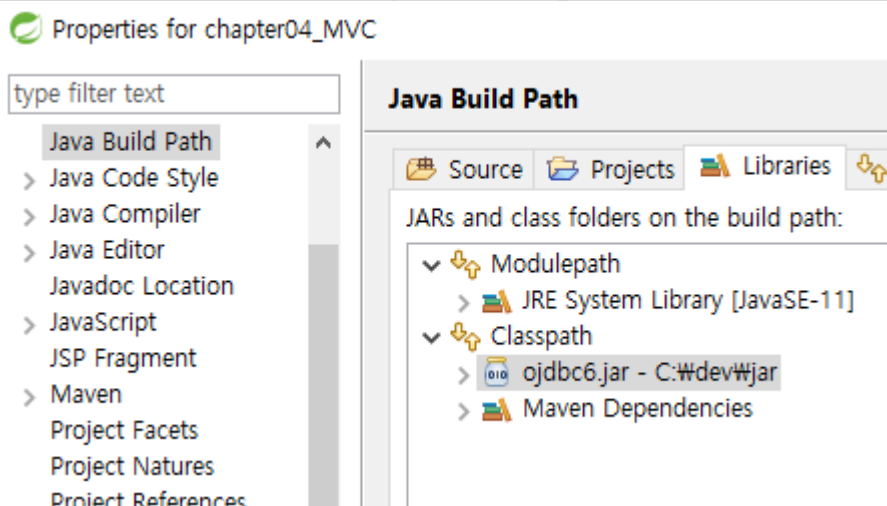
이제 본격적으로 CRUD를 위한 DB 연결을 해볼 예정이다.

이번 챕터를 들어가기 전 [인코딩과 라이브러리 변경&추가는 이전 챕터를 참고](#)하면 된다.

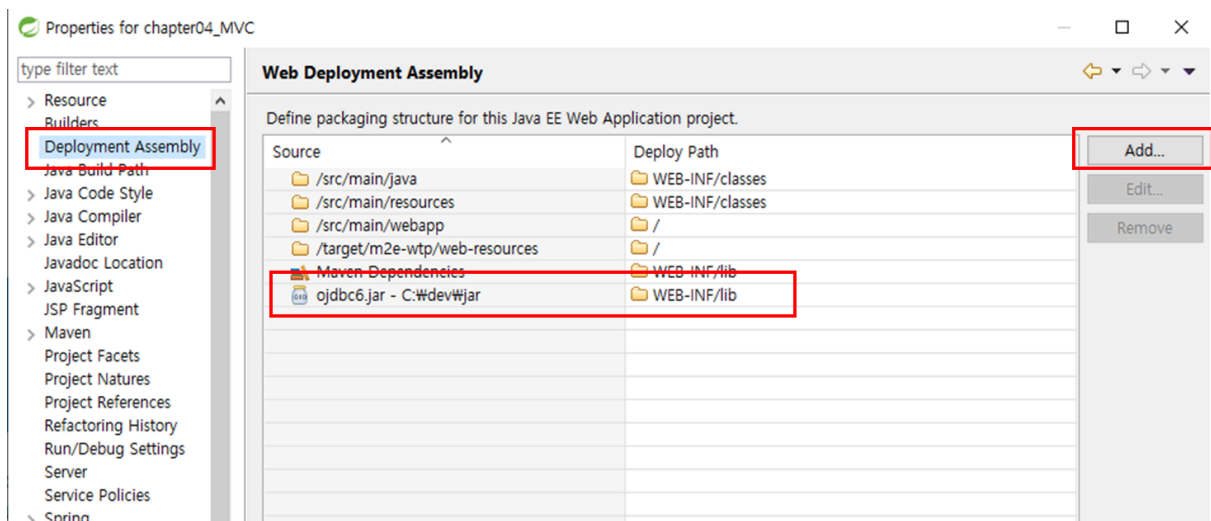
1. ojdbc 추가



!!! ojdbc.jar 파일은 maven으로 추가 시 간헐적인 에러를 발생하므로 물리 파일도 추가해주도록 한다.



빌드패스 해 준 뒤



Deployment Assembly 메뉴에서 빌드패스 한 jar 를 추가해준다.!!!

2. maven - dependency 사용 방법

각 프로젝트의 pom.xml 파일을 열어보면 중간 태그 중에 <dependencies> 태그를 확인할 수 있다.

내부 태그들을 보면 모두 <dependency></dependency> 로 이루어져 있다.

즉 필요한 라이브러리(디펜던시) 들을 가져와서 하나씩 추가해주면 된다.

추가하기 위해 사이트로 이동한다. <https://mvnrepository.com/>

The screenshot shows the Maven Repository website. At the top, there's a search bar with the text "Search for groups, artifacts, categories". Below the search bar, the "Indexed Artifacts (37.1M)" section features a line graph showing the growth of projects from 2008 to 2018. The graph shows a steady increase, reaching approximately 14 million projects by 2018. Below the graph, the "Popular Categories" section lists several categories: Testing Frameworks & Tools, Android Packages, Logging Frameworks, Java Specifications, JSON Libraries, and JVM Languages. On the right side, the "What's New in Maven" section highlights three recent releases: SonarQube Java :: Plugin, Core Nordic Common Library, and SonarQube Java :: Checks Test Tool Kit. Each release includes its group ID, artifact ID, version, and the date of the last release.

← → ↻ <https://mvnrepository.com>

MVN REPOSITORY

Search for groups, artifacts, categories

Indexed Artifacts (37.1M)


Projects (millions)


Year


Popular Categories

- Testing Frameworks & Tools
- Android Packages
- Logging Frameworks
- Java Specifications
- JSON Libraries
- JVM Languages

What's New in Maven

**SonarQube Java :: Plugin**
[org.sonarsource.java » sonar-java-plugin](#) » 7.30.1.34514
Code Analyzer for Java
Last Release on Jan 4, 2024

**Core Nordic Common Library.**
[no.nordicsemi.android.common » core](#) » 1.9.4
Nordic Android Common Libraries
Last Release on Jan 4, 2024

**SonarQube Java :: Checks Test Tool Kit**
[org.sonarsource.java » java-checks-testkit](#) » 7.30.1.34514

검색 창에 필요한 라이브러리를 적는다.

Repository

- Central 111
- Sonatype 31
- Spring Lib M 16
- Spring Plugins 15
- Gradle Plugins 6
- JCenter 6
- IBiblio 4
- ImageJ Public 4

Group

- com.github 15
- io.github 13
- de.plushnikov 10
- org.jetbrains 8
- org.projectlombok 8
- com.thinkinglogic 4
- dev.pcms 3
- io.freefair 3

Category

- Maven Plugins 6
- Bill of materials 1

lombok

Found 135 results

Sort: **relevance** | popular | newest

- 1. Project Lombok**
[org.projectlombok » lombok](#)

Lombok is a Java library that provides annotations to simplify Java development by automating the generation of boilerplate code, hashCode, and toString methods, as well as a facility for automatic resource management. It aims to reduce the amount of manual errors.

Last Release on Sep 20, 2023
- 2. Lombok Maven Plugin**
[org.projectlombok » lombok-maven-plugin](#)

Maven Plugin for Project Lombok

Last Release on Apr 2, 2021
- 3. Lombok Mapstruct Binding**
[org.projectlombok » lombok-mapstruct-binding](#)

Binding for Lombok and Mapstruct, to allow them to cooperate.

Last Release on Dec 4, 2020
- 4. Collection of Lombok Related Gradle Plugins**
[io.freefair.gradle » lombok-plugin](#)

Collection of Lombok related Gradle plugins

lombok 에 대한 것도 여러가지가 나오기 때문에 잘 확인해야 한다.

사용할 내용을 눌러서 들어간다.

[Home](#) » [org.projectlombok](#) » lombok

Project Lombok

Lombok is a Java library that provides annotations to simplify Java development by automating the generation of boilerplate code. Key features include automatic generation of getters, setters, equals, hashCode, and toString methods, as well as a facility for automatic resource management. It aims to reduce the amount of manual coding, thereby streamlining the codebase and reducing potential for errors.

License	MIT
Categories	Code Generators
Tags	lombok codegen code generator
Ranking	#13 in MvnRepository (See Top Artifacts) #1 in Code Generators
Used By	22,145 artifacts

[Central \(51\)](#)
[Lombok \(7\)](#)
[Codemc Public \(1\)](#)
[ICM \(5\)](#)

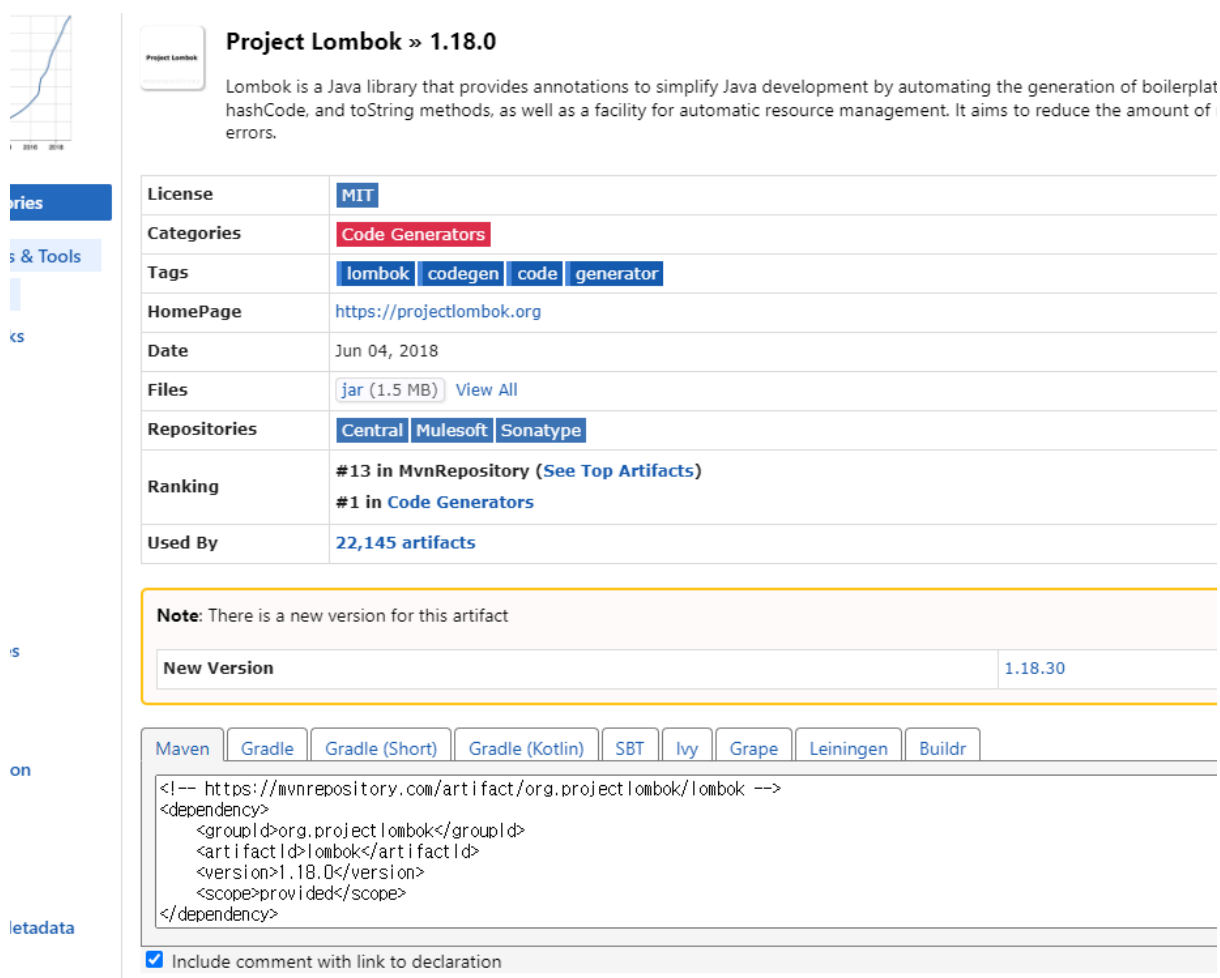
	Version	Vulnerabilities	Repository	Usages	Date
1.18.x	1.18.30		Central	1,590	Sep 20, 2023
	1.18.28		Central	1,569	May 25, 2023
	1.18.26		Central	1,862	Feb 03, 2023
	1.18.24		Central	4,081	Apr 07, 2022
	1.18.22		Central	3,292	Oct 07, 2021
	1.18.20		Central	2,919	Apr 02, 2021
	1.18.18		Central	872	Jan 28, 2021
	1.18.16		Central	1,943	Oct 15, 2020
	1.18.14		Central	211	Oct 08, 2020
	1.18.12		Central	3,200	Feb 07, 2020

최신 버전 순 내림차순 정렬이 되어있고, 사용량 및 적용 날짜까지 나와있다.

여기서 주의할 점은 최신 버전이 무조건 좋은 것은 아니라는 점이다.

개발하는 환경과 맞는 버전을 이용한다.

사용할 버전을 골랐으면 클릭해서 상세 페이지로 이동한다.



Project Lombok » 1.18.0

Lombok is a Java library that provides annotations to simplify Java development by automating the generation of boilerplate hashCode, and toString methods, as well as a facility for automatic resource management. It aims to reduce the amount of errors.

License	MIT
Categories	Code Generators
Tags	lombok codegen code generator
HomePage	https://projectlombok.org
Date	Jun 04, 2018
Files	jar (1.5 MB) View All
Repositories	Central Mulesoft Sonatype
Ranking	#13 in MvnRepository (See Top Artifacts) #1 in Code Generators
Used By	22,145 artifacts

Note: There is a new version for this artifact

New Version 1.18.30

Maven Gradle Gradle (Short) Gradle (Kotlin) SBT Ivy Grape Leiningen Buildr

```
<!-- https://mvnrepository.com/artifact/org.project.lombok/lombok -->
<dependency>
  <groupId>org.project.lombok</groupId>
  <artifactId>lombok</artifactId>
  <version>1.18.0</version>
  <scope>provided</scope>
</dependency>
```

☒ Include comment with link to declaration

그러면 아래쪽에 Maven 탭이 있고 안에 디펜던시 내용이 있으니 복사해서 pom.xml에 그대로 붙여 넣고 maven update를 해주면 된다.

```

111
112         <!-- Test -->
113     <dependency>
114         <groupId>junit</groupId>
115         <artifactId>junit</artifactId>
116         <version>4.12</version>
117         <scope>test</scope>
118     </dependency>
119
120     <!-- https://mvnrepository.com/artifact/org.projectlombok/lombok -->
121     <dependency>
122         <groupId>org.projectlombok</groupId>
123         <artifactId>lombok</artifactId>
124         <version>1.18.0</version>
125         <scope>provided</scope>
126     </dependency>
127
128 </dependencies>
129 <build>
130     <plugins>
131         <plugin>
132             <artifactId>maven-eclipse-plugin</artifactId>

```

3. dependency 추가

mybatis와 DB 등을 연결하기 위한 디펜던시들을 추가한다.

1. DB 관련 라이브러리

- jdbc6 ⇒ 오라클과 자바 연동 라이브러리
 - <https://mvnrepository.com/artifact/com.oracle.database.jdbc/jdbc6>

```

<!-- https://mvnrepository.com/artifact/com.oracle.database.jdbc/jdbc6 -
→
<dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>jdbc6</artifactId>
    <version>11.2.0.4</version>
</dependency>

```

2. 스프링 관련 라이브러리

- spring-tx ⇒ 트랜잭션을 사용하기 위한 라이브러리 (5.0.7)

- <https://mvnrepository.com/artifact/org.springframework/spring-tx>
- spring-jdbc ⇒ DB와 연동하기 위한 라이브러리 (5.0.7)
 - <https://mvnrepository.com/artifact/org.springframework/spring-jdbc>
- spring-test ⇒ 단위 테스트를 진행하기 위한 라이브러리 (5.0.7)
 - <https://mvnrepository.com/artifact/org.springframework/spring-test>



위 3개의 라이브러리의 버전은 스프링과 맞춰 줄 것이기 때문에 maven repository에서 아무거나 받아와도 된다.
 * 5.0.7로 적어도 되고, `${org.springframework-version}` 이용해서 동적으로 적용 할 수도 있다.

```
<!-- https://mvnrepository.com/artifact/org.springframework/spring-tx →
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-tx</artifactId>
  <version>${org.springframework-version}</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework/spring-jdbc --
>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-jdbc</artifactId>
  <version>${org.springframework-version}</version>
</dependency>
<!-- https://mvnrepository.com/artifact/org.springframework/spring-test --
>
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-test</artifactId>
  <version>${org.springframework-version}</version>
  <scope>test</scope>
</dependency>
```

3. MyBatis 관련 라이브러리 추가

- HikariCP ⇒ 대용량 데이터 처리 속도를 빠르게 하기 위한 라이브러리 (2.7.8)
 - <https://mvnrepository.com/artifact/com.zaxxer/HikariCP>
- MyBatis ⇒ DB 프레임워크 (3.4.6)
 - <https://mvnrepository.com/artifact/org.mybatis/mybatis>
- mybatis-spring ⇒ MyBatis를 사용하기 위한 스프링 라이브러리 (1.3.2)
 - <https://mvnrepository.com/artifact/org.mybatis/mybatis-spring>
- Log4jdbc ⇒ 데이터베이스 실행 시, 단위 테스트를 위한 라이브러리 (1.16)
 - <https://mvnrepository.com/artifact/org.bgee.log4jdbc-log4j2/log4jdbc-log4j2-jdbc4.1>

```
<!-- https://mvnrepository.com/artifact/com.zaxxer/HikariCP →  
<dependency>  
  <groupId>com.zaxxer</groupId>  
  <artifactId>HikariCP</artifactId>  
  <version>2.7.8</version>  
</dependency>  
  
<!-- https://mvnrepository.com/artifact/org.mybatis/mybatis →  
<dependency>  
  <groupId>org.mybatis</groupId>  
  <artifactId>mybatis</artifactId>  
  <version>3.4.6</version>  
</dependency>  
  
<!-- https://mvnrepository.com/artifact/org.mybatis/mybatis-spring →  
<dependency>  
  <groupId>org.mybatis</groupId>  
  <artifactId>mybatis-spring</artifactId>  
  <version>1.3.2</version>  
</dependency>  
  
<!-- https://mvnrepository.com/artifact/org.bgee.log4jdbc-log4j2/log4jdbc-  
-log4j2-jdbc4.1 →  
<dependency>
```



```
<groupId>org.bgee.log4jdbc-log4j2</groupId>
<artifactId>log4jdbc-log4j2-jdbc4.1</artifactId>
<version>1.16</version>
</dependency>
```

4. 데이터베이스 관련 설정

1. log4jdbc

MyBatis는 내부적으로 JDBC의 PreparedStatement를 이용해서 SQL을 처리한다.

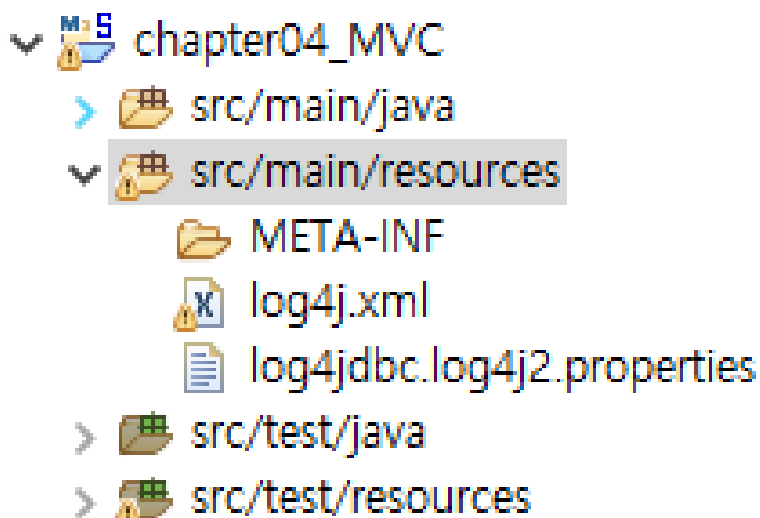
따라서 SQL에 전달되는 파라미터는 JDBC에서와 같이 '?' 로 치환되어서 처리되는데 이를 로그에서

적용된 데이터를 보기 위해서는 **log4jdbc-log4j2** 라이브러리를 사용한다.

2. log4jdbc 파일 추가

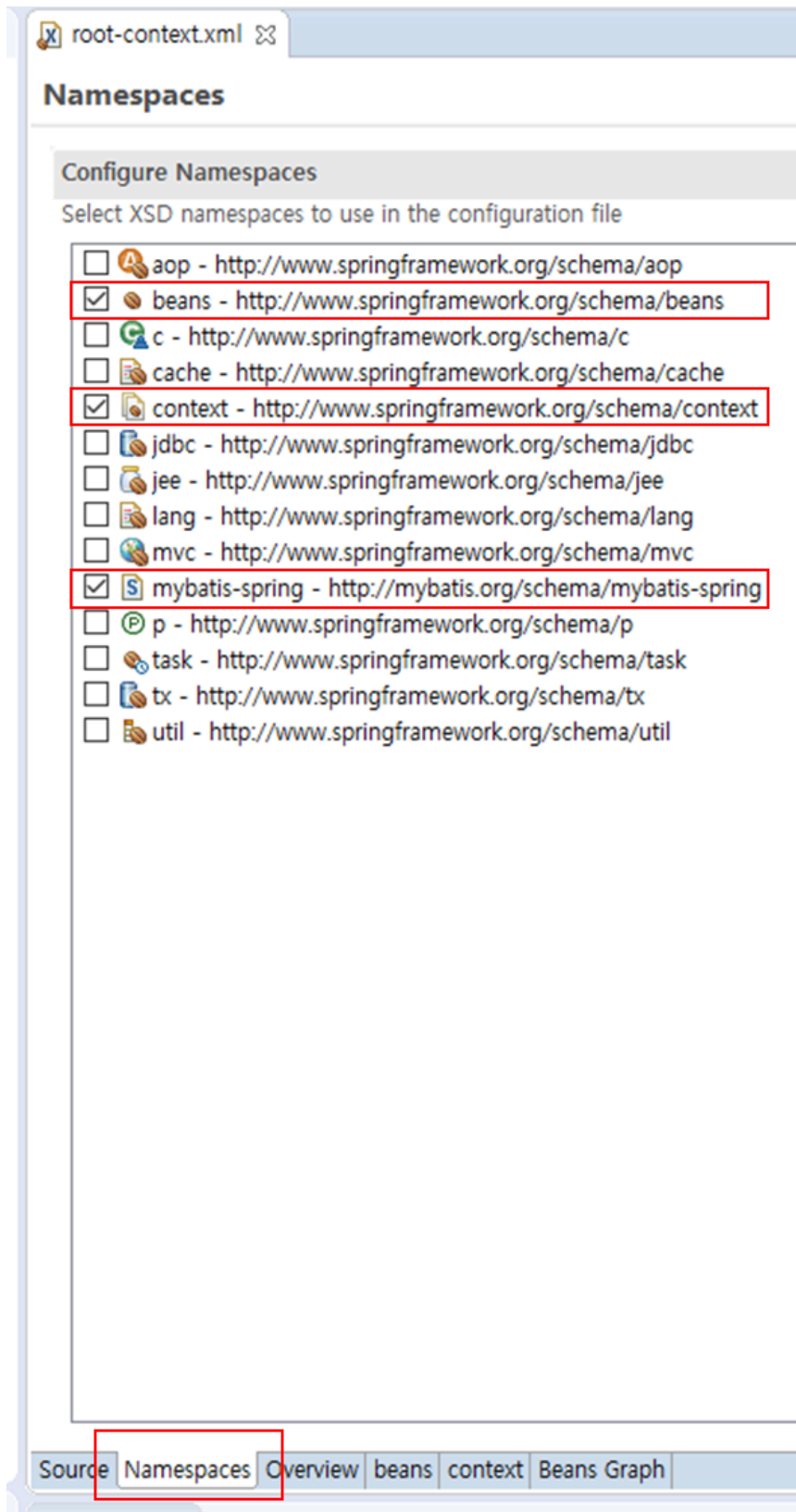
log4jdbc.log4j2.properties

위 파일을 src/main/resources 아래에 추가해주도록 한다.



3. root-context.xml 내용 수정

DB와 관련된 접속 내용들은 root-context.xml에 추가하도록 한다.



해당 bean을 선택하면 Source 안에 추가가 되어 사용할 수 있게 된다. 이제 DB에서 사용할 내용을 추가한다.

root-context.xml

```

<!-- Root Context: defines shared resources visible to all other web components →
<bean id="hikariConfig" class="com.zaxxer.hikari.HikariConfig">
    <!-- log4jdbc →
    <property name="driverClassName"
        value="net.sf.log4jdbc.sql.jdbcapi.DriverSpy"></property>
    </bean>
    <property name="jdbcUrl"
        value="jdbc:log4jdbc:oracle:thin:@localhost:1521:XE"></property>
    <property name="username" value="scott"></property>
    <property name="password" value="tiger"></property>
</bean>

<!-- HikariCP configuration →
<bean id="dataSource" class="com.zaxxer.hikari.HikariDataSource" destroy-method="close">
    <constructor-arg ref="hikariConfig"></constructor-arg>
</bean>

<bean id="sqlSessionFactory" class="org.mybatis.spring.SqlSessionFactoryBean">
    <property name="dataSource" ref="dataSource"></property>
</bean>

<mybatis-spring:scan base-package="org.joonzis.mapper"/>

```