ABB Java Development Training

Spring MVC with JDBC framework

MySQL Database

MySQL is an open-source relational **database** management system (RDBMS)
Please refer to:
http://dev.mysql.com/doc

DAO (Data Access Object) Class

DAO class is Java class which provides abstraction of how accesses to certain table
It is one of the most design pattern in Java.

Data Access Object Pattern or DAO pattern is used to separate low level data accessing API or operations from high level business services. Following are the participants in Data Access Object Pattern.

- **Data Access Object Interface** - This interface defines the standard operations to be performed on a model object(s).
- **Data Access Object concrete class** - This class implements above interface. This class is responsible to get data from a data source which can be database / xml or any other storage mechanism.
- **Model Object or Value Object** - This object is simple POJO containing get/set methods to store data retrieved using DAO class.

Model Object Class

Is a Java class represents certain database table. Model class consists of:

- Properties which represent table column name
- Getter/setter methods

Choosing an approach for JDBC database access

You can choose among several approaches to form the basis for your JDBC database access. In addition to three flavors of the JdbcTemplate, a new SimpleJdbcInsert and SimplejdbcCall approach optimizes database metadata, and the RDBMS Object style takes a more object-oriented approach similar to that of JDO Query design. Once you start using one of these approaches, you can still mix and match to include a feature from a different approach. All

approaches require a JDBC 2.0-compliant driver, and some advanced features require a JDBC 3.0 driver.

- **JdbcTemplate** is the classic Spring JDBC approach and the most popular. This "lowest level" approach and all others use a JdbcTemplate under the covers.

- **NamedParameterJdbcTemplate** wraps a JdbcTemplate to provide named parameters instead of the traditional JDBC "?" placeholders. This approach provides better documentation and ease of usewhen you have multiple parameters for an SQL statement. Spring Framework Reference Documentation 4.2.6.RELEASE Spring Framework 395

- **SimpleJdbcInsert** and **SimpleJdbcCall** optimize database metadata to limit the amount of necessary configuration. This approach simplifies coding so that you only need to provide the name of the table or procedure and provide a map of parameters matching the column names. This only works if the database provides adequate metadata. If the database doesn't provide this metadata, you will have to provide explicit configuration of the parameters.

- **RDBMS Objects** including MappingSqlQuery, SqlUpdate and StoredProcedure requires you to create reusable and thread-safe objects during initialization of your data access layer. This approach is modeled after JDO Query wherein you define your query string, declare parameters, and compile the query. Once you do that, execute methods can be called multiple times with various parameter values passed in.

Spring ORM

The spring-orm module provides integration layers for popular object-relational mapping APIs, including JPA, JDO, and Hibernate. Using the spring-orm module you can use all of these O/Rmapping frameworks in combination with all of the other features Spring offers, such as the simple declarative transaction management feature mentioned previously.

javax.sql.DataSource;

```
@Bean
public DataSource getDataSource() {
    DriverManagerDataSource dataSource = new DriverManagerDataSource();
    dataSource.setDriverClassName("com.mysql.jdbc.Driver");
    dataSource.setUrl("jdbc:mysql://localhost:3306/abbtraining");
    dataSource.setUsername("root");
    dataSource.setPassword("admin");

    return dataSource;
}


@Bean
public ContactDAO getContactDAO() {
    return new ContactDAOImpl(getDataSource());
}
```

JSTL (JSP Standard Tag Library)

The JavaServer Pages Standard Tag Library (JSTL) is a collection of useful JSP tags which encapsulates core functionality common to many JSP applications.

JSTL has support for common, structural tasks such as iteration and conditionals, tags for manipulating XML documents, internationalization tags, and SQL tags. It also provides a framework for integrating existing custom tags with JSTL tags.

The JSTL tags can be classified, according to their functions, into following JSTL tag library groups that can be used when creating a JSP page:

- **Core Tags**
- **Formatting tags**
- **SQL tags**
- **XML tags**
- **JSTL Functions**

- Core Tags:
- The core group of tags are the most frequently used JSTL tags. Following is the syntax to include JSTL Core library in your JSP:

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```