

18 웹 MVC

1

MVC의 개요

2

MVC의 패턴 구조

3

MVC 패턴 구현 방법

4

[웹 쇼핑몰] 게시판 만들기

학습목표

- MVC의 개념을 이해합니다.
- MVC 패턴 구조를 이해합니다.
- MVC 패턴 구현 방법을 익힙니다.
- 웹 쇼핑몰의 게시판을 만듭니다.

1. MVC의 개요

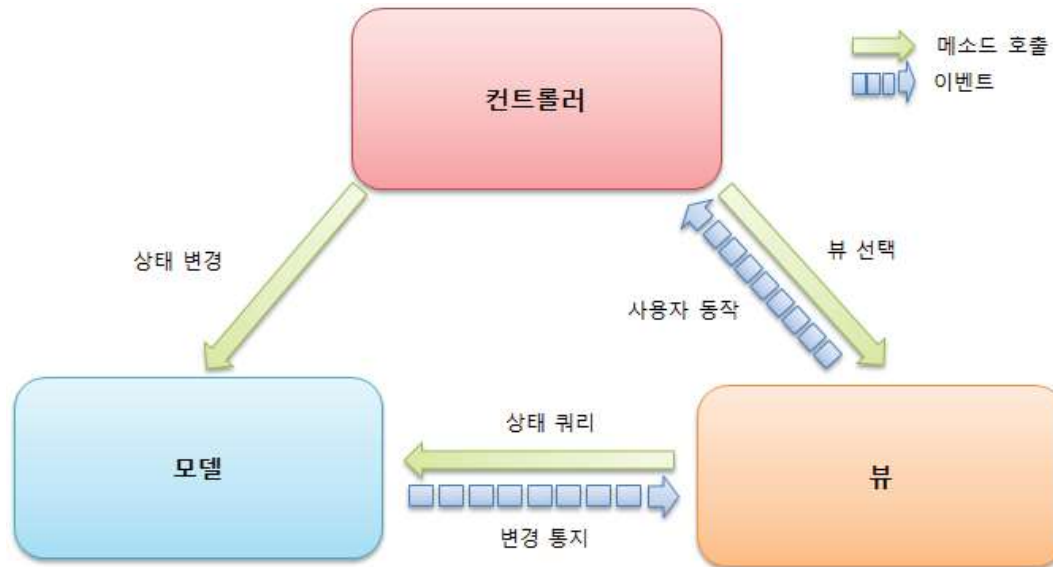
❖ MVC

- Model, View, Controller의 약자로, 웹 애플리케이션을 비즈니스 로직, 프레젠테이션로직, 데이터로 분리하는 디자인 패턴
- 웹 애플리케이션에서는 일반적으로 애플리케이션을 비즈니스 로직, 프레젠테이션, 요청 처리 데이터로 분류
 - 비즈니스 로직은 애플리케이션의 데이터, 즉 고객, 제품, 주문 정보의 조작에 사용
 - 프레젠테이션은 애플리케이션이 사용자에게 어떻게 표시되는지, 즉 위치, 폰트, 크기
 - 요청 처리 데이터는 비즈니스 로직과 프레젠테이션 파트를 함께 묶는 것

1. MVC의 개요

❖ MVC 패턴의 구성 요소

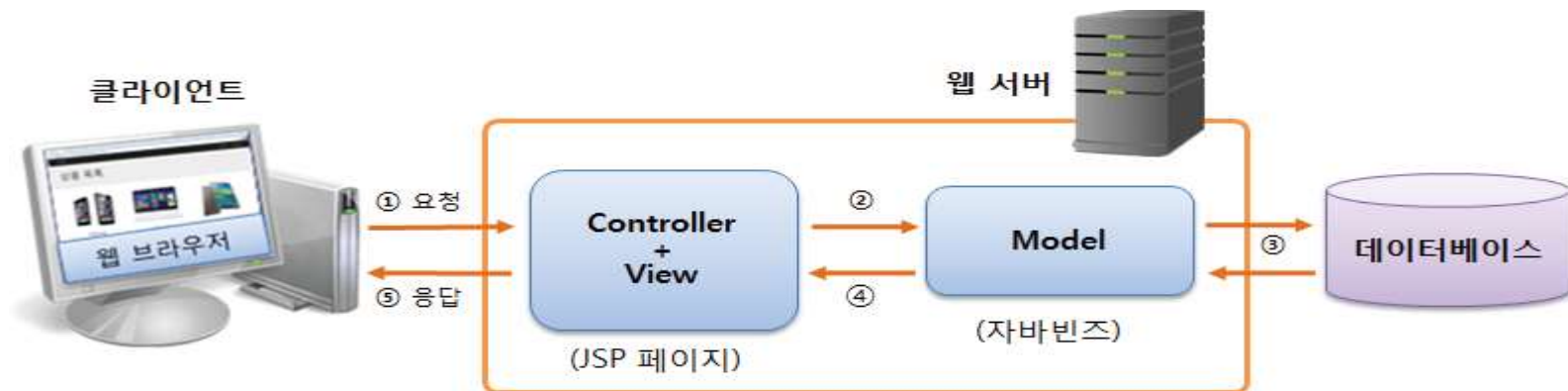
- 모델(model): 애플리케이션의 데이터와 비즈니스 로직을 담는 객체
- 뷰(view): 사용자에게 모델의 정보(데이터)를 보여주는 역할. 비즈니스 로직을 포함하지 않으며, 하나의 모델을 다양한 뷰에서 사용
- 컨트롤러(controller): 모델과 뷰 사이에 어떤 동작이 있을 때 조정하는 역할
웹으로부터 받은 요청에 가장 적합한 모델을 생성하는 것을 처리하는 역할과 사용자에게 응답하는 적절한 뷰를 선택하여 해당 모델을 전달하는 역할



2. MVC 패턴 구조

❖ 2.1 모델 1

- 모델 1은 기존의 JSP로만 구현한 웹 애플리케이션으로, 웹 브라우저의 요청을 JSP 페이지가 받아서 처리하는 구조
- JSP 페이지에 비즈니스 로직을 처리하는 코드와 웹 브라우저에 결과를 출력하는 코드가 섞이는 것
- 모델 1에서는 JSP가 핵심 역할을 수행

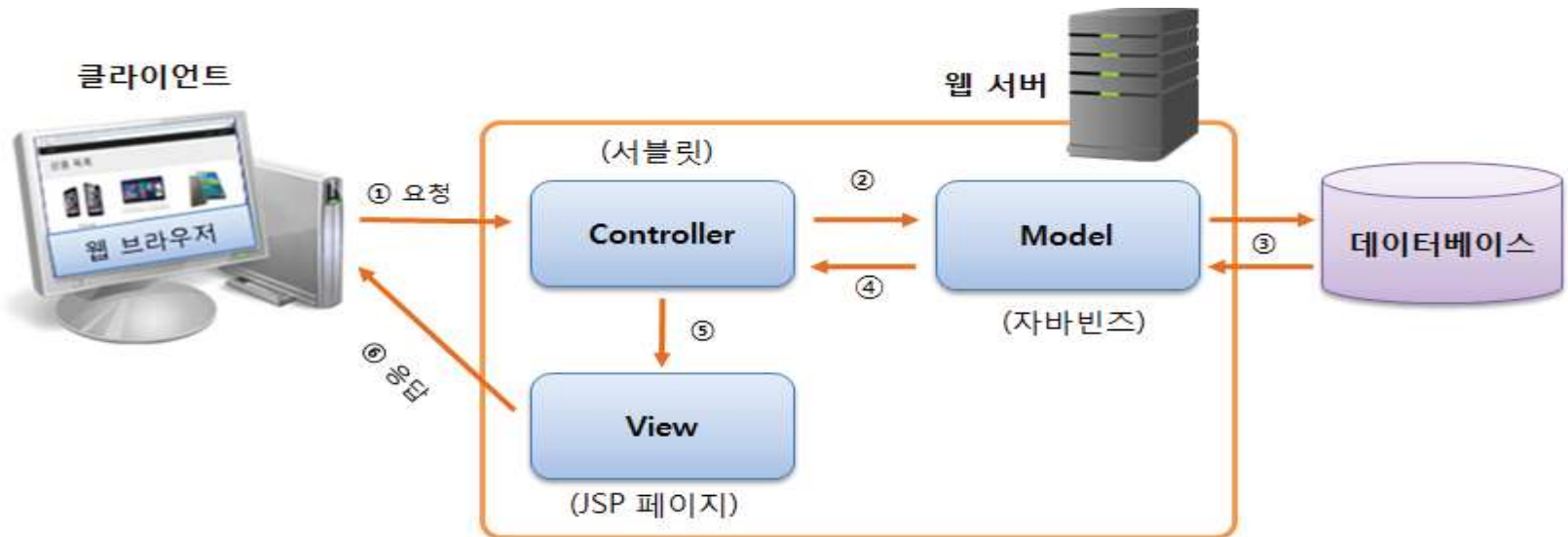


장점	단점
<ul style="list-style-type: none">• 구조가 단순하여 개발자의 수준이 낮아도 쉽게 익힐 수 있어 구현이 용이합니다.• 개발 초기에 복잡한 설정이 필요 없어 빠른 개발이 가능합니다.• 개발 속도가 빠릅니다.	<ul style="list-style-type: none">• 출력을 위한 뷰 코드와 로직 처리를 위한 자바 코드(컨트롤러)가 섞여 있어 분업이 용이하지 않습니다.• 코드가 복잡하여 유지 보수가 어렵습니다.• 자바 코드와 JSP, HTML이 섞이기 때문에 코드가 복잡합니다.

2. MVC 패턴 구조

❖ 2.2 모델 2

- 모델 2는 클라이언트의 요청 처리, 응답 처리, 비즈니스 로직 처리 부분을 모듈화한 구조
- 웹 브라우저의 요청이 들어오면 모든 처리를 JSP 페이지가 담당하는 모델 1과 달리, 요청에 대한 로직을 처리할 자바빈즈나 자바 클래스인 모델, 요청 결과를 출력하는 JSP 페이지인 뷰, 모든 흐름을 제어는 서블릿인 컨트롤러로 나뉘어 웹 브라우저가 요청한 작업을 처리
- 모델 2에서는 서블릿이 중요한 역할



2. MVC 패턴 구조

❖ 2.2 모델 2

- ① 웹 브라우저가 웹 서버에 웹 애플리케이션 실행을 요청하면 웹 서버는 요청을 처리할 수 있는 컨트롤러(서블릿)를 찾아서 요청을 전달
- ② 컨트롤러(서블릿)는 모델 자바 객체의 메소드를 호출
- ③ 데이터를 가공하여 값 객체를 생성하거나 JDBC를 사용하여 데이터베이스와의 인터렉션을 통해 값 객체 생성
- ④ 업무 수행을 마친 결과 값을 컨트롤러에 반환
- ⑤ 컨트롤러는 모델로부터 받은 결과 값을 뷰에 전달
- ⑥ JSP는 전달받은 값을 참조해서 출력할 결과를 만들어 웹 서버에 전달하고, 웹 브라우저는 웹서버로부터 결과 값을 받아 화면에 출력

장점	단점
<ul style="list-style-type: none">• 출력을 위한 뷰 코드와 로직 처리를 위한 자바 코드를 분리하기 때문에 모델 1보다 코드가 간결합니다.• 뷰와 로직 처리에 대한 분업이 용이합니다.• 기능에 따라 분리되어 있기 때문에 유지 보수가 용이합니다.• 확장이 용이합니다.	<ul style="list-style-type: none">• 구조가 복잡하여 습득하기 어렵고 작업량이 많습니다.• 개발 초기에 설정이 필요한 부분이 모델 1보다 많기 때문에 실질적인 작업을 시작하기까지 시간이 걸립니다.• 코드가 분리됨으로써 관리해야 할 파일이 많아집니다.

3. MVC 패턴 구현 방법

❖ web.xml 파일에 서블릿 구성하기

- <servlet> 요소로 서블릿 클래스 등록하기
 - <servlet>은 웹 애플리케이션에서 사용될 기본 서블릿 객체와 매개변수를 설정하는 요소

```
<servlet>
  <servlet-name>서블릿 이름</servlet-name>
  <servlet-class>서블릿 클래스(패키지 이름.클래스)</servlet-class>
  [<init-param>
    <param-name>매개변수 이름</param-name>
    <param-value>매개변수 값</param-value>
  </init-param>
</servlet>
```


3. MVC 패턴 구현 방법

[<servlet> 요소로 서블릿 클래스를 등록한 예]

```
<web-app>
<servlet>
  <servlet-name>myController</servlet-name>
  <servlet-class>ch18.com.controller.MyController</servlet-class>

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

3. MVC 패턴 구현 방법

❖ web.xml 파일에 서블릿 구성하기

- <servlet-mapping> 요소로 요청 URL 패턴 설정하기
 - <servlet-mapping>은 웹 브라우저에서 요청되는 URL과 서블릿 클래스를 매핑하기 위해 URL 패턴을 설정하는 요소

```
<servlet-mapping>
  <servlet-name>서블릿 이름</servlet-name>
  <url-pattern>요청할 URL 패턴</url-pattern>
</servlet-mapping>
```

[<servlet-mapping> 요소로 요청 URL 패턴을 등록한 예]

```
<servlet-mapping>
  <servlet-name>myController</servlet-name>
  <url-pattern>/</url-pattern>
</servlet-mapping>
```

3. MVC 패턴 구현 방법

❖ 컨트롤러 생성하기

■ 서블릿 클래스 생성하기

- 서블릿 클래스는 HttpServlet 클래스를 확장하여 생성
- 생성된 서블릿 클래스는 웹 브라우저에서 전송되는 GET 방식과 POST 방식에 따라 각각 doGet(), doPost() 메소드를 통해 요청 작업을 수행한 후 웹 브라우저에 응답

```
public class 서블릿 이름 extends HttpServlet {  
    @Override  
    public void doGet(HttpServletRequest request, HttpServletResponse  
        response) throws ServletException, IOException {  
        //Get 방식으로 전송되는 요청을 처리  
    }  
    @Override  
    public void doPost(HttpServletRequest request, HttpServletResponse  
        response) throws ServletException, IOException {  
        //Post 방식으로 전송되는 요청을 처리  
    }  
}
```

3. MVC 패턴 구현 방법

❖ 컨트롤러 생성하기

■ 페이지 이동하기

- 서블릿 클래스에서 웹 브라우저로부터 요청된 처리 결과를 보여줄 응답 페이지로 이동
- 이때 현재 뷰 페이지에서 이동할 뷰 페이지에 요청 정보를 그대로 전달하며, 뷰 페이지가 이동해도 처음에 요청된 URL을 계속 유지하기 위해 포워딩 방식을 사용

```
RequestDispatcher dispatcher = request.getRequestDispatcher("JSP 페이지");  
dispatcher.forward(request, response);
```

3. MVC 패턴 구현 방법

[컨트롤러인 서블릿 클래스 MyController 생성 예]

```
public class MyController extends HttpServlet {  
    protected void doGet(HttpServletRequest request,  
                           HttpServletResponse response)  
        throws ServletException, IOException {  
        request.setAttribute("message", "Hello! Java Server Page.");  
        RequestDispatcher rd = request.getRequestDispatcher("view.jsp");  
        rd.forward(request, response);  
    }  
}
```

3. MVC 패턴 구현 방법

❖ 모델 생성하기

- 모델은 웹 애플리케이션의 비즈니스 로직을 포함하는 데이터로 웹 애플리케이션의 상태
- 모델은 데이터베이스에서 데이터를 가져오거나, 웹 애플리케이션에 필요한 서비스를 수행하는 간단한 자바 클래스로 자바빈즈를 의미
- 자바빈즈는 데이터를 담은 멤버 변수인 프로퍼티와 데이터를 가져오거나 저장하는 Getter/Setter() 메소드로 구성

3. MVC 패턴 구현 방법

❖ 뷰 생성하기

- 뷰는 웹 브라우저의 요청을 처리한 결과를 사용자에게 보여주는 JSP 페이지를 의미
- 뷰는 JSP가 제공하는 태그를 사용하여 컨트롤러가 전송한 모델 데이터를 웹 브라우저에 출력

[뷰인 view.jsp 페이지 생성 예]

```
//view.jsp
<%@ page contentType="text/html; charset=utf-8"%>
<%
    String msg = (String)request.getAttribute("message");
    out.println(msg);
%>
```

Hello! java Server Page.

3. MVC 패턴 구현 방법

예제 18-1 MVC를 적용한 로그인 인증하기

JSPBook/WebContent/WEB-INF/web.xml

```
01 <web-app>
02     ... (생략) ...
03     <servlet>
04         <servlet-name>myController</servlet-name>
05         <servlet-class>ch18.com.controller.ControllerServlet</servlet-class>
06     </servlet>
07
08     <servlet-mapping>
09         <servlet-name>myController</servlet-name>
10         <url-pattern>/ch18/ControllerServlet</url-pattern>
11     </servlet-mapping>
12 </web-app>
```


3. MVC 패턴 구현 방법

■ 모델 생성하기:

JSPBook/src/ch18/com/model/LoginBean.java

```
01 package ch18.com.model;
02
03 public class LoginBean {
04     private String id;
05     private String password;
06
07     public String getId() {
08         return id;
09     }
10
11     public void setId(String id) {
12         this.id = id;
13     }
14
15     public String getPassword() {
16         return password;
17     }
18
19     public void setPassword(String password) {
20         this.password = password;
21     }
22
23     public boolean validate() {
24         if (password.equals("admin"))
25             return true;
26         else
27             return false;
28     }
29 }
```

3. MVC 패턴 구현 방법

- 컨트롤러 생성하기:

JSPBook/src/ch18/com/controller/ControllerServlet.java

```
01 package ch18.com.controller;
02
03 import java.io.IOException;
04 import javax.servlet.RequestDispatcher;
05 import javax.servlet.ServletException;
06 import javax.servlet.http.HttpServlet;
07 import javax.servlet.http.HttpServletRequest;
08 import javax.servlet.http.HttpServletResponse;
09
10 import ch18.com.model.LoginBean;
11
12 public class ControllerServlet extends HttpServlet {
13
14     private static final long serialVersionUID = 1L;
15
16     protected void doPost(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
17         response.setContentType("text/html; charset=utf-8");
```

3. MVC 패턴 구현 방법

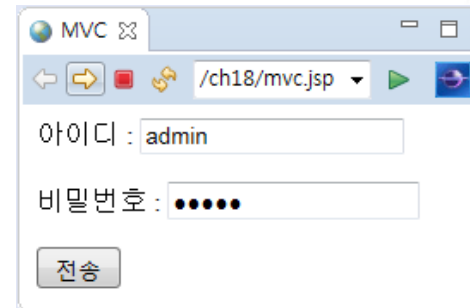
```
18
19     String id = request.getParameter("id");
20     String password = request.getParameter("passwd");
21
22     LoginBean bean = new LoginBean();
23     bean.setId(id);
24     bean.setPassword(password);
25     request.setAttribute("bean", bean);
26
27     boolean status = bean.validate();
28
29     if (status) {
30         RequestDispatcher rd = request.getRequestDispatcher("mvc_
            success.jsp");
31         rd.forward(request, response);
32     } else {
33         RequestDispatcher rd = request.getRequestDispatcher("mvc_error.
            jsp");
34         rd.forward(request, response);
35     }
36 }
37
38 @Override
39 protected void doGet(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
40     doPost(req, resp);
41 }
42 }
```

3. MVC 패턴 구현 방법

■ 뷰 생성하기

JSPBook/WebContent/ch18/mvc.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>MVC</title>
05 </head>
06 <body>
07     <form method="post" action="ControllerServlet">
08         <p> 아이디 : <input type="text" name="id">
09         <p> 비밀번호 : <input type="password" name="passwd">
10         <p> <input type="submit" value="전송">
11     </form>
12 </body>
13 </html>
```

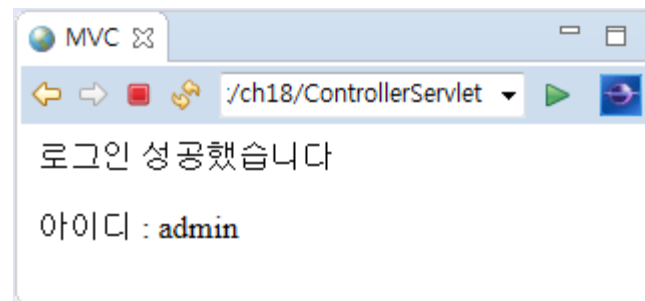


The screenshot shows a web browser window titled 'MVC'. The address bar displays '/ch18/mvc.jsp'. The form contains two input fields: '아이디 : admin' and '비밀번호 : ' followed by five dots. A '전송' button is located below the password field.

3. MVC 패턴 구현 방법

JSPBook/WebContent/ch18/mvc_success.jsp

```
01 <% page contentType="text/html; charset=utf-8"%>
02 <% page import="ch18.com.model.LoginBean"%>
03 <html>
04 <head>
05 <title>MVC</title>
06 </head>
07 <body>
08     <p>로그인 성공했습니다
09     <p><%
10         LoginBean bean = (LoginBean) request.getAttribute("bean");
11         out.print("아이디 : " + bean.getId());
12     %>
13 </body>
14 </html>
```



3. MVC 패턴 구현 방법

JSPBook/WebContent/ch18/mvc_error.jsp

```
01 <% page contentType="text/html; charset=utf-8"%>
02 <html>
03 <head>
04 <title>MVC</title>
05 </head>
06 <body>
07     <p>아이디와 비밀번호를 확인해주세요
08     <% include file="mvc.jsp"%>
09 </body>
10 </html>
```

4. [웹 쇼핑몰] 게시판 만들기

The image displays three browser windows illustrating the development of a web board system.

Top-Left Window: Shows the board list page. The URL is `http://localhost:8080/WebMarket/Board/BoardList.do?pageNum=1`. The page title is "게시판" (Board). It contains a table with the following data:

번호	제목	작성일	조회	글쓴이
5	[웹 쇼핑몰] 웹 게시판 만들기	2018/05/15(12:39:22)	3	홍길순
4	MVC 패턴 구현 방법	2018/05/15(12:39:01)	1	홍길순
3	MVC 패턴 구조	2018/05/15(12:38:39)	1	홍길순
2	MVC의 개요			

Below the table is a search bar with a dropdown menu labeled "제목에서" and a "검색" button. The footer shows "© WebMarket".

Top-Right Window: Shows the board form page. The URL is `http://localhost:8080/WebMarket/Board/BoardForm.do`. The page title is "게시판" (Board). It contains a form with the following fields:

- 성명: 홍길순
- 제목: subject
- 내용: content

The footer shows "© WebMarket".

Bottom Window: Shows the board form page with sample data. The URL is `http://localhost:8080/WebMarket/Board/BoardForm.do?num=5&pageNum=1`. The page title is "게시판" (Board). It contains a form with the following fields:

- 성명: 홍길순
- 제목: [웹 쇼핑몰] 웹 게시판 만들기
- 내용: 웹 쇼핑몰의 게시판을 만듭니다.

Below the form are three buttons: "삭제" (Delete), "수정" (Modify), and "목록" (List). The footer shows "© WebMarket".

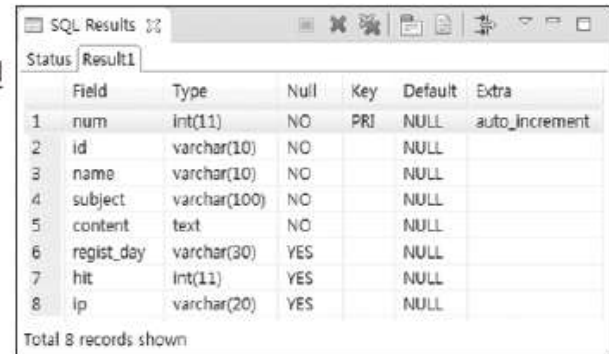
4. [웹 쇼핑몰] 게시판 만들기

예제 18-2 웹 쇼핑몰의 데이터베이스에 게시판 관리 테이블 만들기

- 데이터베이스 커넥션 설정하기:
 - 커넥션 이름은 Market_Conn, 데이터베이스 이름은 WebMarketDB를 사용하여 Data Source Explorer에서 실행
- 회원 관리 테이블 생성하기

WebMarket/WebContent/resources/sql/board.sql

```
01 CREATE TABLE board (  
02     num int not null auto_increment, //게시글 순번  
03     id varchar(10) not null, //회원 아이디  
04     name varchar(10) not null, //회원 이름  
05     subject varchar(100) not null, //게시글 제목  
06     content text not null, //게시글 내용  
07     regist_day varchar(30), //게시글 등록 일자  
08     hit int, //게시글 조회 수  
09     ip varchar(20), //게시글 등록 시 IP  
10     PRIMARY KEY (num) //게시글 순번을 고유 키로 설정  
11 )default CHARSET=utf8;
```



The screenshot shows a window titled 'SQL Results' with a tab labeled 'Result1'. It displays the structure of the 'board' table with 8 records. The columns are Field, Type, Null, Key, Default, and Extra.

	Field	Type	Null	Key	Default	Extra
1	num	int(11)	NO	PRI	NULL	auto_increment
2	id	varchar(10)	NO		NULL	
3	name	varchar(10)	NO		NULL	
4	subject	varchar(100)	NO		NULL	
5	content	text	NO		NULL	
6	regist_day	varchar(30)	YES		NULL	
7	hit	int(11)	YES		NULL	
8	ip	varchar(20)	YES		NULL	

Total 8 records shown

4. [웹 쇼핑몰] 게시판 만들기

예제 18-3 MVC 기반 웹 게시판의 기본 페이지 준비하기

■ 메뉴 페이지 수정하기

WebMarket/WebContent/menu.jsp

```
01 <%@ page contentType="text/html; charset=utf-8"%>
02 ... (생략) ...
03         <li class="nav-item"><a class="nav-link" href="<c:url
           value="/editProduct.jsp?edit=delete"/>">상품 삭제</a></li>
04         <li class="nav-item"><a class="nav-link" href="<c:url
           value="/BoardListAction.do?pageNum=1"/>">게시판</a></li>
05 ... (생략) ...
```

4. [웹 쇼핑몰] 게시판 만들기

- web.xml 파일에 추가 작성하기

WebMarket/WebContent/WEB-INF/web.xml

```
01 <?xml version="1.0" encoding="UTF-8"?>
02 <web-app>
03     ...(생략)...
04     <servlet>
05         <servlet-name>BoardController</servlet-name>
06         <servlet-class>mvc.controller.BoardController</servlet-class>
07     </servlet>
08
09     <servlet-mapping>
10         <servlet-name>BoardController</servlet-name>
11         <url-pattern>*.do</url-pattern>
12     </servlet-mapping>
13 </web-app>
```

4. [웹 쇼핑몰] 게시판 만들기

■ 데이터베이스 연결 클래스 생성하기

WebMarket/src/mvc/database/DBConnection.java

```
01 package mvc.database;
02
03 import java.sql.Connection;
04 import java.sql.SQLException;
05 import java.sql.DriverManager;
06
07 public class DBConnection {
08
09     public static Connection getConnection() throws SQLException,
10         ClassNotFoundException {
11
12         Connection conn = null;
13
14         String url = "jdbc:mysql://localhost:3306/WebMarketDB";
15         String user = "root";
16         String password = "1234";
17
18         Class.forName("com.mysql.jdbc.Driver");
19         conn = DriverManager.getConnection(url, user, password);
20
21         return conn;
22     }
```

4. [웹 쇼핑몰] 게시판 만들기

■ 게시판 데이터 클래스 생성하기

WebMarket/src/mvc/model/BoardDTO.java

```
01 package mvc.model.board;
02
03 public class BoardDTO {
04     private int num;           //순번
05     private String id;         //등록자 아이디
06     private String name;       //등록자 이름
07     private String subject;    //등록 제목
08     private String content;    //등록 내용
09     private String regist_day; //등록 일자
10     private int hit;           //조회 수
11     private String ip;         //IP 주소
12     //기본 생성자
13     public BoardDTO() {
14         super();
15     }
16     //Getter() 메소드와 Setter() 메소드
17     public int getNum() {
18         return num;
19     }
20     ...(생략)...
21     public void setIp(String ip) {
22         this.ip = ip;
23     }
24 }
```

4. [웹 쇼핑몰] 게시판 만들기

예제 18-4 웹 게시판에 등록된 글 목록 보기

■ 컨트롤러 작성하기

WebMarket/src/mvc/controller/BoardController.java

```
01 package mvc.controller;
02
03 import java.io.IOException;
04 import java.util.ArrayList;
05 import java.util.List;
06
07 import javax.servlet.RequestDispatcher;
08 import javax.servlet.ServletException;
09 import javax.servlet.http.HttpServlet;
10 import javax.servlet.http.HttpServletRequest;
11 import javax.servlet.http.HttpServletResponse;
12
13 import mvc.model.BoardDAO;
14 import mvc.model.BoardDTO;
15
16 public class BoardController extends HttpServlet {
17     private static final long serialVersionUID = 1L;
18     static final int LISTCOUNT = 5;
19
20     public void doGet(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
21         doPost(request, response);
22     }
```

4. [웹 쇼핑몰] 게시판 만들기

```
23
24     public void doPost(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
25         String RequestURI = request.getRequestURI();
26         String contextPath = request.getContextPath();
27         String command = RequestURI.substring(contextPath.length());
28
29         response.setContentType("text/html; charset=utf-8");
30         request.setCharacterEncoding("utf-8");
31
32         if (command.equals("/BoardListAction.do")) { //등록된 글 목록 페이지 출력하기
33             requestBoardList(request);
34             RequestDispatcher rd = request.getRequestDispatcher("./board/
    list.jsp");
35             rd.forward(request, response);
36         }
37     }
38
39     public void requestBoardList(HttpServletRequest request){ //등록된 글 목록
    가져오기
40         ... (생략) ...
41     }
42 }
```

4. [웹 쇼핑몰] 게시판 만들기

■ 모델 작성하기

WebMarket/src/mvc/model/BoardDAO.java

```
01 package mvc.model;
02
03 import java.sql.Connection;
04 import java.sql.PreparedStatement;
05 import java.sql.ResultSet;
06 import java.util.ArrayList;
07
08 import mvc.database.DBConnection;
09
10 public class BoardDAO {
11
12     private static BoardDAO instance;
13
14     private BoardDAO() {
15     }
16
17     public static BoardDAO getInstance() {
18         if (instance == null)
19             instance = new BoardDAO();
20         return instance;
21     }
22     //board 테이블의 레코드 개수
23     public int getListCount() {
24         ... (생략) ...
25     }
26     //board 테이블의 레코드 가져오기
27     public ArrayList<BoardDTO> getBoardList(int page, int limit, String
28         items, String text) {
29         ... (생략) ...
30     }
31 }
```

4. [웹 쇼핑몰] 게시판 만들기

- 뷰 작성하기

- /WebContent/ 폴더에 board 폴더를 만든 후 이 폴더에 list.jsp 파일 생성
- list.jsp 파일의 내용은 예제 소스 파일 참고

4. [웹 쇼핑몰] 게시판 만들기

예제 18-5 웹 게시판에 글 등록하기

■ 컨트롤러 작성하기

WebMarket/src/mvc/controller/board/BoardController.java

```
01 package mvc.controller;
02 ... (생략) ...
03
04 public class BoardController extends HttpServlet {
05     ... (생략) ...
06     public void doPost(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
07         ... (생략) ...
08         if (command.equals("/BoardListAction.do")) {
09             requestBoardList(request);
10             RequestDispatcher rd = request.getRequestDispatcher("./board/
                list.jsp");
11             rd.forward(request, response);
12         } else if (command.equals("/BoardWriteForm.do")) { //글 등록 페이지 출력
13             requestLoginName(request);
14             RequestDispatcher rd = request.getRequestDispatcher("./board/
                writeForm.jsp");
15             rd.forward(request, response);
16         } else if (command.equals("/BoardWriteAction.do")) { //새로운 글 등록
17             requestBoardWrite(request);
```

4. [웹 쇼핑몰] 게시판 만들기

```
18         RequestDispatcher rd = request.getRequestDispatcher
           ("/BoardListAction.do");
19         rd.forward(request, response);
20     }
21 }
22 ... (생략) ...
23 public void requestLoginName(HttpServletRequest request){
           //인증된 사용자명 가져오기
24     ... (생략) ...
25 }
26
27 public void requestBoardWrite(HttpServletRequest request){ //새로운 글 등록하기
28     ... (생략) ...
29 }
30 }
```

4. [웹 쇼핑몰] 게시판 만들기

■ 모델 작성하기

WebMarket/srcmvc/controller/board/BoardDAO.java

```
01 package mvc.model;
02 ...
03 public class BoardDAO {
04     ... (생략) ...
05     public String getLoginNameById(String id) { //member 테이블에서 인증된 id의
        사용자명 가져오기
06         ... (생략) ...
07     }
08
09     public void insertBoard(BoardDTO board) { //board 테이블에 새로운 글 삽입하기
10         ... (생략) ...
11     }
12 }
```

4. [웹 쇼핑몰] 게시판 만들기

- 뷰 작성하기
 - /WebContent/board/ 폴더에 writeForm.jsp 파일 생성
 - writeForm.jsp 파일의 내용은 예제 소스 파일 참고

4. [웹 쇼핑몰] 게시판 만들기

예제 18-6 웹 게시판에 등록된 글 확인하기

■ 컨트롤러 작성하기

WebMarket/srcmvc/controller/board/BoardController.java

```
01 package mvc.controller;
02 ... (생략) ...
03 public class BoardController extends HttpServlet {
04     ... (생략) ...
05     public void doPost(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
06         ... (생략) ...
07         } else if (command.equals("/BoardWriteAction.do")) {
08             requestBoardWrite(request);
09             RequestDispatcher rd = request.getRequestDispatcher
                ("/BoardListAction.do");
10             rd.forward(request, response);
11         }
12         else if (command.equals("/BoardViewAction.do")) { //선택된 글 상자 페이지
            가져오기
13             requestBoardView(request);
14             RequestDispatcher rd = request.getRequestDispatcher("/BoardView.
                do");
15             d.forward(request, response);
16         } else if (command.equals("/BoardView.do")) { //글 상세 페이지 출력하기
17             RequestDispatcher rd = request.getRequestDispatcher("/board/
                view.jsp");
18             rd.forward(request, response);
19         }
20     }
21     ... (생략) ...
22     public void requestBoardView(HttpServletRequest request){ //선택된 글 상세
        페이지 가져오기
23         ... (생략) ...
24     }
25 }
```

4. [웹 쇼핑몰] 게시판 만들기

■ 모델 작성하기

WebMarket/srcmvc/controller/board/BoardDAO.java

```
01 package mvc.model;
02 ...생략...
03 public class BoardDAO {
04     ...생략...
05     public void updateHit(int num) { //선택된 글의 조회 수 증가시키기
06         ...생략...
07     }
08
09     public BoardDTO getBoardByNum(int num, int page) { //선택된 글 상세 내용 가져오기
10         ...생략...
11     }
12 }
```

4. [웹 쇼핑몰] 게시판 만들기

- 뷰 작성하기
 - /WebContent/board/ 폴더에 view.jsp 파일 생성
 - view.jsp 파일의 내용은 예제 소스 파일 참고

4. [웹 쇼핑몰] 게시판 만들기

예제 18-7 웹 게시판에 등록된 글 수정하기

■ 컨트롤러 작성하기

WebMarket/srcmvc/controller/board/BoardController.java

```
01 package mvc.controller;
02 ... (생략) ...
03 public class BoardController extends HttpServlet {
04     ... (생략) ...
05     public void doPost(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
06         ... (생략) ...
07         } else if (command.equals("/BoardView.do")) {
08             RequestDispatcher rd = request.getRequestDispatcher("./board/
                view.jsp");
09             rd.forward(request, response);
10         } else if (command.equals("/BoardUpdateAction.do")) { //선택된 글 수정하기
11             requestBoardUpdate(request);
12             RequestDispatcher rd = request.getRequestDispatcher
                ("/BoardListAction.do");
13             rd.forward(request, response);
14         }
15     }
16     ... (생략) ...
17     public void requestBoardUpdate(HttpServletRequest request){ //선택된 글
        내용 수정하기
18         ... (생략) ...
19     }
20 }
```


4. [웹 쇼핑몰] 게시판 만들기

WebMarket/srcmvc/controller/board/BoardDAO.java

```
01 package mvc.model;
02 ... (생략) ...
03 public class BoardDAO {
04     ... (생략) ...
05     public void updateBoard(BoardDTO board) { //선택된 글 내용 수정하기
06         ... (생략) ...
07     }
08 }
```

4. [웹 쇼핑몰] 게시판 만들기

예제 18-8 웹 게시판에 등록된 글 삭제하기

■ 컨트롤러 작성하기

WebMarket/srcmvc/controller/board/BoardController.java

```
01 package mvc.controller;
02 ... (생략) ...
03 public class BoardController extends HttpServlet {
04     ... (생략) ...
05     public void doPost(HttpServletRequest request, HttpServletResponse
        response) throws ServletException, IOException {
06         ... (생략) ...
07         } else if (command.equals("/BoardUpdateAction.do")) {
08             requestBoardUpdate(request);
09             RequestDispatcher rd = request.getRequestDispatcher
                ("/BoardListAction.do");
10             rd.forward(request, response);
11         } else if (command.equals("/BoardDeleteAction.do")) { //선택된 글 삭제하기
12             requestBoardDelete(request);
13             RequestDispatcher rd = request.getRequestDispatcher
                ("/BoardListAction.do");
14             rd.forward(request, response);
15         }
16     }
17     ... (생략) ...
18     public void requestBoardDelete(HttpServletRequest request){ //선택된 글 삭
        제하기
```

4. [웹 쇼핑몰] 게시판 만들기

■ 모델 작성하기

WebMarket/srcmvc/controller/board/BoardDAO.java

```
01 package mvc.model;
02 ...(생략)...
03 public class BoardDAO {
04     ...(생략)...
05     public void deleteBoard(int num) { //선택된 글 삭제하기
06         ...(생략)...
07     }
08 }
```

