Spring Framework

✓ 원리를 알면 IT가 맛있다

Spring Framework for Beginners

chapter 04.

Database 연동

❑ 학습목표

- 스프링의 데이터베이스 연동지원
- 스프링의 JDBC지원- Template 클래스 이용
- 스프링의 JDBC지원- DaoSupport 클래스 이용
- Connection을 위한 DataSource 설정
- JdbcTemplate 클래스 사용 방법

❑ 1] 스프링의 데이터베이스 연동 지원

- 스프링은 JDBC를 비롯한 ORM 프레임워크를 직접적으로 지원하고 있기 때문에 간단하게 JDBC뿐만 아니라 ORM 프레임워크를 손쉽게 스프링과 연동할 수 있다.

- 스프링은 JDBC, ORM 프레임워크등의 다양한 기술을 이용해서 손쉽게 DAO클래스를 구현할 수 있도록 지원한다.

❑ 1) 스프링의 데이터베이스 연동 지원- jdbc 사용

- Jdbc기술의 사용 : pom.xml 에 dependcy추가
- 오라클 드라이버, spring-jdbc프레임워크, dbcp2 커넥션 풀추가

```
<!--
https://mvnrepository.com/artifact/com.jslso
lucoes/ojdbc6 -->
<dependency>
    <groupId>com.jslsolucoes</groupId>
    <artifactId>ojdbc6</artifactId>
    <version>11.2.0.1.0</version>
</dependency>
```


```
String driver="oracle.jdbc.driver.OracleDriver";
String url ="jdbc:oracle:thin:@localhost:1521:orcl";
String userid = "scott";
String passwd = "tiger";
public TestDAO(){
    //드라이버로딩
    try {
        Class.forName(driver);
    } catch (ClassNotFoundException e) {
        // TODO Auto-generated catch block
        e.printStackTrace();
    }
}

public ArrayList<TestDTO> select(){
    ArrayList<TestDTO> list = new ArrayList<TestDTO>();
      Connection con = null;
      PreparedStatement pstmt = null;
      ResultSet rs = null;
      try{
      con =
DriverManager.getConnection(url, userid, passwd);
String sql ="select * from test" ;
    pstmt = con.prepareStatement(sql);
    rs = pstmt.executeQuery();  // select 는 executeQuery()
    while(rs.next()){
        int n = rs.getInt("num");
        String n2 = rs.getString("username");
```

❑ 2) Template 클래스와 DaoSupport 클래스 지원

- Template 클래스 지원
  - 개발자가 중복된 코드를 입력해야 하는 성가신 작업을 감소시킨다.

  예> JDBC : JdbcTemplate
  iBatis : SqlMapClientTemplate
  MyBatis : SqlSessionTemplate
  Hibernate : HibernateTemplate

- DAOSupport 클래스 지원
  - DAO에서 기본적으로 필요로 하는 기능을 제공한다.
  - DaoSupport 클래스를 상속받아 DAO 클래스를 구현한 뒤, DaoSupport 클래스가 제공하는 기능을 사용하여 보다 편리하게 코드를 작성할 수 있게 된다.

  예> JDBC : JdbcDaoSupport
  iBatis : SqlMapClientDaoSupport
  MyBatis : SqlSessionDaoSupport
  Hibernate : HibernateDaoSupport

❑ 3] 예외 클래스 지원

- 의미있는 예외 클래스를 제공한다.
- 스프링은 데이터베이스 처리 과정에서 발생한 예외가 왜 발생했는지를 좀 더 구체적으로 확인할 수 있도록 하기 위해, 데이터베이스 처리와 관련된 예외클래스를 제공한다.
- 데이터베이스 처리과정에서 SQLException이 발생하면 스프링이 제공하는 예외클래스 중 알맞은 예외클래스로 변환하여 예외를 발생시킨다.
- 스프링의 모든 예외 클래스들은 DataAccessException을 상속 받는다.
  (RuntimeException 계열이다.)
- 주요 예외 클래스는 다음과 같다.
  - DuplicateKeyException
  - DataRetrievalFailureException
  - PermissionDeniedDataAccessException
  - BadSqlGrammerException
  - TypeMismatchDataAccessException

❑ 4) DataSource 설정

- 스프링은 DataSource를 통해서 Connection을 제공한다.
  따라서 DataSource 정보를 설정해야 된다.
- 스프링은 다음과 같은 3가지 설정 방식을 제공한다.
  가. 커넥션 풀을 이용한 DataSource 설정
  나. JNDI를 이용한 DataSource 설정
  다. DriverManager를 이용한 DataSource설정 (실습용)

```
<bean class="org.springframework.beans.factory.config.PropertyPlaceholderConfigurer">
    <property name="location">
      <value>classpath:db.properties</value>
    </property>
  </bean>
<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
   destroy-method="close">
   <property name="driverClassName" value="${driver}" />
   <property name="url" value="${url}"/>
   <property name="username" value="${username}" />
   <property name="password" value="${password}" />
</bean>
```

Dbcp2 DataSource-커넥션풀 추가

```
<dependency>
<groupId>org.springframework</groupId>
<artifactId>spring-jdbc</artifactId>
<version>3.1.4.RELEASE</version>
</dependency>

<dependency>
<groupId>org.apache.commons</groupId>
<artifactId>commons-dbcp2</artifactId>
<version>2.1</version>
</dependency>
```

TestMain.java TestDAO.java 05_Day1_dataSource_CHUL/pom.xml

```
1 jdbc.driver=oracle.jdbc.driver.OracleDriver
2 jdbc.url=jdbc:oracle:thin:@localhost:1521:orcl
3 jdbc.userid=scott
4 jdbc.passwd=tiger
```

Select XSD namespaces to use in the configuration file

- ☑ aop - http://www.springframework.org/schema/aop
- ☑ beans - http://www.springframework.org/schema/beans
- ☐ c - http://www.springframework.org/schema/c
- ☐ cache - http://www.springframework.org/schema/cache
- ☑ context - http://www.springframework.org/schema/context
- ☐ jdbc - http://www.springframework.org/schema/jdbc

```
public class TestDAO {
@Autowired
DataSource dataSource;

    public ArrayList<TestDTO> select(){
        ArrayList<TestDTO> list = new ArrayList<TestDTO>();
          Connection con = null;
          PreparedStatement pstmt = null;
          ResultSet rs = null;
          try{
          con = dataSource.getConnection();
    String sql ="select * from test" ;
        pstmt = con.prepareStatement(sql);
```

Dbcp2 DataSource- 컨넥션 풀을 이용한 connection얻기

```
        http://www.springframework.org/schema/aop http://www.springframework.org/schema/aop/spring-
<context:annotation-config></context:annotation-config>
<!-- 1.jdbc.properties등록 -->
<context:property-placeholder location="classpath:com/config/jdbc.properties"/>
<!-- 2. dbcp2 생성 :jdbc.properites 이용 :커넥션풀-->
<bean id="myDataSource" class="org.apache.commons.dbcp2.BasicDataSource" destroy-method="close()">
    <property name="driverClassName" value="${jdbc.driver}"></property>
    <property name="url" value="${jdbc.url}"></property>
    <property name="username" value="${jdbc.userid}"></property>
    <property name="password" value="${jdbc.passwd}"></property>
</bean>

<bean id="testDAO" class="com.dao.TestDAO"></bean>
<bean id="testDTO" class="com.dto.TestDTO"/>
</beans>
```

```
public class TestDAO {
@Autowired
DataSource dataSource;

    public ArrayList<TestDTO> select(){
        ArrayList<TestDTO> list = new ArrayList<TestDTO>();
          Connection con = null;
          PreparedStatement pstmt = null;
          ResultSet rs = null;
          try{
          con = dataSource.getConnection();
    String sql ="select * from test" ;
        pstmt = con.prepareStatement(sql);
```

❑ 5) JdbcTemplate 클래스 이용

- JdbcTemplate 클래스는 DataSource를 필요로 한다.
- JdbcTemplate 클래스를 빈으로 설정하고 미리 설정한 DataSource 빈을 dataSource 프로퍼티로 전달한다.

```
<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
   destroy-method="close">
   <property name="driverClassName" value="${driver}" />
   <property name="url" value="${url}"/>
   <property name="username" value="${username}" />
   <property name="password" value="${password}" />
</bean>
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
 <property name="dataSource" ref="dataSource" />
</bean>
```

❑ 6) JdbcTemplate 을 사용하는 DAO 클래스 작성

- DAO작성시 JdbcTemplate클래스를 주입 받을 수 있도록 프로퍼티와 setter메소드 또는 생성자를 작성한다.
- JdbcTemplate클래스가 제공하는 메소드를 이용해서 DAO를 작성한다.
- Select 쿼리 결과 집합을 매핑할 커스텀 RowMapper를 작성한다.
- DAO클래스를 빈으로 설정하고 미리 설정한 JdbcTemplate빈을 전달 받도록 설정한다.

```
<bean id="dataSource" class="org.apache.commons.dbcp.BasicDataSource"
   destroy-method="close">
   <property name="driverClassName" value="${driver}" />
   <property name="url" value="${url}"/>
   <property name="username" value="${username}" />
   <property name="password" value="${password}" />
</bean>
<bean id="jdbcTemplate" class="org.springframework.jdbc.core.JdbcTemplate">
 <property name="dataSource" ref="dataSource" />
</bean>
<bean id="deptDAO" class="com.dao.DeptDAO">
  <property name="jdbcTemplate" ref="jdbcTemplate" />
</bean>
```

❑ 5) Template 클래스

- 데이터베이스 연동을 위한 Connection에 관련된 코드 및 예외처리등 반복적인 코드를 제거 할 수 있도록 지원된다.

- 스프링은 다음과 같은 의 Template 클래스를 제공한다.

가. JdbcTemplate

- Object queryForObject(String sql, Object[] args, RowMapper rowMapper)
- Object queryForObject (String sql, RowMapper rowMapper)
- int queryForInt(String sql, Object[] args)
- int queryForInt(String sql)
- …

❑ 5) JdbcTemplate API – select : multi row

- List query(String sql, Object[] args, RowMapper rowMapper)
  - PreparedStatement를 이용해서 select 수행할 경우
- List query(String sql, RowMapper rowMapper)
  - 정적 SQL을 이용해서 select 수행할 경우

RowMapper 인터페이스

- ResultSet에서 값을 가져와 원하는 타입으로 매핑할 때 사용된다.

```
public interface RowMapper{
        Object mapRow(ResultSet rs, int rowNum) throws SQLException;
}
```

❑ 1) 스프링의 데이터베이스 연동 지원-JdbcTempate 이용

JdbcTemplate class를 이용한 JdbcTemplate.query(“sql”, RowMapper, 검색어)

```
33 private JdbcTemplate jdbcTemplate;
34         public TestDAO(DataSource dataSource) {
35             jdbcTemplate= new JdbcTemplate(dataSource);
36         }
37     public List<TestDTO> select(){
38         List<TestDTO> list =jdbcTemplate.query("select * from test",
39                 new RowMapper<TestDTO>() {
40                     public TestDTO mapRow(ResultSet rs, int roNum) throws SQLException {
41       public void insert(int num, String name, String address){
42           jdbcTemplate.update("insert into test (num, username, address) values (?,?,?)",
43           num, name, address);
44
45         }//end insert
46
47         });
48
49         return list;
50     }//end select
```

```
<!-- 1.jdbc.properties등록 -->
<context:property-placeholder location="classpath:com/config/jdbc.properties"/>
<!-- 2. dbcp2 생성 :jdbc.properites 이용 :커넥션풀-->
<bean id="myDataSource" class="org.apache.commons.dbcp2.BasicDataSource" destroy-method
    <property name="driverClassName" value="${jdbc.driver}"></property>
    <property name="url" value="${jdbc.url}"></property>
    <property name="username" value="${jdbc.userid}"></property>
    <property name="password" value="${jdbc.passwd}"></property>
</bean>

<bean id="testDAO" class="com.dao.TestDAO">
    <constructor-arg name="dataSource" ref="myDataSource"></constructor-arg>
</bean>
<bean id="testDTO" class="com.dto.TestDTO"/>
```

하나의 결과를 가져오는 조회 sql문의 실행
jdbcTemplate.queryForObject("sq문", 리턴클래스 타입);

```
public int selectCount(){

    Integer count= jdbcTemplate.queryForObject("select count(*) from test", Integer.class);
    return count;
}//end select
public List<TestDTO> select(){
```

❑ 1) 스프링의 데이터베이스 연동 지원-JdbcTempate 이용

JdbcTemplate class를 이용한 JdbcTemplate.query(“sql”, RowMapper, 검색어)
JdbcTemplate.update(“sql”, 변수, 변수…….)

```
.
public void insert(int num, String name, String address){
    jdbcTemplate.update("insert into test (num, username, address) values (?,?,?)",
    num, name, address);

}//end insert
```

- 스프링의 데이터베이스 연동지원
- 스프링의 JDBC지원- Template 클래스 이용
- 스프링의 JDBC지원- DaoSupport 클래스 이용
- Connection을 위한 DataSource 설정
- JdbcTemplate 클래스 사용 방법

Thank you