

SiteMesh/Ibatis/Mybatis



SiteMesh

- SiteMesh의 가장 큰 장점은 매우 쉽게 웹 페이지들을 구성할 수 있는 방법을 제공하는 것이다.
SiteMesh 웹 페이지의 화면 구성에 관한 프레임워크로 대형 웹사이트에서 많은 웹 페이지가 일관된 사용자 룩앤필, 페이지이동, 페이지 레이아웃을 갖도록 하는데 도움을 준다.
- Decorate 디자인 패턴: SiteMesh는 웹서버로부터 웹브라우저로 제공되는 원본 콘텐츠를 파싱하고, 콘텐츠로부터 속성과 데이터를 추출하여 적절한 최종 결과물을 생산해낸다. 예를들면 JSP페이지 수행결과가 웹브라우저에 가기 전에 중간에 가로채서 문서의 내용을 변경할 수 있게 해준다.
- Composite 디자인 패턴: SiteMesh는 전체 웹페이지를 통채로 어느 다른 페이지의 일부 페이지에 삽입할 수 있게해준다. 이는 Sever_side include 방법과 비슷하긴 하나 HTML 문서가 하나의 가상 페이지로 만들어지기 위하여 내부적으로 수정된다. 간단한 포털 형태의 웹사이트는 이 방법을 이용하여 빨리 쉽게 만들 수 있다.
- SiteMesh는 Java 2, Servlet, JSP, XML 기반 위에 동작하였지만 자바기반이 다른 다른 웹 아키텍처도 잘 통합된다. CGI (Perl, Python etc), PHP, Cold Fusion 등등

동작

- SiteMesh는 Servlet 스펙 중에 좀 덜 알려진 page filter를 구현하였다.
만일 현재 날짜와 시간을 출력하는 JSP가 하나 있다고 하자.
보통 그 JSP를 웹 브라우저를 통해 호출하게 되면, 해당 요청은 웹 어플리케이션 서버로 도착한 뒤, 웹 어플리케이션 서버가 해당 JSP 수행하고 결과인 HTML문서를 웹 브라우저에게 돌려준다. SiteMesh는 일종의 웹페이지의 filter 처럼 동작하여, 아까 수행 결과인 HTML문서를 웹 브라우저에게 보내기 전에 변경을 한다

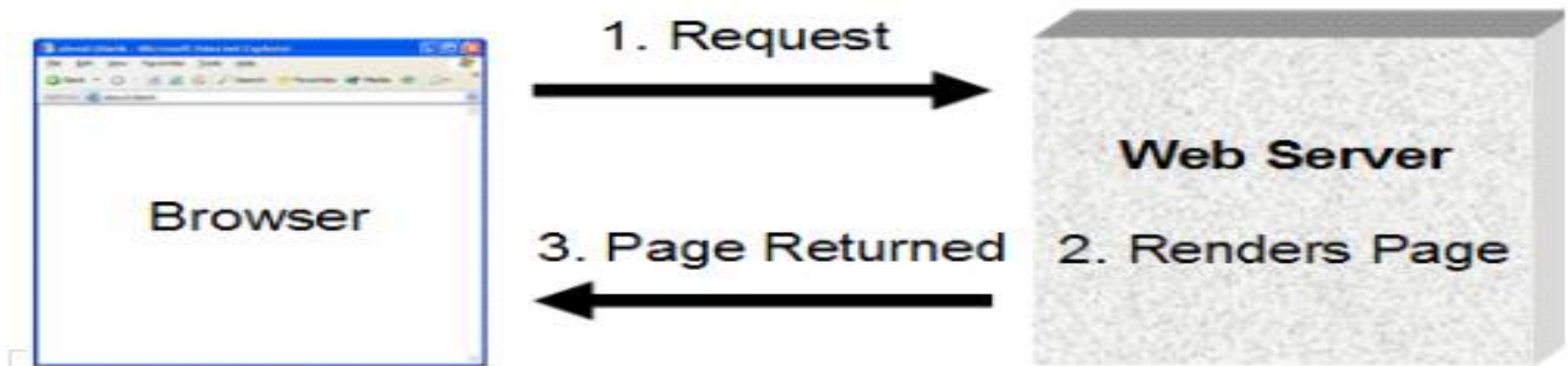


Figure 1. 일반적인 웹 페이지 렌더링

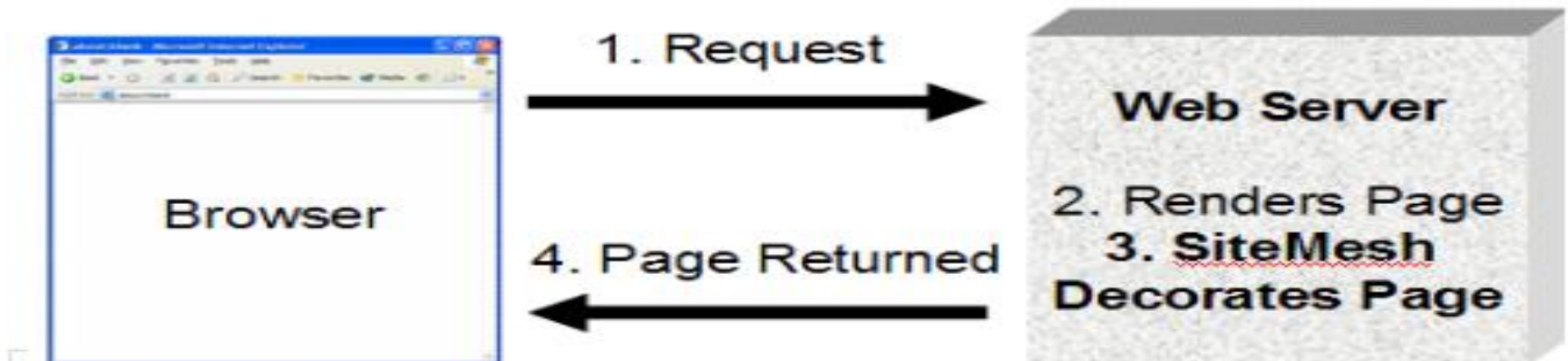


Figure 2. SiteMesh 페이지 렌더링

SiteMesh 태그 라이브러리

장식자 태그, 페이지 태그	장식자 페이지를 생성할 때 사용되는 태그
<decorator:head />	장식될 페이지의 <head>태그의 내용을 삽입
<decorator:body />	장식될 페이지의 <body>태그의 내용을 삽입
<decorator:title />	장식될 페이지의 <title>태그의 내용을 삽입
<decorator:getProperty />	장식이 완료된 HTML 페이지의 <body> 태그 내에 이벤트 핸들러를 생성하기 위해 사용
<decorator:usePage />	장식자 페이지에서 장식될 페이지의 페이지 객체를 얻을 수 있게 함
페이지 태그	장식자 페이지 내에서 다른 장식자를 포함할 때 사용
<page:applyDecorator />	현재 장식자 페이지 내에 장식될 페이지와 장식자를 지정하여 삽입한다.
<page:param>	<page:applyDecorator /> 사용시 해당 장식자에게 파라미터를 전달하기 위해 사용

SiteMesh설치

- 다운로드 : <http://wiki.sitemesh.org/> 또는 <http://www.opensymphony.com/> 에서 jar파일을 다운로드 받은 후에 WEB-INF/lib에 복사
- 구현기초
 - web.xml파일에 SiteMeshFilter설정 추가
 - SiteMesh설정파일 작성
 - 데코레이터 설정파일 작성
 - 데코레이터 JSP작성

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xmlns="http://java.sun.com/xml/ns/javaee"  
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee  
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"  
id="chap19" version="3.0">
```

```
<filter>
```

```
<filter-name>sitemesh</filter-name>
```

```
<filter-class>com.opensymphony.sitemesh.webapp.SiteMeshFilter</filter-class>
```

```
</filter>
```

```
<filter-mapping>
```

```
<filter-name>sitemesh</filter-name>
```

```
<url-pattern>/*</url-pattern>
```

```
</filter-mapping>
```

```
</web-app>
```

sitemesh.xml

```
<sitemesh>
```

```
<property name="decorators-file" value="/WEB-INF/decorators.xml" />
```

```
<excludes file="${decorators-file}" />
```

```
<page-parsers>
```

```
<parser content-type="text/html"
```

```
class="com.opensymphony.module.sitemesh.parser.HTMLPageParser" />
```

```
</page-parsers>
```

```
<decorator-mappers>
```

```
<mapper
```

```
class="com.opensymphony.module.sitemesh.mapper.ConfigDecoratorMapper">
```

```
<param name="config" value="${decorators-file}" />
```

```
</mapper>
```

```
</decorator-mappers>
```

```
</sitemesh>
```

ConfigDecoratorMapper파일을 이용하여 데코레이터 설정

excludes태그는 데코레이터를 적용하지 않을 경로 정보를 담고있는 파일 지정

decorators.xml

```
<?xml version="1.0" encoding="euc-kr" ?>
<decorators defaultdir="/decorator">
    <decorator name="main" page="hello_decorator.jsp">
        <pattern>/hello.jsp</pattern>
        <pattern>/hello/*</pattern>
    </decorator>
    <decorator name="news" page="/decorator2/news_decorator.jsp">
        <pattern>/news/*</pattern>
    </decorator>
    <excludes>
        <pattern>/hello/exclude.jsp</pattern>
        <pattern>/news/exclude.jsp</pattern>
    </excludes>
</decorators>
```

패턴 태그는 데코레이터가 적용될 적용될 경로 패턴을 지정할 때 사용
excludes태그는 데코레이터 경로를 제외할 대상지정

decorator/ hello_decorator.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces="true" %>
<%@ taglib prefix="decorator"
uri="http://www.opensymphony.com/sitemesh/decorator" %>
<html lang="ko">
<head>
<title>19장 <decorator:title /> </title> // 데코레이터가 적용될 응답화면 타이틀
<style type="text/css">
#footer {color:red}
</style>
<decorator:head /> // 데코레이터가 적용될 응답화면 head내용 title은 제외
</head>
<body>
상단 공통 메뉴
<hr/>
<decorator:body /> // 데코레이터가 적용될 응답화면의 body태그 내용
<hr/>
<div id="footer">하단 내용</div>
</body>
</html>
```

decorator2/news_decorator.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces="true" %>
<%@ taglib prefix="decorator"
uri="http://www.opensymphony.com/sitemesh/decorator" %>
<html lang="ko">
<head>
<title>19장 <decorator:title /> </title>
<style type="text/css">
#footer {color:red}
</style>
<decorator:head />
</head>
<body>
뉴스 홈 | 스포츠 | 경제 | 연애
<hr/>
<decorator:body />
<hr/>
<div id="footer">뉴스 푸터</div>
</body>
</html>
```

hello/exclude.jsp

```
<%@ page contentType= "text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces= "true" %>
<html lang= "ko">
<head>
<title>헬로우 제외</title>
</head>
<body>
<div class= "content">헬로우 제외 페이지!</div>
</body>
</html>
```

hello/main.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces="true" %>
<html lang="ko">
<head>
<title>헬로우 메인</title>
</head>
<body>
<div class="content">헬로우 메인 페이지!</div>
</body>
</html>
```

news/exclude.jsp

```
<%@ page contentType= "text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces= "true" %>
<html lang= "ko">
<head>
<title>뉴스 제외</title>
</head>
<body>
<div class= "content">뉴스 제외 페이지!</div>
</body>
</html>
```

news/main.jsp

```
<%@ page contentType= "text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces= "true" %>
<html lang= "ko">
<head>
<title>뉴스 메인</title>
</head>
<body>
<div class= "content">뉴스 메인 페이지!</div>
</body>
</html>
```

hello.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%@ page trimDirectiveWhitespaces="true" %>
<html lang="ko">
<head>
<title>안녕하세요</title>
<style type="text/css">
.content {font-size: 12pt}
</style>
</head>
<body>
<div class="content">안녕하세요!</div>
</body>
</html>
```


ORM 프레임워크인 iBATIS

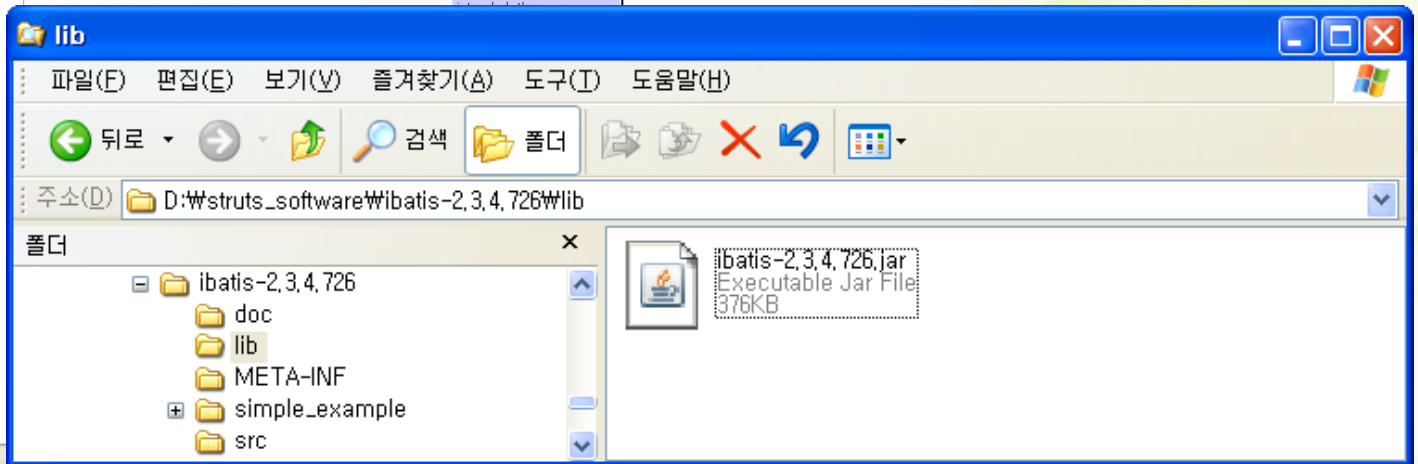
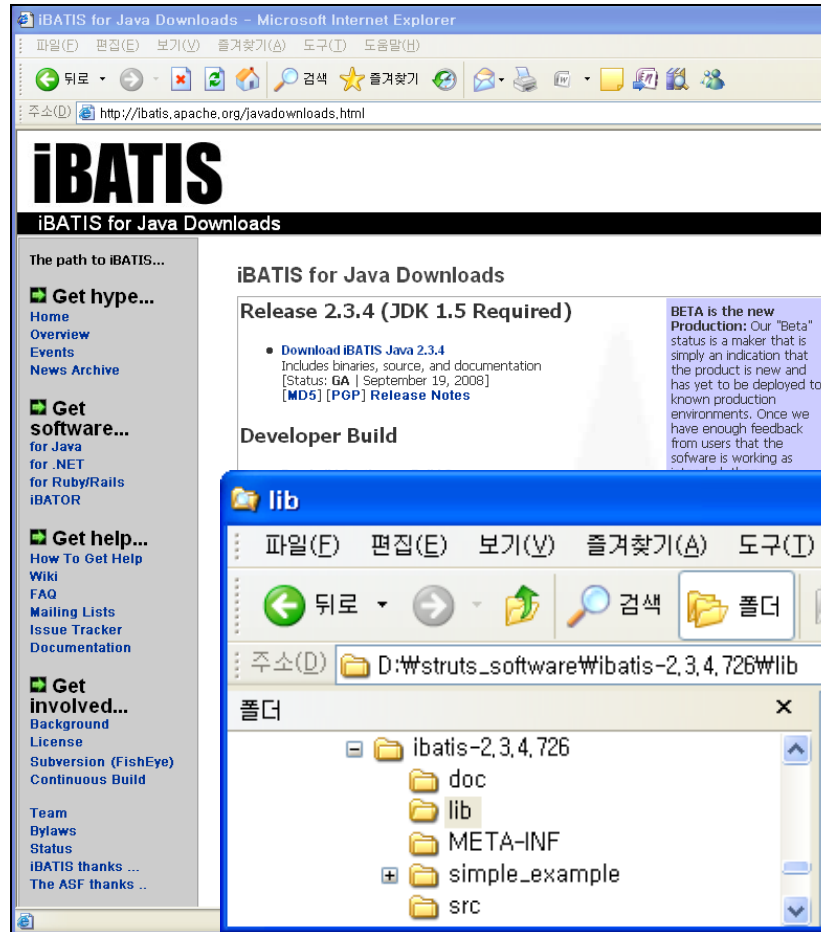
- ORM(Object Relational Mapping) 프레임워크는 데이터베이스와 객체와의 관계를 맵핑시켜 퍼시스턴스 로직처리를 도와주는 프레임워크이다.
- XML 파일에 맵핑 정보를 기술하여 데이터베이스의 테이블과 자바 객체를 맵핑하여 데이터베이스에 생성, 조회, 수정, 삭제(CRUD)작업을 도와주는 역할을 한다.
- ORM 프레임워크는 퍼시스턴스 층에 생산성을 높여 주기 위한 프레임워크로 JDBC를 사용할 때보다 약 60% 정도의 코드만으로 프로그램 작성이 가능해 진다.

작업 절차

- JDBC 드라이버 설치
- DBCP 사용을 위한 jar 파일 설치하기
- iBATIS 설치하기
- iBATIS 사용한 JDBC 프로젝트 작성하기
- 데이터베이스 연결 정보를 담는 프로퍼티 파일 작성하기
- SQLMaps 설정파일인 SqlMapConfig.xml 파일 작성하기

iBATIS 다운받아 설치하기

- <http://ibatis.apache.org/>
- 현재는 mybatis로 변경 됨



데이터베이스 연결 정보를 담은 프로퍼티 파일 작성

- 액션 클래스에서 데이터베이스를 연결하고 iBATIS를 사용하도록 하기 위해서 몇 가지 파일들을 생성해서 설정을 해주어야 한다.
- 먼저 데이터베이스 연결 정보를 담은 dbconnect.properties을 작성해 보자.

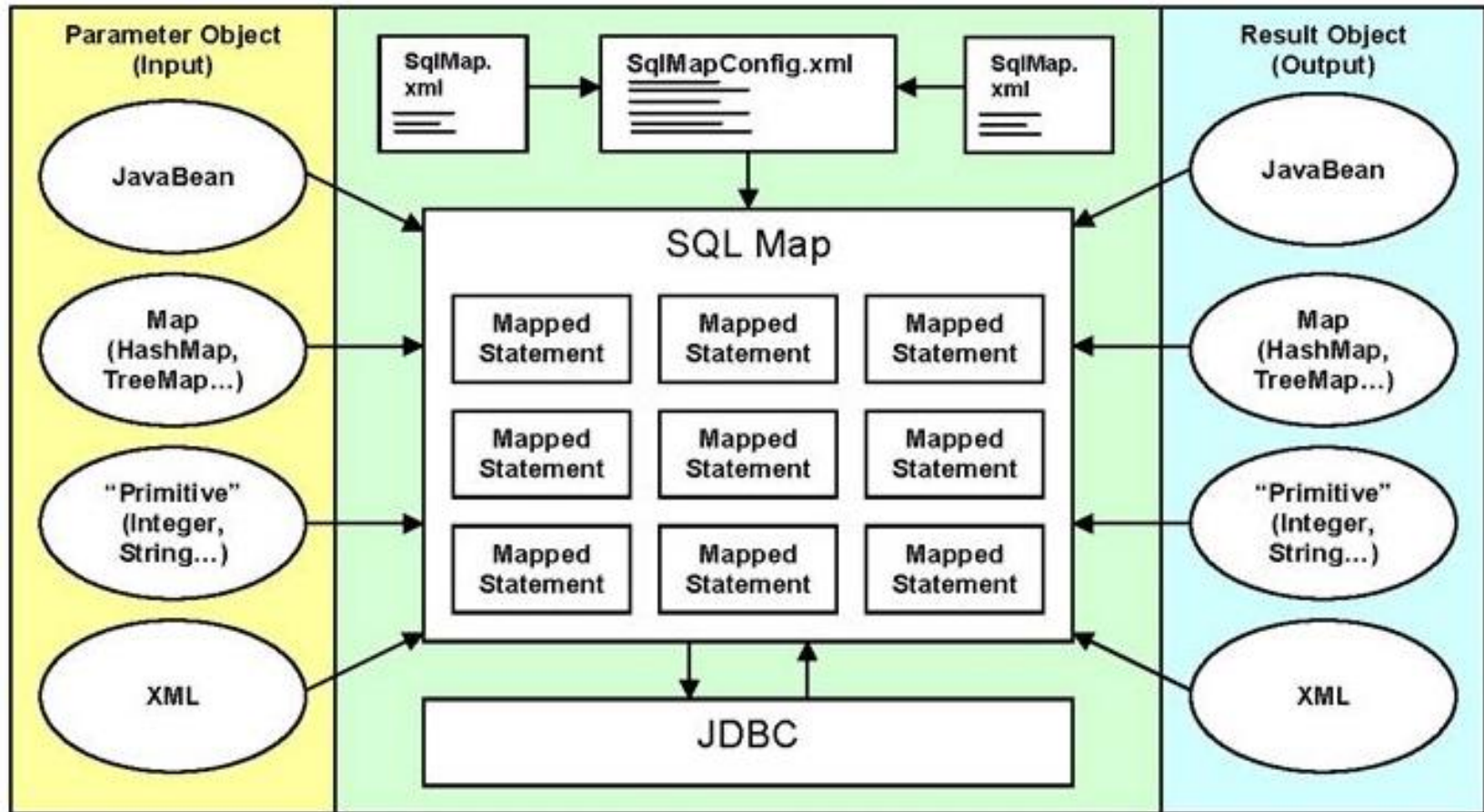
mysql

```
driver=com.mysql.jdbc.Driver  
url=jdbc:mysql://localhost:3306/test  
username=root  
password=mysql
```

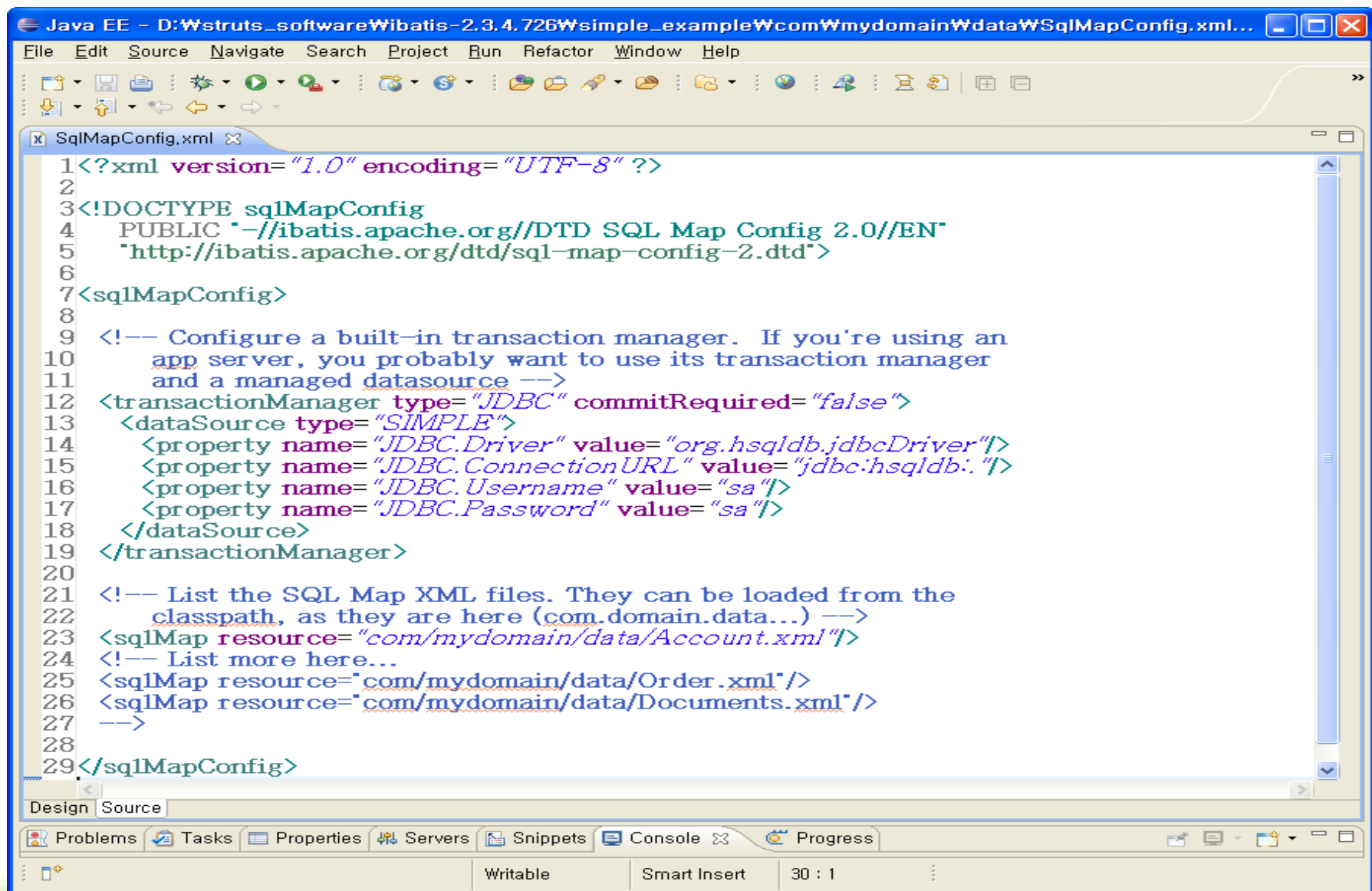
Oracle

- driver=oracle.jdbc.driver.OracleDriver
- url=jdbc:oracle:thin:@localhost:1521:orcl
- username=scott
- password=tiger

SQLMaps 설정파일 : SqlMapConfig.xml

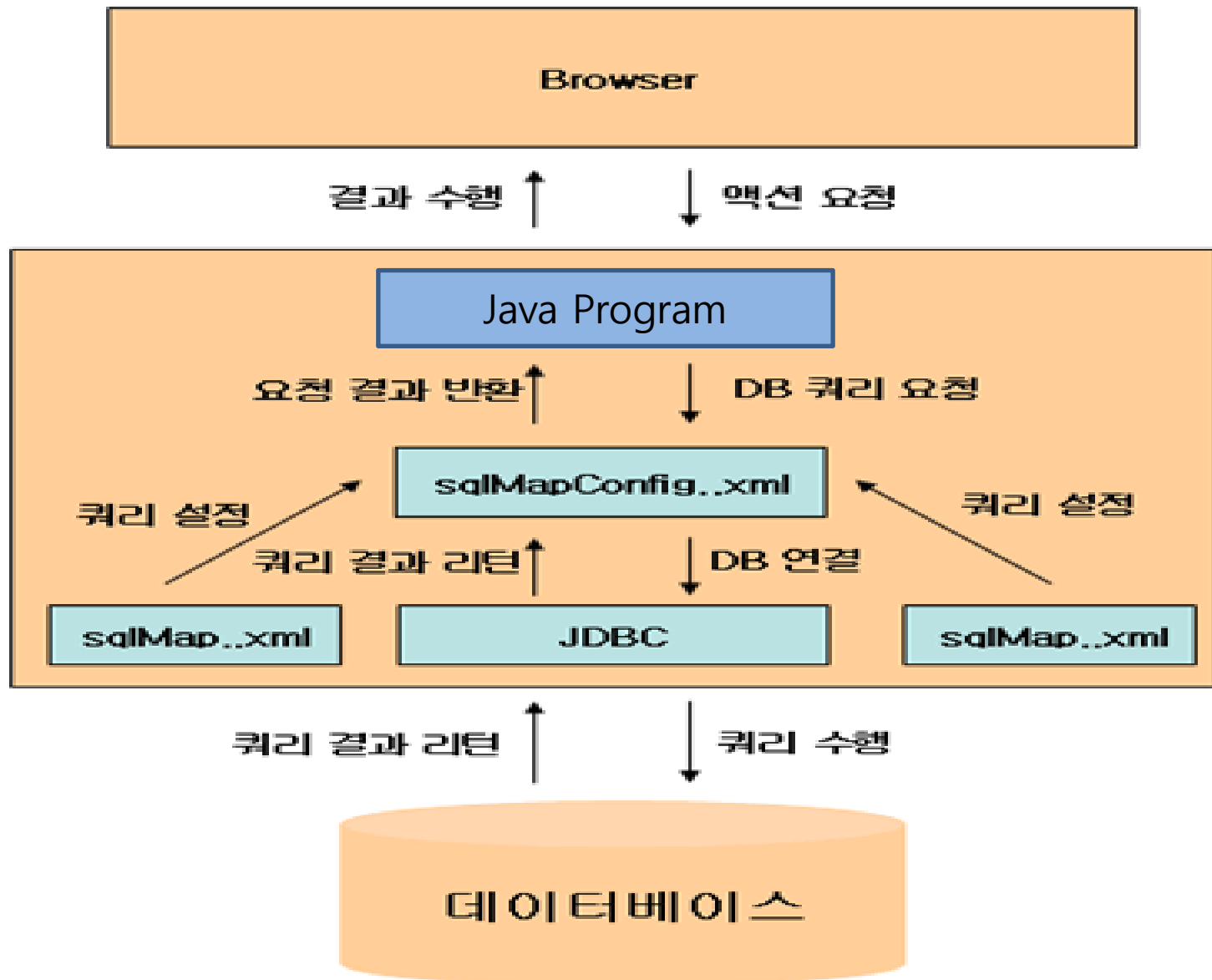


SQLMaps 설정파일 : SqlMapConfig.xml



```
1<?xml version="1.0" encoding="UTF-8" ?>
2
3<!DOCTYPE sqlMapConfig
4  PUBLIC "-//ibatis.apache.org//DTD SQL Map Config 2.0//EN"
5  "http://ibatis.apache.org/dtd/sql-map-config-2.dtd">
6
7<sqlMapConfig>
8
9  <!-- Configure a built-in transaction manager. If you're using an
10   app server, you probably want to use its transaction manager
11   and a managed datasource -->
12  <transactionManager type="JDBC" commitRequired="false">
13    <dataSource type="SIMPLE">
14      <property name="JDBC.Driver" value="org.hsqldb.jdbcDriver"/>
15      <property name="JDBC.ConnectionURL" value="jdbc:hsqldb:."/>
16      <property name="JDBC.Username" value="sa"/>
17      <property name="JDBC.Password" value="sa"/>
18    </dataSource>
19  </transactionManager>
20
21  <!-- List the SQL Map XML files. They can be loaded from the
22   classpath, as they are here (com.domain.data...) -->
23  <sqlMap resource="com/mydomain/data/Account.xml"/>
24  <!-- List more here...
25  <sqlMap resource="com/mydomain/data/Order.xml"/>
26  <sqlMap resource="com/mydomain/data/Documents.xml"/>
27  -->
28
29</sqlMapConfig>
```

iBatis가 적용한 Spring 웹 애플리케이션의 흐름도

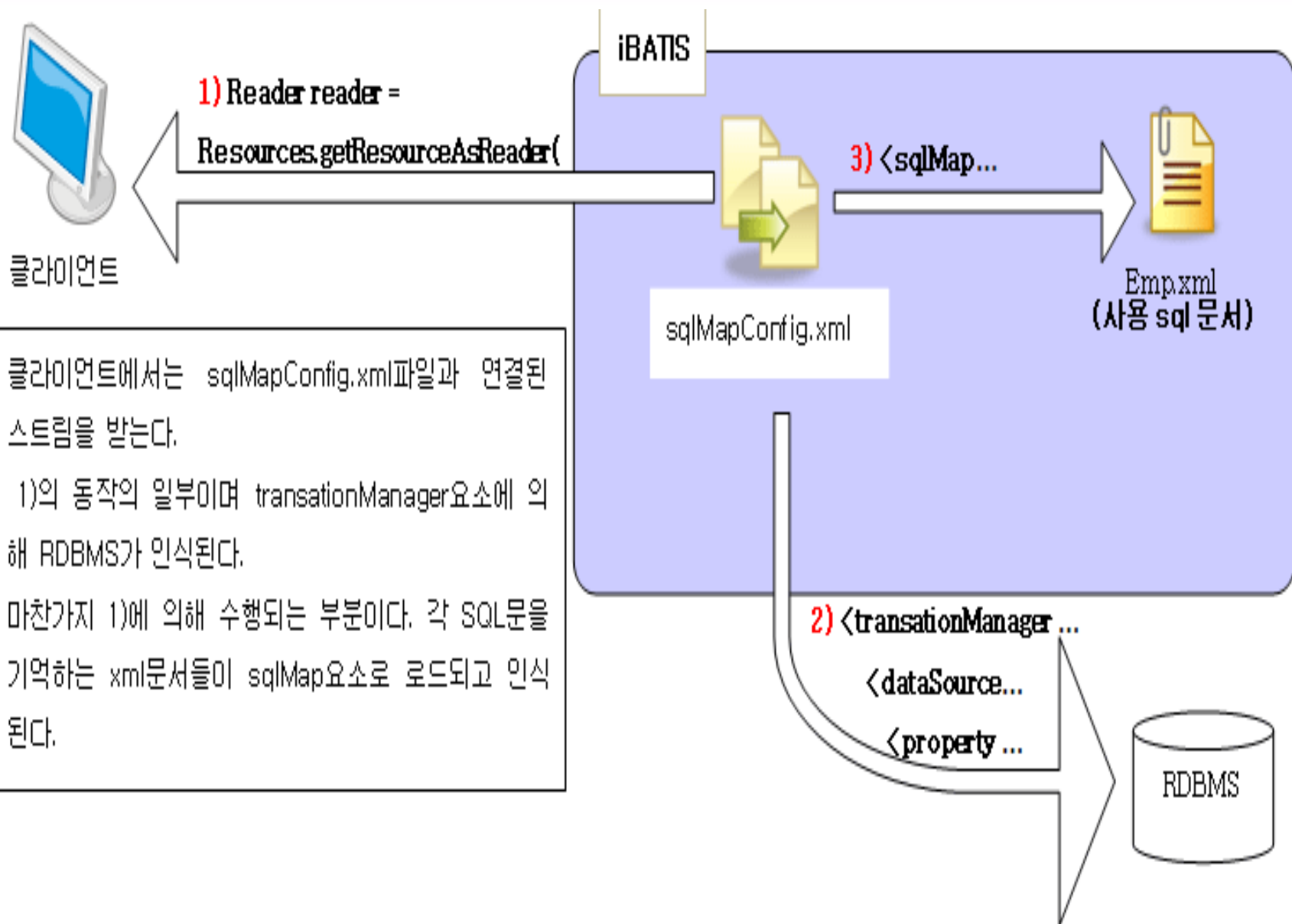


SQL Map API

iBATIS란

- ❖클래스를 테이블로 필드를 컬럼으로 직접 매핑하지 않고 SQL 구문과 파라미터와 결과를 클래스 클래스에 매핑
- ❖쿼리를 담은 내용의 XML과 그 쿼리를 실행시키는 자바 코드로 유연하게 클래스와 테이블을 매핑
- ❖클래스의 프로퍼티와 데이터베이스의 테이블간의 파라미터와 결과값을 매핑하는 책임을 맡고 있기에 SQL 매퍼라고 부름

iBATIS의 동작원리



iBatis를 이용한 jsp간단한 예제

```
create table member3 (  
    id varchar2(10) primary key,  
    password varchar2(10));
```

Member.java

```
package model;  
public class Member {  
    private String id;        private String password;  
    public Member(){}  
    public Member(String id,String pass){  
        this.id = id;        this.password=pass;  
    }  
    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; }  
}
```

dbconnect.properties

driver=oracle.jdbc.driver.OracleDriver
url=jdbc:oracle:thin:@127.0.0.1:1521:xe
username=scott
password=tiger

SqlMapConfig.xml

```
<?xml version="1.0" encoding="utf-8" ?>
<!DOCTYPE sqlMapConfig PUBLIC "-//iBATIS.com//DTD SQL Map Config 2.0//EN"
"http://www.ibatis.com/dtd/sql-map-config-2.dtd">
<sqlMapConfig>
  <properties resource="dbconnect.properties"/>
  <transactionManager type="JDBC">
    <dataSource type="SIMPLE">
      <property name="JDBC.Driver" value="${driver}"/>
      <property name="JDBC.ConnectionURL" value="${url}"/>
      <property name="JDBC.Username" value="${username}"/>
      <property name="JDBC.Password" value="${password}"/>
      <property name="JDBC.DefaultAutoCommit" value="true"/>
    </dataSource>
  </transactionManager>
  <sqlMap resource="member.xml"/>
</sqlMapConfig>
```

member.xml

```
<?xml version="1.0" encoding="utf-8"?>
<!DOCTYPE sqlMap PUBLIC "-//ibatis.apache.org//DTD SQL Map 2.0//EN"
"http://ibatis.apache.org/dtd/sql-map-2.dtd">
<sqlMap namespace="member">
<typeAlias alias="MEMBER" type="model.Member"/>
<resultMap id="memberResult" class="MEMBER">
    <result property="id" column="id"/>
    <result property="password" column="password"/>
</resultMap>
<select id="memberOne" resultMap="memberResult" parameterClass="String">
    select id,password from member3 where id = #id#
</select>
<select id="listMember" resultMap="memberResult">
    select * from member3
</select>
<insert id="addMember" parameterClass="MEMBER">
    insert into member3 (id,password) values(#id#,#password#)
</insert>
<update id="updateMember" parameterClass="MEMBER">
    update member3 set password=#password# where id=#id#
</update>
<delete id="delMember" parameterClass="String">
    delete from member3 where id=#id#
</delete>
</sqlMap>
```

SqlMapLocator.java

```
package util;
import java.io.IOException;
import java.io.Reader;
import com.ibatis.common.resources.Resources;
import com.ibatis.sqlmap.client.SqlMapClient;
import com.ibatis.sqlmap.client.SqlMapClientBuilder;
public class SqlMapLocator {
    public static SqlMapClient getMapper(){
        SqlMapClient sqlMapper;
        try{ Reader reader = Resources.getResourceAsReader("SqlMapConfig.xml");
            sqlMapper = SqlMapClientBuilder.buildSqlMapClient(reader);
            reader.close();
        }catch(IOException e){
            throw new RuntimeException( "Error while building the SqlMapClient
                instance."+e,e);
        }
        return sqlMapper;
    }
}
```

MemberDAO.java

```
package model; import util.SqlMapLocator; import java.util.*; import java.sql.*;
public class MemberDAO {
    public Member getMemberOne(String id){
        Member member = null;
        try { member =
            (Member)SqlMapLocator.getMapper().queryForObject("memberOne",id);
        }catch(SQLException e){      System.out.println(e.getMessage());  }
        return member;      }
    public void addMember(Member man){
        try{ SqlMapLocator.getMapper().update("addMember",man);
        }catch(SQLException e){      System.out.println(e.getMessage());  }      }
    public List<Member> list(){
        List<Member> list = null;
        try{ list = SqlMapLocator.getMapper().queryForList("listMember");
        }catch(SQLException e){      System.out.println(e.getMessage());  }
        return list;      }
    public void updateMember(Member man){
        try{ SqlMapLocator.getMapper().update("updateMember",man);
        }catch(SQLException e){      System.out.println(e.getMessage());  }
    }
    public void delMember(String id){
        try{ SqlMapLocator.getMapper().delete("delMember",id);
        }catch(SQLException e){      System.out.println(e.getMessage());  }
    }
}
```

index.html

```
<script> location.href="loginForm.jsp" </script>
```

joinForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
```

<title>회원관리 시스템 회원가입 페이지</title>

```
<form name= "joinform" action="joinProc.jsp" method="post">
```

|

<td colspan= "2" align=center>

호원가입 페이지

0 0 □ :	<input name="id" type="text"/>
----------	--------------------------------

| 비밀번호 : | |
|

<td colspan= "2" align=center>

`회원가입 `

`다시작성`

</table>

</form>

joinProc.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ page import="model.*" %>
<jsp:useBean id="memberDAO" class="model.MemberDAO"/>

<%
    request.setCharacterEncoding("utf-8");
    String id=request.getParameter("id");
    String pass=request.getParameter("pass");

    Member member = new Member(id,pass);
    memberDAO.addMember(member);
%>
<script>
alert('회원이입이 완료되었습니다.');
```

location.href='loginForm.jsp';

```
</script>
```

loginForm.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
```

<title>회원관리 시스템 로그인 페이지</title>

```
<form name= "loginform" action= "loginProc.jsp" method= "post">
```

|
 <td colspan="2" align="center"> | |

로그인 페이지

</td>

0 0 □ :	<input name="id" type="text"/>
----------	--------------------------------

| 비밀번호 : | |
|
 <td colspan="2" align="center"> | |

`로그인 `

`회원가입`

</td>

</form>

loginProc.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ page import="model.*" %>
<jsp:useBean id="memberDAO" class="model.MemberDAO"/>

<%
    String id=request.getParameter("id");
    String pass=request.getParameter("pass");
    Member member = memberDAO.getMemberOne(id);

    if(member != null){
        if (pass.equals(member.getPassword())){
            session.setAttribute("id",id);
            out.println("<script>");
            out.println("location.href='main.jsp'");
            out.println("</script>");
        }
    }
    out.println("<script>");
    out.println("alert('로그인 정보가 다릅니다.')");
    out.println("location.href='loginForm.jsp'");
    out.println("</script>");
%>
```

main.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%
String id=null;

if (session.getAttribute("id")!=null){
    id=(String)session.getAttribute("id");
}else{
    out.println("<script>");
    out.println("location.href='loginForm.jsp'");
    out.println("</script>");
}
%>
<title>회원관리 시스템 메인 페이지</title>
<h3><%=id %> 로 로그인하셨습니다.</h3>
<%if(id.equals("admin")){%>
<a href="member_list.jsp">관리자모드 접속(회원 목록 보기)</a>
<%}else{%>
<a href="loginForm.jsp">로그인 화면</a>
<%} %>
```

member_del.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<jsp:useBean id="memberDAO" class="model.MemberDAO"/>
```

```
<%
```

```
String id=null;
if ((session.getAttribute("id")==null) ||
    (!((String)session.getAttribute("id")).equals("admin")) {
    out.println("<script>");
    out.println("location.href='loginForm.jsp'");
    out.println("</script>");
}
```

```
String delete_id=request.getParameter("id");
```

```
memberDAO.delMember(delete_id);
```

```
%>
```

```
<script>
```

```
location.href="member_list.jsp";
```

```
</script>
```

member_info.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%> <%@ page import="model.*" %>
<jsp:useBean id="memberDAO" class="model.MemberDAO"/>
<%
    String id=null;
    if ((session.getAttribute("id")==null)
        ||(!((String)session.getAttribute("id")).equals("admin"))){
        out.println("<script>location.href='loginForm.jsp'</script>");
    }
    String info_id=request.getParameter("id");
    Member member = memberDAO.getMemberOne(info_id);
%>
<center>
<form name="infoform" action="updateProc.jsp" method="post">
    <table border=1 width=300>
        <input type="hidden" name="id" value="<%= member.getId() %>"/>
        <tr align=center><td>아이디 : </td><td><%= member.getId() %></td></tr>
        <tr align=center><td>비밀번호 : </td>
            <td><input type="text" value="<%=member.getPassword() %>"
                name="pass"></td></tr>
        <tr align=center><td colspan=2><a href="member_list.jsp">리스트</a>&nbsp;  
            <a href="javascript:infoform.submit()">수정</a></td></tr></table>
    </form>
</center>
```

member list.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<jsp:useBean id="memberDAO" class="model.MemberDAO"/>
<%@ page import="java.util.*,model.*" %>
<% String id=null;
    if ((session.getAttribute("id")==null) ||
        (!((String)session.getAttribute("id")).equals("admin")))) {
        out.println("<script>location.href='loginForm.jsp'</script>");
    }
    List<Member> list = memberDAO.list();
%>
<title>회원관리 시스템 관리자모드(회원 목록 보기)</title> <center>
<table border=1 width=300>
    <tr align=center> <td colspan=2>회원 목록</td> </tr>
    <% for(Member man:list){ %>
        <tr align=center><td><a href="member_info.jsp?id=<%=man.getId() %>"
            <%=man.getId() %></a>
            </td>
            <td><a href="member_del.jsp?id=<%=man.getId() %>">삭제</a></td>
        </tr>
    <%} %>
</table>
<a href="loginForm.jsp">로그인 화면</a>
</center>
```


updateProc.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
pageEncoding="UTF-8"%>
<%@ page import="model.*" %>
<jsp:useBean id="memberDAO" class="model.MemberDAO"/>
<%
    String id=null;
    if ((session.getAttribute("id")==null) ||
        (!((String)session.getAttribute("id")).equals("admin")))) {
        out.println("<script>");
        out.println("location.href='loginForm.jsp'");
        out.println("</script>");
    }
    String up_id=request.getParameter("id");
    String up_pass=request.getParameter("pass");

    Member member = new Member(up_id,up_pass);
    memberDAO.updateMember(member);
%>
<script>
alert('수정이 완료되었습니다.');
```

location.href='loginForm.jsp';

```
</script>
```

Mybatis

configuration.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE configuration PUBLIC "-//mybatis.org//DTD Config 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-config.dtd">
<configuration>
<properties resource="dbconnect.properties"/>
<typeAliases>
    <typeAlias alias="member" type="model.Member" />
</typeAliases>
<environments default="development">
    <environment id="development">
        <transactionManager type="JDBC" />
        <dataSource type="POOLED">
            <property name="driver" value="${driver}" />
            <property name="url" value="${url}" />
            <property name="username" value="${username}" />
            <property name="password" value="${password}" />
        </dataSource>
    </environment>
</environments>
<mappers>
    <mapper resource="member.xml" />
</mappers>
</configuration>
```

member.xml

```
<?xml version="1.0" encoding="UTF-8" ?>
<!DOCTYPE mapper PUBLIC "-//mybatis.org//DTD Mapper 3.0//EN"
"http://mybatis.org/dtd/mybatis-3-mapper.dtd">
<mapper namespace="member">
  <resultMap type="member" id="ResultMap">
    <result property="id" column="id"/>
    <result property="password" column="password"/>
  </resultMap>
  <select id="memberOne" resultMap="ResultMap" parameterType="String">
    select id,password from member3 where id = #{id}
  </select>
  <select id="listMember" resultMap="ResultMap">
    select * from member3
  </select>
  <insert id="addMember" parameterType="member">
    insert into member3 (id,password) values(#{id},#{password})
  </insert>
  <update id="updateMember" parameterType="member">
    update member3 set password=#{password} where id=#{id}
  </update>
  <delete id="delMember" parameterType="String">
    delete from member3 where id=#{id}
  </delete>
</mapper>
```

MemberDAO.java

```
package model;
import java.io.*;      import java.util.List;
import org.apache.ibatis.io.Resources;
import org.apache.ibatis.session.SqlSession;
import org.apache.ibatis.session.SqlSessionFactory;
import org.apache.ibatis.session.SqlSessionFactoryBuilder;
public class MemberDAO {
    private SqlSession getSession() throws IOException {
        String src = "configuration.xml";
        Reader reader = Resources.getResourceAsReader(src);
        SqlSessionFactory sqlMapper = new SqlSessionFactoryBuilder().build(reader);
        SqlSession session = sqlMapper.openSession(true);
        return session;
    }
    public Member getMemberOne(String id){
        SqlSession session = null;
        Member member = null;
        try{
            session = getSession();
            member = (Member)session.selectOne("memberOne",id);
        }catch(Exception e){          System.out.println(e.getMessage());  }
        return member;
    }
}
```

```
public void addMember(Member man){
    SqlSession session = null;
    try{        session = getSession();
                session.insert("addMember",man);
    }catch(Exception e){System.out.println(e.getMessage());}
}
public List<Member> list(){
    List<Member> list = null;    SqlSession session = null;
    try{        session = getSession();
                list = session.selectList("listMember");
    }catch(Exception e){System.out.println(e.getMessage());}
    return list;
}
public void updateMember(Member man){
    SqlSession session = null;
    try{        session = getSession();
                session.update("updateMember",man);
    }catch(Exception e){System.out.println(e.getMessage());}
}
public void delMember(String id){
    SqlSession session = null;
    try{        session = getSession();
                session.delete("delMember",id);
    }catch(Exception e){System.out.println(e.getMessage());}
}
}
```