MYONGJI UNIVERSITY

Servlet & JSP 3: MVC 활용

2012년 웹프로그래밍

권동섭

참조 문헌

- Java EE 6 Tutorial
  - http://docs.oracle.com/javaee/6/tutorial/doc/
- Apache Tomcat 7 Document: First Webapp Tutorial
  - http://tomcat.apache.org/tomcat-7.0-doc/appdev/index.html
- Servlet & JSP Tutorial in CoreServlet.com
  - http://courses.coreservlets.com/Course-Materials/csajsp2.html

Outline

- Basic
  - **Servlet & JSP 소개**
  - **JSP 기본 구문**
  - **Request를 이용한 폼 입력 처리 방법**
  - **Session을 이용한 로그인**

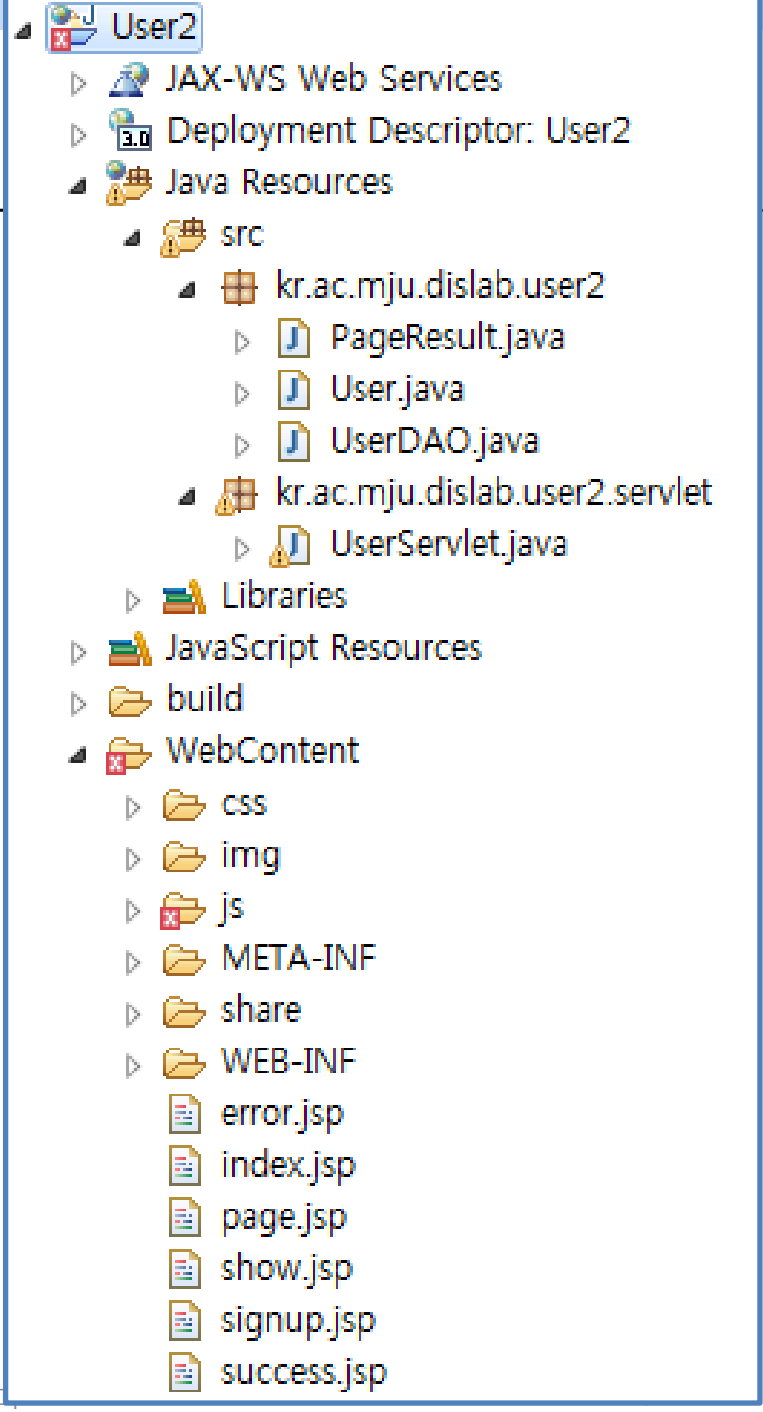
- Advanced
  - **JDBC를 이용한 데이터베이스 연동**
  - Data Source를 이용한 DB Connection Pooling
  - DAO 패턴
  - Java Bean 이용
  - Servlet과 JSP의 연결

USER1: JDBC + JSP 예제의 문제점?

- 동일한 코드의 반복?
  - DRY (Don't Repeat Yourself!)
- 코드의 재사용 불가능
- HTML코드에 비하여 Java코드가 너무 많고 복잡함
  - 가분수 코드?
- DB 관리 코드가 흩어져 있음

실습: User2

- Model
  - Java Bean: User, PageResult
  - DAO: UserDAO
- Controller
  - UserServlet
- View
  - index.jsp
  - page.jsp
  - show.jsp
  - signup.jsp
  - success.jsp
  - error.jsp



MYONGJI UNIVERSITY

JAVA BEAN

Java Bean

- 재사용 가능한 자바 컴포넌트 (클래스)
  - **Serializable** 해야 함

```
implements java.io.Serializable
```

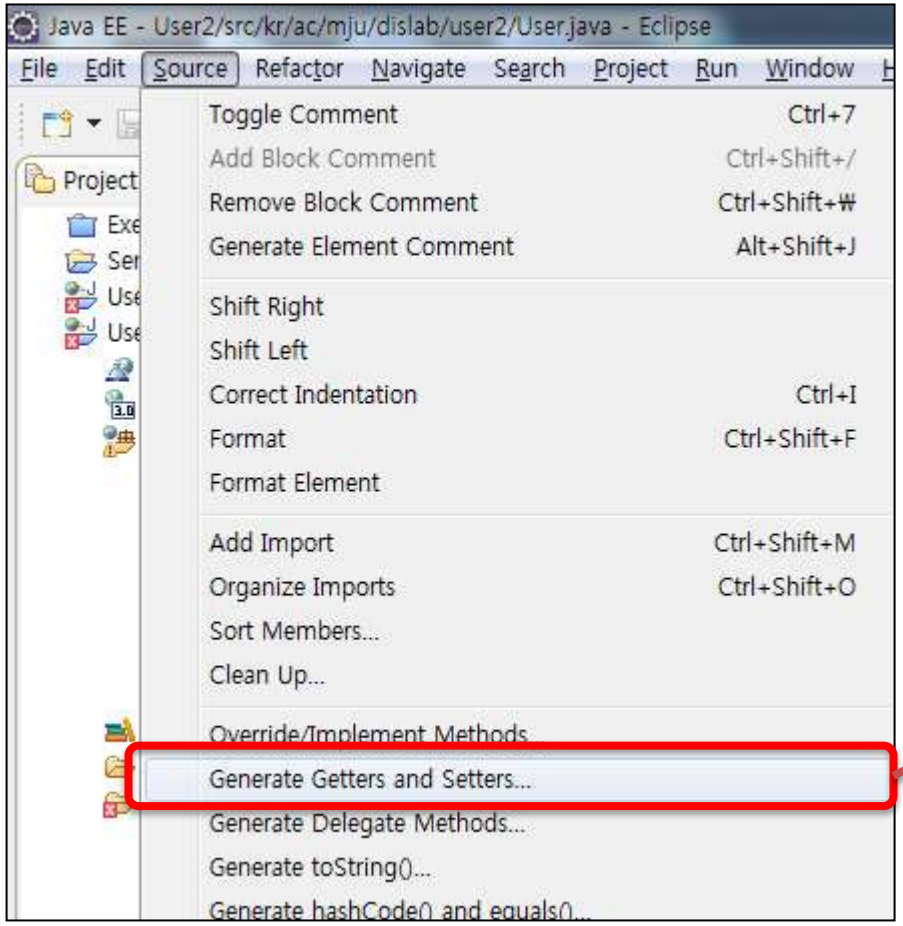
  - **Zero-argument constructor** 가 있어야 함

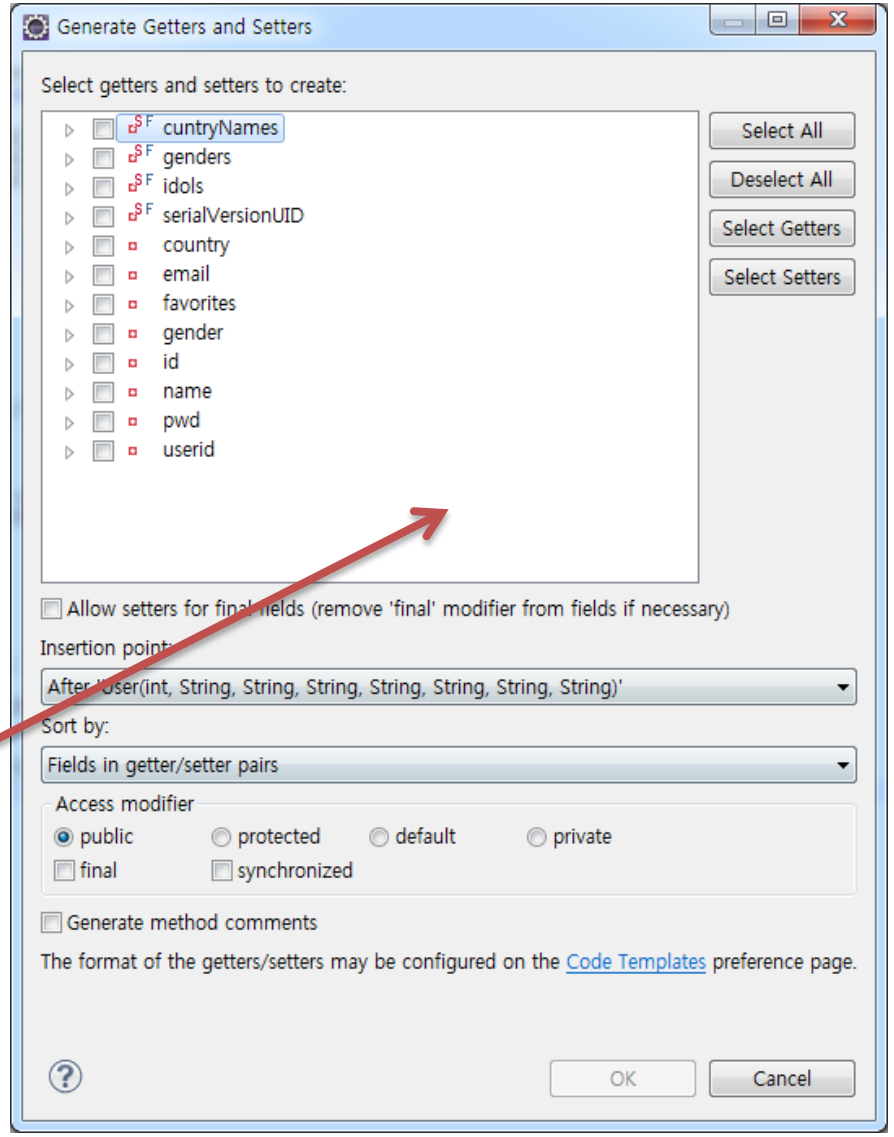
해야 할 일이 없으면 빈 생성자 라도 만들어 두면 됨.

  - 모든 property는 **getter, setter**로 접근 가능해야 함 (public Field는 없어야 함!)

Eclipse를 이용하면 한 번에 자동 생성 가능함.

Eclipse의 소스 생성 기능 활용 가능





Servlet/JSP에서 Bean의 활용

- JSP로 전달 가능한 데이터는 모두 Java Bean
- 일반적으로 DB Entity 들은 Bean 클래스로 나타남
  - ER 모델링이나 UML (Class diagram) 모델링 활용 가능
- getter, setter 외에도 각 클래스에 적합한 책임(기능) 부여 (코드 모듈화, 캡슐화, 추상화)

User.java : 사용자 모델 (Bean)

```java
package kr.ac.mju.dislab.user2;

import java.util.*;

public class User implements java.io.Serializable {
    private static final long serialVersionUID = 2193897931951340673L;

    private static final String[] cuntryNames = {"한국", "미국", "영국", "일본", "중국"};
    private static final String[] idols = {"아이유", "카라", "소녀시대", "2NE1", "씨스타"};
    private static final String[][] genders = {{"M", "남성"}, {"F", "여성"}};

    private int id;
    private String userid;
    private String name;
    private String pwd;
    private String email;
    private String country;
    private String gender;
    private String favorites;

    // No-arg constructor 가 있어야 한다.
    public User() {
    }
```

```
27    public User(int id, String userid, String name, String pwd, String email,
28            String country, String gender, String favorites) {
29        super();
30        this.id = id;
31        this.userid = userid;
32        this.name = name;
33        this.pwd = pwd;
34        this.email = email;
35        this.country = country;
36        this.gender = gender;
37        this.favorites = favorites;
38    }
39
40    // getter & setter 가 있어야 한다. (Eclipse 에서 자동 생성 가능)
41    public int getId() {
42        return id;
43    }
44
45    public void setId(int id) {
46        this.id = id;
47    }
48
49    public String getUserid() {
50        return userid;
51    }
52
53    public void setUserid(String userid) {
54        this.userid = userid;
55    }
56
57    public String getName() {
58        return name;
59    }
60
61    public void setName(String name) {
62        this.name = name;
63    }
64
65    public String getPwd() {
66        return pwd;
67    }
```

```
69      public void setPwd(String pwd) {
70          this.pwd = pwd;
71      }
72
73      public String getEmail() {
74          return email;
75      }
76
77      public void setEmail(String email) {
78          this.email = email;
79      }
80
81      public String getCountry() {
82          return country;
83      }
84
85      public void setCountry(String country) {
86          this.country = country;
87      }
88
89      public String getGender() {
90          return gender;
91      }
92
93      public void setGender(String gender) {
94          this.gender = gender;
95      }
96
97      public String getFavorites() {
98          return favorites;
99      }
00
01      public void setFavorites(String favorites) {
02          this.favorites = favorites;
03      }
```

```
105     public String getGenderStr() {
106         if (gender.equals("M")) {
107             return "남성";
108         } else {
109             return "여성";
110         }
111     }
112
113     public String[] getCountryNames() {
114         return cuntryNames;
115     }
116
117     public String checkCountry(String countryName) {
118         return (countryName.equals(country)) ? "selected" : "";
119     }
120
121     public String[] getIdols() {
122         return idols;
123     }
124
125     public List<String> getFavoriteList() {
126         List<String> favoriteList = null;
127         if (favorites != null && favorites.length() > 0) {
128             favoriteList = Arrays.asList(StringUtils.split(favorites, ","));
129         }
130         return favoriteList;
131     }
132
133     public String checkIdol(String idolName) {
134         List<String> favoriteList = getFavoriteList();
135         return (favoriteList != null && favoriteList.contains(idolName))? "checked" : "";
136     }
137
138     public String[][] getGenders() {
139         return genders;
140     }
141
142     public String checkGender(String genderName) {
143         return (genderName.equals(gender)) ? "checked" : "";
144     }
145 }
```

PageResult.java : 객체를 페이지 형태로 접근

```java
package kr.ac.mju.dislab.user2;

import java.util.*;

public class PageResult<T> implements java.io.Serializable{
    private static final long serialVersionUID = -1826830567659349558L;

    private List<T> list;
    private int numItemsInPage;
    private int numItems;
    private int numPages;
    private int page;

    private final static int delta = 5;
    public List<T> getListUsers() {
        return list;
    }

    public PageResult(int numItemsInPage, int page) {
        super();
        this.numItemsInPage = numItemsInPage;
        this.page = page;
        numItems = 0;
        numPages = 0;
        list = new ArrayList<T>();
    }

    public List<T> getList() {
        return list;
    }
```

```
31
32      public int getNumItemsInPage() {
33          return numItemsInPage;
34      }
35
36      public int getNumItems() {
37          return numItems;
38      }
39
40      public void setNumItems(int numItems) {
41          this.numItems = numItems;
42          numPages = (int) Math.ceil((double)numItems / (double)numItemsInPage);
43      }
44      public int getNumPages() {
45          return numPages;
46      }
47
48      public int getPage() {
49          return page;
50      }
51
52      public int getStartPageNo() {
53          return (page <= delta) ? 1: page - delta;
54      }
55
56      public int getEndPageNo() {
57          int endPageNo = getStartPageNo() + (delta * 2) + 1;
58
59          if (endPageNo > numPages) {
60              endPageNo = numPages;
61          }
62
63          return endPageNo;
64      }
65  }
```

참고: Java Collection

Interface	Hash Table	Resizable Array	Balanced Tree	Linked List	Hash Table + Linked List
Set	HashSet		TreeSet		LinkedHashSet
List		ArrayList		LinkedList	
Deque		ArrayDeque		LinkedList	
Map	HashMap		TreeMap		LinkedHashMap

- 참고: http://docs.oracle.com/javase/7/docs/technotes/guides/collections/overview.html
- 자료구조+기본 알고리즘을 클래스 라이브러리로 제공
- 효과적인 재사용 및 변경이 가능하도록 함
- Object-Orient 설계 기법을 익히기 위한 좋은 예제

MYONGJI UNIVERSITY

DATA SOURCE를 활용한 DB 접속

JNDI Data Source

- 기존의 문제점
  - DB 접속 코드가 여러 군데 나누어져 있음
  - 개발 서버와 운영 서버의 DB 종류, 설정 등이 다를 수 있는데, 이 때마다 소스 코드를 바꿔야 하나?
- JNDI (Java Naming and Directory Interface)
  - Java에서 이름으로 특정 리소스를 찾는 방법
  - DB 접속 설정 등은 별도의 설정 파일에 저장하고, 소스 코드에서는 JNDI를 통해 이용 가능함
- JNDI를 이용한 Data Source 사용 장점
  - 소스 코드와 DB 접속 설정의 분리
  - 개발 서버, 운영 서버 등 별로 각각의 DB 설정 사용 가능

Apache Tomcat JNDI Datasource HOW-TO

- Tomcat 버전 별로 다를 수 있으므로 버전 확인 필수!
- http://tomcat.apache.org/tomcat-7.0-doc/jndi-datasource-examples-howto.html

1. Tomcat 서버 디렉토리의 **context.xml**에 리소스 설정
2. 애플리케이션의 WEB-INF/**web.xml**에 참조 설정
3. 소스 코드에서 데이터 소스 사용

참고: DBCP (DB Connection Pooling)

- 데이터베이스 connection을 반환하지 않고 재사용하여 성능을 향상시키는 방법
- DBCP 라이브러리를 사용하는 것이 일반적
  - Apache Commons-DBCP: http://commons.apache.org/dbcp/
  - C3P0 - http://sourceforge.net/projects/c3p0
- Tomcat, Spring등은 Apache DBCP 개선 버전을 제공
  - Tomcat JDBC Connection Pool http://tomcat.apache.org/tomcat-7.0-doc/jdbc-pool.html

context.xml : 서버 설정

```
32      <Resource name="jdbc/WebDB"
33                auth="Container"
34                type="javax.sql.DataSource"
35                factory="org.apache.tomcat.jdbc.pool.DataSourceFactory"
36                testWhileIdle="true"
37                testOnBorrow="true"
38                testOnReturn="false"
39                validationQuery="SELECT 1"
40                validationInterval="30000"
41                timeBetweenEvictionRunsMillis="30000"
42                maxActive="100"
43                minIdle="10"
44                maxWait="10000"
45                initialSize="10"
46                removeAbandonedTimeout="60"
47                removeAbandoned="true"
48                logAbandoned="true"
49                minEvictableIdleTimeMillis="30000"
50                jmxEnabled="true"
51                jdbcInterceptors=
52    "org.apache.tomcat.jdbc.pool.interceptor.ConnectionState;org.apache.tomcat.jdbc.pool.interceptor.StatementFinalizer"
53                username="web"
54                password="asdf"
55                driverClassName="com.mysql.jdbc.Driver"
56                url="jdbc:mysql://localhost:3306/web2012"
57                />
58      <!--
59       <Resource name="jdbc/WebDB" auth="Container" type="javax.sql.DataSource"
60                 maxActive="100" maxIdle="30" maxWait="10000"
61                 username="web" password="asdf" driverClassName="com.mysql.jdbc.Driver"
62                 url="jdbc:mysql://localhost:3306/web2012"/>
63                 -->
64
65  </Context>
```

WEB-INF/web.xml : 웹 애플리케이션 설정

```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <web-app xmlns:xsi="http://www.w3.org/2001/XMLS
3    <display-name>User2</display-name>
4    <welcome-file-list>
5      <welcome-file>user</welcome-file>
6      <welcome-file>index.html</welcome-file>
7      <welcome-file>index.htm</welcome-file>
8      <welcome-file>index.jsp</welcome-file>
9      <welcome-file>default.html</welcome-file>
10     <welcome-file>default.htm</welcome-file>
11     <welcome-file>default.jsp</welcome-file>
12   </welcome-file-list>
13   <resource-ref>
14     <description>DB Connection</description>
15     <res-ref-name>jdbc/WebDB</res-ref-name>
16     <res-type>javax.sql.DataSource</res-type>
17     <res-auth>Container</res-auth>
18   </resource-ref>
19 </web-app>
```

Data Source 활용 코드 예제

```
public static DataSource getDataSource() throws NamingException {
    Context initCtx = null;
    Context envCtx = null;

    // Obtain our environment naming context
    initCtx = new InitialContext();
    envCtx = (Context) initCtx.lookup("java:comp/env");

    // Look up our data source
    return (DataSource) envCtx.lookup("jdbc/WebDB");
}
```

```
DataSource ds = getDataSource();

conn = ds.getConnection();
stmt = conn.createStatement();
```

MYONGJI UNIVERSITY

DAO를 이용한 DATABASE 연동

Data Access Object (DAO)

- 물리적인 DB와 비즈니스 로직(모델) 클래스와의 직접적인 연결을 피하는 것이 바람직
- DB 접속 위한 코드들을 별도의 클래스로 추출 (CRUD)
  - CRUD: **C**REATE/**R**EAD/**U**PDATE/**D**ELETE
- 본 강의에서는 가장 단순한 형태의 DAO 클래스만 생성

- 실제 대형 프로젝트에서는 이러한 역할을 위한 별도의 프레임워크를 사용하는 경우가 많음
- ORM (Object-Relation Mapping): 클래스와 DB를 연결
  - iBatis, Hibernate, JPA (Java Persistence API) 등

UserDAO.java : User관련 DB 처리 추상화

```java
package kr.ac.mju.dislab.user2;

import java.sql.*;

import javax.naming.*;
import javax.sql.*;

public class UserDAO {
    public static DataSource getDataSource() throws NamingException {
        Context initCtx = null;
        Context envCtx = null;

        // Obtain our environment naming context
        initCtx = new InitialContext();
        envCtx = (Context) initCtx.lookup("java:comp/env");

        // Look up our data source
        return (DataSource) envCtx.lookup("jdbc/WebDB");
    }

    public static PageResult<User> getPage(int page, int numItemsInPage)
            throws SQLException, NamingException {
        Connection conn = null;
        Statement stmt = null;
        ResultSet rs = null;

        if (page <= 0) {
            page = 1;
        }

        DataSource ds = getDataSource();
        PageResult<User> result = new PageResult<User>(numItemsInPage, page);
```

```
        int startPos = (page - 1) * numItemsInPage;

        try {
            conn = ds.getConnection();
            stmt = conn.createStatement();

            // users 테이블: user 수 페이지수 개산
            rs = stmt.executeQuery("SELECT COUNT(*) FROM users ORDER BY name");
            rs.next();

            result.setNumItems(rs.getInt(1));

            rs.close();
            rs = null;
            stmt.close();
            stmt = null;

            // users 테이블 SELECT
            stmt = conn.createStatement();
            rs = stmt.executeQuery("SELECT * FROM users ORDER BY name LIMIT " + startPos + ", " + numItemsInPage);

            while(rs.next()) {
                result.getList().add(new User(rs.getInt("id"),
                            rs.getString("userid"),
                            rs.getString("name"),
                            rs.getString("pwd"),
                            rs.getString("email"),
                            rs.getString("country"),
                            rs.getString("gender"),
                            rs.getString("favorites")
                        ));
            }
        } finally {
            // 무슨 일이 있어도 리소스를 제대로 종료
            if (rs != null) try{rs.close();} catch(SQLException e) {}
            if (stmt != null) try{stmt.close();} catch(SQLException e) {}
            if (conn != null) try{conn.close();} catch(SQLException e) {}
        }

        return result;
    }
```

```java
77    public static User findById(int id) throws NamingException, SQLException{
78        User user = null;
79
80        Connection conn = null;
81        PreparedStatement stmt = null;
82        ResultSet rs = null;
83
84        DataSource ds = getDataSource();
85
86        try {
87            conn = ds.getConnection();
88
89            // 질의 준비
90            stmt = conn.prepareStatement("SELECT * FROM users WHERE id = ?");
91            stmt.setInt(1, id);
92
93            // 수행
94            rs = stmt.executeQuery();
95
96            if (rs.next()) {
97                user = new User(rs.getInt("id"),
98                        rs.getString("userid"),
99                        rs.getString("name"),
100                       rs.getString("pwd"),
101                       rs.getString("email"),
102                       rs.getString("country"),
103                       rs.getString("gender"),
104                       rs.getString("favorites"));
105           }
106       } finally {
107           // 무슨 일이 있어도 리소스를 제대로 종료
108           if (rs != null) try{rs.close();} catch(SQLException e) {}
109           if (stmt != null) try{stmt.close();} catch(SQLException e) {}
110           if (conn != null) try{conn.close();} catch(SQLException e) {}
111       }
112
113       return user;
114   }
```

```java
    public static boolean create(User user) throws SQLException, NamingException {
        int result;
        Connection conn = null;
        PreparedStatement stmt = null;
        ResultSet rs = null;

        DataSource ds = getDataSource();

        try {
            conn = ds.getConnection();

            // 질의 준비
            stmt = conn.prepareStatement(
                    "INSERT INTO users(userid, name, pwd, email, country, gender, favorites) " +
                    "VALUES(?, ?, ?, ?, ?, ?, ?)"
                    );
            stmt.setString(1,  user.getUserid());
            stmt.setString(2,  user.getName());
            stmt.setString(3,  user.getPwd());
            stmt.setString(4,  user.getEmail());
            stmt.setString(5,  user.getCountry());
            stmt.setString(6,  user.getGender());
            stmt.setString(7,  user.getFavorites());

            // 수행
            result = stmt.executeUpdate();
        } finally {
            // 무슨 일이 있어도 리소스를 제대로 종료
            if (rs != null) try{rs.close();} catch(SQLException e) {}
            if (stmt != null) try{stmt.close();} catch(SQLException e) {}
            if (conn != null) try{conn.close();} catch(SQLException e) {}
        }

        return (result == 1);
    }
```

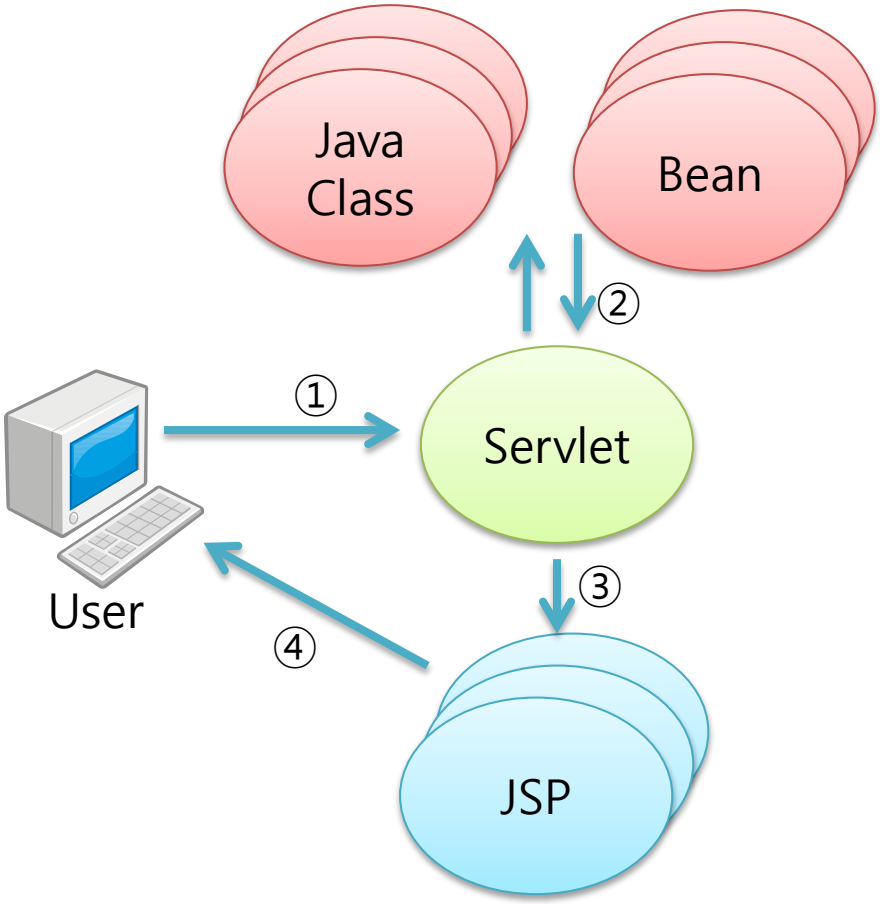
```
152    public static boolean update(User user) throws SQLException, NamingException {
153        int result;
154        Connection conn = null;
155        PreparedStatement stmt = null;
156        ResultSet rs = null;
157
158        DataSource ds = getDataSource();
159
160        try {
161            conn = ds.getConnection();
162
163            // 질의 준비
164            stmt = conn.prepareStatement(
165                    "UPDATE users " +
166                    "SET  userid=?, name=?, email=?, country=?, gender=?, favorites=? " +
167                    "WHERE id=?"
168                    );
169            stmt.setString(1,  user.getUserid());
170            stmt.setString(2,  user.getName());
171            stmt.setString(3,  user.getEmail());
172            stmt.setString(4,  user.getCountry());
173            stmt.setString(5,  user.getGender());
174            stmt.setString(6,  user.getFavorites());
175            stmt.setInt(7,  user.getId());
176
177            // 수행
178            result = stmt.executeUpdate();
179        } finally {
180            // 무슨 일이 있어도 리소스를 제대로 종료
181            if (rs != null) try{rs.close();} catch(SQLException e) {}
182            if (stmt != null) try{stmt.close();} catch(SQLException e) {}
183            if (conn != null) try{conn.close();} catch(SQLException e) {}
184        }
185
186        return (result == 1);
187    }
```

```
189     public static boolean remove(int id) throws NamingException, SQLException {
190         int result;
191         Connection conn = null;
192         PreparedStatement stmt = null;
193         ResultSet rs = null;
194 
195         DataSource ds = getDataSource();
196 
197         try {
198             conn = ds.getConnection();
199 
200             // 질의 준비
201             stmt = conn.prepareStatement("DELETE FROM users WHERE id=?");
202             stmt.setInt(1,  id);
203 
204             // 수행
205             result = stmt.executeUpdate();
206         } finally {
207             // 무슨 일이 있어도 리소스를 제대로 종료
208             if (rs != null) try{rs.close();} catch(SQLException e) {}
209             if (stmt != null) try{stmt.close();} catch(SQLException e) {}
210             if (conn != null) try{conn.close();} catch(SQLException e) {}
211         }
212 
213         return (result == 1);
214     }
215 }
```

MYONGJI UNIVERSITY

MVC

MVC: Servlet + JSP



Model (Business Logic)

프로그램의 핵심 기능, 데이터 처리 등을 재활용 가능하도록 모듈화함. 웹 프로그래밍이 아닌 곳에서도 활용할 수 있도록 설계

Controller

사용자의 입력을 모델에 전달하여 처리하고, 그 결과를 뷰에 전달하여 연결해 줌

View

최종 결과물로 나올 HTML을 생성함

Servlet과 JSP를 이용한 MVC 방법

- Servlet
  - 사용자로 부터의 request 처리
  - 서블릿 메소드 내부에는 복잡한 로직을 최소화하고, 별도의 클래스 라이브러리(Business Logic)를 활용
  - 결과를 **Bean** 인스턴스로 받음
  - Bean을 request, session, application 등에 저장
  - 적절한 JSP 페이지로 **Forward: RequestDispatcher**
- JSP
  - HTML 결과 생성
  - Bean의 결과를 이용: **EL (Expression Language)**활용
  - Java 코드를 최소화 하는 것이 바람직
  - Scriptlet (<% %>)대신 **JSTL (JSP Tag Library)** 사용

Data Sharing Scope

- Request: forward하는 페이지까지만 데이터 전달 (기본)
- Session: 세션 내에서는 계속 유지 됨. redirect 주로 사용
- Application: 전체 사용자가 공유

- 주의
  - 리소스 소모 등의 측면에서 가능하면 적은 수준에서 공유하는 것이 바람직함
  - Session이나 Application의 경우 Thread Safe 문제가 있으므로 필요하면 동기화를 해야 함

UserServlet.java

- 컨트롤러

```java
package kr.ac.mju.dislab.user2.servlet;

import java.io.IOException;
import java.sql.SQLException;
import java.util.*;

import javax.naming.NamingException;
import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import org.apache.commons.lang3.StringUtils;

import kr.ac.mju.dislab.user2.*;

/**
 * Servlet implementation class User
 */
@WebServlet("/user")
public class UserServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public UserServlet() {
        super();
    }


    private int getIntFromParameter(String str, int defaultValue) {
        int id;

        try {
            id = Integer.parseInt(str);
        } catch (Exception e) {
            id = defaultValue;
        }
        return id;
    }
```

```java
    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException
        String op = request.getParameter("op");
        String actionUrl = "";
        boolean ret;

        int id = getIntFromParameter(request.getParameter("id"), -1);

        if (op == null && id > 0) {
            op = "show";
        }

        try {
            if (op == null || op.equals("index")) {
                int page = getIntFromParameter(request.getParameter("page"), 1);

                PageResult<User> users = UserDAO.getPage(page, 10);
                request.setAttribute("users", users);
                request.setAttribute("page", page);
                actionUrl = "index.jsp";
            } else if (op.equals("show")) {
                User user = UserDAO.findById(id);
                request.setAttribute("user", user);

                actionUrl = "show.jsp";
            } else if (op.equals("update")) {
                User user = UserDAO.findById(id);
                request.setAttribute("user", user);
                request.setAttribute("method", "PUT");

                actionUrl = "signup.jsp";
            } else if (op.equals("delete")) {
                ret = UserDAO.remove(id);
                request.setAttribute("result", ret);

                if (ret) {
                    request.setAttribute("msg", "사용자 정보가 삭제되었습니다.");
                    actionUrl = "success.jsp";
                } else {
                    request.setAttribute("error", "사용자 정보 삭제에 실패했습니다.");
                    actionUrl = "error.jsp";
                }
```

```
89
90              } else if (op.equals("signup")) {
91                  request.setAttribute("method", "POST");
92                  request.setAttribute("user", new User());
93                  actionUrl = "signup.jsp";
94              } else {
95                  request.setAttribute("error", "알 수 없는 명령입니다");
96                  actionUrl = "error.jsp";
97              }
98          } catch (SQLException | NamingException e) {
99              request.setAttribute("error", e.getMessage());
100             e.printStackTrace();
101             actionUrl = "error.jsp";
102         }
103
104         RequestDispatcher dispatcher = request.getRequestDispatcher(actionUrl);
105         dispatcher.forward(request,  response);
106
107     }
108
109
110     private boolean isRegisterMode(HttpServletRequest request) {
111         String method = request.getParameter("_method");
112         return method == null || method.equals("POST");
113     }
114
115     /**
116      * @see HttpServlet#doPost(HttpServletRequest request, HttpServletResponse response
117      */
```

```java
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletExceptio
    boolean ret = false;
    String actionUrl;
    String msg;
    User user = new User();

    request.setCharacterEncoding("utf-8");

    String userid = request.getParameter("userid");
    String pwd = request.getParameter("pwd");
    String pwd_confirm = request.getParameter("pwd_confirm");
    String name = request.getParameter("name");
    String email = request.getParameter("email");
    String country = request.getParameter("country");
    String gender = request.getParameter("gender");
    String[] favorites = request.getParameterValues("favorites");
    String favoriteStr = StringUtils.join(favorites, ",");

    List<String> errorMsgs = new ArrayList<String>();

    if (isRegisterMode(request)) {
        if (pwd == null || pwd.length() < 6) {
            errorMsgs.add("비밀번호는 6자 이상 입력해주세요.");
        }

        if (!pwd.equals(pwd_confirm)) {
            errorMsgs.add("비밀번호가 일치하지 않습니다.");
        }
        user.setPwd(pwd);
    } else {
        user.setId(getIntFromParameter(request.getParameter("id"), -1));
    }

    if (userid == null || userid.trim().length() == 0) {
        errorMsgs.add("ID를 반드시 입력해주세요.");
    }

    if (name == null || name.trim().length() == 0) {
        errorMsgs.add("이름을 반드시 입력해주세요.");
    }

    if (gender == null || !(gender.equals("M") || gender.equals("F") )) {
        errorMsgs.add("성별에 적합하지 않은 값이 입력되었습니다.");
    }
```

```
163         user.setName(name);
164         user.setUserid(userid);
165         user.setCountry(country);
166         user.setEmail(email);
167         user.setFavorites(favoriteStr);
168         user.setGender(gender);
169
170         try {
171             if (isRegisterMode(request)) {
172                 ret = UserDAO.create(user);
173                 msg = "<b>" + name + "</b>님의 사용자 정보가 등록되었습니다.";
174             } else {
175                 ret = UserDAO.update(user);
176                 actionUrl = "success.jsp";
177                 msg = "<b>" + name + "</b>님의 사용자 정보가 수정되었습니다.";
178             }
179             if (ret != true) {
180                 errorMsgs.add("변경에 실패했습니다.");
181                 actionUrl = "error.jsp";
182             } else {
183                 request.setAttribute("msg", msg);
184                 actionUrl = "success.jsp";
185
186             }
187         } catch (SQLException | NamingException e) {
188             errorMsgs.add(e.getMessage());
189             actionUrl = "error.jsp";
190         }
191
192         request.setAttribute("errorMsgs", errorMsgs);
193         RequestDispatcher dispatcher = request.getRequestDispatcher(actionUrl);
194         dispatcher.forward(request,  response);
195     }
196
197 }
```

MYONGJI UNIVERSITY

EL & JSTL

JSP에서 Java Bean을 이용하는 방법

- Servlet에서 저장한 Bean을 어떻게 사용하나?
  - request.setAttribute("user", user);
- user.getName( )을 얻기 위한 방법?

- 방법 1) **JSP Action** 활용
  - **<jsp:useBean id="user" type="kr.ac.dislab.user2.User" scope="request"/>**
  - **<jsp:getProperty name="user" property="name"/>**

- 방법 2) **Expression Language** 활용
  - **${user.name}**

Expression Language (EL)

- JSP 2.0 (Servlet 2.4)이후에 들어간 JSP에서 Bean을 쉽게 읽기 위한 스크립팅 방법
- 형태: **${expression}**
- expression에 가능한 것
  - 변수: ${**variable**} , ${**name.property**}
    - name은 request, session, application 순으로 자동으로 검색
    - property는 getter가 존재해야 함
  - 계산식: ${3+2-1}

- 암묵적으로 사용가능한 객체
  - param, header, cookie 등의 객체는 사용 가능

JSTL

- JSP Standard Tag Library
- HTML tag외에 MVC에서 유용하게 사용할 수 있는 태그들을 제공하는 라이브러리
- 스크립틀릿(<% %>) 대신 가능하면 JSTL을 사용

- Download: http://jstl.java.net/
- 설치: WEB-INF/lib 아래에 jar 파일 복사
- 사용법: JSP 파일 상단에 다음 줄 추가 후 사용

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

주요 구문

- 반복: Java의 for(var v: items) 와 유사
  - <c:forEach var="*name*" items="${*el*}">...</c:forEach>
- 조건
  - <c:if test="${*el*}">...</c:if>
  - <c:choose><c:when test="test">...</c:when>...</c:choose>
- 출력
  - <c:out value="${el}"/>
    - Escape: "<" 등의 문자를 entity(예: "<")로 변환
    - **Cross-site Scripting (XSS) 공격을 차단**할 수 있음

index.jsp : 사용자 목록

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>회원목록</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <link href="css/base.css" rel="stylesheet">
  <script src="js/jquery-1.8.2.min.js"></script>
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
<jsp:include page="share/header.jsp">
  <jsp:param name="current" value="Sign Up"/>
</jsp:include>

  <div class="container">
    <div class="row">
      <div class="span12 page-info">
        <div class="pull-left">
          Total <b>${users.numItems }</b> users
        </div>
        <div class="pull-right">
          <b>${users.page }</b> page / total <b>${users.numPages }</b> pages
        </div>
      </div>
    </div>
```

```
<table class="table table-bordered table-stripped">
  <thead>
    <tr>
      <th>ID</th>
      <th>Name</th>
      <th>Email</th>
      <th>Gender</th>
      <th>Country</th>
      <th></th>
    </tr>
  </thead>
  <tbody>
  <c:forEach var="user" items="${users.list }">
    <tr>
      <td><a href="user?id=${user.id}"><c:out value="${user.userid}"/></a></td>
      <td><c:out value="${user.name}"/></td>
      <td><c:out value="${user.email}"/></td>
      <td><c:out value="${user.genderStr}"/></td>
      <td><c:out value="${user.country}"/></td>
      <td><a href="user?op=update&id=${user.id}"
        class="btn btn-mini">modify</a> <a href="#"
        class="btn btn-mini btn-danger" data-action="delete"
        data-id="${user.id}">delete</a></td>
    </tr>
  </c:forEach>
  </tbody>
</table>

<jsp:include page="page.jsp">
  <jsp:param name="currentPage" value="${users.page}"/>
  <jsp:param name="url" value="user"/>
  <jsp:param name="startPage" value="${users.startPageNo}"/>
  <jsp:param name="endPage" value="${users.endPageNo}"/>
  <jsp:param name="numPages" value="${users.numPages}"/>
</jsp:include>
```

```
65      <div class="form-action">
66        <a href="user?op=signup" class="btn btn-primary">Sign Up</a>
67      </div>
68    </div>
69  <jsp:include page = "share/footer.jsp" />
70  </body>
71  <script>
72    $("a[data-action='delete']").click(function() {
73      if (confirm("정말로 삭제하시겠습니까?")) {
74        location = 'user?op=delete&id=' + $(this).attr('data-id');
75      }
76      return false;
77    });
78  </script>
79  </html>
```

page.jsp : 페이지 번호 선택 부분 모듈화

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<c:set var="currentPage" value="${param.currentPage}" />
<c:set var="url" value="${param.url}" />
<c:set var="startPage" value="${param.startPage}" />
<c:set var="endPage" value="${param.endPage}" />
<c:set var="numPages" value="${param.numPages}" />
<div class="pagination pagination-centered">
  <ul>
    <c:choose>
      <c:when test="${1 >= currentPage}">
        <li class="disabled"><a href="#">&laquo;</a></li>
      </c:when>
      <c:otherwise>
        <li><a href="${url}?page=${currentPage - 1}">&laquo;</a></li>
      </c:otherwise>
    </c:choose>

```

```
    <c:forEach var="i" begin="${startPage}"  end="${endPage}">
      <c:choose>
        <c:when test="${i == currentPage}">
          <li class="active"><a href="${url}?page=${i}">${i}</a></li>
        </c:when>
        <c:otherwise>
          <li><a href="${url}?page=${i}">${i}</a></li>
        </c:otherwise>
      </c:choose>
    </c:forEach>

    <c:choose>
      <c:when test="${currentPage >= numPages}">
        <li class="disabled"><a href="#">&raquo;</a></li>
      </c:when>
      <c:otherwise>
        <li><a href="${url}?page=${currentPage + 1}">&raquo;</a></li>
      </c:otherwise>
    </c:choose>
  </ul>
</div>
```

show.jsp : 회원 정보 상세보기

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" import="java.util.*"  import="java.sql.*"
    import="org.apache.commons.lang3.StringUtils"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>회원목록</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <link href="css/base.css" rel="stylesheet">
  <script src="js/jquery-1.8.2.min.js"></script>
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
<jsp:include page="share/header.jsp">
  <jsp:param name="current" value="Sign Up"/>
</jsp:include>
```

```
<div class="container">
    <div>
      <h3><c:out value="${user.name}"></c:out></h3>
      <ul>
        <li>User ID: <c:out value="${user.userid}"/></li>
        <li>Country: <c:out value="${user.country}"/></li>
        <li>Email: <a href="mailto:${user.email}"><c:out value="${user.email}"/></a></li>
        <li>Gender: ${user.genderStr }</li>
        <li>Favorites: <c:out value="${user.favorites }"/></li>
      </ul>
    </div>

    <div class="form-actions">
      <a href="user" class="btn">목록으로</a>
      <a href="user?op=update&id=${user.id}" class="btn btn-primary">수정</a>
      <a href="#" class="btn btn-danger" data-action="delete" data-id="${user.id}" >삭제</a>
    </div>
    <script>
      $("a[data-action='delete']").click(function() {
        if (confirm("정말로 삭제하시겠습니까?")) {
          location = 'user?op=delete&id=' + $(this).attr('data-id');
        }
        return false;
      });
    </script>
  </div>
</body>
</html>
```

signup.jsp : 회원 가입 폼 (가입/수정 공용)

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>회원목록</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <link href="css/base.css" rel="stylesheet">
  <script src="js/jquery-1.8.2.min.js"></script>
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
<jsp:include page="share/header.jsp">
  <jsp:param name="current" value="Sign Up"/>
</jsp:include>

 <div class="container">
   <div>
      <form class="form-horizontal" action="user" method="POST">
      <fieldset>
        <legend class="legend">Sign Up</legend>
        <c:if test="${method == 'PUT'}">
          <input type="hidden" name="id" value="${user.id }"/>
          <input type="hidden" name="_method" value="PUT"/>
        </c:if>
```

```
28        <div class="control-group">
29          <label class="control-label" for="userid">ID</label>
30          <div class="controls">
31            <input type="text" name="userid" value="${user.userid}">
32          </div>
33        </div>
34
35        <div class="control-group">
36          <label class="control-label" for="name">Name</label>
37          <div class="controls">
38            <input type="text" placeholder="홍길동" name="name" value="${user.name}">
39          </div>
40        </div>
41
42        <c:if test="${method == 'POST'}">
43          <%-- 신규 가입일 때만 비밀번호 입력창을 나타냄 --%>
44          <div class="control-group">
45            <label class="control-label" for="pwd">Password</label>
46            <div class="controls">
47              <input type="password" name="pwd">
48            </div>
49          </div>
50
51          <div class="control-group">
52            <label class="control-label" for="pwd_confirm">Password Confirmation</label>
53            <div class="controls">
54              <input type="password" name="pwd_confirm">
55            </div>
56          </div>
57        </c:if>
58        <div class="control-group">
59          <label class="control-label" for="email">E-mail</label>
60          <div class="controls">
61            <input type="email" placeholder="hong@abc.com" name="email" value="${user.email}">
62          </div>
63        </div>
```

```
65        <div class="control-group">
66          <label class="control-label">Country</label>
67          <div class="controls">
68            <select name="country">
69              <c:forEach var="countryName" items="${user.countryNames}">
70                <option ${user.checkCountry(countryName)}>${countryName }</option>
71              </c:forEach>
72            </select>
73          </div>
74        </div>
75
76        <div class="control-group">
77          <label class="control-label">Gender</label>
78          <div class="controls">
79            <c:forEach var="gender" items="${user.genders}">
80              <label class="radio">
81                <input type="radio" value="${gender[0]}" name="gender" ${user.checkGender(gender[0])}/>
82                ${gender[1]}
83              </label>
84            </c:forEach>
85          </div>
86        </div>
87
88        <div class="control-group">
89          <label class="control-label">Favorites</label>
90          <div class="controls">
91            <c:forEach var="idolName" items="${user.idols}">
92              <label class="checkbox">
93                <input type="checkbox" name="favorites" value="${idolName}" ${user.checkIdol(idolName)}/>
94                ${idolName}
95              </label>
96            </c:forEach>
97          </div>
98        </div>
```

```
100          <div class="form-actions">
101            <a href="user" class="btn">목록으로</a>
102            <c:choose>
103              <c:when test="${method=='POST'}">
104                <input type="submit" class="btn btn-primary" value="가입">
105              </c:when>
106              <c:otherwise>
107                <input type="submit" class="btn btn-primary" value="수정">
108              </c:otherwise>
109            </c:choose>
110          </div>
111        </fieldset>
112        </form>
113      </div>
114    </div>
115  </body>
116  </html>
```

success.jsp : 가입/수정/삭제 등이 성공했을 때

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>회원목록</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <link href="css/base.css" rel="stylesheet">
  <script src="js/jquery-1.8.2.min.js"></script>
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
<jsp:include page="share/header.jsp">
  <jsp:param name="current" value="Sign Up"/>
</jsp:include>
  <div class="container">
    <div class="alert alert-success">
      ${msg}
    </div>
    <div class="form-action">
      <a href="user" class="btn">목록으로</a>
    </div>
  </div>
</body>
</html>
```

error.jsp : 가입/수정/삭제 등에서 에러 출력

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>회원목록</title>
  <link href="css/bootstrap.min.css" rel="stylesheet">
  <link href="css/base.css" rel="stylesheet">
  <script src="js/jquery-1.8.2.min.js"></script>
  <script src="js/bootstrap.min.js"></script>
</head>
<body>
<jsp:include page="share/header.jsp">
  <jsp:param name="current" value="Sign Up"/>
</jsp:include>
  <div class="container">
    <div class="alert alert-error">
      <c:out value="${error}"/>

      <c:if test="${errorMsg != null || errorMsg.size() > 0 }">
        <h3>Errors:</h3>
        <ul>
          <c:forEach var="msg" items="${errorMsgs}">
            <li>${msg}</li>
          </c:forEach>
        </ul>
      </c:if>
    </div>
  </div>
<jsp:include page = "share/footer.jsp" />

</body>
</html>
```

과제 3: Servlet/JSP + JDBC

- JSP를 이용하여 간단한 게시판을 작성하시오.
  - JSP만으로 작성해도 무방함
  - JSP + Servlet으로 작성하는 것을 권함
- 기능
  - 페이지 단위로 목록 보기, 글 내용 보기, 작성, 수정, 삭제
- 제출방법: 소스파일 전체를 "학번.zip" 파일로 압축 제출
  - 테이블 생성을 위한 SQL문 반드시 포함
- 제출기한: 홈페이지 공지

과제 4: 웹 보안 관련 보고서 제출

- 웹 애플리케이션의 보안에 관하여 다음 내용들을 조사하고 그 결과를 보고서로 작성하여 제출하시오.
  - **SQL Injection**과 **Cross-site Scripting** 공격의 정의 및 방어법
  - 사용자의 암호 및 개인 정보를 안전하게 보관하기 위한 방법

- 제출 방법
  - 학교 보고서 표지 사용
  - 제출 파일 형태는 PDF로 (무료 pdf 변환 툴 사용 권장)
  - 제출 파일명: “학번.pdf”

- 제출 기간: 홈페이지 공지