

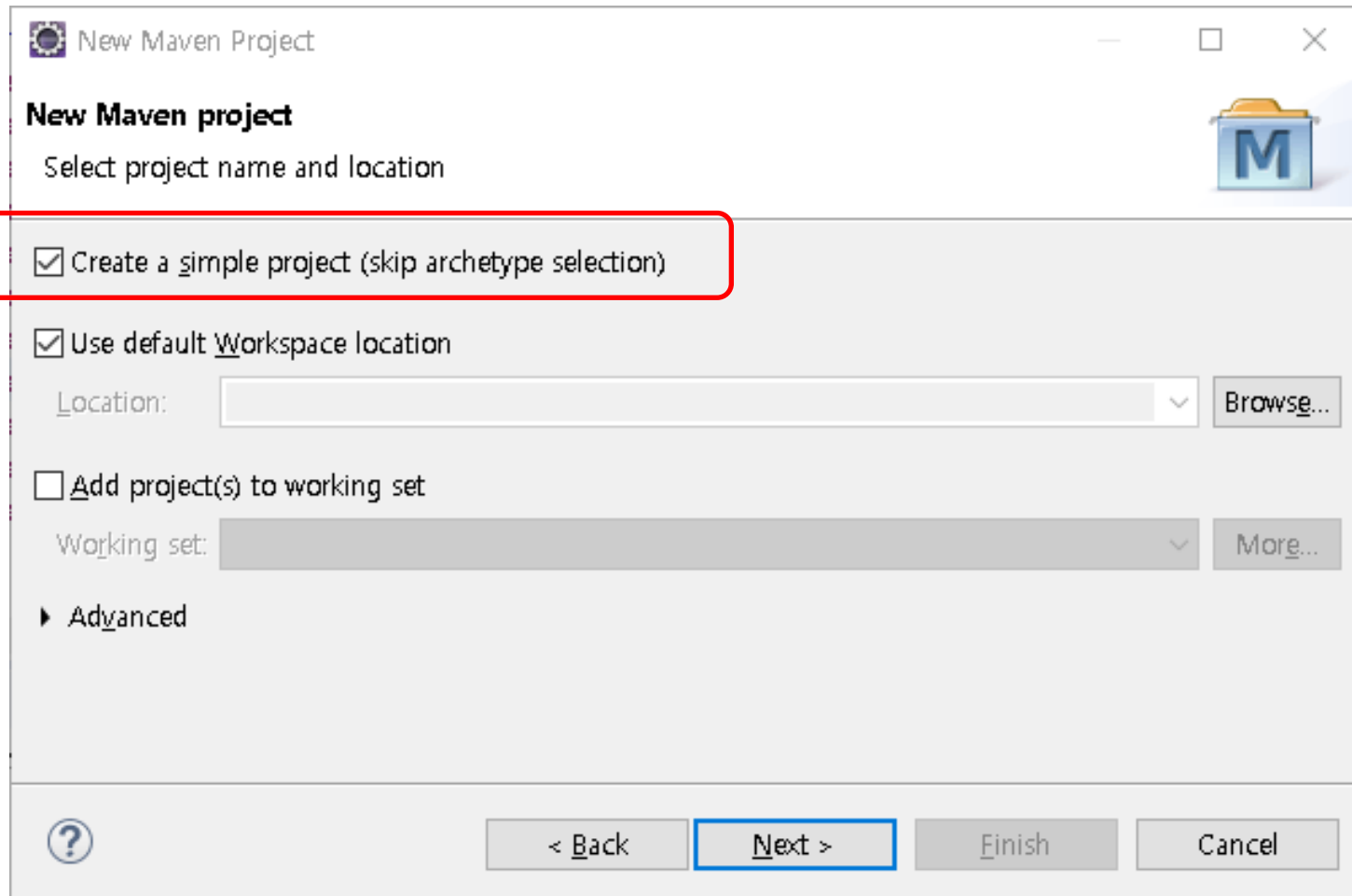
# **MyBatis**

## **- 프로젝트 준비 -**

## 프로젝트 준비

### ❖ Maven 공통 모듈 프로젝트 생성

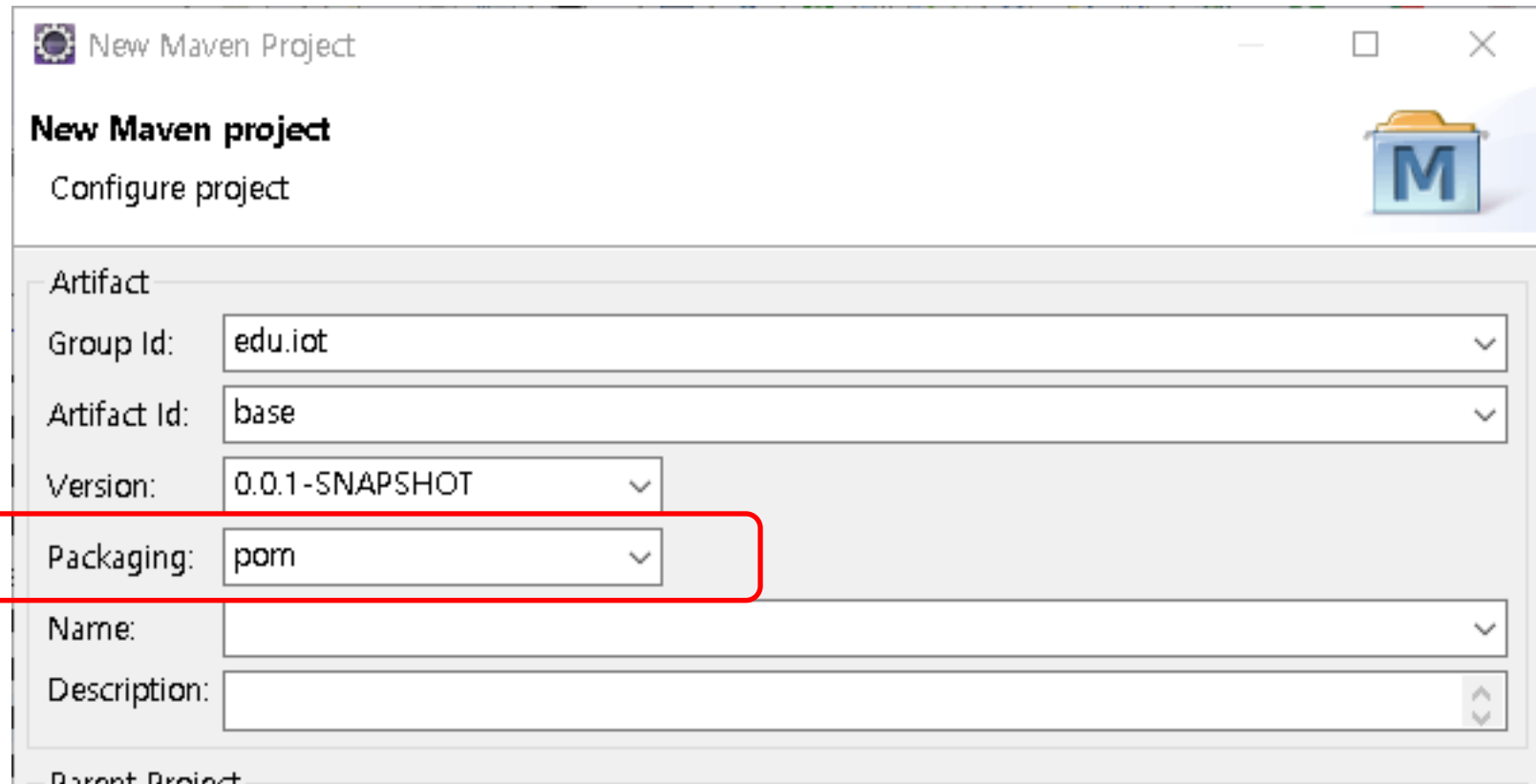
- 반복적으로 사용하는 의존성 재사용
- New > Maven Project



## 프로젝트 준비

### ❖ Maven 공통 모듈 프로젝트 생성

- Group Id : edu.iot
- Artifact Id : base
- Packaging : pom



New Maven Project

**New Maven project**

Configure project

Artifact

Group Id: edu.iot

Artifact Id: base

Version: 0.0.1-SNAPSHOT

Packaging: pom

Name:

Description:

Parent Project:

# 프로젝트 준비

---

## ❖ pom.xml

- 공통 의존성 등록
  - lombok
  - ojdbc6
  - gson
  - junit 4.12

# 프로젝트 준비

## ❖ pom.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
    http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>
  <groupId>edu.iot</groupId>
  <artifactId>base</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>pom</packaging>

  <repositories>
    <repository>
      <id>ojdbc</id>
      <url>
        http://nexus.talanlabs.com/content/repositories/releases/
      </url>
    </repository>
  </repositories>
```

## ❖ pom.xml

```
<dependencies>
  <dependency>
    <groupId>org.projectlombok</groupId>
    <artifactId>lombok</artifactId>
    <version>1.18.2</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>com.oracle</groupId>
    <artifactId>ojdbc</artifactId>
    <version>6</version>
  </dependency>
  <dependency>
    <groupId>com.google.code.gson</groupId>
    <artifactId>gson</artifactId>
    <version>2.8.5</version>
  </dependency>
</dependencies>
```

## ❖ pom.xml

```
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.12</version>
  <scope>test</scope>
</dependency>
</dependencies>

</project>
```

# 프로젝트 준비

---

## ❖ Maven 프로젝트 생성

- Artifact Id : sagittarius
- pom.xml

```
<project ...>
  <groupId>edu.iot</groupId>
  <artifactId>sagittarius</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <parent>
    <groupId>edu.iot</groupId>
    <artifactId>base</artifactId>
    <version>0.0.1-SNAPSHOT</version>
  </parent>
  :
  <dependencies>

  </dependencies>
</project>
```



# 프로젝트 준비

---

## ❖ MyBatis

- <http://www.mybatis.org/mybatis-3/ko/index.html>
- 개발자가 지정한 SQL, 저장프로시저 그리고 몇가지 고급 매핑을 지원하는 퍼시스턴스 프레임워크
- JDBC로 처리하는 상당부분의 코드와 파라미터 설정 및 결과 매핑을 대신
- 데이터베이스 레코드에 원시타입과 Map 인터페이스 그리고 자바 POJO 를 설정해서 매핑하기 위해 XML과 애노테이션을 사용
- base 프로젝트의 pom.xml에 추가

```
<dependency>
  <groupId>org.mybatis</groupId>
  <artifactId>mybatis</artifactId>
  <version>3.4.6</version>
</dependency>
```

## 프로젝트 준비

---

### ❖ MyBatis 설정 파일

- xml 파일로 운영 정보(데이터베이스 접속 정보 포함) 설정
- src/main/resources 소스 폴더에
  - config 패키지 생성
  - database.properties 파일 배치
  - mybatis-config.xml 추가

## 프로젝트 준비

### ❖ mybatis-config.xml 기본 골격 - 태그 순서 준수해야 함

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
    "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
    <!-- 외부 설정 정보 파일 -->
    <properties resource="config/database.properties" />

    <!-- MyBatis 기본 설정 내용 -->
    <settings></settings>

    <!-- 모델 객체의 별칭 설정 -->
    <typeAliases></typeAliases>

    <!-- 데이터베이스 접속 정보 설정 -->
    <environments default="development"></environments>

    <!-- Mapper 파일 위치 설정 -->
    <mappers></mappers>
</configuration>
```

### ❖ mybatis-config.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration
  PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
  "http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>

  <properties resource="config/database.properties" />

  <settings>
    <setting name="cacheEnabled" value="false" />
    <setting name="mapUnderscoreToCamelCase" value="true" />
    <setting name="useGeneratedKeys" value="false" />
  </settings>
```

### ❖ mybatis-config.xml

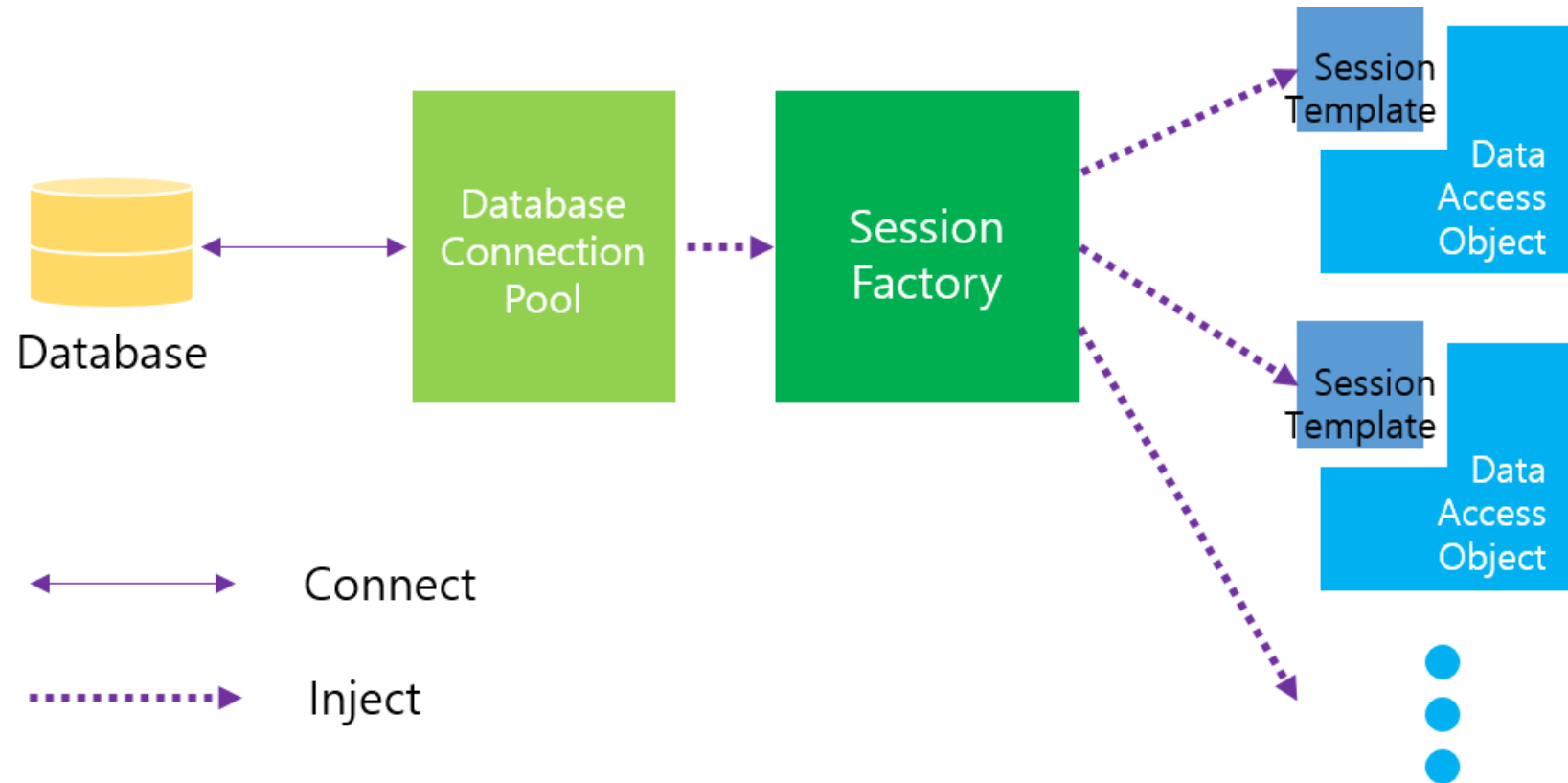
```
<typeAliases></typeAliases>

<environments default="development">
  <environment id="development">
    <transactionManager type="JDBC" />
    <dataSource type="POOLED">
      <property name="driver" value="${driver}" />
      <property name="url" value="${url}" />
      <property name="username" value="${username}" />
      <property name="password" value="${password}" />
    </dataSource>
  </environment>
</environments>

<mappers></mappers>
</configuration>
```

# 프로젝트 준비

## ❖ MyBatis 주요 객체



### ❖ MyBatis 주요 객체

- `SqlSessionFactoryBuilder`
  - MyBatis 설정을 읽어 `SqlSessionFactory` 객체 생성
- `SqlSessionFactory`
  - `SqlSession` 객체 생성 및 관리
- `SqlSession`
  - 데이터베이스와 연결된 작업공간 관리 객체
  - JDBC의 `Connection`과 같은 역할
  - SQL문 실행 및 트랜잭션 관리
    - `commit()`, `rollback()`
  - 사용 완료 후 반드시 `close()` 호출
    - `try with resource`로 처리

### ❖ Session 객체 (database 패키지)

```
public class Session {
    static private SqlSessionFactory sqlSessionFactory;
    static {
        try {
            String resource = "config/mybatis-config.xml";
            Reader reader = Resources.getResourceAsReader(resource);
            sqlSessionFactory =
                new SqlSessionFactoryBuilder().build(reader);

        } catch (Exception e) {
            e.printStackTrace();
        }
    }

    public static SqlSession getSession() {
        return sqlSessionFactory.openSession();
    }
}
```



# 프로젝트 준비

---

## ❖ 확인

```
public class App {  
    public static void main(String[] args) {  
        SqlSession session = Session.getSession();  
  
        System.out.println("MyBatis 준비 완료");  
        session.close();  
    }  
}
```

- 성공 확인 후 mybatis-config.xml의 내용을 XML 템플릿의 New XML File 카테고리로 추가