

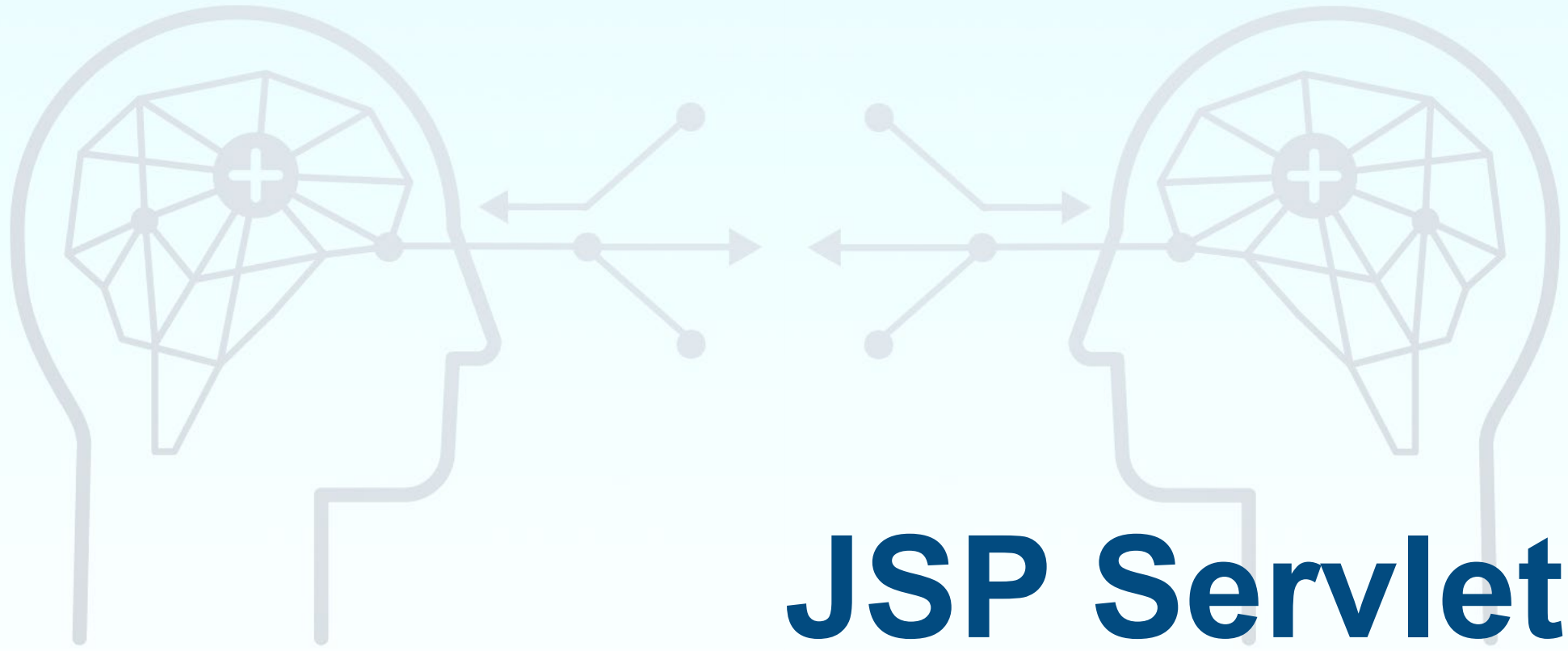


[한국폴리텍대학 성남캠퍼스 인공지능소프트웨어과]

JSP Servlet

2023. 5

김 형 오



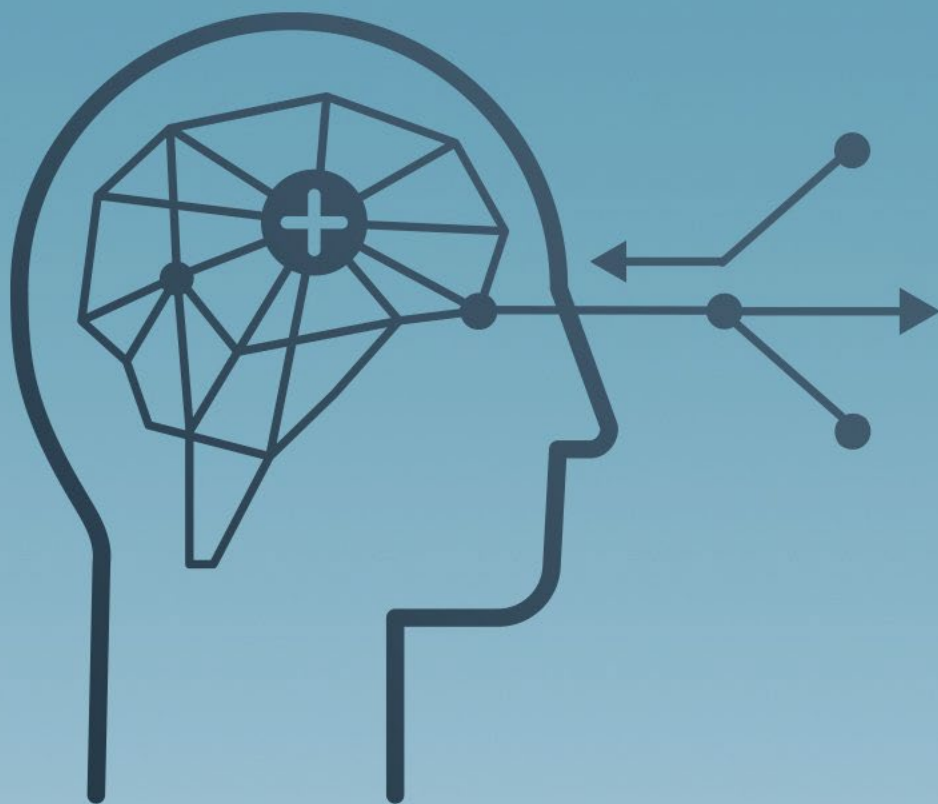
JSP Servlet

DEEP
LEARNING

MACHINE LEARNING BASED
ON ARTIFICIAL NEURAL NETWORKS

DEEP
LEARNING

MACHINE LEARNING BASED
ON ARTIFICIAL NEURAL NETWORKS



DEEP LEARNING

MACHINE LEARNING BASED
ON ARTIFICIAL NEURAL NETWORKS



목차

A table of Contents

#1, Database 설정

#2, 게시판 회원 가입



Part 1, DataBase 설정


참고문헌

<https://shinny.tistory.com/77>

H2 데이터베이스

H2 Database Engine

h2database.com/html/main.html



[Translate](#)

Search:

[Home](#)
[Download](#)
[Cheat Sheet](#)

Documentation
[Quickstart](#)
[Installation](#)
[Tutorial](#)
[Features](#)
[Security](#)
[Performance](#)
[Advanced](#)

Reference
[Commands](#)
[Functions](#)
[Aggregate](#) • [Window](#)


H2 Database Engine


Welcome to H2, the Java SQL database. The main features of H2 are:

- Very fast, open source, JDBC API
- Embedded and server modes; in-memory databases
- Browser based Console application
- Small footprint: around 2.5 MB jar file size

Download

Version 2.1.214 (2022-06-13)

 [Windows Installer \(6.7 MB\)](#)

 [All Platforms \(zip, 9.5 MB\)](#)

[All Downloads](#)

Support

[Stack Overflow \(tag H2\)](#)

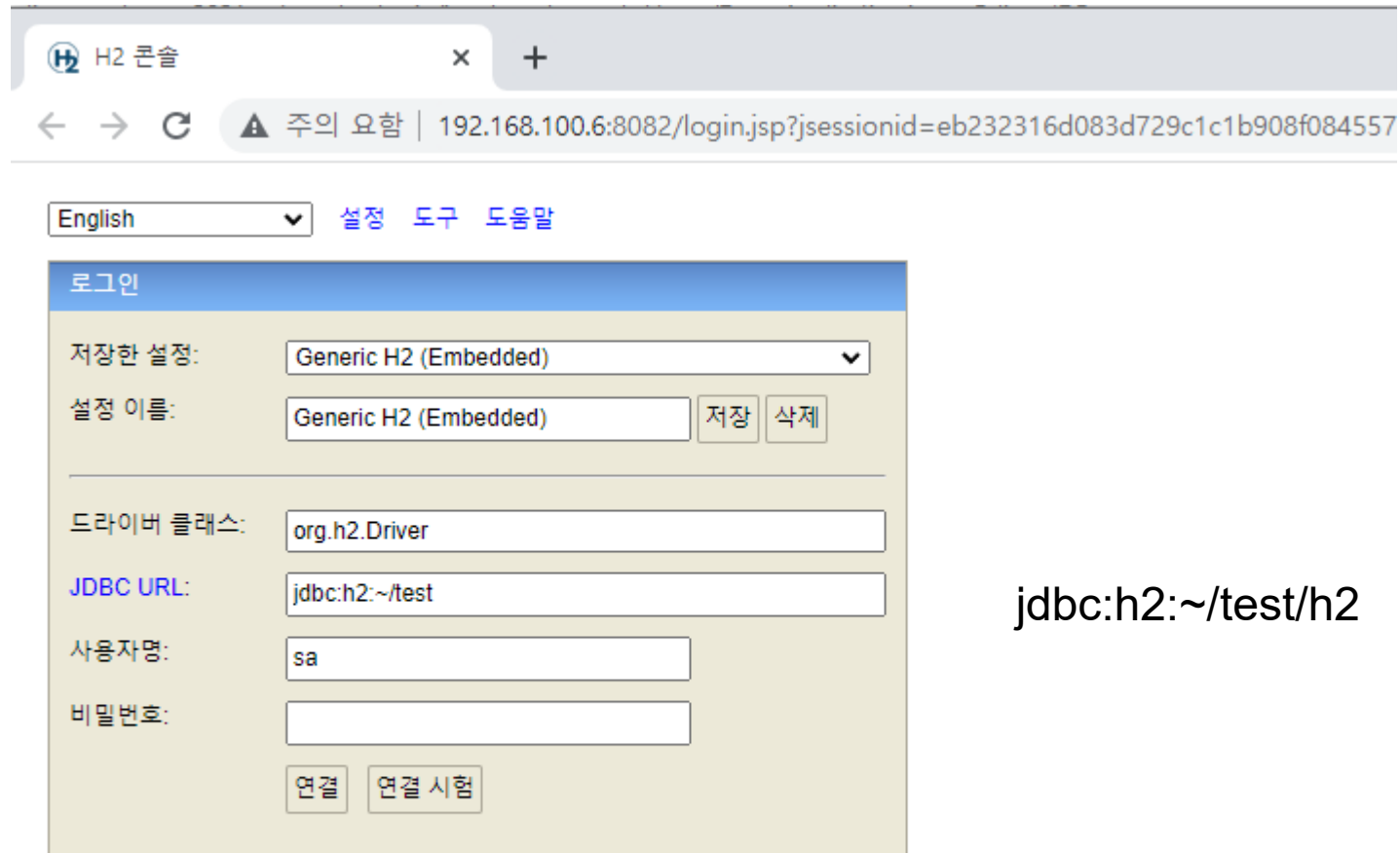
[Google Group](#)

For non-technical issues, use:
[dbsupport at h2database.com](#)

Features

- Very fast, open source, JDBC API
- Embedded and server modes; disk-based or in-memory databases
- Transaction support, multi-version concurrency
- Browser based Console application

H2 데이터베이스 설치 확인



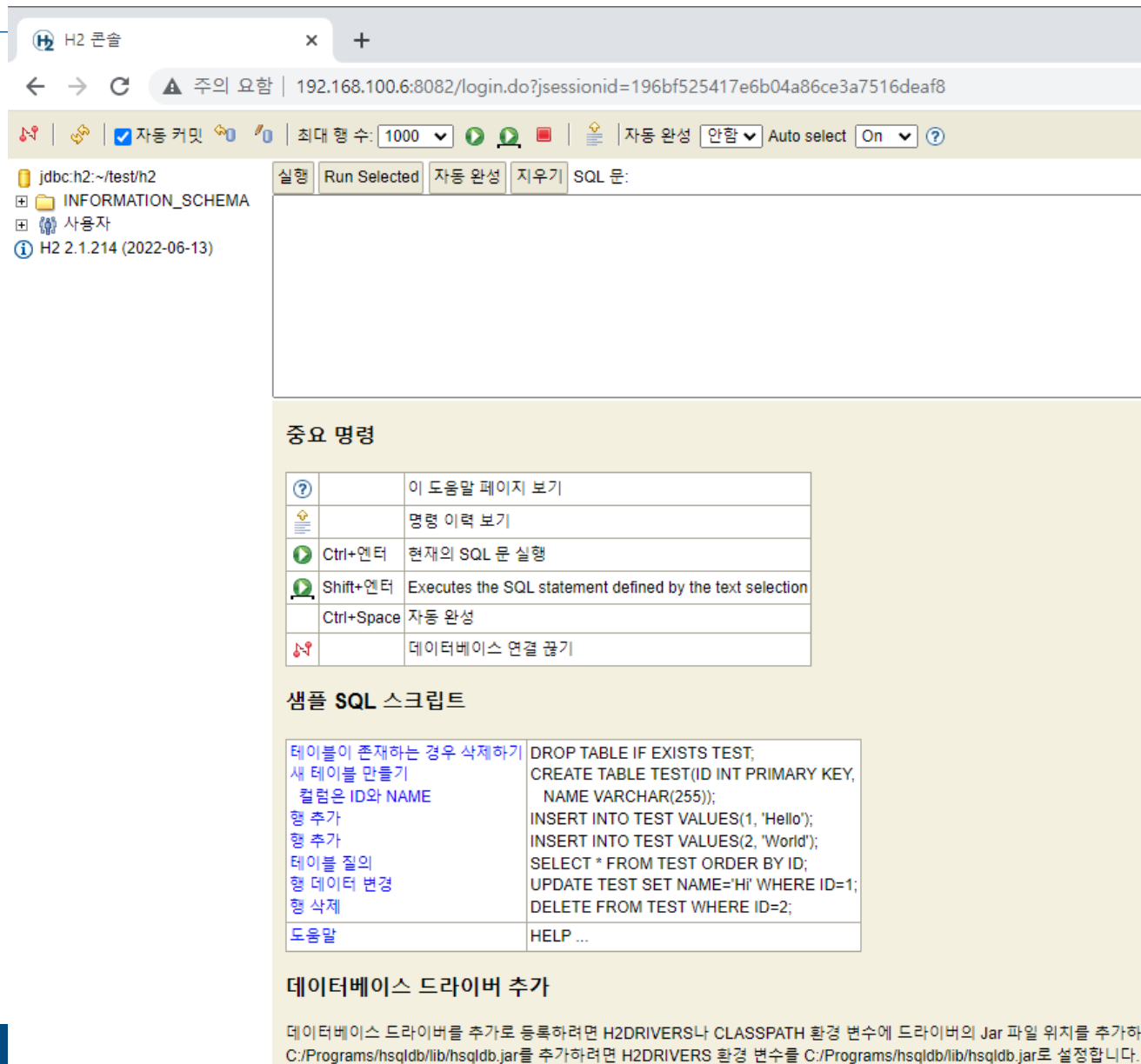
The screenshot shows the H2 console web interface in a browser. The address bar displays the URL: 192.168.100.6:8082/login.jsp?jsessionId=eb232316d083d729c1c1b908f084557. The page has a language dropdown set to 'English' and links for '설정' (Settings), '도구' (Tools), and '도움말' (Help). The '로그인' (Login) form includes the following fields:

- 저장한 설정: Generic H2 (Embedded) (dropdown menu)
- 설정 이름: Generic H2 (Embedded) (text input) with '저장' (Save) and '삭제' (Delete) buttons.
- 드라이버 클래스: org.h2.Driver (text input)
- JDBC URL: jdbc:h2:~/test (text input)
- 사용자명: sa (text input)
- 비밀번호: (empty text input)

At the bottom of the form are '연결' (Connect) and '연결 시험' (Test Connection) buttons.

jdbc:h2:~/test/h2

H2 데이터베이스



H2 콘솔

주의 요함 | 192.168.100.6:8082/login.do?sessionId=196bf525417e6b04a86ce3a7516deaf8

자동 커밋 | 최대 행 수: 1000 | 자동 완성 | Auto select On

jdbc:h2:~/test/h2

INFORMATION_SCHEMA

사용자

H2 2.1.214 (2022-06-13)

실행 | Run Selected | 자동 완성 | 지우기 | SQL 문:

중요 명령

?	이 도움말 페이지 보기
	명령 이력 보기
Ctrl+엔터	현재의 SQL 문 실행
Shift+엔터	Executes the SQL statement defined by the text selection
Ctrl+Space	자동 완성
	데이터베이스 연결 끊기

샘플 SQL 스크립트

테이블이 존재하는 경우 삭제하기	DROP TABLE IF EXISTS TEST;
새 테이블 만들기	CREATE TABLE TEST(ID INT PRIMARY KEY,
컬럼은 ID와 NAME	NAME VARCHAR(255));
행 추가	INSERT INTO TEST VALUES(1, 'Hello');
행 추가	INSERT INTO TEST VALUES(2, 'World');
테이블 질의	SELECT * FROM TEST ORDER BY ID;
행 데이터 변경	UPDATE TEST SET NAME='Hi' WHERE ID=1;
행 삭제	DELETE FROM TEST WHERE ID=2;
도움말	HELP ...

데이터베이스 드라이버 추가

데이터베이스 드라이버를 추가로 등록하려면 H2DRIVERS나 CLASSPATH 환경 변수에 드라이버의 Jar 파일 위치를 추가합니다.
C:/Programs/hsqldb/lib/hsqldb.jar를 추가하려면 H2DRIVERS 환경 변수를 C:/Programs/hsqldb/lib/hsqldb.jar로 설정합니다.

H2 데이터베이스

H2 콘솔

← → ↺ 주의 요함 | 192.168.100.6:8082/login.do?jsessionid=196bf525417e6b04a86ce3a7516deaf8

한국어 ▼ 설정 도구 도움말

로그인

저장한 설정: Generic H2 (Embedded) ▼

설정 이름: Generic H2 (Embedded) 저장 삭제

드라이버 클래스: org.h2.Driver

JDBC URL: jdbc:h2:tcp://localhost/~test/h2

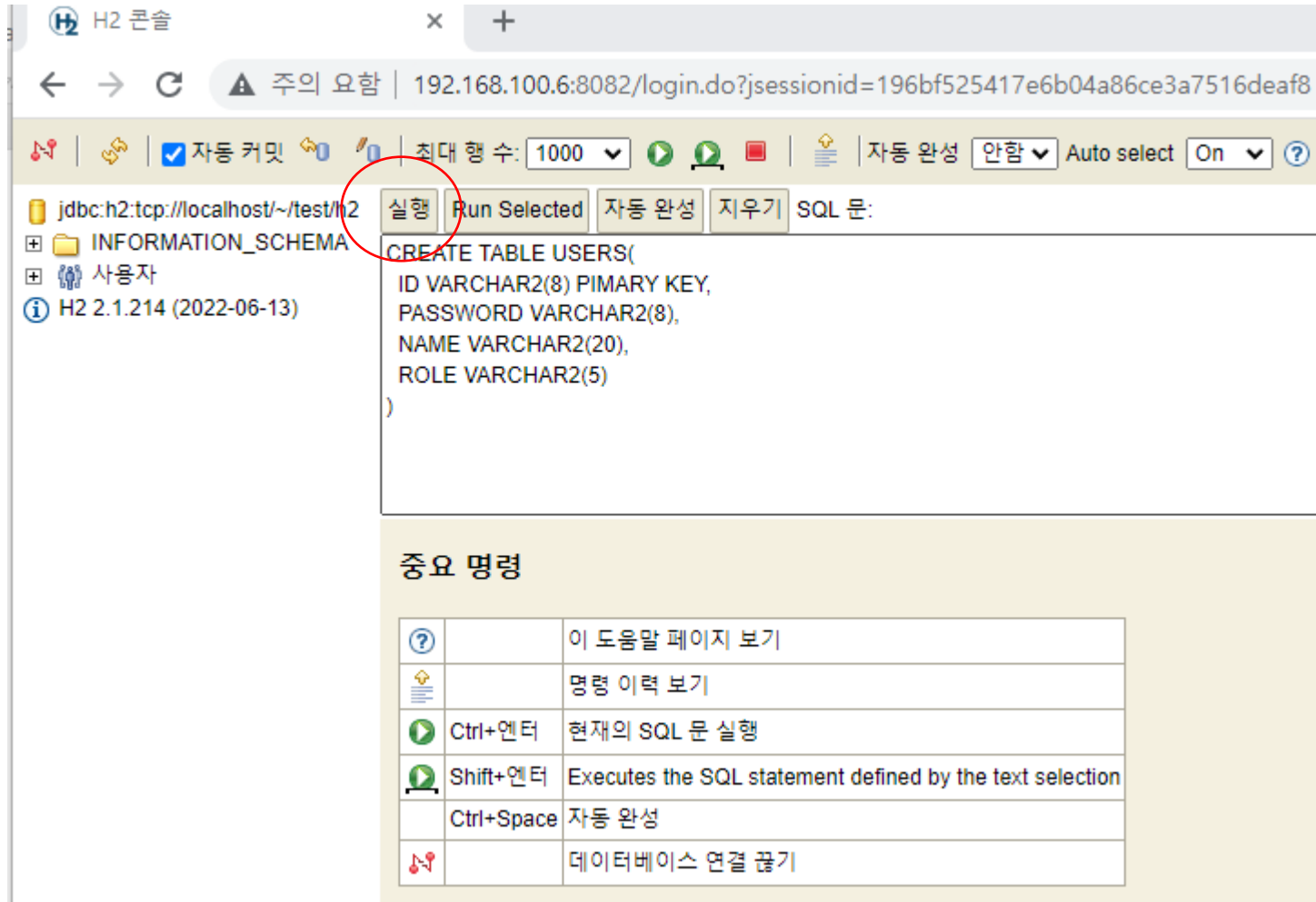
사용자명: sa

비밀번호:

연결 연결 시험

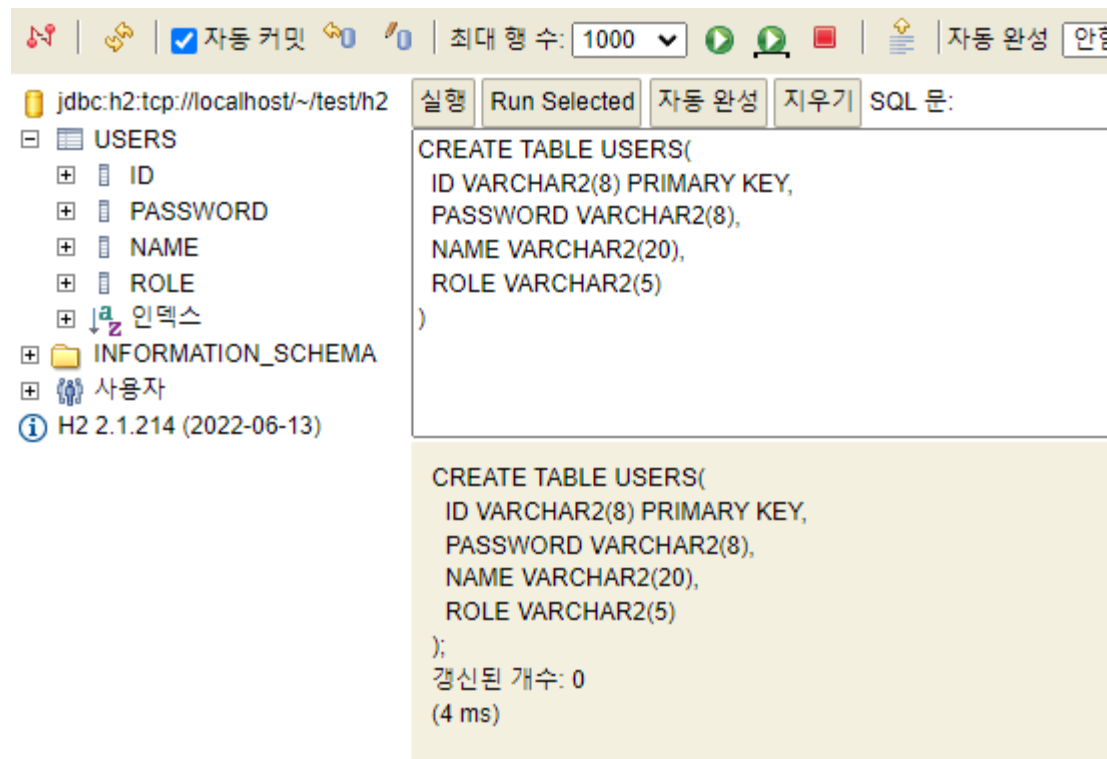
jdbc:h2:tcp://localhost/~test/h2

회원 테이블 만들기



```
CREATE TABLE USERS(  
  ID VARCHAR2(8) PRIMARY KEY,  
  PASSWORD VARCHAR2(8),  
  NAME VARCHAR2(20),  
  ROLE VARCHAR2(5)  
)
```

회원 테이블 만들기



회원 테이블 만들기

```
INSERT INTO USERS VALUES('admin','admin','admin','Admin')
```

```
INSERT INTO USERS VALUES('guest','guest','guest','User')
```

```
SELECT * FROM USERS
```

The screenshot shows a database management interface. On the left, a tree view displays the database structure, including a table named 'USERS' with columns 'ID', 'PASSWORD', 'NAME', and 'ROLE'. The main area on the right shows the execution of the SQL query 'SELECT * FROM USERS;'. Below the query, the results are displayed in a table format, showing two rows of data: 'admin' and 'guest'. The interface also includes a toolbar with various icons and buttons, and a status bar at the bottom indicating the number of rows and execution time.

jdbc:h2:tcp://localhost/~test/h2 | 실행 | Run Selected | 자동 완성 | 지우기 | SQL 문: | 자동 완성 | 안함 | Auto s

최대 행 수: 1000

USERS

- ID
- PASSWORD
- NAME
- ROLE
- 인덱스

INFORMATION_SCHEMA

사용자

H2 2.1.214 (2022-06-13)

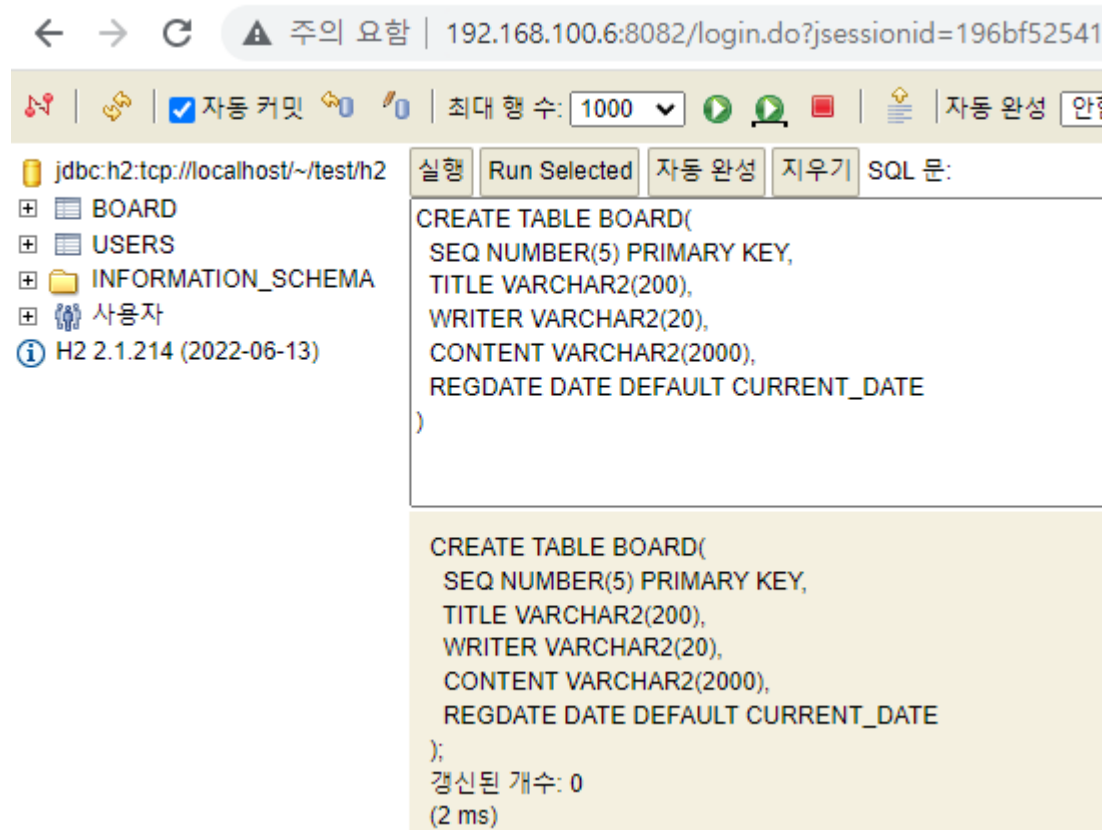
SELECT * FROM USERS;

ID	PASSWORD	NAME	ROLE
admin	admin	admin	Admin
guest	guest	guest	User

(2 행, 4 ms)

편집

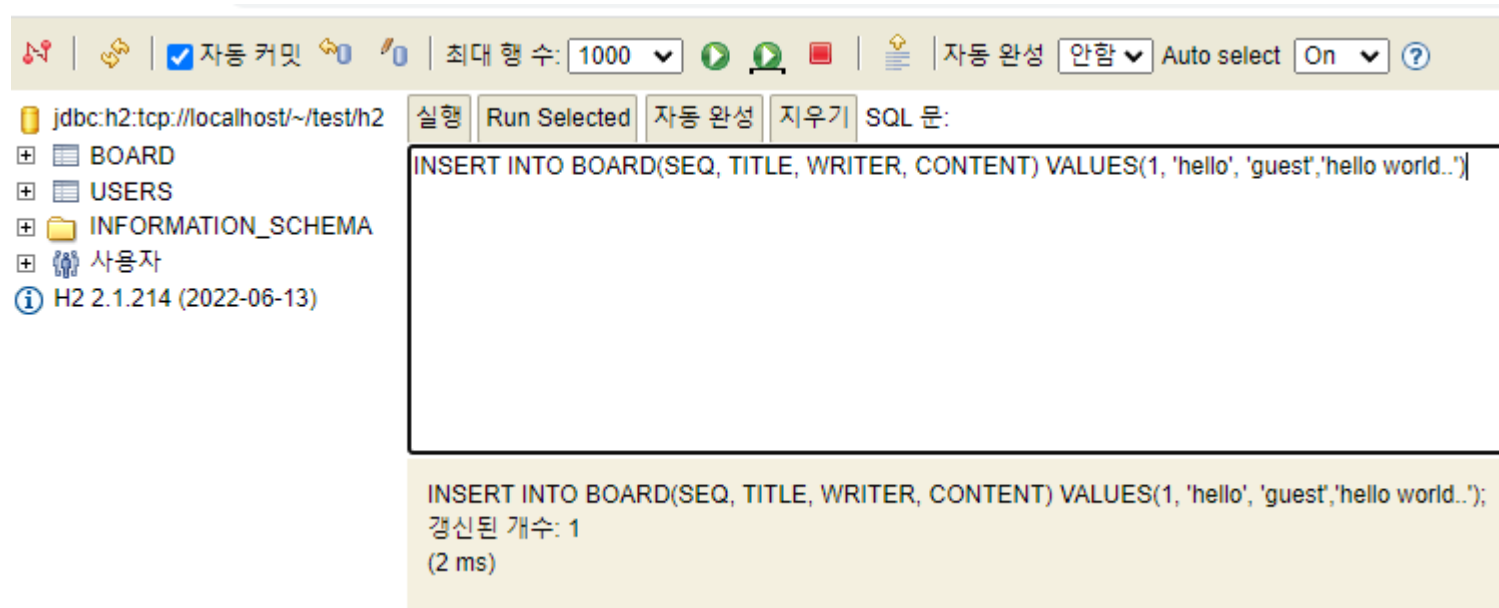
게시글 테이블 만들기



```
CREATE TABLE BOARD(  
  SEQ NUMBER(5) PRIMARY KEY,  
  TITLE VARCHAR2(200),  
  WRITER VARCHAR2(20),  
  CONTENT VARCHAR2(2000),  
  REGDATE DATE DEFAULT CURRENT_DATE  
)
```

게시글 테이블 만들기

```
INSERT INTO BOARD(SEQ, TITLE, WRITER, CONTENT) VALUES(1, 'hello', 'guest','hello world..')
```



게시글 테이블 만들기

jdbc:h2:tcp://localhost/~test/h2

- BOARD
- USERS
- INFORMATION_SCHEMA
- 사용자
- H2 2.1.214 (2022-06-13)

실행 Run Selected 자동 완성 지우기 SQL 문:

```
SELECT * FROM BOARD
```

```
SELECT * FROM BOARD;
```

SEQ	TITLE	WRITER	CONTENT	REGDATE
1	hello	guest	hello world..	2023-05-29

(1 row, 2 ms)

편집

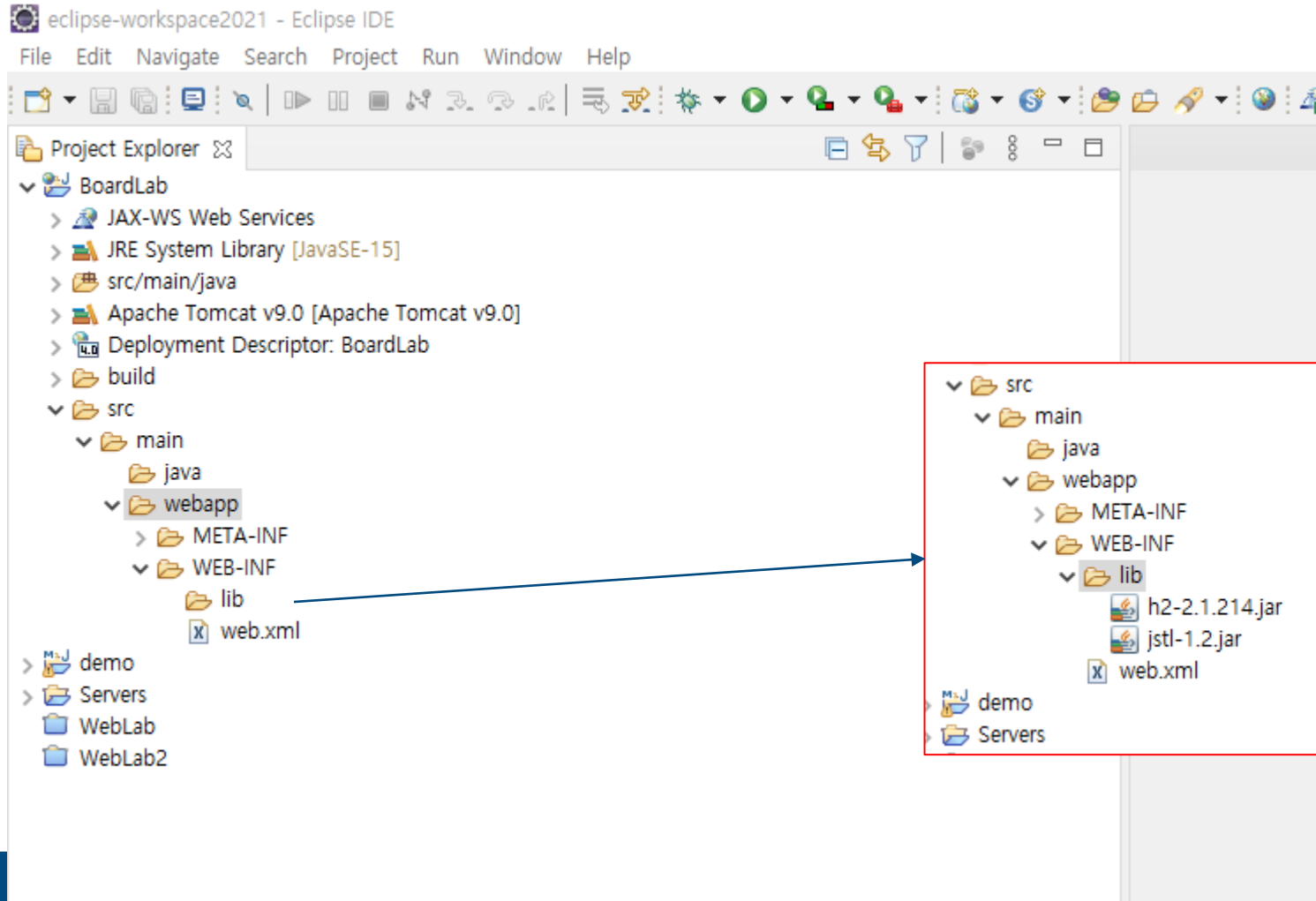


Part 2, 게시판 회원 가입

참고문헌
<https://shinny.tistory.com/77>

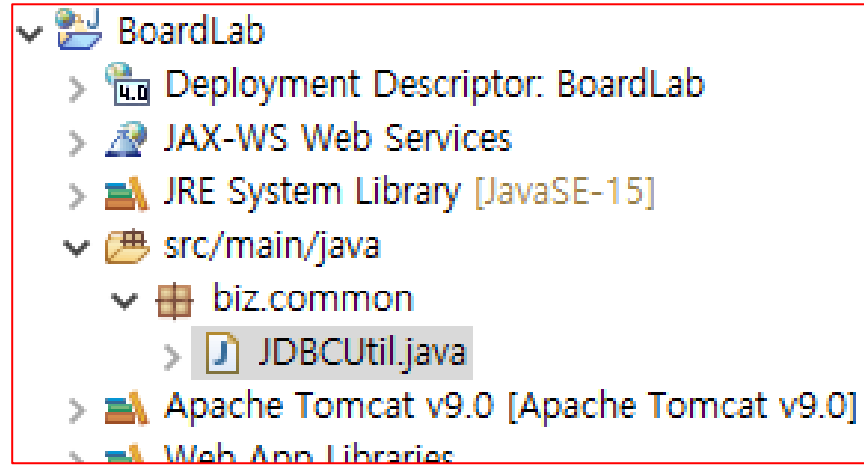
프로젝트 생성

- File -> New -> Dynamic Web Project
- JSTL 라이브러리, JDBC 라이브러리 추가



JDBC 연결

- “biz.common” 패키지 생성
- “JDBCUtil.java” 클래스 생성



JDBC 연결

JDBCUtil.java

```
package biz.common;
```

```
import java.sql.Connection;
```

```
import java.sql.DriverManager;
```

```
import java.sql.ResultSet;
```

```
import java.sql.Statement;
```

```
public class JDBCUtil {
```

```
    public static Connection getConnection() {
```

```
        try {
```

```
            Class.forName("org.h2.Driver");
```

```
            String url = "jdbc:h2:tcp://localhost/~test/h2";
```

```
            return DriverManager.getConnection(url, "sa", "");
```

```
        } catch (Exception e) {
```

```
        }
```

```
        return null;
```

```
    }
```

JDBC 연결

JDBCUtil.java

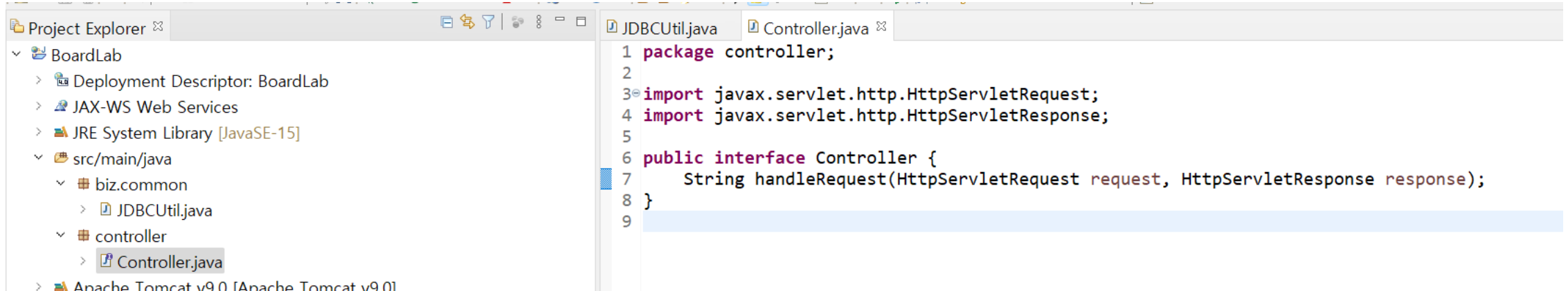
```
public static void close(Statement stmt, Connection conn) {  
    try {  
        if(stmt != null)  
            stmt.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        stmt = null;  
    }  
    try {  
        if(conn != null)  
            conn.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        conn = null;  
    }  
}
```

JDBC 연결

JDBCUtil.java

```
public static void close(ResultSet rs, Statement stmt, Connection conn) {  
    try {  
        if(rs != null)  
            rs.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        stmt = null;  
    }  
    try {  
        if(stmt != null)  
            stmt.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        stmt = null;  
    }  
    try {  
        if(conn != null)  
            conn.close();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        conn = null;  
    }  
}
```

Controller Interface 생성



Controller Interface 생성

Controller.java

```
package controller;
```

```
import javax.servlet.http.HttpServletRequest;
```

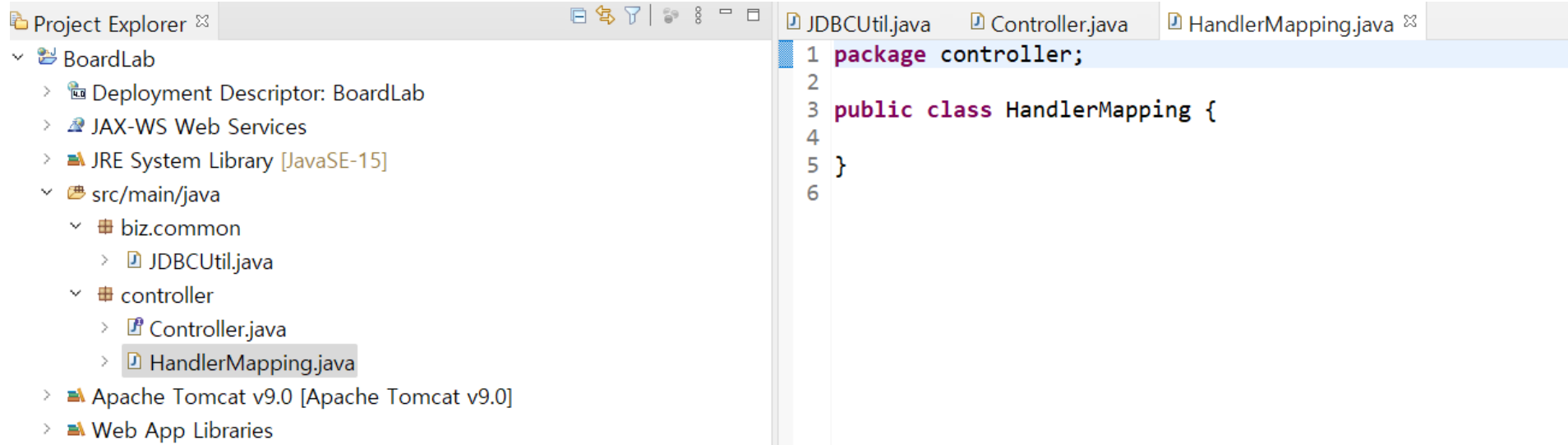
```
import javax.servlet.http.HttpServletResponse;
```

```
public interface Controller {
```

```
    String handleRequest(HttpServletRequest request, HttpServletResponse response);
```

```
}
```


HandlerMapping 생성



HandlerMapping 생성

HandlerMapping.java

```
package controller;
```

```
import java.util.HashMap;
```

```
import java.util.Map;
```

```
public class HandlerMapping {  
    private Map<String, Controller> mappings;  
    public HandlerMapping() {  
        mappings = new HashMap<String, Controller>();  
    }  
    public Controller getController(String path) {  
        return mappings.get(path);  
    }  
}
```

DispatcherServlet 생성

DispatcherServlet.java

```
public class DispatcherServlet extends HttpServlet {  
    private static final long serialVersionUID = 1L;  
  
    private HandlerMapping mapping;  
    public DispatcherServlet() {  
        super();  
        // TODO Auto-generated constructor stub  
    }  
  
    @Override  
    public void init() throws ServletException {  
        mapping = new HandlerMapping();  
    }  
}
```

DispatcherServlet 생성

DispatcherServlet.java

```
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws  
ServletException, IOException {  
    process(request, response);  
}
```

```
/**  
 * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response)  
 */  
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws  
ServletException, IOException {  
    request.setCharacterEncoding("EUC-KR");  
    process(request, response);  
}
```

DispatcherServlet 생성

DispatcherServlet.java

```
protected void process(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    String uri = request.getRequestURI();
    String path = uri.substring(uri.lastIndexOf("/"));

    Controller ctrl = mapping.getController(path);
    String viewPage = ctrl.handleRequest(request, response);

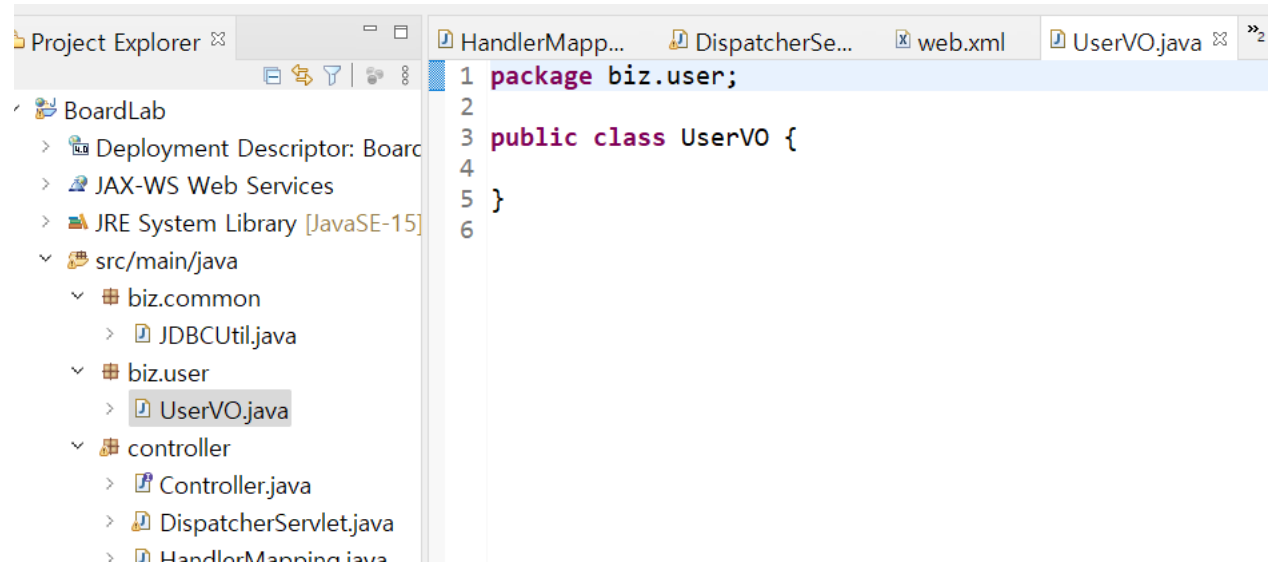
    RequestDispatcher rd = request.getRequestDispatcher(viewPage);
    rd.forward(request, response);
}
}
```

Web.xml 수정

web.xml

```
<welcome-file-list>
  <welcome-file>index.html</welcome-file>
  <welcome-file>index.htm</welcome-file>
  <welcome-file>index.jsp</welcome-file>
  <welcome-file>default.html</welcome-file>
  <welcome-file>default.htm</welcome-file>
  <welcome-file>default.jsp</welcome-file>
</welcome-file-list>
<servlet>
<servlet-name>dispatcher</servlet-name>
<servlet-class>controller.DispatcherServlet</servlet-class>
</servlet>
<servlet-mapping>
<servlet-name>dispatcher</servlet-name>
<url-pattern>*.do</url-pattern>
</servlet-mapping>
</web-app>
```

UserVO Class 생성



UserVO Class 생성

UserVO.java

```
package biz.user;

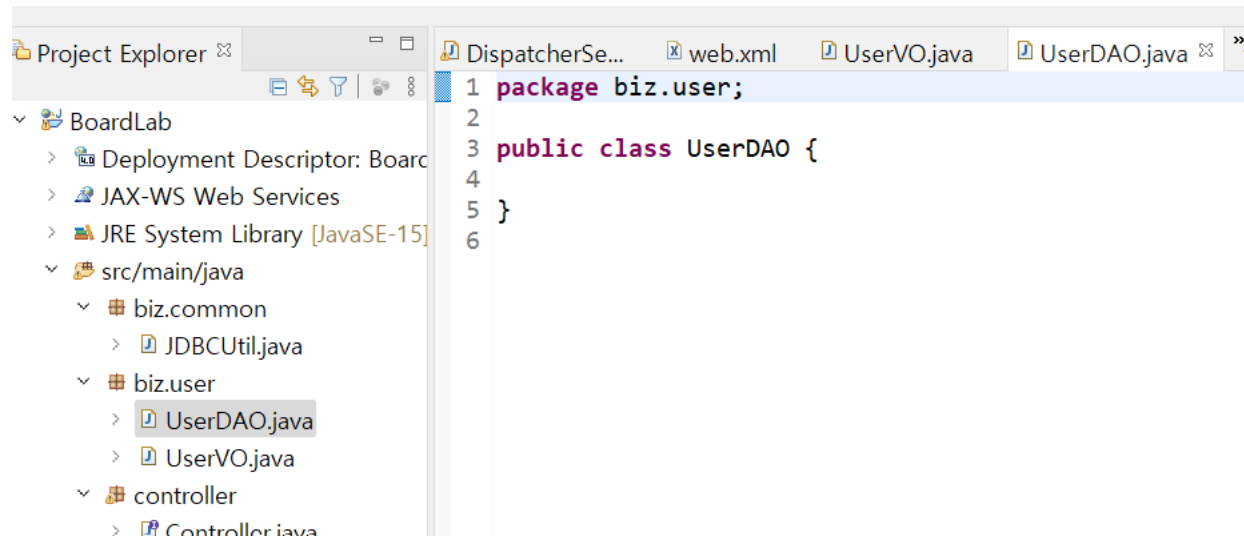
public class UserVO {
    private String id;
    private String password;
    private String name;
    private String role;
    public String getId() {
        return id;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getPassword() {
        return password;
    }
    public void setPassword(String password) {
        this.password = password;
    }
}
```

UserVO Class 생성

UserVO.java

```
public String getName() {  
    return name;  
}  
public void setName(String name) {  
    this.name = name;  
}  
public String getRole() {  
    return role;  
}  
public void setRole(String role) {  
    this.role = role;  
}  
@Override  
public String toString() {  
    return "UserVO [id=" + id + ", password=" + password + ", name=" + name + ",  
        role=" + role + "];"  
}  
}
```

UserDAO Class 생성



UserDAO Class 생성

UserDAO.java

```
package biz.user;
```

```
import java.sql.Connection;
```

```
import java.sql.PreparedStatement;
```

```
import java.sql.ResultSet;
```

```
import biz.common.JDBCUtil;
```

```
public class UserDAO {
```

```
    private Connection conn;
```

```
    private PreparedStatement stmt;
```

```
    private ResultSet rs;
```

```
    private static String USER_INSERT=
```

```
        "insert into users (id, password, name, role) "+
```

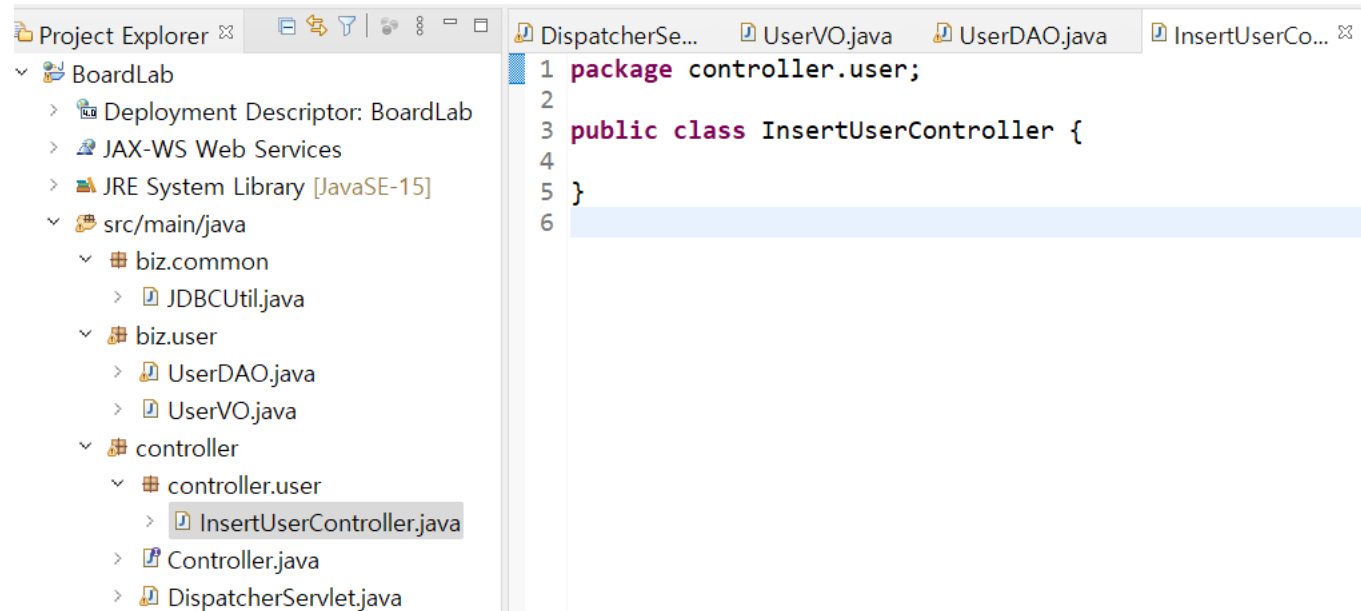
```
        "values (?, ?, ?, ?)";
```

UserDAO Class 생성

UserDAO.java

```
public void insertUser(UserVO vo) {  
    try {  
        conn = JDBCUtil.getConnection();  
        stmt = conn.prepareStatement(USER_INSERT);  
        stmt.setString(1, vo.getId());  
        stmt.setString(2, vo.getPassword());  
        stmt.setString(3, vo.getName());  
        stmt.setString(4, vo.getRole());  
        stmt.executeUpdate();  
    } catch (Exception e) {  
        e.printStackTrace();  
    } finally {  
        JDBCUtil.close(stmt, conn);  
    }  
}
```

InsertUserController Class 생성



InsertUserController Class 생성

InsertUserController.java

```
package controller.user;

import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import controller.Controller;

public class InsertUserController implements Controller {
    @Override
    public String handleRequest(HttpServletRequest request, HttpServletResponse
response) {
        String id = request.getParameter("id");
        return null;
    }
}
```


Login.html 생성

Login.html

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
<title>Insert title here</title>
</head>
<body>
<h1>로그인</h1>
<hr/>
<form action="login.do" method="post">
  <table border="1">
    <tr>
      <td>아이디</td>
      <td><input type="text" name="id"/>
    </tr>
```

Login.html 생성

Login.html

```
<tr>
  <td>비밀번호</td>
  <td><input type="password" name="password"/>
</tr>
<tr>
  <td colspan="2"><input type="submit" value="login"/>
</tr>
</table>
</form>
<br/>
<a href="insertUser.html">회원가입</a>
</body>
</html>
```

insertUser.html 생성

insertUser.html

```
<!DOCTYPE html>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
<title>Insert title here</title>
</head>
<body>
<h1>회원가입</h1>
<hr/>
<form action="insertUser.do" method="post">
  <table border="1">
    <tr>
      <td>아이디</td>
      <td><input type="text" name="id"/>
    </tr>
```

insertUser.html 생성

insertUser.html

```
<tr>
  <td>비밀번호</td>
  <td><input type="password" name="password"/>
</tr>
<tr>
  <td>이름</td>
  <td><input type="text" name="name"/>
</tr>
<tr>
  <td>권한</td>
  <td>
    <input type="radio" name="role" value="Admin" checked="checked"/> 관리자
    <input type="radio" name="role" value="User"/> 사용자
  </td>
</tr>
<tr>
  <td colspan="2"><input type="submit" value="회원가입"/>
</td>
</tr>
</table>
```

insertUser.html 생성

insertUser.html

```
</form>  
<br/>  
<a href="insertUser.html">회원가입</a>  
</body>  
</html>
```



Part 3,

로그인 인증

참고문헌

<https://shinny.tistory.com/77>

LoginController.java

LoginController.java

```
public class LoginController implements Controller {
    @Override
    public String handleRequest(HttpServletRequest request, HttpServletResponse response) {
        String id = request.getParameter("id");
        String password = request.getParameter("password");
        UserVO vo = new UserVO();
        vo.setId(id);
        vo.setPassword(password);

        UserDao dao = new UserDao();
        UserVO user = dao.getUser(vo);

        if(user != null) {
            HttpSession session=request.getSession();
            session.setAttribute("user", user);
            return "ok.jsp";
        }else {
            return "login.html";
        }
    }
}
```

LogoutController.java

LogoutController.java

```
public class LogoutController implements Controller {  
    @Override  
    public String handleRequest(HttpServletRequest request, HttpServletResponse response) {  
        HttpSession session = request.getSession();  
        session.invalidate();  
        return "login.html";  
    }  
}
```


Ok.jsp

Ok.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"  
    pageEncoding="EUC-KR"%>  
<!DOCTYPE html>  
<html>  
<head>  
<meta charset="EUC-KR">  
<title>Insert title here</title>  
</head>  
<body>  
<h1>로그인</h1>  
</body>  
</html>
```

HandlerMapping 추가

HandlerMapping.java

```
public HandlerMapping() {  
    mappings = new HashMap<String, Controller>();  
    mappings.put("/insertUser.do", new insertUserController());  
    mappings.put("/login.do", new LoginController());  
    mappings.put("/logout.do", new LogoutController());  
}  
public Controller getController(String path) {  
    return mappings.get(path);  
}
```

UserDAO 추가

UserDAO.java

```
public UserVO getUser(UserVO vo) {
    UserVO user = null;
    try {
        conn = JDBCUtil.getConnection();
        stmt = conn.prepareStatement(USER_GET);
        stmt.setString(1, vo.getId());
        stmt.setString(2, vo.getPassword());
        rs=stmt.executeQuery();
        if(rs.next()) {
            user = new UserVO();
            user.setId(rs.getString("ID"));
            user.setPassword(rs.getString("PASSWORD"));
            user.setName(rs.getString("NAME"));
            user.setRole(rs.getString("ROLE"));
        }
    } catch (Exception e) {
        e.printStackTrace();
    } finally {
        JDBCUtil.close(rs, stmt, conn);
    }
    return user;
}
```

경청해주셔서 감사합니다.