

Menu

[Struts 1](#)[Struts 2](#)[Spring](#)[Hibernate](#)[Ant](#)[Log4j](#)[Spring](#) > Spring SimpleJdbcTemplate**Spring SimpleJdbcTemplate**

To use the SimpleJdbcTemplate you need to use JDK 1.5 or higher. SimpleJdbcTemplate takes advantage of the Java 5 language features like varargs, autoboxing, generics and covariant returns.

```

01. package com.vaannila.dao;
02.
03. import java.sql.ResultSet;
04. import java.sql.SQLException;
05.
06. import javax.sql.DataSource;
07.
08. import org.springframework.jdbc.core.simple.ParameterizedRowMapper;
09. import org.springframework.jdbc.core.simple.SimpleJdbcTemplate;
10.
11. import com.vaannila.domain.Forum;
12.
13. public class ForumDAOImpl implements ForumDAO {
14.
15.     private SimpleJdbcTemplate simpleJdbcTemplate;
16.
17.     public void setDataSource(DataSource dataSource) {
18.         this.simpleJdbcTemplate = new SimpleJdbcTemplate(dataSource);
19.     }
20.
21.     @Override
22.     public void insertForum(Forum forum) {
23.         String query = "INSERT INTO FORUMS (FORUM_ID, FORUM_NAME,
24.             FORUM_DESC) VALUES (?, ?, ?)";
25.         simpleJdbcTemplate.update(query, forum.getForumId(), forum
26.             .getForumName(), forum.getForumDesc());
27.     }
28.
29.     @Override
30.     public Forum selectForum(int forumId) {
31.         String query = "SELECT * FROM FORUMS WHERE FORUM_ID=?";
32.         return simpleJdbcTemplate.queryForObject(query, new
33.             ParameterizedRowMapper<Forum>() {
34.                 @Override
35.                 public Forum mapRow(ResultSet resultSet, int rowNum)
36.                     throws SQLException {
37.                     return new Forum(resultSet.getInt("FORUM_ID"),
38.                         resultSet.getString("FORUM_NAME"),
39.                         resultSet.getString("FORUM_DESC"));
40.                 }
41.             }, forumId);
42.     }
43. }

```

The following snippet shows the insertForum() method using the JdbcTemplate.

```

01. public void insertForum(Forum forum) {
02.     String query = "INSERT INTO FORUMS (FORUM_ID, FORUM_NAME, FORUM_DESC)
03.         VALUES (?, ?, ?)";
04.     jdbcTemplate.update(query, new Object[] {
05.         Integer.valueOf(forum.getForumId()),
06.         forum.getForumName(), forum.getForumDesc() });
07. }

```

When using SimpleJdbcTemplate you can use the variable length arguments instead of an Object array. There is no need to explicitly typecast int to Integer.

```

01. @Override
02. public void insertForum(Forum forum) {
03.     String query = "INSERT INTO FORUMS (FORUM_ID, FORUM_NAME, FORUM_DESC)
04.         VALUES (?, ?, ?)";
05.     simpleJdbcTemplate.update(query, forum.getForumId(), forum
06.         .getForumName(), forum.getForumDesc());
07. }

```

The following snippet shows the selectForum() method using the JdbcTemplate.

```

01. public Forum selectForum(int forumId) {
02.     String query = "SELECT * FROM FORUMS WHERE FORUM_ID=?";
03.     return (Forum) jdbcTemplate.queryForObject(query, new Object[] {
04.         Integer.valueOf(forumId) },
05.         new RowMapper() {
06.             public Object mapRow(ResultSet resultSet, int rowNum) throws
07.                 SQLException {
08.                 return new Forum(resultSet.getInt("FORUM_ID"),
09.                     resultSet.getString("FORUM_NAME"),
10.                     resultSet.getString("FORUM_DESC"));
11.             }
12.         });
13. }

```

Here when using SimpleJdbcTemplate you need not explicitly typecast the return object, the ParameterizedRowMapper object's type parameter will be taken by default. The return type of the mapRow() method is Forum instead of Object because in Java 5 you can have covariant return

Subscribe

[RSS](#)[Email](#)

types. Since the statement parameter can be of variable length the `forumId` is specified at the end of the list.

```
01. @Override
02. public Forum selectForum(int forumId) {
03.     String query = "SELECT * FROM FORUMS WHERE FORUM_ID=?";
04.     return simpleJdbcTemplate.queryForObject(query, new
05.         ParameterizedRowMapper<Forum>() {
06.             @Override
07.             public Forum mapRow(ResultSet resultSet, int rowNum) throws
08.                 SQLException {
09.                 return new Forum(resultSet.getInt("FORUM_ID"),
10.                     resultSet.getString("FORUM_NAME"),
11.                     resultSet.getString("FORUM_DESC"));
12.             }
13.         }, forumId);
14. }
```

You can download and try the example here.

Source : [Download](#)

Source + Lib : [Download](#)