Spring SimpleJdbcTemplate Example

Spring 3 JDBC supports the java 5 feature var-args (variable argument) and autoboxing by the help of SimpleJdbcTemplate class.

SimpleJdbcTemplate class wraps the JdbcTemplate class and provides the update method where we can pass arbitrary number of arguments.

syntax of update method of SimpleJdbcTemplate class

```
1.  int update(String sql,Object... parameters)
```

Employee.java

This class contains 3 properties with constructors and setter and getters.

```
1.  package com.soften;
2.
3.  public class Employee {
4.  private int id;
5.  private String name;
6.  private float salary;
7.  //no-arg and parameterized constructors
8.  //getters and setters
9.  }
```

EmployeeDao.java

It contains one property SimpleJdbcTemplate and one method update. In such case, update method will update only name for the corresponding id. If you want to update the name and salary both, comment the above two lines of code of the update method and uncomment the 2 lines of code given below.

```
1.  package com.soften;
2.
3.  import org.springframework.jdbc.core.simple.SimpleJdbcTemplate;
4.  public class EmpDao {
5.  SimpleJdbcTemplate template;
6.
7.  public EmpDao(SimpleJdbcTemplate template) {
8.      this.template = template;
9.  }
10. public int update (Emp e){
```

```
11.     String query="update employee set name=? where id=?";
12.     return template.update(query,e.getName(),e.getId());
13.
14.     //String query="update employee set name=?,salary=? where id=?";
15.     //return template.update(query,e.getName(),e.getSalary(),e.getId());
16.     }
17.
18.     }
```

applicationContext.xml

The **DriverManagerDataSource** is used to contain the information about the database such as driver class name, connnection URL, username and password.

There are a property named **datasource** in the SimpleJdbcTemplate class of DriverManagerDataSource type. So, we need to provide the reference of DriverManagerDataSource object in the SimpleJdbcTemplate class for the datasource property.

Here, we are using the SimpleJdbcTemplate object in the EmployeeDao class, so we are passing it by the constructor but you can use setter method also.

```
1.      <?xml version="1.0" encoding="UTF-8"?>
2.      <beans
3.        xmlns="http://www.springframework.org/schema/beans"
4.        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5.        xmlns:p="http://www.springframework.org/schema/p"
6.        xsi:schemaLocation="http://www.springframework.org/schema/beans
7.      http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">
8.
9.      <bean id="ds" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
10.     <property name="driverClassName" value="oracle.jdbc.driver.OracleDriver" />
11.     <property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />
12.     <property name="username" value="system" />
13.     <property name="password" value="oracle" />
14.     </bean>
15.
16.     <bean id="jtemplate" class="org.springframework.jdbc.core.simple.SimpleJdbcTemplate">
17.     <constructor-arg ref="ds"></constructor-arg>
18.     </bean>
19.
20.     <bean id="edao" class="com.soften.EmpDao">
```

```
21.         <constructor-arg>
22.         <ref bean="jtemplate"/>
23.         </constructor-arg>
24.         </bean>
25.
26.         </beans>
```

SimpleTest.java

This class gets the bean from the applicationContext.xml file and calls the update method of EmpDao class.

```
1.          package com.soften;
2.
3.          import org.springframework.beans.factory.BeanFactory;
4.          import org.springframework.beans.factory.xml.XmlBeanFactory;
5.          import org.springframework.core.io.ClassPathResource;
6.          import org.springframework.core.io.Resource;
7.
8.          public class SimpleTest {
9.          public static void main(String[] args) {
10.
11.             Resource r=new ClassPathResource("applicationContext.xml");
12.             BeanFactory factory=new XmlBeanFactory(r);
13.
14.             EmpDao dao=(EmpDao)factory.getBean("edao");
15.             int status=dao.update(new Emp(23,"Tarun",35000));
16.             System.out.println(status);
17.         }
18.         }
```