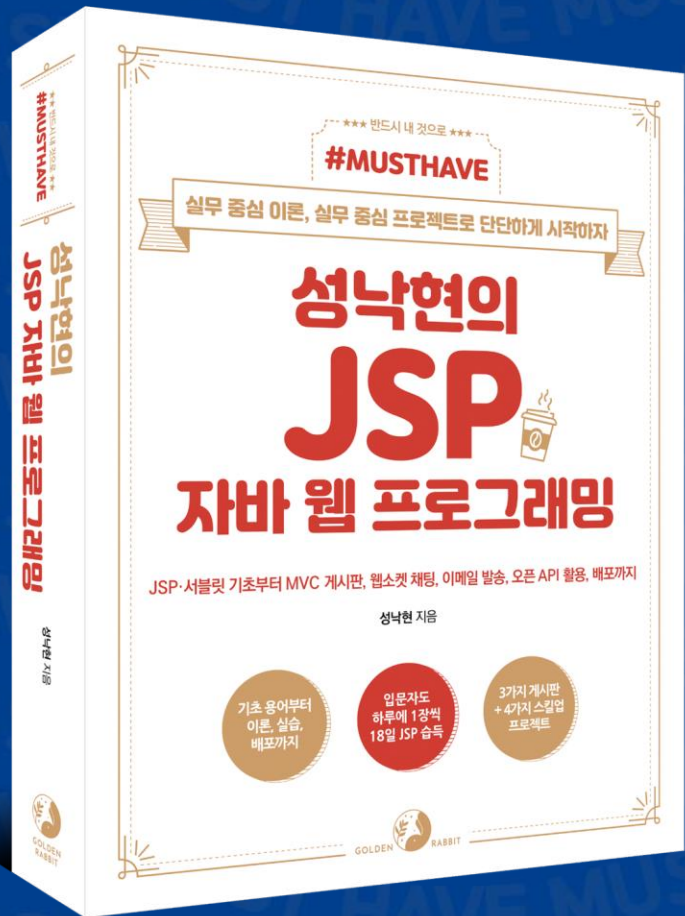


Chapter

14

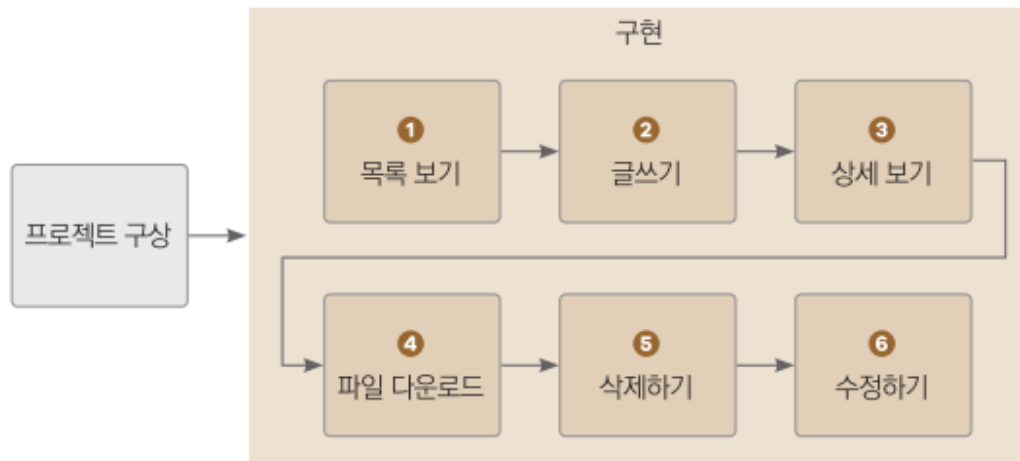
모델2 방식(MVC 패턴) 자료실형 게시판 만들기



■ 학습 목표

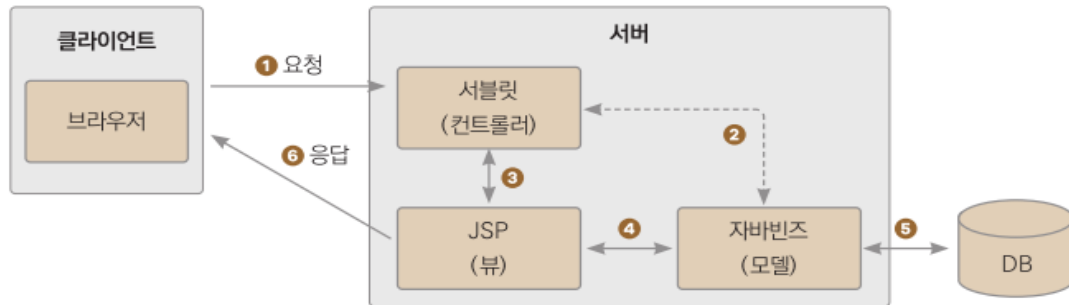
- 이번 단계(10~13장)에서 학습한 내용을 기반으로 MVC 패턴을 적용한 모델2 방식의 게시판을 제작해보겠습니다.
- 회원인증 없이 누구나 사용할 수 있고, 파일 첨부과 다운로드 기능도 제공합니다.

■ 학습 순서



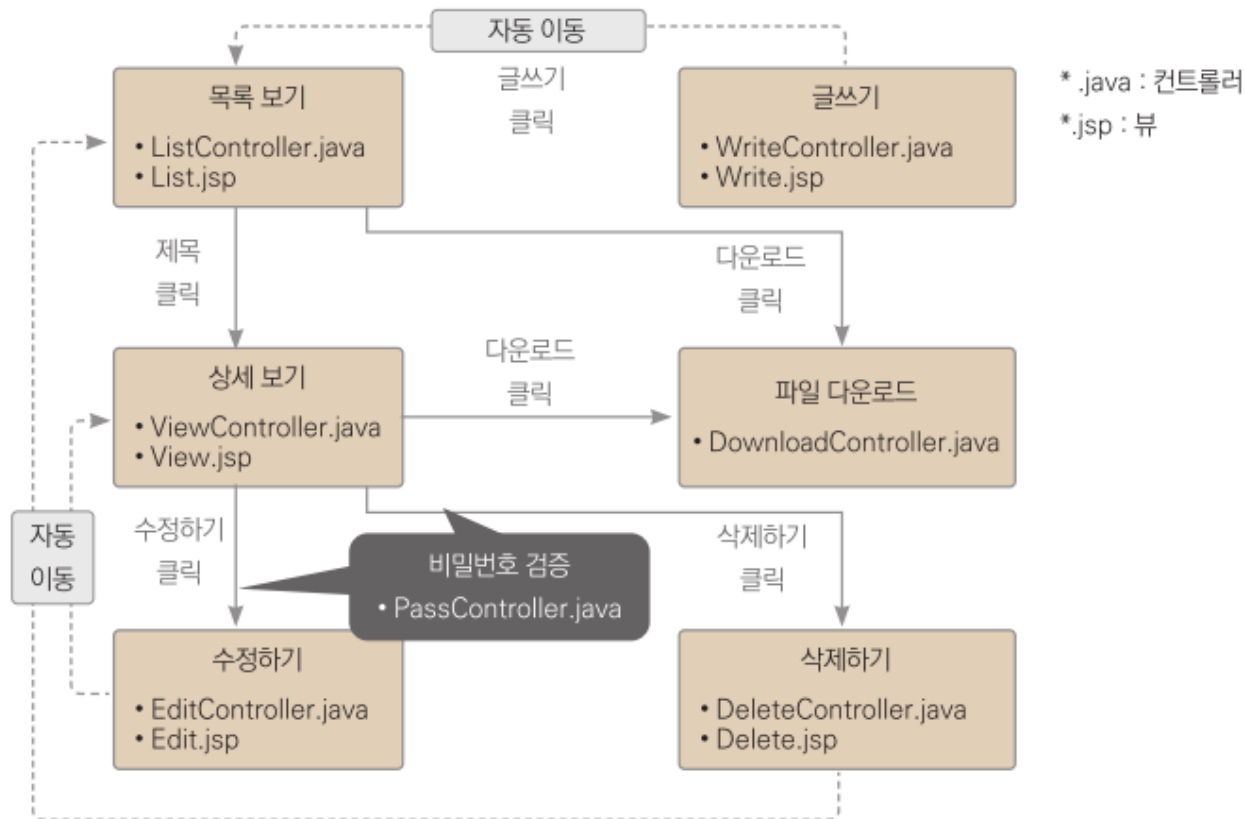
EL, JSTL, 파일업로드, 서블릿을 종합적으로 사용하여 자료실형 게시판 제작

- 비회원제
 - 회원인증 없이 누구나 글을 작성
 - 대신 글쓰기 시 비밀번호 입력이 필수
 - 비밀번호를 통해 수정이나 삭제 가능
- 자료실
 - 글쓰기 시 파일을 첨부 가능
 - 파일 첨부 시 정해진 용량 이상은 업로드할 수 없음
 - 첨부된 파일을 다운로드할 수 있음
- Model2 방식, 즉 MVC 패턴
 - 모델-뷰-컨트롤러 구성이 핵심



14.1 프로젝트 구상

■ 자료실형 게시판의 프로세스



14.1 프로젝트 구상

■ 기능별 요청명 정의

- 서블릿으로 개발할 때는 기능별로 요청명과 서블릿 클래스명을 먼저 정의하는것이 좋음

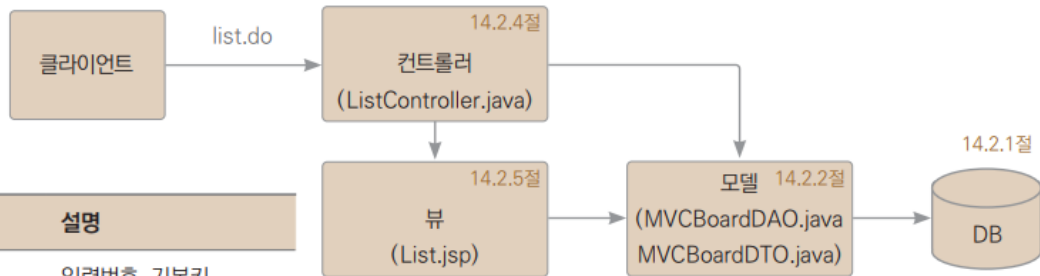
기능	매핑 방법	요청명	컨트롤러(서블릿)	뷰(JSP) 경로
목록 보기	web.xml	/mvcboard/list.do	ListController	/14MVCBoard/List.jsp
글쓰기	web.xml	/mvcboard/write.do	WriteController	/14MVCBoard/Write.jsp
상세 보기	애너테이션	/mvcboard/view.do	ViewController	/14MVCBoard/View.jsp
비밀번호 검증	애너테이션	/mvcboard/pass.do	PassController	/14MVCBoard/Pass.jsp
수정	애너테이션	/mvcboard/edit.do	EditController	/14MVCBoard/Edit.jsp
삭제	애너테이션	필요 없음	PassController	필요 없음
다운로드	애너테이션	/mvcboard/download.do	DownloadController	필요 없음

- 삭제 : 비밀번호 검증이 완료되면 즉시 삭제되므로 요청명, 뷰 필요없음
- 다운로드 : 요청 즉시 다운로드 되므로 뷰 필요없음

■ 목록 보기 처리 프로세스 및 테이블 정의서

테이블명 : mvcboard

컬럼명	데이터 타입	null 허용	키	기본값	설명
idx	number	N	기본키		일련번호. 기본키
name	varchar2(50)	N			작성자 이름
title	varchar2(200)	N			제목
content	varchar2(2000)	N			내용
postdate	date	N		sysdate	작성일
ofile	varchar2(200)	Y			원본 파일명
sfile	varchar2(30)	Y			저장된 파일명
downcount	number	N		0	다운로드 횟수
pass	varchar2(50)	N			비밀번호
visitcount	number	N		0	조회수



- 비회원제 게시판이므로 id 컬럼 대신 name 과 pass 컬럼으로 대체됨
- 파일명은 원본과 저장된 파일명을 구분해서 저장함(12장에서 설명했음)
- 단 첨부파일 없이도 작성되어야 하므로 null을 허용하는 컬럼으로 지정



14.2 목록 보기

▣ DTO 및 DAO 클래스 생성

예제 14-1] Java Resources/model2/mvcboard/MVCBoardDTO.java

```
package model2.mvcboard;

public class MVCBoardDTO {
    // 멤버 변수 선언 ❶
    private String idx;
    private String name;
    private String title;
    private String content;
    private java.sql.Date postdate;
    private String ofile;
    private String sfile;
    private int downcount;
    private String pass;
    private int visitcount;

    // 게터/세터 ❷
    public String getIdx() {
        return idx;
    }
}
```

- ❶ mvcboard 테이블의 컬럼명과 동일하게 멤버변수 선언
- ❷ getter / setter는 이클립스의 자동생성 메뉴 사용

생성자는 필요없음

■ DTO 및 DAO 클래스 생성

예제 14-2] Java Resources/model2/mvcboard/MVCBoardDAO.java

```
public class MVCBoardDAO extends DBConnPool { // 커넥션 풀 상속
    public MVCBoardDAO() {
        super();
    }

    // 검색 조건에 맞는 게시물의 개수를 반환합니다.
    public int selectCount(Map<String, Object> map) {
        int totalCount = 0;
        // 쿼리문 준비
        String query = "SELECT COUNT(*) FROM mvcboard";
        // 검색 조건이 있다면 WHERE절로 추가
        if (map.get("searchWord") != null) {
            query += " WHERE " + map.get("searchField") + " "
                + " LIKE '%" + map.get("searchWord") + "%'";
        }
    }
}
```

DBCP(커넥션풀)을 사용하기 위해 상속

기존 회원제 게시판에서 사용했던 메서드와 동일함. 테이블명만 mvcboard 로 변경.(아래 부분 생략)

■ DTO 및 DAO 클래스 생성

예제 14-2] Java Resources/model2/mvcboard/MVCBoardDAO.java(계속)

```
// 검색 조건에 맞는 게시물 목록을 반환합니다(페이징 기능 지원).
public List<MVCBoardDTO> selectListPage(Map<String,Object> map) {
    List<MVCBoardDTO> board = new Vector<MVCBoardDTO>();
    // 쿼리문 준비
    String query = " "
        + "SELECT * FROM ( "
        + "    SELECT Tb.*, ROWNUM rNum FROM ( "
        + "        SELECT * FROM mvcboard ";

    // 검색 조건이 있다면 WHERE절로 추가
    if (map.get("searchWord") != null)
    {
        query += " WHERE " + map.get("searchField")
            + " LIKE '%" + map.get("searchWord") + "%' ";
    }

    query += "        ORDER BY idx DESC "
        + "    ) Tb "
        + " ) "
        + " WHERE rNum BETWEEN ? AND ?"; // 게시물 구간은 인파라미터로..
```

게시물의 목록을 페이지의 구간에 맞춰 인출하는 메서드로 회원제 게시판과 동일함.(아래 부분 생략)



14.2 목록 보기

■ 진입 화면 작성 및 매핑

- 서블릿은 요청명을 통해 실행해야 하므로 진입페이지를 제작

예제 14-3] webapp/14MVCBoard/Default.jsp

```
<body>
  <h2>파일 첨부형 게시판</h2>
  <a href="../mvcboard/list.do">게시판 목록 바로가기</a>
</body>
```

파일 첨부형 게시판

[게시판 목록 바로가기](#)

예제 14-4] webapp/WEB-INF/web.xml

```
<servlet>
  <servlet-name>MVCBoardList</servlet-name>  ← 서블릿 이름
  <servlet-class>model2.mvcboard.ListController</servlet-class> ← 서블릿 클래스
</servlet>
<servlet-mapping>
  <servlet-name>MVCBoardList</servlet-name>  ← 서블릿 이름
  <url-pattern>/mvcboard/list.do</url-pattern> ← 요청명
</servlet-mapping>
```



14.2 목록 보기

■ 컨트롤러(서블릿) 작성

예제 14-5] Java Resources/model2/mvcboard/ListController.java

```
public class ListController extends HttpServlet { ❶
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException { ❷
        // DAO 생성 ❸
        MVCBoardDAO dao = new MVCBoardDAO();

        int totalCount = dao.selectCount(map); // 게시물 개수 ❹

        ServletContext application = getServletContext();
        int pageSize = Integer.parseInt(application.getInitParameter(
            "POSTS_PER_PAGE"));
        int blockPage = Integer.parseInt(application.getInitParameter(
            "PAGES_PER_BLOCK")); ❷
        map.put("start", start);
        map.put("end", end);

        List<MVCBoardDTO> boardLists = dao.selectListPage(map);
```

❶ HttpServlet 을 상속해서 서블릿 정의

❹ 게시물의 갯수 카운트

❷ 서블릿에서 application 내장객체를 사용할때는 getServletContext() 메서드로 얻어와야 함

계산된 페이지의 구간을 Map 컬렉션에 추가한 후 게시물 인출

■ 컨트롤러(서블릿) 작성

예제 14-5] Java Resources/model2/mvcboard/ListController.java

```
// 뷰에 전달할 매개변수 추가
String pagingImg = BoardPage.pagingStr(totalCount, pageSize,
    blockPage, pageNum, "../mvcboard/list.do");
    // 바로가기 영역 HTML 문자열
map.put("pagingImg", pagingImg);
map.put("totalCount", totalCount);
map.put("pageSize", pageSize);
map.put("pageNum", pageNum);

// 전달할 데이터를 request 영역에 저장 후 List.jsp로 포워드 ⑫
req.setAttribute("boardLists", boardLists); ⑬
req.setAttribute("map", map);
req.getRequestDispatcher("/14MVCBoard/List.jsp").forward(req, resp);
}
```

⑪ 처리를 완료한 여러가지 데이터를 Map 컬렉션에 저장(페이지 바로가기, 게시물 개수 등)

⑫ View로 전달하기 위해 request 영역에 저장한 후 JSP로 포워드



14.2 목록 보기

■ 뷰(JSP) 만들기

예제 14-6] webapp/14MVCBoard/List.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@ taglib prefix="c" uri="jakarta.tags.core" %>
<c:choose>
    <c:when test="${ empty boardLists }"> <!-- 게시물이 없을 때 --> ③
        <tr>
            <td colspan="6" align="center">
                등록된 게시물이 없습니다^^*
            </td>
        </tr>
    </c:when>
    <c:otherwise> <!-- 게시물이 있을 때 --> ④
        <c:forEach items="${ boardLists }" var="row" varStatus="loop">
            <td> <!-- 번호 --> ⑤
                ${ map.totalCount - (((map.pageNum-1) * map.pageSize) + loop.
index)}
            </td>
```

서블릿 게시판에서는 JSTL을 사용 하
므로 taglib 지시어 선언

③ JSTL의 <c:choose> 태그로 게시물
이 없는 경우와

④ 있는 경우에 대한 조건을 설정. 게시
물이 있다면 <c:forEach> 태그로 갯수
만큼 반복 출력

⑤ 가상번호는 현재 페이지 번호를 적
용하여 계산한 후 EL로 출력

■ 뷰(JSP) 만들기

예제 14-6] webapp/14MVCBoard/List.jsp(계속)

```

<td> <!-- 첨부 파일 -->
<c:if test="${ not empty row.ofile }">
    <a href="../mvcboard/download.do?ofile=${ row.ofile }&sfile=${
row.sfile }&idx=${ row.idx }">[Down]</a> ⑦
</c:if>
</td>
</tr>
</c:forEach>
</c:otherwise>
</c:choose>
<tr align="center">
<td>
    ${ map.pagingImg } ⑧
</td>
<td width="100"><button type="button"
    onclick="location.href='../mvcboard/write.do';">글쓰기</button>
</td> ⑨
</tr>

```

⑦ 첨부파일이 있는 경우에만 다운로드 링크 보임 처리

⑧ 페이지 바로가기 링크 출력

■ 뷰(JSP) 만들기

예제 14-6] webapp/14MVCBoard/List.jsp(계속)

● 가상번호 계산하기

`${ 전체 게시물 수 - (((현재 페이지 번호 - 1) * 페이지 사이즈) + varStatus의 index값) }`

- 첫 번째 게시물 : $5 - (((1-1) * 10) + 0) = 5$
- 두 번째 게시물 : $5 - (((1-1) * 10) + 1) = 4$

파일 첨부형 게시판 - 목록 보기(List)

번호	제목	작성자	조회수	작성일	첨부
5	자료실 제목5입니다.	대조영	0	2021-08-04	
4	자료실 제목4입니다.	강감찬	0	2021-08-04	
3	자료실 제목3입니다.	이순신	0	2021-08-04	
2	자료실 제목2입니다.	장보고	0	2021-08-04	
1	자료실 제목1입니다.	김유신	0	2021-08-04	

1

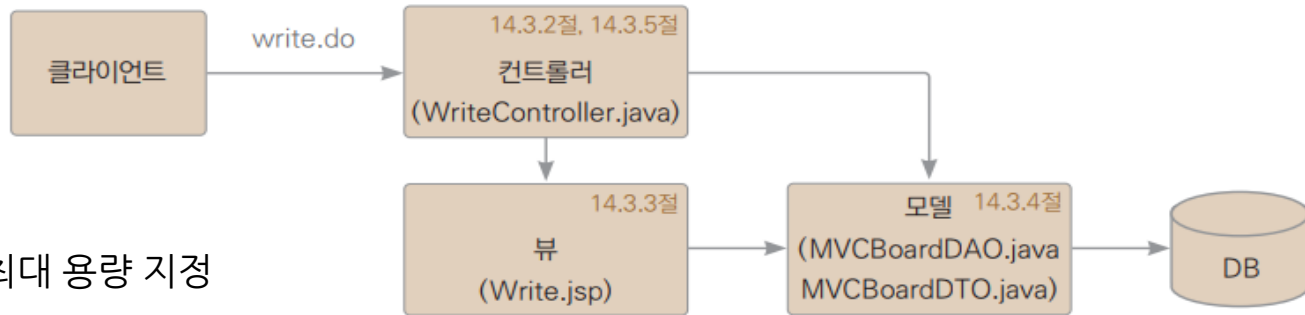
검색 폼

목록

하단 메뉴

■ 글쓰기 처리 프로세스 및 요청명 매핑

- 자료실형 게시판이므로 파일 업로드 기능도 함께 구현해야 함



- 매핑 시 첨부파일의 최대 용량 지정

```
<servlet>
  <servlet-name>MVCBoardWrite</servlet-name>
  <servlet-class>model2.mvcboard.WriteController</servlet-class>
  <multipart-config>①
    <max-file-size>1048576</max-file-size> <!-- 1MB -->
    <max-request-size>10485760</max-request-size> <!-- 10MB -->
  </multipart-config>
</servlet>
```

<max-file-size>
: 개별 파일의 최대 크기를 지정

<max-request-size>
: 전체 파일의 최대 크기를 지정



14.3 글쓰기

■ 컨트롤러 작성 1 - 작성 폼으로 진입

- 글쓰기 폼으로 진입할때는 doGet() 메서드 오버라이딩

예제 14-8] Java Resources/model2/mvcboard/WriteController.java

```
public class WriteController extends HttpServlet {  
    @Override  
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)  
        throws ServletException, IOException { ❶  
        req.getRequestDispatcher("/14MVCBoard/Write.jsp").forward(req, resp);  
    }  
}
```

■ 뷰 작성 예제 14-9] webapp/14MVCBoard/Write.jsp

- <form> 태그의 method 속성은 **post**, enctype 속성은 **multipart/form-data**로 설정
- 일반적인 글쓰기 폼이므로 코드는 생략

■ 모델 작성(DAO에 기능 추가)

- 글쓰기 폼에서 입력한 내용과 첨부파일을 insert 하는 메서드 추가

예제 14-10] Java Resources/model2/mvcboard/MVCBoardDAO.java

```
public int insertWrite(MVCBoardDTO dto) { ❶
    int result = 0;
    try {
        String query = "INSERT INTO mvcboard ( "
            + " idx, name, title, content, ofile, sfile, pass) "
            + " VALUES ( "
            + " seq_board_num.NEXTVAL,?,?,?,?,?,?)"; ❷

        psmt = con.prepareStatement(query); ❸
        psmt.setString(1, dto.getName());
        psmt.setString(2, dto.getTitle());
        psmt.setString(3, dto.getContent());
        psmt.setString(4, dto.getOfile());
        psmt.setString(5, dto.getSfile());
```

❶ 입력한 품값을 저장한 DTO객체를 매개변수로 받음

❷ insert 쿼리문 작성

❸ 인파라미터 설정 및 실행

■ 컨트롤러 작성 2 - 폼값 처리

- 글쓰기 처리에는 doPost() 메서드 오버라이딩

예제 14-11] Java Resources/model2/mvcboard/WriteController.java

```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp)
    throws ServletException, IOException {
    String saveDirectory = req.getServletContext().getRealPath("/Uploads");

    String originalFileName = "";
    try {
        originalFileName = FileUtil.uploadFile(req, saveDirectory);
    }
    catch (Exception e) {
        // 파일 업로드 실패 ③
        JSFunction.alertLocation(resp, "파일 업로드 오류입니다.",
            "../mvcboard/write.do");
        return;
    }
}
```

Uploads 디렉토리의 물리적 경로를
얻어온 후 유틸리티 클래스를 통해 파
일 업로드 처리

■ 컨트롤러 작성 2 - 품값 처리(계속)

예제 14-11] Java Resources/model2/mvcboard/WriteController.java

```
// 품값을 DTO에 저장 ④
MVCBoardDTO dto = new MVCBoardDTO();
dto.setName(req.getParameter("name"));
dto.setTitle(req.getParameter("title"));
if (originalFileName != "") { ⑤
    // 첨부 파일이 있을 경우 파일명 변경 ⑥
    String savedFileName = FileUtil.renameFile(saveDirectory,
originalFileName);

    dto.setOfile(originalFileName); // 원래 파일 이름
    dto.setSfile(savedFileName); // 서버에 저장된 파일 이름 ⑦
}

// DAO를 통해 DB에 게시 내용 저장 ⑧
MVCBoardDAO dao = new MVCBoardDAO();
int result = dao.insertWrite(dto);
dao.close();
```

④ 품값을 DTO에 저장

⑤ 첨부파일이 있는 경우 파일명을
변경한 후 DTO에 파일명 저장

⑧ 테이블에 insert 처리

■ Javascript 유틸리티 메서드 추가

예제 14-12] Java Resources/Utils/JFunction.java

```
public static void alertLocation(HttpServletResponse resp, String msg,
String url) {①
    try {
        resp.setContentType("text/html;charset=UTF-8"); ②
        PrintWriter writer = resp.getWriter(); ③
        String script = ""
            + "<script>"
            + "    alert('" + msg + "');"
            + "    location.href='" + url + "';"
            + "</script>";
        writer.print(script); ④
    }
    catch (Exception e) {}
}
```

① 서블릿에서 즉시 출력하여 실행할 수 있는 Javascript 코드 추가

alert()로 알림창을 띄운 후 페이지를 이동

알림창의 띄운 후 뒤로 이동하는 유틸리티 메서드는 동일한 형식으로 구현

■ 동작 확인

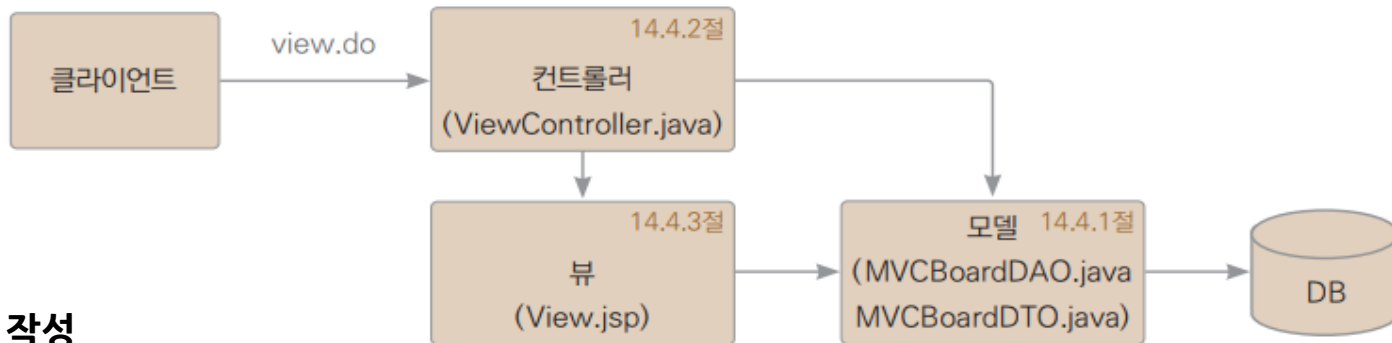
파일 첨부형 게시판 - 글쓰기(Write)

작성자	<input type="text" value="성낙현"/>
제목	<input type="text" value="글쓰기 Test"/>
내용	<div>글을 작성해보겠습니다. 파일도 첨부하겠습니다. 단, 1MB 이하로 선택해주세요. 비밀번호는 1234로 지정하겠습니다.</div>
첨부 파일	<div>파일 선택</div> <div>유경이랑.png</div> <div>1MB 이하 파일 선택</div>
비밀번호	<input type="password" value="...."/>
<div>작성 완료</div> <div>RESET</div> <div>목록 바로가기</div>	

파일 첨부형 게시판 - 목록 보기(List)

<div>제목 ▼</div> <div></div> <div>검색하기</div>					
번호	제목	작성자	조회수	작성일	첨부
6	글쓰기 Test	성낙현	1	2021-08-04	[Down]
5	자료실 제목5입니다.	대조영	2	2021-08-04	
4	자료실 제목4입니다.	강감찬	0	2021-08-04	

■ 상세보기 처리 프로세스



■ 모델 작성

예제 14-13] Java Resources/model2/mvcboard/MVCBoardDAO.java

```
public MVCBoardDTO selectView(String idx) {  
    MVCBoardDTO dto = new MVCBoardDTO(); // DTO 객체 생성  
    String query = "SELECT * FROM mvcboard WHERE idx=?"; // 쿼리문 템플릿 준비  
    try {  
        pstmt = con.prepareStatement(query); // 쿼리문 준비  
        pstmt.setString(1, idx); // 인파라미터 설정  
        rs = pstmt.executeQuery(); // 쿼리문 실행
```

게시물 조회를 위한 메서드

■ 모델 작성

예제 14-13] Java Resources/model2/mvcboard/MVCBoardDAO.java(계속)

```
// 주어진 일련번호에 해당하는 게시물의 조회수를 1 증가시킵니다. ②
public void updateVisitCount(String idx) {
    String query = "UPDATE mvcboard SET " ③
        + " visitcount=visitcount+1 "
        + " WHERE idx=?";

    try {
        psmt = con.prepareStatement(query);
        psmt.setString(1, idx);
        psmt.executeQuery();
    }
```

게시물 조회수 증가를 위한 메서드

■ 컨트롤러 작성

예제 14-14] Java Resources/model2/mvcboard/ViewController.java

```
@WebServlet("/mvcboard/view.do") ❶
public class ViewController extends HttpServlet {
    @Override
    protected void service(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        // 게시물 불러오기
        MVCBoardDAO dao = new MVCBoardDAO(); ❷
        String idx = req.getParameter("idx"); ❸
        dao.updateVisitCount(idx); // 조회수 1 증가 ❹
        MVCBoardDTO dto = dao.selectView(idx); ❺
        dao.close();
    }
}
```

모델 클래스에서 작성한 2개의 메서드를 호출해서 게시물 인출 및 조회수 증가

■ 컨트롤러 작성

예제 14-14] Java Resources/model2/mvcboard/ViewController.java(계속)

```
// 첨부 파일 확장자 추출 및 이미지 타입 확인 ⑦
String ext = null, fileName = dto.getSfile();
if(fileName!=null) {
    ext = fileName.substring(fileName.lastIndexOf(".") + 1);
}
String[] mimeStr = {"png", "jpg", "gif"}; ⑧
List<String> mimeTypeList = Arrays.asList(mimeStr); ⑨
boolean isImage = false;
if(mimeTypeList.contains(ext)) { ⑩
    isImage = true;
}

// 게시물(dto) 저장 후 뷰로 포워드
req.setAttribute("dto", dto); ⑪
req.setAttribute("isImage", isImage);
req.getRequestDispatcher("/14MVCBoard/View.jsp").forward(req, resp); ⑫
```

첨부 파일의 확장자를 추출해서 이미지인 경우 태그로 웹 브라우저에 보여줄 것임

■ 뷰 작성

예제 14-15] webapp/14MVCBoard/View.jsp

```
<!-- 게시물 정보 -->
<tr>
  <td>번호</td> <td>${ dto.idx }</td>
  <td>작성자</td> <td>${ dto.name }</td>
</tr>
<tr>
  <td>내용</td>
  <td colspan="3" height="100">${ dto.content }
    <c:if test="${ not empty dto.ofile and isImage eq true }">
      <br>
    </c:if> ②
```

각 항목은 EL을 통해 내용 출력

첨부파일은 이미지인 경우
태그로 화면에 표시

■ 뷰 작성

예제 14-15] webapp/14MVCBoard/View.jsp(계속)

```
<tr>
  <td>첨부 파일</td>
  <td>
    <c:if test="${ not empty dto.ofile }"> ③
      ${ dto.ofile }
      <a href="../../mvcboard/download.do?ofile=${ dto.ofile }&sfile=${ dto.s
file }&idx=${ dto.idx }">
        [다운로드]
      </a> ④
    </c:if>
  </td>
  <td>다운로드수</td>
  <td>${ dto.downcount }</td>
</tr>
```

첨부파일이 있는 경우 다운로드 링크 표시.

링크에는 원본 파일명, 저장된 파일명, 일련번호가 매개변수로 전달됨

■ 뷰 작성

예제 14-15] webapp/14MVCBoard/View.jsp(계속)

```
<tr>
  <td colspan="4" align="center">
    <button type="button" onclick="location.href='../mvcboard/pass.
do?mode=edit&idx=${ param.idx }';">
      수정하기
    </button>
    <button type="button" onclick="location.href='../mvcboard/pass.
do?mode=delete&idx=${ param.idx }';">
      삭제하기
    </button>
    <button type="button" onclick="location.href='../mvcboard/list.do';">
      목록 바로가기
    </button>
  </td>
</tr>
```


수정, 삭제, 목록 바로가기 버튼

수정, 삭제의 경우 패스워드 검증을
먼저 진행해야 하므로 동일한 요청
명을 사용하되 mode만 서로 다른
상태임

동작확인

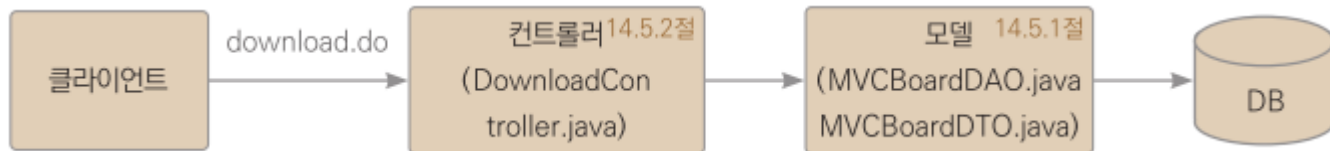
- 목록에서 원하는 글의 제목을 클릭
- 첨부한 파일이 이미지라면 아래와 같이 화면에 출력

파일 첨부형 게시판 - 상세 보기(View)

번호	110	작성자	성낙현
작성일	2022-10-18	조회수	44
제목	글쓰기 Test		
내용	<p>글을 작성해보겠습니다. 파일도 첨부하겠습니다. 단, 1MB 이하로 선택해주세요. 비밀번호는 1234로 지정하겠습니다.</p> 		
첨부파일	유겸이랑.png [다운로드]	다운로드수	4

[수정하기](#)
[삭제하기](#)
[목록 바로가기](#)

파일 다운로드 처리 프로세스



모델 작성

예제 14-16] Java Resources/model2/mvcboard/MVCBoardDAO.java

```
// 다운로드 횟수를 1 증가시킵니다.  
public void downCountPlus(String idx) { ❶  
    String sql = "UPDATE mvcboard SET "  
        + " downcount=downcount+1 " ❷  
        + " WHERE idx=? ";  
    try {  
        psmt = con.prepareStatement(sql);  
        psmt.setString(1, idx);  
        psmt.executeUpdate();  
    }  
    catch (Exception e) {}  
}
```

파일 다운로드 횟수를 1증가시킴



14.5 파일 다운로드

■ 컨트롤러 작성

- 13장의 Download.jsp와 거의 동일함

예제 14-17] Java Resources/fileupload/FileUtil.java

// 명시한 파일을 찾아 다운로드합니다.

```
public static void download(HttpServletRequest req, HttpServletResponse resp,
    String directory, String sfileName, String ofilename) { ❶
    String sDirectory = req.getServletContext().getRealPath(directory); ❷
    try {
        // 파일을 찾아 입력 스트림 생성 ❸
        File file = new File(sDirectory, sfileName);
        InputStream iStream = new FileInputStream(file);
        // 한글 파일명 깨짐 방지
        String client = req.getHeader("User-Agent"); ❹
        if (client.indexOf("WOW64") == -1) { ❺
            ofilename = new String(ofilename.getBytes("UTF-8"), "ISO-8859-1");
        }
        else {
            ofilename = new String(ofilename.getBytes("KSC5601"), "ISO-8859-1");
        }
    }
}
```

❶ 파일 다운로드를 위한 메서드

❷ 디렉토리의 물리적 경로 가져오기

❸ 경로로 파일 객체 생성

❹ 웹브라우저가 익스플로러인지 구분
하여 한글 깨짐 처리

■ 컨트롤러 작성

예제 14-17] Java Resources/fileupload/FileUtil.java

```
// 파일 다운로드용 응답 헤더 설정 ⑥
resp.reset();
resp.setContentType("application/octet-stream");
resp.setHeader("Content-Disposition",
               "attachment; filename=\"" + ofileName + "\"");
resp.setHeader("Content-Length", "" + file.length() );
// response 내장 객체로부터 새로운 출력 스트림 생성 ⑧
OutputStream oStream = resp.getOutputStream();

// 출력 스트림에 파일 내용 출력 ⑨
byte b[] = new byte[(int)file.length()];
int readBuffer = 0;
while ( (readBuffer = iStream.read(b)) > 0 ) {
    oStream.write(b, 0, readBuffer);
}
```

⑥ 다운로드 창을 띄우기 위한 응답헤더 설정. 여기서 원본 파일명으로 변경.

⑧ 출력 스트림을 생성 하여 로컬로 파일 다운로드



14.5 파일 다운로드

■ 컨트롤러 작성

예제 14-18] Java Resources/model2/mvcboard/DownloadController.java

```
@WebServlet("/mvcboard/download.do")
public class DownloadController extends HttpServlet{
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        // 매개변수 받기 ❶
        String ofile = req.getParameter("ofile"); // 원본 파일명
        String sfile = req.getParameter("sfile"); // 저장된 파일명
        String idx = req.getParameter("idx");      // 게시물 일련번호

        // 파일 다운로드
        FileUtil.download(req, resp, "/Uploads", sfile, ofile); ❷

        // 해당 게시물의 다운로드 수 1 증가
        MVCBoardDAO dao = new MVCBoardDAO();
        dao.downCountPlus(idx); ❸
        dao.close();
    }
}
```

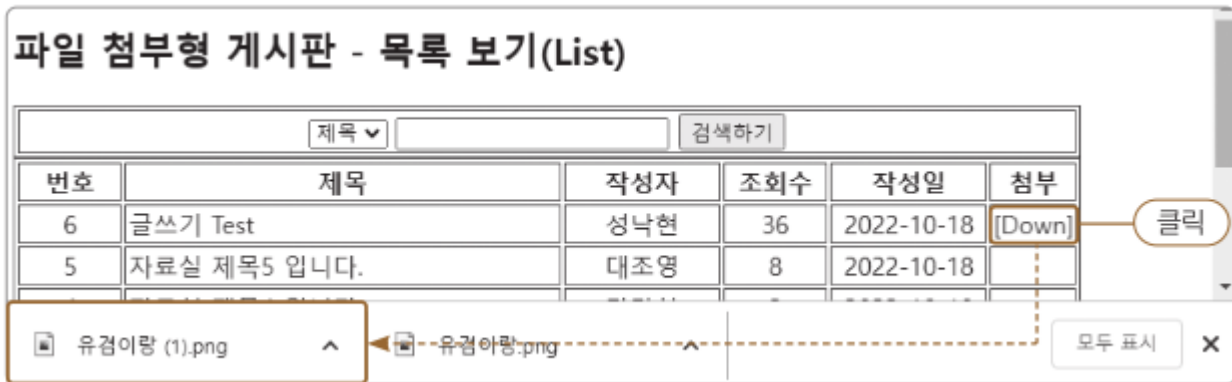
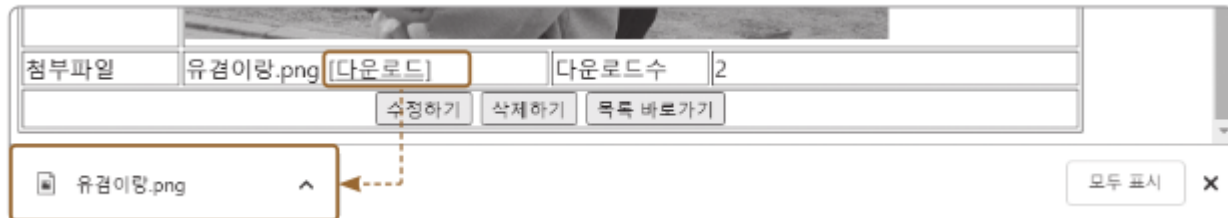
❶ 다운로드 링크로 전달된 파라미터 받기

❷ 파일 다운로드

❸ 다운로드 수 증가

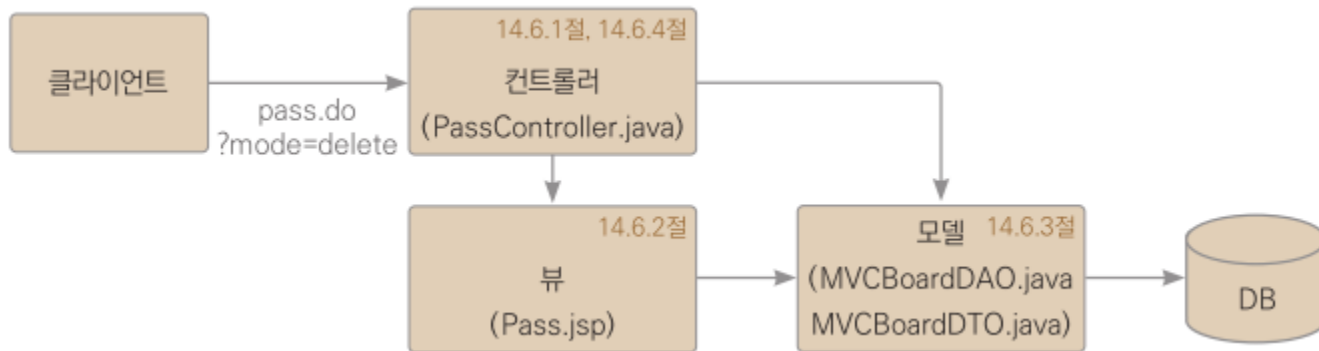
■ 동작 확인

- 상세보기 혹은 목록의 [다운로드] 링크를 클릭



■ 수정 및 삭제를 위한 요청명 확인

- 요청명 둘다 pass.do로 동일하고, 매개변수 중 mode의 값만 다름
- 즉 비밀번호 검증을 먼저 진행해야 함



▣ 요청명 / 서블릿 매핑

예제 14-19] Java Resources/model2/mvcboard/PassController.java

```
@WebServlet("/mvcboard/pass.do") ❶
public class PassController extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        req.setAttribute("mode", req.getParameter("mode")); ❷
        req.getRequestDispatcher("/14MVCBoard/Pass.jsp").forward(req, resp); ❸
    }
}
```

❶ 애너테이션으로 요청명과 매핑

❷ ❸ request 영역에 저장 후 포워드

▣ 뷰 작성

- 비밀번호 입력용 뷰로 쓰기 페이지를 조금 수정해서 제작

파일 첨부형 게시판 - 비밀번호 검증(Pass)

비밀번호

검증하기

RESET

목록 바로가기

▣ 뷰 작성

예제 14-20] webapp/14MVCBoard/Pass.jsp

```
<h2>파일 첨부형 게시판 - 비밀번호 검증(Pass)</h2>
<form name="writeFrm" method="post" action="../mvcboard/pass.do" onsubmit=
"return validateForm(this);"> ②
<input type="hidden" name="idx" value="${ param.idx }" />
<input type="hidden" name="mode" value="${ param.mode }" /> ③
<table border="1" width="90%">
  <tr>
    <td>비밀번호</td>
    <td>
      <input type="password" name="pass" style="width:100px;" />
    </td>
  </tr>
</table>
```

② <form> 태그

③ 게시물의 일련번호와 수정, 삭제를 판단하기 위한 mode를 hidden으로 세팅

■ 모델 작성

예제 14-21] Java Resources/model2/mvcboard/MVCBoardDAO.java

// 입력한 비밀번호가 지정한 일련번호의 게시물의 비밀번호와 일치하는지 확인합니다.

```
public boolean confirmPassword(String pass, String idx) {  
    boolean isCorr = true;  
    try {  
        String sql = "SELECT COUNT(*) FROM mvcboard WHERE pass=? AND idx=?"; ①  
        pstmt = con.prepareStatement(sql);  
        pstmt.setString(1, pass);  
        pstmt.setString(2, idx);  
        rs = pstmt.executeQuery();  
        rs.next();  
        if (rs.getInt(1) == 0) {  
            isCorr = false;  
        }  
    }  
    catch (Exception e) {  
        isCorr = false; ②  
    }  
}
```

① 일련번호와 패스워드에 일치하는
게시물이 있는지 확인하는 쿼리문

② 일치하는 게시물이 없거나 예외
발생 시에는 검증실패로 처리

■ 모델 작성

예제 14-21] Java Resources/model2/mvcboard/MVCBoardDAO.java(계속)

```
// 지정한 일련번호의 게시물을 삭제합니다.  
public int deletePost(String idx) {  
    int result = 0;  
    try {  
        String query = "DELETE FROM mvcboard WHERE idx=?";  
        pstmt = con.prepareStatement(query);  
        pstmt.setString(1, idx);  
        result = pstmt.executeUpdate(); ③  
    }  
    catch (Exception e) {  
        System.out.println("게시물 삭제 중 예외 발생");  
        e.printStackTrace();  
    }  
    return result;  
}  
}
```

③ 일련번호를 통해 게시물을 삭제하는 delete 쿼리문 작성 및 실행

■ 컨트롤러 작성

예제 14-22] Java Resources/fileupload/FileUtil.java

// 지정한 위치의 파일을 삭제합니다.

```
public static void deleteFile(HttpServletRequest req,
    String directory, String filename) {
    String sDirectory = req.getServletContext().getRealPath(directory); ❶
    File file = new File(sDirectory + File.separator + filename); ❷
    if (file.exists()) { ❸
        file.delete(); ❹
    }
}
```

❶ 해당 디렉토리에 파일이 존재하는 경우 삭제

예제 14-23] Java Resources/model2/mvcboard/PassController.java

```
@WebServlet("/mvcboard/pass.do")
public class PassController extends HttpServlet {
    ... 생략 ...
}
```

패스워드 검증 페이지의 매핑 처리

■ 컨트롤러 작성

예제 14-23] Java Resources/model2/mvcboard/PassController.java(계속)

```
// 매개변수 저장 ②
String idx = req.getParameter("idx");
String mode = req.getParameter("mode");
String pass = req.getParameter("pass");

// 비밀번호 확인
MVCBoardDAO dao = new MVCBoardDAO();
boolean confirmed = dao.confirmPassword(pass, idx); ③
dao.close();

if (confirmed) { // 비밀번호 일치 ④
    if (mode.equals("edit")) { // 수정 모드 ⑤
        HttpSession session = req.getSession(); ⑥
        session.setAttribute("pass", pass); ⑦
        resp.sendRedirect("../mvcboard/edit.do?idx=" + idx); ⑧
    }
}
```

③ 비밀번호 검증

④ 비밀번호가 일치하고

⑤ 수정 모드인 경우

⑥ 세션 내장객체를 얻어와서

⑦ 세션영역에 비밀번호를 저장

⑧ 수정 페이지로 이동

■ 컨트롤러 작성

예제 14-23] Java Resources/model2/mvcboard/PassController.java(계속)

```
        else if (mode.equals("delete")) { // 삭제 모드 ⑨
            dao = new MVCBoardDAO();
            MVCBoardDTO dto = dao.selectView(idx); ⑩
            int result = dao.deletePost(idx); // 게시물 삭제 ⑪
            dao.close();
            if (result == 1) { // 게시물 삭제 성공 시 첨부 파일도 삭제 ⑫
                String saveFileName = dto.getSfile();
                FileUtil.deleteFile(req, "/Uploads", saveFileName);
            }
            JSFunction.alertLocation(resp, "삭제되었습니다.",
                                    "../mvcboard/list.do"); ⑬
        }
    }
    else { // 비밀번호 불일치 ⑭
        JSFunction.alertBack(resp, "비밀번호 검증에 실패했습니다.");
    }
}
}
```

⑨ 삭제 모드인 경우

⑩ 기존 게시물의 내용을 가져옴

⑪ 게시물 삭제

⑫ 첨부파일도 같이 삭제

비밀번호를 session 영역에 저장한 이유

⑦에서 비밀번호를 session 영역에 저장한 이유가 무엇일까요? 수정하기 페이지의 요청명은 “edit.do?idx=일련번호” 형태입니다. 그런데 만약 사용자가 이 URL 패턴을 이미 알고 있다면 비밀번호 검증 없이도 곧바로 수정하기 페이지에 접속할 수 있게 됩니다.

비밀번호 검증을 건너 뛰고 수정하기 페이지에 접속했다면 게시물이 수정되면 안 됩니다. 따라서 검증이 완료된 비밀번호를 session 영역에 저장해놓고, 수정 시 저장된 비밀번호가 없다면 정상적인 경로로 접속하지 않은 것으로 판단하는 것입니다. 수정하기용 컨트롤러를 작성하는 14.7.5절에서 다시 설명하겠습니다.

동작 확인

파일 첨부형 게시판 - 비밀번호 검증(Pass)

비밀번호

1

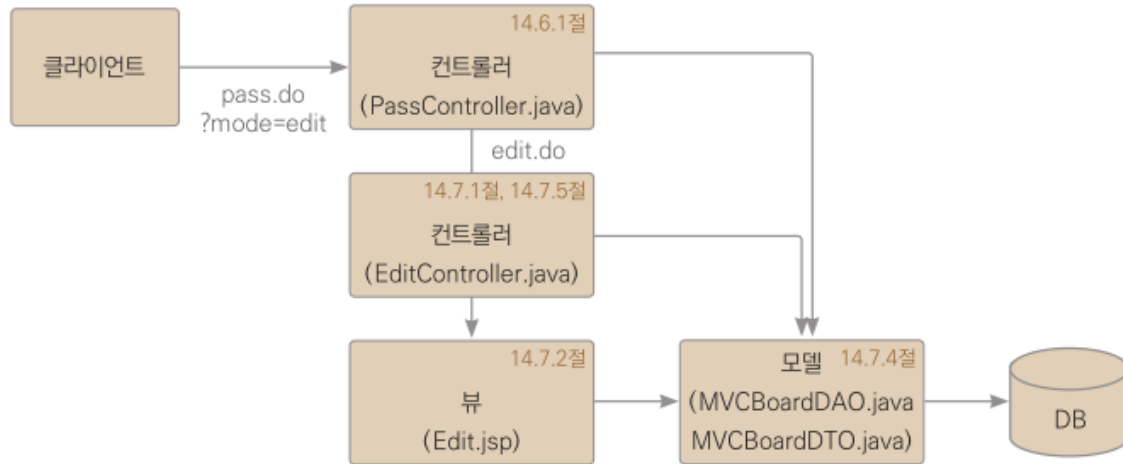
2

검증하기 RESET 목록 바로가기



■ 수정하기 처리 프로세스

- 비밀번호 검증은 앞에서 진행했으므로 검증 통과 후의 로직만 구현





14.7 수정하기

■ 요청명 / 서블릿 매핑

예제 14-24] Java Resources/model2/mvcboard/EditController.java

```
@WebServlet("/mvcboard/edit.do")
@MultipartConfig(①
    maxFileSize = 1024 * 1024 * 1,
    maxRequestSize = 1024 * 1024 * 10
)
public class EditController extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        String idx = req.getParameter("idx"); ②
        MVCBoardDAO dao = new MVCBoardDAO();
        MVCBoardDTO dto = dao.selectView(idx); ③
        req.setAttribute("dto", dto); ④
        req.getRequestDispatcher("/14MVCBoard/Edit.jsp").forward(req, resp); ⑤
    }
}
```

① 서블릿 매핑과 파일 업로드를 위한 Multipart 설정

② 일련번호를 파라미터로 받음

③ 게시물을 인출한 후

④ request 영역에 저장하고

⑤ View로 포워드

■ 뷰 작성

예제 14-25] webapp/14MVCBoard/Edit.jsp

```
<h2>파일 첨부형 게시판 - 수정하기(Edit)</h2>
<form name="writeFrm" method="post" enctype="multipart/form-data"
      action="../mvcboard/edit.do" onsubmit="return validateForm(this);"> ②
  <input type="hidden" name="idx" value="${ dto.idx }"/>
  <input type="hidden" name="prevOfile" value="${ dto.ofile }" /> ③
  <input type="hidden" name="prevSfile" value="${ dto.sfile }" />
  <td>제목</td>
  <td>
    <input type="text" name="title"
           style="width:90%;" value="${ dto.title }" />
  </td>
</tr>
<tr>
  <td>내용</td>
  <td>
    <textarea name="content" style="width:90%;height:100px;">${ dto.
content }</textarea>
  </td>
```

- ② 수정을 위한 <form> 태그 정의.
action 속성만 변경되고 글쓰기와 동일
- ③ 일련번호와 파일명을 hidden 입력
상자로 세팅

나머지 부분은 글쓰기 폼에 제목, 내용
등을 채움

■ 중간 동작 확인

- 비밀번호 입력 후 일치하면 수정페이지로 이동

파일 첨부형 게시판 - 비밀번호 검증(Pass)

비밀번호	<input type="password" value="...."/>
<input type="button" value="검증하기"/>	<input type="button" value="RESET"/> <input type="button" value="목록 바로가기"/>

파일 첨부형 게시판 - 수정하기(Edit)

작성자	<input type="text" value="김유신"/>
제목	<input type="text" value="자르실 제목1입니다."/>
내용	<div></div>
첨부 파일	<input type="button" value="파일 선택"/> <input type="button" value="선택된 파일 없음"/>
<input type="button" value="작성 완료"/> <input type="button" value="RESET"/> <input type="button" value="목록 바로가기"/>	

■ 모델 작성

예제 14-26] Java Resources/model2/mvcboard/MVCBoardDAO.java

```
// 게시글 데이터를 받아 DB에 저장되어 있던 내용을 갱신합니다(파일 업로드 지원).
public int updatePost(MVCBoardDTO dto) { ❶
    int result = 0;
    try {
        // 쿼리문 템플릿 준비
        String query = "UPDATE mvcboard"
            + " SET title=?, name=?, content=?, ofile=?, sfile=? "
            + " WHERE idx=? and pass=?"; ❷

        // 쿼리문 준비
        pstmt = con.prepareStatement(query);
        pstmt.setString(6, dto.getIdx());
        pstmt.setString(7, dto.getPass());

        // 쿼리문 실행
        result = pstmt.executeUpdate(); ❸
    }
}
```

❶ 게시물 수정을 위한 메서드

❷ update 쿼리문 작성

❸ 인파라미터 설정 및 쿼리실행

■ 컨트롤러 작성

예제 14-27] Java Resources/model2/mvcboard/EditController.java

```
@WebServlet("/mvcboard/edit.do")
@MultipartConfig(
    maxFileSize = 1024 * 1024 * 1,
    maxRequestSize = 1024 * 1024 * 10
)
public class EditController extends HttpServlet {
    private static final long serialVersionUID = 1L;

    @Override
    protected void doPost(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        // 1. 파일 업로드 처리 =====
        // 업로드 디렉터리의 물리적 경로 확인 ❶
        String saveDirectory = req.getServletContext().getRealPath("/Uploads");
```

서블릿 매핑과 Multipart 설정

❶ 디렉토리의 물리적 경로 확인

■ 컨트롤러 작성

예제 14-27] Java Resources/model2/mvcboard/EditController.java

```
// 파일 업로드 ②
String originalFileName = "";
try {
    originalFileName = FileUtil.uploadFile(req, saveDirectory);
}
catch (Exception e) {
    JSFunction.alertBack(resp, "파일 업로드 오류입니다.");
    return;
}

// 2. 파일 업로드 외 처리 =====
// 수정 내용을 매개변수에서 얻어옴 ③
String idx = req.getParameter("idx");
String prev0file = req.getParameter("prev0file");
String prevSfile = req.getParameter("prevSfile");
```

② 파일 업로드

③ 전송된 품값 받기

■ 컨트롤러 작성

예제 14-27] Java Resources/model2/mvcboard/EditController.java

```
// 비밀번호는 session에서 가져옴 ④
HttpSession session = req.getSession();
String pass = (String)session.getAttribute("pass");

// DTO에 저장 ⑤
MVCBoardDTO dto = new MVCBoardDTO();
dto.setIdx(idx);

// 원본 파일명과 저장된 파일 이름 설정 ⑥
if (originalFileName != "") {
    String savedFileName = FileUtil.renameFile(saveDirectory,
        originalFileName);

    dto.setOfile(originalFileName); // 원래 파일 이름
    dto.setSfile(savedFileName); // 서버에 저장된 파일 이름

    // 기존 파일 삭제 ⑦
    FileUtil.deleteFile(req, "/Uploads", prevSfile);
}
```

④ 비밀번호 검증시 세션 영역에 저장했던 값 가져오기

⑤ DTO에 값 저장

⑥ 수정시 첨부한 파일이 있으면 저장된 파일 이름을 변경

⑦ 기존에 등록된 파일은 삭제

■ 컨트롤러 작성

예제 14-27] Java Resources/model2/mvcboard/EditController.java

```
else {  
    // 첨부 파일이 없으면 기존 이름 유지 ⑧  
    dto.setOfile(prevOfile);  
    dto.setSfile(prevSfile);  
}  
  
// DB에 수정 내용 반영 ⑨  
MVCBoardDAO dao = new MVCBoardDAO();  
int result = dao.updatePost(dto);  
dao.close();  
  
// 성공 or 실패?  
if (result == 1) { // 수정 성공 ⑩  
    session.removeAttribute("pass");  
    resp.sendRedirect("../mvcboard/view.do?idx=" + idx);  
}  
else { // 수정 실패 ⑪  
    JSFunction.alertLocation(resp, "비밀번호 검증을 다시 진행해주세요.",  
        "../mvcboard/view.do?idx=" + idx);  
}
```

⑧ 수정시 첨부한 파일이 없다면 hidden 으로 설정했던 값을 통해 기존 파일명 유지

⑨ update 쿼리문 실행

⑩ 성공 혹은 실패에 따른 페이지 이동

■ 동작 확인

파일 첨부형 게시판 - 수정하기(Edit)

작성자	김수정
제목	첫 번째 게시물을 수정합니다.
내용	첫 번째 게시물을 수정합니다. 파일도 첨부합니다.
첨부 파일	<div>파일 선택</div> <div>2</div> <div>3</div> <div>김이랑엄마랑.png</div>

작성 완료

RESET

목록 바로가기

파일 첨부형 게시판 - 상세 보기(View)

번호	133	작성자	김수정
작성일	2022-10-18	조회수	20
제목	첫 번째 게시물을 수정합니다.		
내용	첫 번째 게시물을 수정합니다. 파일도 첨부합니다.		

첨부파일

김이랑엄마랑.png

[다운로드]

다운로드수

2

수정하기

삭제하기

목록 바로가기

■ 핵심요약

- 비회원제 게시판에서는 비밀번호가 식별자 역할을 하게 되므로, 인증이 필요한 모든 곳에서 아이디와 같이 사용됨
- 서블릿을 이용할 때는 기능별 요청명을 미리 정의해두는 것이 좋음. 만약 애너테이션으로 매핑을 한다면, 요청명만으로 컨트롤러를 찾을 수 있도록 연관있는 명칭을 사용해야 함
- 모델1 방식과 모델2 방식을 비교해보면서 본인의 업무에 무엇을 선택하면 좋을지 스스로 판단해 보세요.

