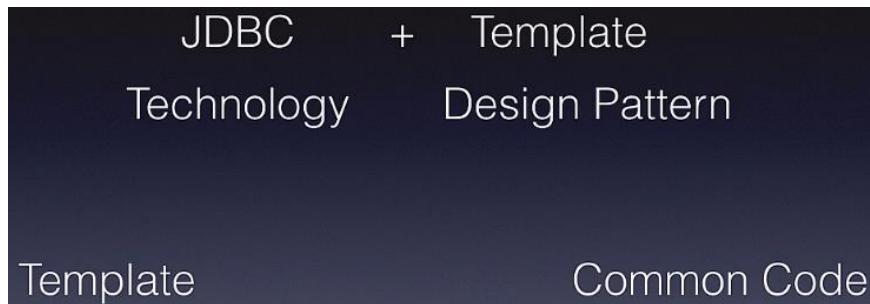


1. 소개

가. 스프링 JDBC(Java DataBase Connectivity)

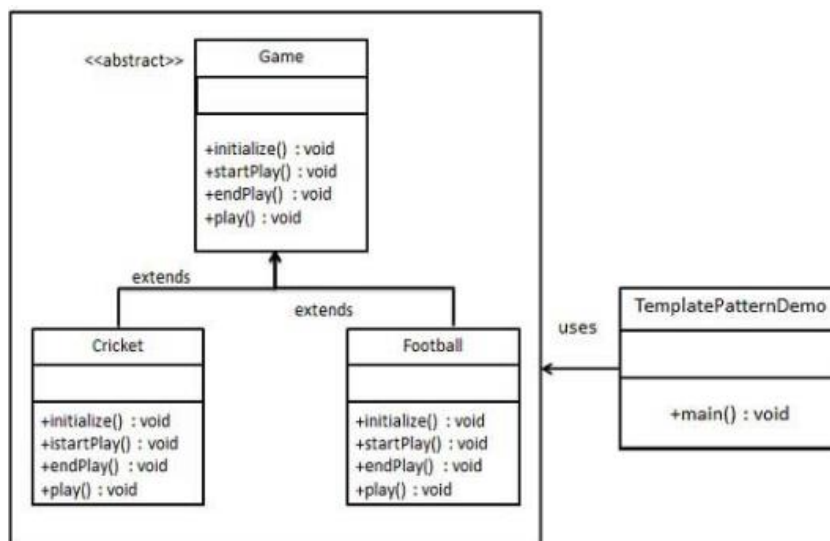


나. JDBC 템플릿



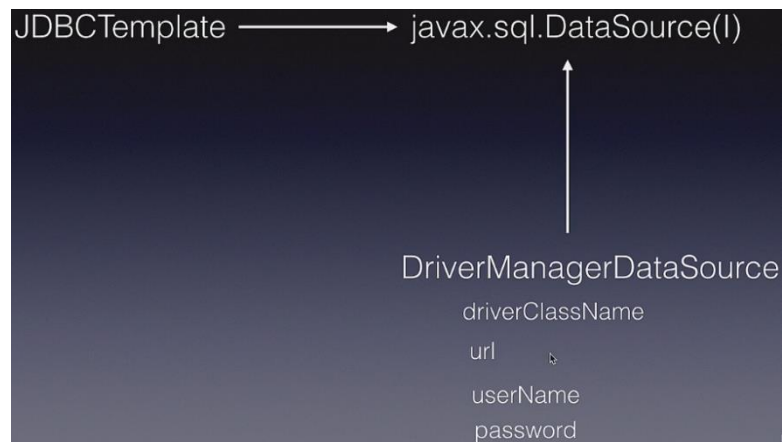
다. 템플릿 패턴

- 추상클래스가 정의된 방식/템플릿을 노출해서 메소드를 호출
- 하위(서브)클래스가 필요할 때마다 메소드 구현체를 override 할 수 있지만, 호출은 추상 클래스에 의해 정의된 것과 동일한 방식
- 행동패턴에 속함
- 예시 이미지



- 1 단계: 추상클래스 만들기
- 2단계: 구현 클래스 만들기
- 3단계: 데모클래스 실행 및 결과보기

라. JDBC 템플릿



마. 제공 메소드

```

update(String sql) int

update(String sql, Object...args) int

insert, update and delete
  
```

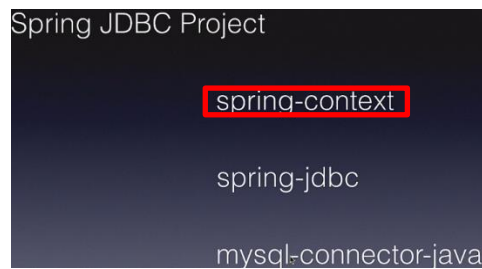
2. 테이블 준비

가. 스크립트 파일: 테이블 생성 및 조회

나. SQL DEVELOPER 실행

3. 메이븐 프로젝트 만들기

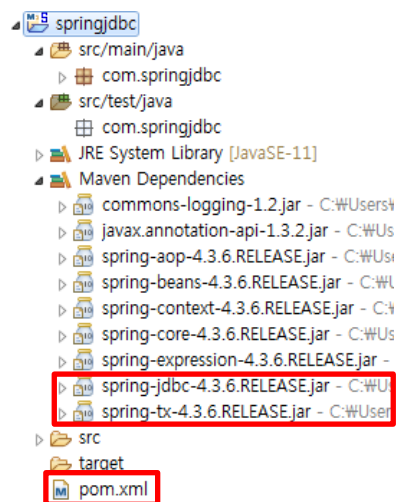
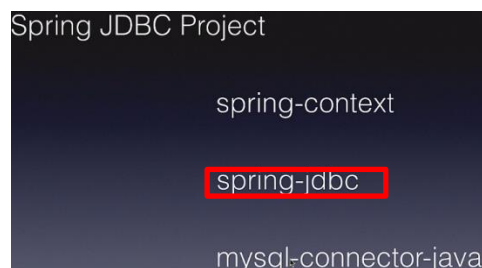
가. 스프링 컨텍스트(Spring-Context) 설정



나. 프로젝트 생성

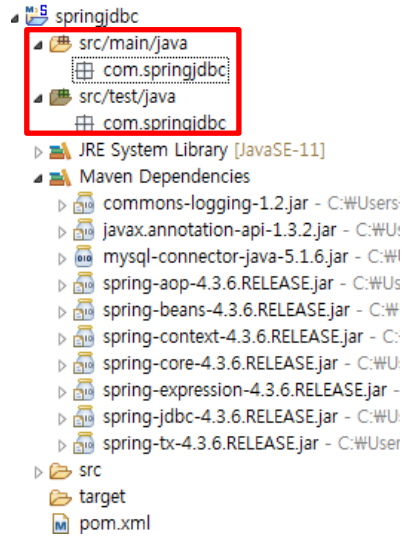
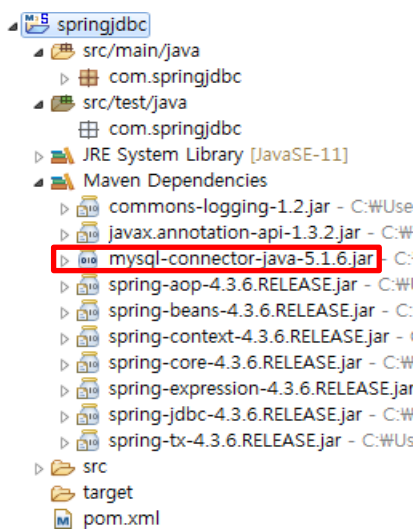
다. pom 파일 부분 수정

라. 스프링 JDBC(Spring-JDBC) 설정



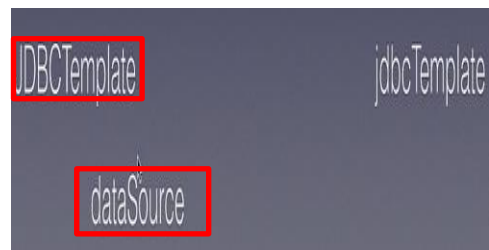
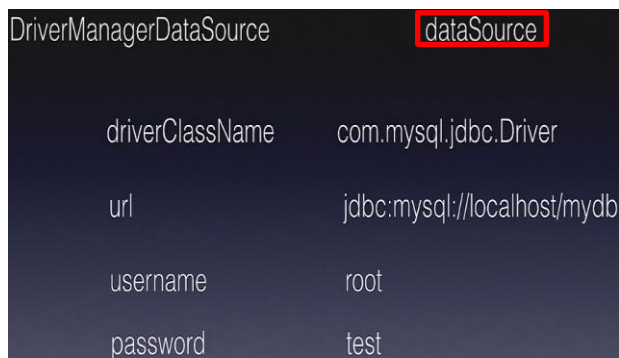
```
<dependencies>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-core</artifactId>
    <version>${springframework.version}</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-context</artifactId>
    <version>${springframework.version}</version>
  </dependency>
  <dependency>
    <groupId>org.springframework</groupId>
    <artifactId>spring-jdbc</artifactId>
    <version>${springframework.version}</version>
  </dependency>
</dependencies>
```

마. 데이터베이스 드라이버 설정



4. JDBC 템플릿을 사용하는 절차

가. 드라이버매니저 데이터소스 설정



나. 테스트 클래스 만들고 JDBC 템플릿 사용

5. 데이터소스와 JDBC 템플릿 설정

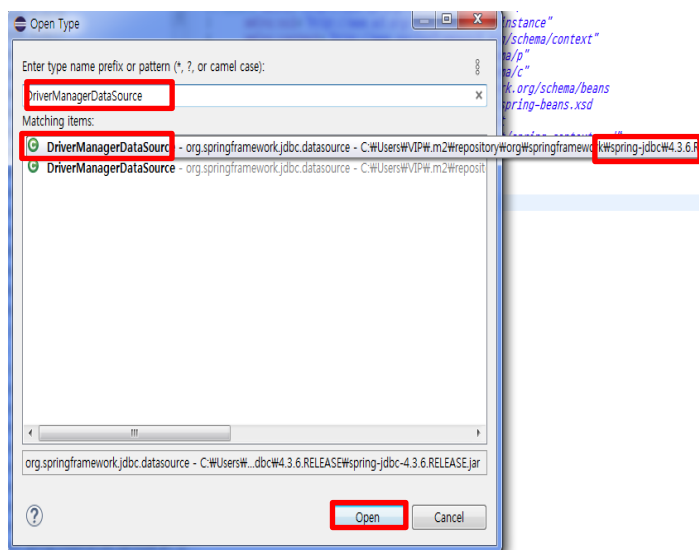
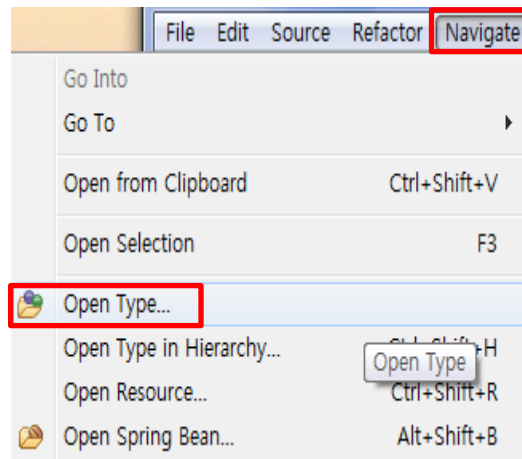


DriverManagerDataSource dataSource

```

driverClassName    com.mysql.jdbc.Driver
url                jdbc:mysql://localhost/mydb
username           root
password           test

```



config.xml DriverManagerDataSource.class

```

20 * Copyright 2002-2015 the original author or authors.
16
17 package org.springframework.jdbc.datasource;
18
19 import java.sql.Connection;
20 import java.sql.DriverManager;
21 import java.sql.SQLException;
22 import java.util.Properties;
23
24 import org.springframework.util.Assert;
25 import org.springframework.util.ClassUtils;
26
27 /**
28  * Simple implementation of the standard JDBC {@link java
29  * configuring the plain old JDBC {@link java.sql.DriverM
30

```

config.xml DriverManagerDataSource.class JdbcTemplate.class

```

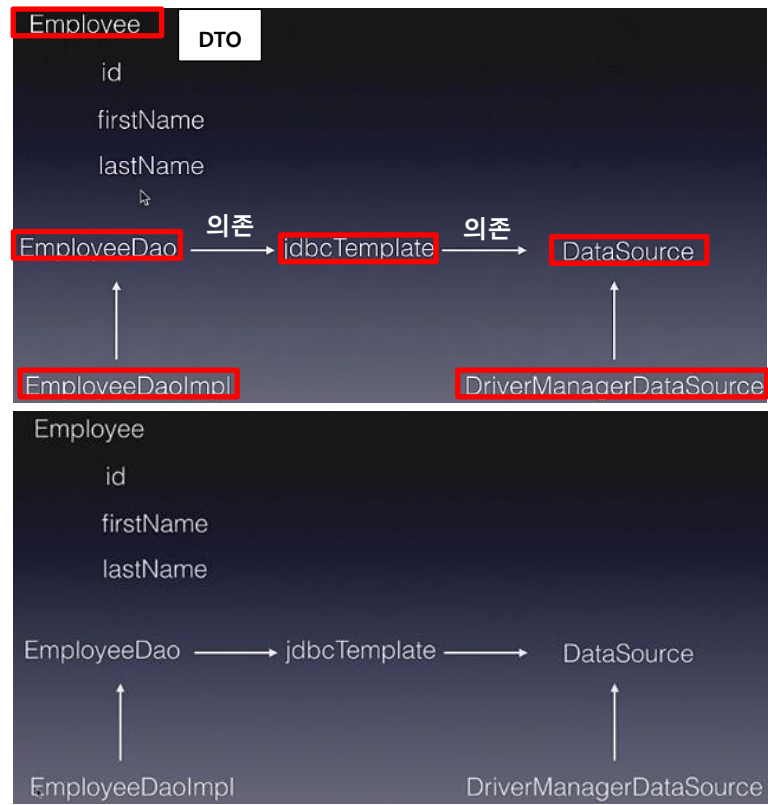
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:p="http://www.springframework.org/schema/p"
6       xmlns:c="http://www.springframework.org/schema/c"
7       xsi:schemaLocation="http://www.springframework.org/schema/beans
8                           http://www.springframework.org/schema/beans/spring-beans.xsd
9                           http://www.springframework.org/schema/context
10                          http://www.springframework.org/schema/context/spring-context.xsd">
11
12     <bean
13         class="org.springframework.jdbc.datasource.DriverManagerDataSource"
14         name="dataSource"
15         p:driverClassName="com.mysql.jdbc.Driver"
16         p:url="jdbc:mysql://localhost/mykhtdb"
17         p:username="root"
18         p:password="root" />
19
20     <bean class="org.springframework.jdbc.core.JdbcTemplate"
21         name="jdbcTemplate" p:dataSource-ref="dataSource" />
22
23 </beans>

```

6. JDBC 템플릿을 사용해 데이터 입력 테스트

7. 처리 설명

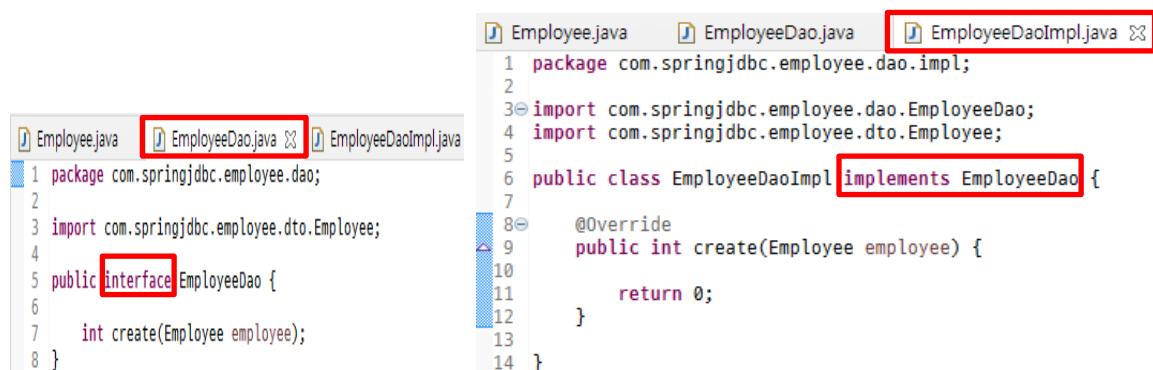
8. DTO(Data Transfer Object), DAO(Data Access Object) 클래스 만들기





9. create 메소드 구현

10. 설정파일 만들기



11. 테스트 만들고 실행하기

12. 어플리케이션 흐름도

13. 레코드(row) 업데이트

가. 업데이트 구문

```
update employee set firstname=?,lastname=? where id=?
```

나. 적용

14. 레코드(row) 삭제

가. 삭제 구문

```
delete from employee where id=?
```

나. 적용

15. 조회(select) 소개

가. 개요

JdbcTemplate

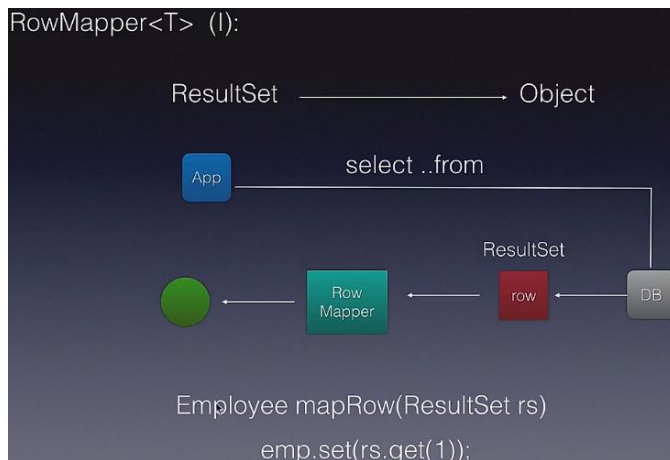
```
queryForObject(String sql, RowMapper<T> rowMapper, Object... args);  
query(String sql, RowMapper<T> rowMapper);
```

질의문 ??? 동적조건
args: <T> 리턴 타입: 단일 객체
리턴 타입: 컬렉션 객체

나. RowMapper<T>

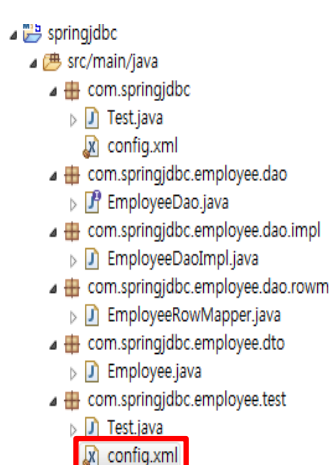
- 스프링 프레임워크에서 제공하는 구현이 필요한 인터페이스
- RowMapper는 JDBC의 ResultSet과 매핑

다. 정리

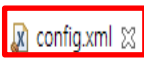


16. 읽기 메소드와 로매퍼(RowMapper) 만들기

- 17. 매퍼 구현
- 18. 테스트
- 19. 여러 레코드 읽기
- 20. JDBCTemplate 자동 와이어링(auto-wiring)



springjdbc
├── src/main/java
│ ├── com.springjdbc
│ │ ├── Test.java
│ │ └── config.xml
│ ├── com.springjdbc.employee.dao
│ │ ├── EmployeeDao.java
│ │ └── EmployeeDaoImpl.java
│ ├── com.springjdbc.employee.dao.impl
│ │ ├── EmployeeDaoImpl.java
│ │ ├── EmployeeRowMapper.java
│ │ ├── Employee.java
│ │ └── EmployeeTest.java
│ └── Test.java
└── config.xml

 config.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:p="http://www.springframework.org/schema/p"
6       xmlns:c="http://www.springframework.org/schema/c"
7       xsi:schemaLocation="http://www.springframework.org/schema/beans
8       http://www.springframework.org/schema/beans/spring-beans.xsd
9       http://www.springframework.org/schema/context
10      http://www.springframework.org/schema/context/spring-context.xsd">
11
12     <bean
13         class="org.springframework.jdbc.datasource.DriverManagerDataSource"
14         name="dataSource"
15         p:driverClassName="com.mysql.jdbc.Driver"
16         p:url="jdbc:mysql://localhost/mykhtdb"
17         p:username="root"
18         p:password="root" />
19
20     <bean class="org.springframework.jdbc.core.JdbcTemplate"
21         name="jdbcTemplate" p:dataSource-ref="dataSource" />
22
23     <!-- <bean class="com.springjdbc.employee.dao.impl.EmployeeDaoImpl"
24         name="employeeDao">
25         <property name="jdbcTemplate">
26             <ref bean="jdbcTemplate" />
27         </property>
28     </bean> -->
29
30 </beans>
```

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <beans xmlns="http://www.springframework.org/schema/beans"
3       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4       xmlns:context="http://www.springframework.org/schema/context"
5       xmlns:p="http://www.springframework.org/schema/p"
6       xmlns:c="http://www.springframework.org/schema/c"
7       xsi:schemaLocation="http://www.springframework.org/schema/beans
8       http://www.springframework.org/schema/beans/spring-beans.xsd
9       http://www.springframework.org/schema/context
10      http://www.springframework.org/schema/context/spring-context.xsd">
11
12     <context:component-scan
13         base-package="com.springjdbc.employee.dao.impl" />
14
15     <bean
16         class="org.springframework.jdbc.datasource.DriverManagerDataSource"
17         name="dataSource" p:driverClassName="com.mysql.jdbc.Driver"
18         p:url="jdbc:mysql://localhost/mykhtdb" p:username="root"
19         p:password="root" />
20
21     <bean class="org.springframework.jdbc.core.JdbcTemplate"
22         name="jdbcTemplate" p:dataSource-ref="dataSource" />
23
24     <!-- <bean class="com.springjdbc.employee.dao.impl.EmployeeDaoImpl" name="
25         <property name="jdbcTemplate"> <ref bean="jdbcTemplate" /> </propert
26
27 </beans>

```

21. 요약

Spring JDBC

No Repetitive Code	No Checked Exceptions
JdbcTemplate	update
DataSource	queryForObject
driver Class	query
url	
name	
password	

22. 과제

Spring JDBC

Passenger	create	JdbcTemplate	DataSource
id	read		
firstName	update		
lastName	delete	Test	