배기원

스마트 콘텐츠 융합 응용SW 엔지니어 양성과정(888H)
[강남 M] 2020. 12. 02 ~ 2021. 07. 08 15:30~22:00
김동현 강사 | 진은지 취업담임

남은 시간 11:56:14

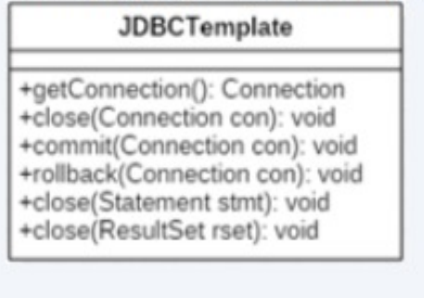
수강생 평가

* 첨부파일의 확장자를 소문자로 등록하세요. - 예시 : test.jpg(O), test.JPG(X)

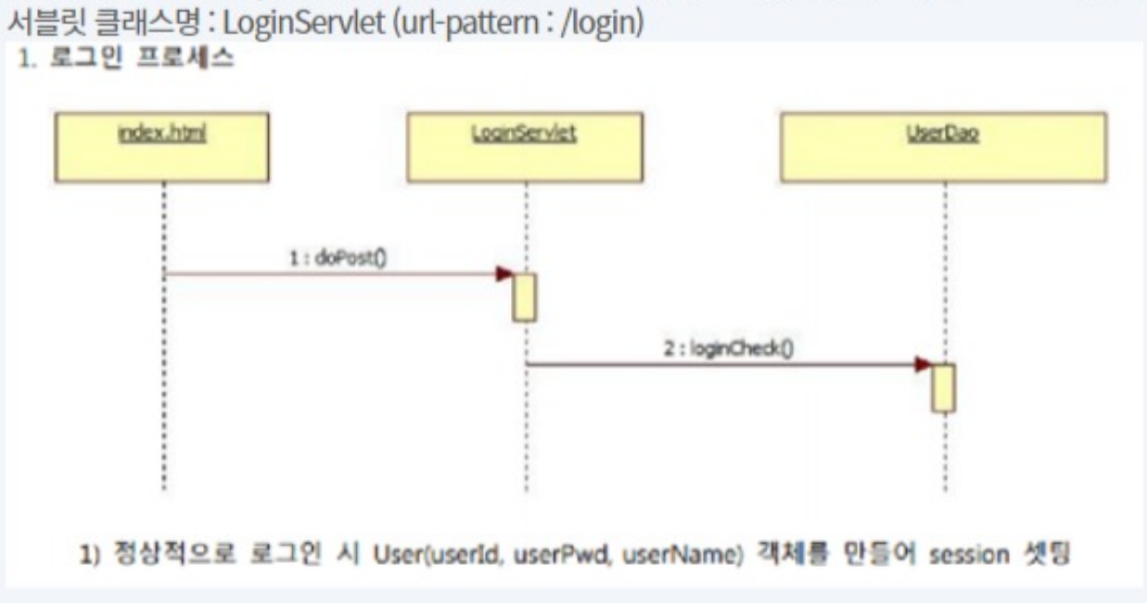
[NCS전공교과] 애플리케이션 설계 (평가자체크리스트) 1회차 ⌄

총점 : 80.0

1. 웹 서버 프로그램에서 MVC Model2 패턴 적용시, 비즈니스 로직을 처리할 모델(Model) 클래스들에 공통으로 사용될 데이터베이스 연동 및 트랜잭션 관리를 담당할 JDBC 템플릿 클래스를 아래의 클래스 다이어그램을 보고 구현하시오.
- 패키지명 : common
- 데이터베이스 url : "jdbc:oracle:thin:@198.121.57.34:1521:xe",
- 사용자계정, 암호 : "userxe", "passxe"



2. 다음의 시퀀스(Sequence) 다이어그램을 보고, 요구한대로 서블릿 클래스 안에 소스 코드를 작성하시오.
서블릿 클래스명 : LoginServlet (url-pattern : /login)



1. 공통모듈로 JDBCTemplate.java 를 구현하시오.
- 오라클과 연동 처리한다. (IP: 128.168.25.30, SID: XE)
- Connection 을 리턴하는 getConnection() 메소드를 구현한다.
- close() 메소드를 구현한다. 각각 Connection, Statement, ResultSet 에 대해 overloading 처리한다.
- commit, rollback 메소드도 구현한다.

요구사항

1-1. JDBCTemplate.java 클래스를 구현하시오.

수강생 답

```
package common;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Properties;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;


public class JDBCTemplate {
          private static String drivercClass="oracle.jdbc.OracleDriver";
          private static String url ="jdbc:oracle:thin:@198.121.57.34:1521:xe";
          private static String user ="uestxe";
          private static String password ="passxe";
          static {
             try {
                Class.forName(drivercClass);

             } catch (ClassNotFoundException e) {
                e.printStackTrace();
             }
          }

             public static Connection getConnection() {
                Connection conn = null;
                try {
                   conn = DriverManager.getConnection(url, user, password);
                   conn.setAutoCommit(false);
                } catch (SQLException e) {
                   e.printStackTrace();
                }

                return conn;
             }

             public static void close(Connection con) {
                try {
                   if(con != null)
                      con.close();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }

             public static void commit(Connection con) {
                try {
                   if(con != null)
                      con.commit();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }

             public static void rollback(Connection con) {
                try {
                   if(con != null)
                      con.rollback();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }
             public static void close(PreparedStatement stmt) {
                try {
                   if(stmt != null)
                      stmt.close();
                } catch (SQLException e) {
                   e.printStackTrace();
                }

             }
             public static void close(ResultSet rset) {
                try {
                   if(rset != null)
                      rset.close();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }
   }
```

답안첨삭

```
package common;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.Properties;

import javax.naming.Context;
import javax.naming.InitialContext;
import javax.naming.NamingException;
import javax.sql.DataSource;


public class JDBCTemplate {
          private static String drivercClass="oracle.jdbc.OracleDriver";
          private static String url ="jdbc:oracle:thin:@198.121.57.34:1521:xe";
          private static String user ="uestxe";
          private static String password ="passxe";
          static {
             try {
                Class.forName(drivercClass);

             } catch (ClassNotFoundException e) {
                e.printStackTrace();
             }
          }

             public static Connection getConnection() {
                Connection conn = null;
                try {
                   conn = DriverManager.getConnection(url, user, password);
                   conn.setAutoCommit(false);
                } catch (SQLException e) {
                   e.printStackTrace();
                }

                return conn;
             }

             public static void close(Connection con) {
                try {
                   if(con != null)
                      con.close();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }

             public static void commit(Connection con) {
                try {
                   if(con != null)
                      con.commit();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }

             public static void rollback(Connection con) {
                try {
                   if(con != null)
                      con.rollback();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }
             public static void close(PreparedStatement stmt) {
                try {
                   if(stmt != null)
                      stmt.close();
                } catch (SQLException e) {
                   e.printStackTrace();
                }

             }
             public static void close(ResultSet rset) {
                try {
                   if(rset != null)
                      rset.close();
                } catch (SQLException e) {
                   e.printStackTrace();
                }
             }
   }
```

소프트웨어 모듈간의 결합도는 줄이고 개별 모듈의 내부 응집도를 높인 JDBC에 대한 공통모듈을 정확하게 구현하였다.

모범답안

```
package common;
import java.sql.*;
public class JDBCTemplate {
public static Connection getConnection() {
Connection con = null;
try {
Class.forName("oracle.jdbc.driver.OracleDriver");
con = DriverManager.getConnection("jdbc:oracle:thin:@198.121.57.34:1521:xe", "userxe", "passxe");
} catch (Exception e) {
e.printStackTrace();
}
return con;
}
public static void close(Connection con) {
try {
if(con != null && !con.isClosed())
con.close();
} catch (Exception e) {
e.printStackTrace();
}
}
public static void close(Statement stmt) {
try {
if(stmt != null && !stmt.isClosed())
stmt.close();
} catch (Exception e) {
e.printStackTrace();
}
}
public static void close(ResultSet rset) {
try {
if(rset != null && !rset.isClosed())
rset.close();
} catch (Exception e) {
e.printStackTrace();
}
}
public static void commit(Connection con) {
try {
if(con != null && !con.isClosed())
con.commit();
} catch (Exception e) {
e.printStackTrace();
}
}
public static void rollback(Connection con) {
try {
if(con != null && !con.isClosed())
con.rollback();
} catch (Exception e) {
e.printStackTrace();
}
}
}
```

2. 시퀀스(Sequence) 다이어그램을 보고, 요구한대로 서블릿 클래스 안에 소스 코드를 작성하시오.

요구사항

2-1 서블릿 클래스명 : LoginServlet (url-pattern : /login)

수강생 답

```
package user.controller;

@WebServlet(urlPatterns = {"/login"})
public class LoginServlet extends HttpServlet {
   private static final long serialVersionUID = 1L;


      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
         String userId = request.getParameter("userId");
         String userPwd = request.getParameter("userPwd");
         User user = new UserDao.loginCheck(userId, userPwd);
         request.getSession().setAttribute("user", user);
         response.sendRedirect("index.html");
      }
   }
```

답안첨삭

```
package user.controller;

@WebServlet(urlPatterns = {"/login"})
public class LoginServlet extends HttpServlet {
   private static final long serialVersionUID = 1L;


      protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
         String userId = request.getParameter("userId");
         String userPwd = request.getParameter("userPwd");
         User user = new UserDao.loginCheck(userId, userPwd);
         request.getSession().setAttribute("user", user);
         response.sendRedirect("index.html");
      }
   }
```

소프트웨어 아키텍처에 따라 정의된 연동 상세 설계 가이드에 따라 타 시스템 연동 상세를 일부만 구현할 수 있다.

모범답안

```
@WebServlet("/login")
public class LoginServlet extends HttpServlet {
//중간 생략
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
// 1. 전송값에 한글이 있을 경우 인코딩처리함
request.setCharacterEncoding("utf-8");
//2. 전송값 꺼내서 User 객체에 저장하기
String userid = request.getParameter("userid");
String userpwd = request.getParameter("userpwd");
//3. 비즈니스 로직 처리하는 클래스의 해당 메소드를 실행하고, 처리 결과를 받음
User user = new UserDao().loginCheck(userid, userpwd);
//4. 받은 결과를 가지고 성공/실패에 대한 뷰페이지(파일)을 선택해서 내보냄
response.setContentType("text/html; charset=utf-8");
if(member != null) {
//로그인 성공시 해당 클라이언트에 대한 세션 객체를 생성함
HttpSession session = request.getSession();
session.setMaxInactiveInterval(10 * 60); //10분뒤에 자동 파기
//세션 객체에 리턴된 결과 User 객체 저장함
session.setAttribute("user", user);
response.sendRedirect("/first/index.jsp");
}else {
response.sendRedirect("views/member/memberError.html");
}
}
```

평가항목

평가내용	평가기준	배점	평가결과
중복개발 회피	재사용성 확보와 중복개발을 회피하기 위하여 전체 시스템 차원의 공통 부분을 식별하여 상세 명세를 작성할 수 있다.	20.0	예
공통 모듈 설계	개발할 응용소프트웨어의 전반적인 기능과 구조를 이해하기 쉬운 공통 모듈로 설계할 수 있다.	20.0	예
타 시스템 연동 설계	소프트웨어 아키텍처에서 정의한 타 시스템 연동 리스트 및 연동 방안을 작성할 수 있다.	20.0	예
타 시스템 연동 상세	소프트웨어 아키텍처에서 따라 선정된 개발 및 운영 환경에 사용될 솔루션에 대하여 상세 설계를 할 수 있다.	20.0	예
오류 예측 및 방안 제시	시스템간의 연동시 발생할 수 있는 오류를 예측하고 대응방안을 제시할 수 있다.	20.0	아니오

평가 메인으로