SpringNamedParameterJdbcTemplate Example

Spring provides another way to insert data by named parameter. In such way, we use names instead of ?(question mark). So it is better to remember the data for the column.

Simple example of named parameter query

```
insert into employee values (:id,:name,:salary)
```

Method of NamedParameterJdbcTemplate class

In this example,we are going to call only the execute method of NamedParameterJdbcTemplate class. Syntax of the method is as follows:

```
pubic T execute(String sql,Map map,PreparedStatementCallback psc)
```

Example of NamedParameterJdbcTemplate class

Employee.java

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

```
package com.soften;

public class Employee {
private int id;
private String name;
private float salary;
//no-arg and parameterized constructors
//getters and setters
}
```

EmployeeDao.java

It contains on property jdbcTemplate and one method save.

```
package com.soften;

import java.sql.PreparedStatement;
import java.sql.SQLException;
import org.springframework.dao.DataAccessException;
```

```java
import org.springframework.jdbc.core.PreparedStatementCallback;
import org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate;
import java.util.*;

public class EmpDao {
NamedParameterJdbcTemplate template;

public EmpDao(NamedParameterJdbcTemplate template) {
      this.template = template;
}
public  void save (Emp e){
String query="insert into employee values (:id,:name,:salary)";

Map<String,Object> map=new HashMap<String,Object>();
map.put("id",e.getId());
map.put("name",e.getName());
map.put("salary",e.getSalary());

template.execute(query,map,new PreparedStatementCallback() {
   @Override
   public Object doInPreparedStatement(PreparedStatement ps)
           throws SQLException, DataAccessException {
      return ps.executeUpdate();
   }
});
}
}
```

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 NamedParameterJdbcTemplate class of DriverManagerDataSource type. So, we need to provide the reference of DriverManagerDataSource object in the NamedParameterJdbcTemplate class for the datasource property.

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

```xml
<?xml version="1.0" encoding="UTF-8"?>
```

```
<beans
    xmlns="http://www.springframework.org/schema/beans"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xmlns:p="http://www.springframework.org/schema/p"
    xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-3.0.xsd">

<bean id="ds" class="org.springframework.jdbc.datasource.DriverManagerDataSource">
<property name="driverClassName" value="oracle.jdbc.driver.OracleDriver" />
<property name="url" value="jdbc:oracle:thin:@localhost:1521:xe" />
<property name="username" value="system" />
<property name="password" value="oracle" />
</bean>

<bean id="jtemplate"
 class="org.springframework.jdbc.core.namedparam.NamedParameterJdbcTemplate">
<constructor-arg ref="ds"></constructor-arg>
</bean>

<bean id="edao" class="com.soften.EmpDao">
<constructor-arg>
<ref bean="jtemplate"/>
</constructor-arg>
</bean>

</beans>
```

SimpleTest.java

This class gets the bean from the applicationContext.xml file and calls the save method.

```
package com.soften;

import org.springframework.beans.factory.BeanFactory;
import org.springframework.beans.factory.xml.XmlBeanFactory;
import org.springframework.core.io.ClassPathResource;
import org.springframework.core.io.Resource;

public class SimpleTest {
    public static void main(String[] args) {

        Resource r=new ClassPathResource("applicationContext.xml");
        BeanFactory factory=new XmlBeanFactory(r);

        EmpDao dao=(EmpDao)factory.getBean("edao");
        dao.save(new Emp(23,"sonoo",50000));

    }
}
```