Spring JDBC With Data Access Object Pattern

1. Overview

In this project, you will develop an Employee Management Application using Spring JDBC With DAO Pattern. This application allows you to perform basic CRUD operations such as inserting, retrieving, and displaying employee records from a MySQL database.

This project will help you:
- Learn how to use Spring JDBC for database interactions.
- Create database tables dynamically.
- Implement Spring's JdbcTemplate for easier data access and manipulation.

2. Project Structure

The project consists of the following components:
1. **Configuration Class (AppConfig.java)**: This class configures the Spring beans for database connection, JdbcTemplate, and EmployeeDAO. [**Already Defined**]
2. **Main Application Class (EmployeeApp.java)**: The entry point of the application that loads the Spring context, performs CRUD operations on employee data. [**This should be defined by you**]
3. **DAO (EmployeeDAO.java)**: A Data Access Object (DAO) class responsible for interacting with the database, performing CRUD operations. [**This should be defined by you**]
4. **Model (Employee.java)**: A model class representing an employee, with properties such as id, name, department, and salary. [**Already Defined**]
5. **Database Initialization**: The application automatically creates the database and the Employee table if they do not exist. [**Already Defined**]

3. Instructions

Step 1: Understand the Provided Classes

Before you begin, ensure you understand the following key components:
- **AppConfig Class**: This class defines the configuration for the data source, JdbcTemplate, and EmployeeDAO. It also handles the creation of the database and table at the application start using the @PostConstruct annotation.
- **EmployeeApp Class**: This is the main class that initializes the Spring context, retrieves the EmployeeDAO bean, and calls methods for CRUD operations (inserting, retrieving, and displaying employees).

- **EmployeeDAO Class**: The DAO class that interacts with the database using JdbcTemplate. It provides methods for inserting a new employee, retrieving all employees, and retrieving a specific employee by ID.
- **Employee Class**: The model class representing the employee entity. It contains the employee's attributes such as id, name, department, and salary.

Step 2: Implement the Main Application Class (EmployeeApp.java)

In the EmployeeApp.java class, you will load the Spring context, retrieve the necessary beans, and perform employee-related database operations.
Tasks:
1. **Load Spring Context**: Use AnnotationConfigApplicationContext to load the context using AppConfig.class.
2. **Retrieve EmployeeDAO Bean**: Retrieve the EmployeeDAO bean from the Spring context using context.getBean(EmployeeDAO.class).
3. **Perform CRUD Operations**: Create an employee using Employee employee = new Employee("John Doe", "IT", 50000) and insert it into the database using employeeDAO.insertEmployee(employee).
4. **Close the Context**: Ensure the Spring context is closed after all operations are performed using context.close().

Step 3: Implement the EmployeeDAO Class (EmployeeDAO.java)

This DAO class is responsible for interacting with the Employee table.
Tasks:
1. **Insert Employee**: Implement the insertEmployee(Employee employee) method to insert an employee into the database using the JdbcTemplate.update() method.
2. **Retrieve All Employees**: Implement the getAllEmployees() method to retrieve all employee records from the database.
3. **Retrieve Employee by ID**: Implement the getEmployeeById(int id) method to retrieve a specific employee by their ID.

Execution Steps to Follow

1. All actions like build, compile, running application, running test cases will be through Command Terminal.
2. To open the command terminal the test takers need to go to the Application menu (Three horizontal lines at left top) -> Terminal -> New Terminal.
3. cd into your backend project folder.
4. To build your project use command:

   mvn clean package -Dmaven.test.skip
5. To launch your application, move into the target folder (cd target). Run the following command to run the application:

   java -jar <your application jar file name>
6. To test any Restful application, the last option on the left panel of IDE, you can find ThunderClient, which is the lightweight equivalent of POSTMAN. Please use 127.0.0.1 instead of localhost to test rest endpoints.
7. Default credentials for MySQL:
   a. Username: root
   b. Password: pass@word1
8. To login to mysql instance: Open new terminal and use following command:
   a. sudo systemctl enable mysql
   b. sudo systemctl start mysql

   NOTE: After typing any of the above commands you might encounter any warnings.

   >> Please note that this warning is expected and can be disregarded. Proceed to the next step.
   c. mysql -u root -p

   The last command will ask for password which is 'pass@word1'
9. Mandatory: Before final submission run the following command:

   mvn test
10. You need to use CTRL+Shift+B - command compulsorily on code IDE, before final submission as well. This will push or save the updated contents in the internal git/repository, and will be used to evaluate the code quality.