



spring 8강-쇼핑몰 관리자

양 명 속

[now4ever7@gmail.com]



목차

- 관리자 등록
- 상품 등록
- 상품 목록



authority table – 권한 테이블

--권한 테이블

--drop table authority cascade constraint;

create table authority

(

authCode varchar2(30) primary key , --권한코드

authName varchar2(100) not null , --권한명

authDesc varchar2(200) null , --권한설명

authLevel number(3), --레벨 순위 --1(ADMIN), 2(ASSISTANT), 3(STAFF)

regdate date default sysdate

);

INSERT INTO authority VALUES ('ADMIN','관리자', 'manager',1, SYSDATE);

INSERT INTO authority VALUES ('ASSISTANT','부관리자', 'assistant manager',2, SYSDATE);

INSERT INTO authority VALUES ('STAFF','스태프', 'staff',3, SYSDATE);



managers table – 관리자 테이블

```
--drop table managers cascade constraint;
```

```
create table managers
```

```
(
```

```
  no          number      primary key,
```

```
  userid      varchar2(20) unique not null,
```

```
  name        varchar2(20) not null,
```

```
  pwd         varchar2(15) not null,
```

```
  authCode    varchar2(30) not null, --권한코드 ADMIN,ASSISTANT,STAFF
```

```
  regdate     date        default sysdate,
```

```
  constraint managers_fk1 foreign key(authCode) references authority(authCode)
```

```
);
```

```
--drop sequence managers_seq;
```

```
create sequence managers_seq
```

```
increment by 1
```

```
start with 1
```

```
nocache;
```

```
INSERT INTO managers VALUES (managers_seq.nextval,'admin','김관리','1', 'ADMIN', SYSDATE);
```

관리자 메뉴

관리자등록

상품등록

상품목록

주문 관리

일별매출

월별매출

기간별매출

관리자 등록

관리자명

관리자ID

중복확인

비밀번호

비밀번호 확인

레벨

선택하세요 ▾

선택하세요

관리자

부관리자

스태프

등록

adminLogin.jsp

```
<div class="simpleForm">
  <form name="frmLogin" id="frmLogin" method="post"
    action="<c:url value='/admin/login/adminLogin.do'/">">
    <fieldset>
      <legend>관리자 로그인</legend>
      <div>
        <label for="userid" class="label">아이디</label>
        <input type="text" name="userid"
          id="userid"
          value="${cookie.admin_ck_userid.value}">
      </div>
      <div>
        <label for="pwd" class="label">비밀번호</label>
        <input type="password" name="pwd"
          id="pwd">
      </div>
      <div class="align_center">
        <input type="submit" value="로그인">
        <input type="checkbox" name="chkSave"
          id="chkSave"
          <c:if test="${!empty cookie.admin_ck_userid}">
            checked
          </c:if>
        <label for="chkSave">아이디 저장하기</label>
      </div>
    </fieldset>
  </form>
</div>
```



adminTop.jsp

```
<div id="headerRight">
<ul class="views">
  <c:if test="${!empty sessionScope.adminUserid}">
    <li><span style="color:gray;font-size: 1em">
      관리자 (${sessionScope.adminUserName}) 님 </span></li>
    <li><a href="<c:url value='/admin/login/logout.do' />">로그아웃</a> | </li>
  </c:if>
  <li><a href="<c:url value='/index.do' />">쇼핑몰 가기</a></li>
</ul>
</div>
```



AdminController

```
package com.herb.app.controller;
```

```
@Controller
```

```
@RequestMapping("/admin")
```

```
public class AdminController {
```

```
    private static final Logger logger
```

```
        =LoggerFactory.getLogger(AdminController.class);
```

```
    @Autowired
```

```
    private ManagerService managerService;
```

```
    @RequestMapping("/adminMain.do")
```

```
    public void adminMain(){
```

```
        logger.info("관리자 메인 페이지 보여주기");
```

```
    }
```




AdminController

```
@RequestMapping(value="/manager/join.do", method=RequestMethod.GET)
public String join_get(Model model){
    logger.info("관리자 등록화면 보여주기");

    List<Map<String, Object>> authList
        =managerService.selectAuthority();
    logger.info("관리자 등록 화면 - 권한 조회 결과, authList.size()={},",
        authList.size());

    model.addAttribute("authList", authList);

    return "admin/manager/register";
}
```

```
@RequestMapping(value="/manager/join.do", method=RequestMethod.POST)
public String join_post(@ModelAttribute ManagerVO vo, Model model){
    logger.info("관리자 등록 처리, 파라미터 vo={}", vo);

    int cnt = managerService.managerInsert(vo);
    logger.info("관리자 등록 결과, cnt={}", cnt);

    String msg="", url="/admin/manager/join.do";
    if(cnt>0){
        msg="관리자 등록되었습니다";
    }else{
        msg="관리자 등록 실패";
    }

    model.addAttribute("msg", msg);
    model.addAttribute("url", url);

    return "common/message";
}
```

```

@RequestMapping("/manager/checkUserId.do")
public String checkId(@RequestParam String userid, Model model){
    logger.info("아이디 중복확인, 파라미터 userid={}", userid);

    int result =0;
    if(userid!=null && !userid.isEmpty()){
        result =managerService.duplicateUserId(userid);
        logger.info("아이디 중복확인 결과, result={}", result);
    }

    model.addAttribute("result", result);
    model.addAttribute("EXIST_ID", ManagerService.EXIST_ID);
    model.addAttribute("NONE_EXIST_ID", ManagerService.NONE_EXIST_ID);

    return "admin/manager/checkUserId";
}

@RequestMapping(value="/login/login.do", method=RequestMethod.GET)
public String login_get(){
    logger.info("로그인 화면 보여주기");

    return "admin/login/login";
}

```

```

@RequestMapping(value="/login/login.do", method=RequestMethod.POST)
public String login_post(@RequestParam String userid,@RequestParam String pwd,
    @RequestParam(required=false) String chkSaveld,
    Model model, HttpServletRequest request, HttpServletResponse response){
    logger.info("로그인 처리, 파라미터 userid={},pwd={}", userid,pwd);
    logger.info("파라미터 chkSaveld={}", chkSaveld);

    String msg="", url="/admin/login/login.do";
    int result =managerService.loginCheck(userid, pwd);
    if(result==ManagerService.LOGIN_OK){
        //로그인 성공
        //관리자 정보 조회하기
        ManagerVO vo =managerService.selectManagerByUserid(userid);
        logger.info("관리자 로그인-관리자정보 조회 결과, vo={}", vo);

        msg=vo.getName() + "님 관리자 로그인되었습니다.";
        url="/admin/adminMain.do";

        //세션에 저장
        HttpSession session=request.getSession();
        session.setAttribute("adminUserid", userid);
        session.setAttribute("adminUserName", vo.getName());
        session.setAttribute("adminAuthCode", vo.getAuthCode());

        //쿠키에 저장
        Cookie ck = new Cookie("ck_admin_userid", userid);
    }
}

```

```

        if(chkSaveId!=null){
            ck.setMaxAge(1000*24*60*60);
            ck.setPath("/");
            response.addCookie(ck);
        }else{
            ck.setMaxAge(0);
            ck.setPath("/");
            response.addCookie(ck);
        }
    }else if(result==ManagerService.PWD_DISAGREE){
        msg="비밀번호가 일치하지 않습니다";
    }else if(result==ManagerService.ID_NONE){
        msg="아이디가 존재하지 않습니다";
    }else{
        msg="로그인 처리 에러";
    }
    model.addAttribute("msg", msg);
    model.addAttribute("url", url);

    return "common/message";
}

```

```
@RequestMapping("/login/logout.do")
public String logout(HttpSession session, Model model){
    logger.info("관리자 로그아웃 처리");

    session.removeAttribute("adminUserId");
    session.removeAttribute("adminUserName");
    session.removeAttribute("adminAuthCode");

    model.addAttribute("msg", "관리자 로그아웃되었습니다");
    model.addAttribute("url", "/admin/login/login.do");

    return "common/message";
}
}
```



manager.xml

```
<mapper namespace="config.mybatis.mapper.oracle.manager">
    <insert id="managerInsert" parameterType="ManagerVO">
        <selectKey keyProperty="no" resultType="int" order="BEFORE">
            select managers_seq.nextval from dual
        </selectKey>
        insert into managers(no, userid, name, pwd, authCode)
        values(#{no}, #{userid}, #{name}, #{pwd}, #{authCode})
    </insert>

    <select id="duplicateUserId" parameterType="string" resultType="int" >
        select count(*) from managers
        where userid=#{userid}
    </select>

    <select id="selectPwdByUserId" parameterType="string" resultType="string">
        select pwd from managers where userid=#{userid}
    </select>

    <select id="selectAuthority" resultType="map">
        select * from authority order by authLevel
    </select>
```



manager.xml

```
<select id="selectManagerByUserId" parameterType="string"
    resultType="ManagerVO" >
    select * from managers
    where userid=#{userid}
</select>

</mapper>
```




AdminLoginInterceptor

@Component

```
public class AdminLoginInterceptor extends HandlerInterceptorAdapter {  
    private static final Logger logger  
        =LoggerFactory.getLogger(AdminLoginInterceptor.class);
```

@Override

```
public void afterCompletion(HttpServletRequest request, HttpServletResponse response,  
Object handler, Exception ex)throws Exception {  
    logger.info("view 생성 후 호출-afterCompletion()");  
}
```

@Override

```
public void postHandle(HttpServletRequest request, HttpServletResponse response, Object  
handler, ModelAndView modelAndView) throws Exception {  
    logger.info("컨트롤러 수행 후 호출 - postHandle()");  
}
```

@Override

```
public boolean preHandle(HttpServletRequest request,  
    HttpServletResponse response, Object handler)  
    throws Exception {  
    logger.info("컨트롤러 수행 전 호출 - preHandle()");
```

//uri 읽어오기

```
String uri = request.getRequestURI();
logger.info("uri={}",uri);
//=>http://localhost:9090/springherb/admin/login/adminLogin.do
//=> uri => /springherb/admin/login/adminLogin.do
```

```
//관리자 로그인 페이지는 걸러낸다
if(uri.indexOf("/admin/login/adminLogin.do")!= -1){
    return true; //에러 처리하지 않는다
}
```

```
HttpSession session = request.getSession();
String adminUserId = (String) session.getAttribute("adminUserId");
```

```
//관리자 로그인되지 않은 경우 에러 처리
if(adminUserId==null || adminUserId.isEmpty()){
    response.setContentType("text/html;charset=utf-8");
    PrintWriter out = response.getWriter();
    out.println("<script>");
    out.println("alert('먼저 관리자 로그인하세요');");
    out.println("location.href='"+request.getContextPath()+"/admin/login/login.do'");
    out.println("</script>");
    return false;
}else{
    return true;
}
```

```
}
```



servlet-context.xml

```
<interceptors>
```

```
    <interceptor>
```

```
        <mapping path="/shop/cart/*" />
```

```
        <mapping path="/shop/order/*" />
```

```
        <mapping path="/member/memberEdit.do" />
```

```
        <mapping path="/member/memberOut.do" />
```

```
        <beans:ref bean ="loginInterceptor" />
```

```
    </interceptor>
```

```
    <interceptor>
```

```
        <mapping path="/admin/*" />
```

```
        <mapping path="/admin/*/*" />
```

```
        <beans:ref bean ="adminLoginInterceptor" />
```

```
    </interceptor>
```

```
</interceptors>
```

```

<%@ include file="../../inc/adminTop.jsp" %>
<script type="text/javascript" src="<c:url value='/js/member2.js' />"></script>
<script src="<c:url value='/jquery/jquery-1.11.2.js' />" type="text/javascript"></script>
<script type="text/javascript">
    $(document).ready(function() {
        $('#wr_submit').click(function () {
            if($("#name").val().length<1){
                alert("이름을 입력하세요");
                $("#name").focus();
                return false;
            }else if(!validate_userid($("#userid").val())){
                alert("아이디는 숫자나 영문 대소문자, 밑줄문자(_)만 가능합니다");
                $("#userid").focus();
                return false;
            }else if($("#pwd").val().length<1){
                alert("비밀번호를 입력하세요");
                $("#pwd").focus();
                return false;
            }else if($("#pwd").val() != $("#pwd2").val()){
                alert("비밀번호가 일치하지 않습니다");
                $("#pwd2").focus();
                return false;
            }
        });
    });
</script>
<div class="divForm">
<form name="frm1" method="post"
action="<c:url value='/admin/manager/join.do' />" >
<fieldset>

```

```

function validate_userid(userid){
    var pattern = new RegExp(/^[a-zA-Z0-9_]+$ /g);
    return pattern.test(userid); //true이면 정규식을 만족,
                                //false이면 에러

    /*
    정규식 /^[a-zA-Z0-9_]+$ /g
    a에서 z 사이의 문자, A~Z사이의 문자, 0 에서 9사이
    의 숫자나 _로 시작하거나 끝나야 한다는 의미
    닫기 대괄호(]) 뒤의 + 기호는 이 패턴이 한 번 또는
    그 이상 반복된다는 의미
    */
}

```

```

<legend>관리자 등록</legend>
<div>
    <label for="name">관리자명</label>
    <input type="text" name="name" id="name" style="ime-mode:active">
</div>
<div>
    <label for="userid">관리자ID</label>
    <input type="text" name="userid" id="userid" style="ime-mode:inactive">&nbsp;
    <input type="button" value="중복확인" id="btnChkId" title="새창열림">
</div>
<div>
    <label for="pwd">비밀번호</label>
    <input type="Password" name="pwd" id="pwd">
</div>
<div>
    <label for="pwd2">비밀번호 확인</label>
    <input type="Password" name="pwd2" id="pwd2">
</div>
<div>
    <label for="authCode">레벨</label>&nbsp;
    <select name="authCode" id="authCode">
        <option value="">선택하세요</option>
        <!-- 반복 시작 -->
        <c:forEach var="map" items="${authList }">
            <option value="${map['AUTHCODE']}">${map['AUTHNAME']}</option>
        </c:forEach>
        <!-- 반복 끝 -->
    </select>
</div>

```

```
<div class="center">  
    <input type="submit" id="wr_submit" value="등록">  
</div>  
</fieldset>
```

```
<input type="hidden" name="chkId" id="chkId">
```

```
</form>  
</div>  
</article>
```

```
<%@ include file="../../inc/adminBottom.jsp" %>
```

상품 등록

상품 등록

관리자 메뉴

관리자등록

상품등록

상품목록

주문 관리

일별매출

월별매출

기간별매출

상품명

카테고리

선택하세요

허브자

아로마오일

허브향초

상품이미지

찾아보기...

가격

요약설명

상세설명

적립금

제조회사

등록

취소

전체 상품 12 건 입니다.

이벤트 상품 조회

▼

검색

<input type="checkbox"/>	상품이미지	상품명	가격	등록일	수정	삭제
<input type="checkbox"/>		페파민트 향초	19,000 원	2014-05-20	수정	삭제
<input type="checkbox"/>		자스민허브차	9,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		로즈마리허브차	8,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		라벤다허브차	7,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		자스민	9,000 원	2014-05-17	수정	삭제

[1] 2 [3]

선택한 상품 삭제

선택한 상품을

▼

등록

상품 목록-이벤트 상품 조회

상품 목록

BEST 상품 8건입니다.

이벤트 상품 조회 인기상품 ▼ 검색

<input type="checkbox"/>	상품이미지	상품명	가격	등록일	수정	삭제
<input type="checkbox"/>		라벤다	7,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		로즈마리	8,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		자스민	9,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		라벤다허브차	7,000 원	2014-05-17	수정	삭제
<input type="checkbox"/>		로즈마리허브차	8,000 원	2014-05-17	수정	삭제



Spring - 커맨드 객체로 List 받기



커맨드 객체로 List 받기

- 스프링 MVC는 List 타입의 프로퍼티에 대한 바인딩도 처리해 줌
- 예) ProductCommand 클래스는 productVo 목록을 갖는 List 타입의 pdItems 프로퍼티를 갖고 있다

```
public class ProductCommand {  
    private List<ProductVO> pdItems;  
    private String address;  
  
    public List<ProductVO> getPdItems() {  
        return pdItems;  
    }  
  
    public void setPdItems(List<ProductVO> pdItems) {  
        this.pdItems = pdItems;  
    }  
  
    public String getAddress() {  
        return address;  
    }  
    public void setAddress(String address) {  
        this.address = address;  
    }  
}
```

커맨드 객체로 List 받기

- List 타입의 프로퍼티에 값을 전달하고 싶으면, 폼에서 "프로퍼티명[인덱스].프로퍼티" 같이 입력 폼의 이름을 구성하면 됨
- 예) 위 코드의 pdItems 프로퍼티에 값을 전달하고 싶으면 아래 코드와 같이 폼을 구성하면 됨

```
<form method="post">
[상품1]
상품번호 - <input type="text" name="pdItems[0].productNo" />
상품명 - <input type="text" name="pdItems[0].productName" />
가격- <input type="text" name="pdItems[0].sellPrice" />
<br/>
[상품2]
상품번호 - <input type="text" name="pdItems[1].productNo" />
상품명 - <input type="text" name="pdItems[1].productName" />
가격- <input type="text" name="pdItems[1].sellPrice" />
...
주소- <input type="text" name="address" />
<input type="submit">
</form>
```

- 품의 <input>이나 <select> 등의 name에 인덱스 값을 포함시키면, List 타입의 프로퍼티에 값을 전달받을 수 있음



커맨드 객체로 List 받기

- 컨트롤러 코드에서는 다음과 같이 커맨드 객체를 @RequestMapping 메서드에 지정해주기만 하면 됨

```
@Controller
@RequestMapping("/product/product.do")
public class ProductController {

    @RequestMapping(method = RequestMethod.POST)
    public String pdSubmit(@ModelAttribute ProductCommand pdCommand) {
        return "product/productCompletion";
    }
}
```

```
<mapper namespace="config.mybatis.mapper.oracle.Product">
  <insert id="productInsert" parameterType="productVo">
    <selectKey keyProperty="productNo" resultType="int" order="BEFORE">
      select products_seq.nextval from dual
    </selectKey>
    insert into products(productno, categoryno, productname, sellprice, company, imageurl,
      explains, description, mileage)
    values(#{productNo}, #{categoryNo}, #{productName}, #{sellPrice}, #{company},
      #{imageURL}, #{explains}, #{description}, #{mileage})
  </insert>
  <!-- 해당 페이지의 레코드만 조회하도록 변경 -->
  <select id="productList" resultType="productVo" parameterType="eventProductVO">
    SELECT *
    FROM (
      SELECT ROWNUM RNUM, ALL_LIST.*
      FROM (
        <choose>
          <when test="eventName != null and eventName !=" ">
            select * from productEventView where eventname=#{eventName}
          </when>
          <otherwise>
            select * from products order by productNo desc
          </otherwise>
        </choose>
      ) ALL_LIST
    )
    <![CDATA[
      WHERE RNUM > #{firstRecordIndex}
      AND RNUM <= #{firstRecordIndex} + #{recordCountPerPage} ]]>
  </select>
```

```

<select id="getTotalRecord" resultType="Integer" parameterType="eventProductVO">
    <choose>
        <when test="eventName != null and eventName !=" ">
            select count(*) from productEventView
            where eventname=#{eventName}
        </when>
        <otherwise>
            select count(*) from products
        </otherwise>
    </choose>
</select>

```

```

<delete id="productDelete" parameterType="int">
    delete from products where productNo=#{productNo}
</delete>

```

```

<select id="eventProductSelect" parameterType="eventProductVO"
    resultType="int">
    select count(*) from eventproduct
    where productno=#{productNo} and eventName=#{eventName}
</select>

```

```

<insert id="eventProductInsert" parameterType="eventProductVO" >
    <selectKey keyProperty="eventProductNo" resultType="int" order="BEFORE">
        select eventproduct_seq.nextval from dual
    </selectKey>
    insert into eventproduct
    values(#{eventProductNo}, #{productNo}, #{eventName})
</insert>

```

```

</mapper>

```

SearchVO

/** 검색 정보를 담고 있는 Bean, 페이징 처리 관련 변수 포함*/

```
public class SearchVO {
```

```
    /** 검색조건 */
```

```
    private String searchCondition = "";
```

```
    /** 검색키워드 */
```

```
    private String searchKeyword = "";
```

```
    /** 검색 사용여부 */
```

```
    private String searchUseYn = "";
```

```
    /** 현재 페이지 */
```

```
    private int currentPage = 1;
```

```
    /**블럭당 보여질 페이지 수, 페이지 사이즈 */
```

```
    private int blockSize; // properties에서 설정
```

```
    /** 시작 인덱스 (curPos) */
```

```
    private int firstRecordIndex = 1;
```

```
    /** 끝 인덱스 */
```

```
    private int lastRecordIndex = 1;
```

```
    /**페이지 별 레코드 갯수 (pageSize) */
```

```
    private int recordCountPerPage;
```

```
....
```




ProductVO

```
public class ProductVO {  
    //멤버 변수-프로퍼티  
    private int productNo;  
    private int categoryNo;  
    private String productName;  
    private int sellPrice;  
    private String company;  
    private String imageUrl;  
    private String explain;  
    private String description;  
    private Timestamp regDate;  
    private int mileage;  
  
    ....  
}
```

```
public class EventProductVO extends SearchVO{  
    private int eventProductNo;  
    private int productNo;  
    private String eventName;  
  
    ...  
}
```

public class ProductDAOMybatis extends SqlSessionDaoSupport implements ProductDAO{

private String namespace="config.mybatis.mapper.oracle.Product";

public int insertProduct(ProductVO bean) {

int key = (Integer) getSession().insert(namespace+ ".productInsert", bean);

return key;

}

public List<ProductVO> listProductAll(SearchVO searchVO) {

//전체 상품 조회

List<ProductVO> alist

= getSession().selectList(namespace+ ".productList", searchVO);

return alist;

}

public int selectEventProduct(EventProductVO epBean){

int count = (Integer) getSession().selectOne(namespace+ ".eventProductSelect", epBean);

return count;

}

//이벤트 상품 등록

public int insertEventProduct(EventProductVO epBean){

int key = (Integer) getSession().insert(namespace+ ".eventProductInsert", epBean);

return key;

}

ProductDAOMybatis

@Override

```
public int selectTotalRecord(SearchVO searchVO) {  
    int count  
    = (Integer) getSqlSession().selectOne(namespace+ ".getTotalRecord", searchVO);  
    return count;  
}
```

```
public int deleteProduct(int pdNo) {  
    int n =    getSqlSession().delete(namespace+ ".productDelete", pdNo);  
    return n;  
}
```

```
}//class
```

- **eventProduct** 테이블에서 먼저 삭제한 후 **products** 테이블에서 삭제
- **On delete cascade** 되어 있다면 **products**만 삭제해도 자동으로 **eventProduct** 도 삭제됨

//이벤트 상품 등록

```
public int insertEventProduct(EventProductVO epBean){
    int key=0;
    int count = productDao.selectEventProduct(epBean);

    //해당 상품이 이미 해당 이벤트로 등록되어 있는 경우에는 insert하지 않고 skip한다
    if(count>0){
        key=1;
    }else if(count==0){ //처음 등록하는 경우에만
        key = productDao.insertEventProduct(epBean);
    }
    return key;
}
```

eventProduct 테이블에 해당 이벤트로 **insert**,
단, 해당상품이 해당 이벤트로 이미 등록되어 있는 경우
에는 **skip(insert하지 말자)**



파일 업로드 처리

- fileUpload.properties

```
file.upload.path=pds_upload
```

```
file.upload.path.test=E:\\d_java\\spws5\\springherb3\\src\\main\\webapp\\pds_upload
```

```
imageFile.upload.path=pd_images
```

```
imageFile.upload.path.test=E:\\d_java\\spws5\\springherb3\\src\\main\\webapp\\pd_images
```

```
file.upload.type=test
```

```
#file.upload.type=deploy
```



context-common.xml

```
<!-- MULTIPART RESOLVERS -->
<bean id="spring.RegularCommonsMultipartResolver"
      class="org.springframework.web.multipart.commons.CommonsMultipartResolver">
    <property name="maxUploadSize" value="100000000" />
    <property name="maxInMemorySize" value="100000000" />
</bean>

<alias name="spring.RegularCommonsMultipartResolver" alias="multipartResolver" />

<util:properties id="fileUploadProperties"
                 location="classpath:/config/props/fileUpload.properties" />
```

```
package com.herb.app.common;

@Component
public class FileUploadWebUtil {
    //자료실 업로드 폴더
    @Resource(name = "fileUploadProperties")
    Properties fileuploadProperties;

    public static final int PATH_FLAG_PDS = 1;
    public static final int PATH_FLAG_IMAGE = 2;

    private static final Logger logger = LoggerFactory.getLogger(FileUploadWebUtil.class);

    public String getUploadPath(HttpServletRequest request, int pathFlag){
        String uploadPath="";

        String upType
        = fileuploadProperties.getProperty("file.upload.type");

        String key = "";
        if(upType.equals("test")){
            if(pathFlag==PATH_FLAG_PDS){
                key = "file.upload.path.test";
            }else{
                key = "imageFile.upload.path.test";
            }
        }
    }
}
```

```
}else{
    if(pathFlag==PATH_FLAG_PDS){
        key = "file.upload.path";
    }else{
        key = "imageFile.upload.path";
    }
}
uploadPath = fileuploadProperties.getProperty(key);

if(!upType.equals("test")){
    uploadPath=request.getSession().getServletContext().getRealPath(uploadPath);
}

logger.debug("upType={}", uploadPath={}, upType, uploadPath);

return uploadPath;
}
```


//파일 업로드 처리

```
public List<Map<String, Object>> fileupload(HttpServletRequest request, int pathFlag)
    throws Exception{
    final MultipartHttpServletRequest multiRequest = (MultipartHttpServletRequest) request;
```

```
    final Map<String, MultipartFile> filesMap = multiRequest.getFileMap();
```

```
    String uploadPath =getUploadPath(request, pathFlag);
```

FileUploadWebUtil

```
    File saveFolder = new File(uploadPath);
```

```
    String fileName = null;
```

```
    // 디렉토리 생성
```

```
    if (!saveFolder.exists() || saveFolder.isFile()) {
        saveFolder.mkdirs();
    }
```

```
    List<Map<String, Object>> resultList = new ArrayList<Map<String,Object>>();
```

```
    Iterator<String> keyIter = filesMap.keySet().iterator();
```

```
    MultipartFile tempFile;
```

```
while (keyIter.hasNext())
{
    String key = keyIter.next();
    tempFile = filesMap.get(key);

    if(!tempFile.isEmpty()){
        long fileSize = tempFile.getSize(); //파일 크기
        String oName = tempFile.getOriginalFilename();
        //변경된 파일 이름
        fileName = Utility.getUniqueFileName(uploadPath, oName);
        logger.debug("fileSize ={}", fileName={}, fileSize, fileName);

        // 파일 전송
        File myfile = new File(uploadPath, fileName);
        tempFile.transferTo(myfile);

        Map<String, Object> resultMap = new HashMap<String, Object>();
        resultMap.put("fileName",fileName);
        resultMap.put("fileSize",fileSize);

        resultList.add(resultMap);
    }//if
}

return resultList;
}
```

```

/**
 * 시스템에서 17자리의TIMESTAMP값을 구한다
 * @return
 */
private static String getTimeStamp() {
    String result = "";

    // 문자열로 변환하기 위한 패턴(년-월-일 시:분:초:밀리초(자정 이후 초))
    String pattern = "yyyyMMddhhmmssSSS";

    SimpleDateFormat sdf = new SimpleDateFormat(pattern);
    /*Timestamp ts = new Timestamp(System.currentTimeMillis());
    result = sdf.format(ts.getTime());*/

    Date today = new Date();
    result = sdf.format(today);
    //result = sdf.format(today.getTime());
    System.out.println("getTimeStamp():"+result);

    return result;
}

```

```
public class Utility {  
    public static final int RECORD_COUNT_PER_PAGE=5; //페이지당 보여질 레코드수  
    public static final int BLOCK_SIZE=10;           //블럭당 보여질 페이지 수  
  
    //파일 업로드시 이름 변경하는 메서드  
    public static String getUniqueFileName(String dirPath, String fileName){  
        //test.txt => test_1.txt => test_2.txt  
        //확장자를 제외한 화일 이름 구하기  
        int idx = fileName.lastIndexOf(".");  
        String fName = fileName.substring(0, idx); //test  
  
        //.확장자 구하기  
        String ext = fileName.substring(idx); //.txt  
  
        boolean fExist=true;  
        int count=0;  
        while(fExist){  
            if(new File(dirPath, fileName).exists()){  
                fileName = fName + "_" + ++count + ext;  
            }else{  
                fExist=false;  
            }  
        }  
    }  
    return fileName;  
}
```

ProductAdminController

```
@Controller
@RequestMapping("/admin/product")
public class ProductAdminController {
    @Autowired CategoryService categoryService;
    @Autowired ProductService productService;

    @Resource(name="fileUploadWebUtil")
    FileUploadWebUtil fileUploadWebUtil;

    private static final Logger logger = LoggerFactory.getLogger(ProductAdminController.class);

    @RequestMapping(value="/productWrite.do", method=RequestMethod.GET)
    public String insertGet(Model model){
        //상품등록 페이지 보여주기 => 카테고리 조회
        //1. 파라미터
        logger.debug("상품 등록 화면 보여주기");

        //2. db작업 - select
        List<CategoryBean> alist= (List<CategoryBean>) categoryService.listCategoryAll();
        logger.debug("상품등록 - 카테고리조회 결과, alist.size()={}", alist.size());

        //3. 결과 저장, 리턴
        model.addAttribute("cgList", alist);
        return "/admin/product/productWrite";
    }
}
```

```
@RequestMapping(value="/productWrite.do", method=RequestMethod.POST)
```

```
public String insertPost(ProductVO bean, HttpServletRequest request){
```

```
    //상품 등록 처리 - db에서 products 테이블에 insert
```

```
    //=> 상품 이미지 파일업로드 처리
```

```
    //1. 파라미터
```

```
    logger.debug("파라미터 : productVo={}", bean);
```

```
    //파일 업로드 처리
```

```
    String fileName="";
```

```
    long fileSize=0;
```

```
    List<Map<String, Object>> resultList = null;
```

```
    try {
```

```
        int pathFlag=FileUploadWebUtil.PATH_FLAG_IMAGE;
```

```
        resultList = fileUploadWebUtil.fileupload(request, pathFlag);
```

```
        for(int i=0;i<resultList.size();i++){
```

```
            Map<String, Object> fileInfoMap = resultList.get(i);
```

```
            fileName = (String) fileInfoMap.get("fileName");
```

```
            fileSize =(Long) fileInfoMap.get("fileSize");
```

```
        }//for
```

```
        logger.debug("파일 업로드 성공, fileName={}, fileSize={}", fileName, fileSize);
```

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

```
        logger.debug("파일 업로드 실패");
```

```
        e.printStackTrace();
```

```
    }
```

```
    bean.setImageURL(fileName);
```

```
    //2. db작업 - insert
```

```
    int n=productService.insertProduct(bean);
```

```
    logger.debug("상품 등록 결과, n={}", n);
```

```
    //3. 결과 저장, 리턴
```

```
    return "redirect:/admin/product/productList.do";
```

```
}
```

ProductAdminController

@RequestMapping("/productList.do")

ProductAdminController

public String pdList(HttpServletRequest request,

@ModelAttribute EventProductVO eventProductVO, Model model){

//관리자 페이지 - 상품 목록 조회

//1. 파라미터

logger.debug("파라미터: eventProductVO={}", eventProductVO);

//2. db작업 - select

//페이징 처리

int recordCountPerPage = Utility.RECORD_COUNT_PER_PAGE;

int blockSize = Utility.BLOCK_SIZE;

logger.debug("recordCountPerPage={}, blockSize={}", recordCountPerPage, blockSize);

eventProductVO.setBlockSize(blockSize);

eventProductVO.setRecordCountPerPage(recordCountPerPage);

/** paging setting */

PaginationInfo paginationInfo = new PaginationInfo();

paginationInfo.setCurrentPageNo(eventProductVO.getCurrentPage());

paginationInfo.setRecordCountPerPage(recordCountPerPage);

paginationInfo.setBlockSize(blockSize);

eventProductVO.setFirstRecordIndex(paginationInfo.getFirstRecordIndex());

logger.debug("셋팅 후 eventProductVO={}", eventProductVO);

```
List<ProductVO> alist=null;
```

ProductAdminController

```
alist = productService.listProductAll(eventProductVO);  
int totalRecord = productService.selectTotalRecord(eventProductVO);
```

```
logger.debug("상품 목록 조회 결과, alist.size()={}", alist.size());
```

```
paginationInfo.setTotalRecordCount(totalRecord);
```

```
//3. 결과 저장, 리턴
```

```
model.addAttribute("pdList", alist);  
model.addAttribute("pagingInfo", paginationInfo);
```

```
return "/admin/product/productList";
```

```
}
```



```
@RequestMapping("/productDeleteMulti.do")
```

```
public String deleteMulti(@ModelAttribute ProductListVO pdListVo,  
                           HttpServletRequest request, Model model){
```

ProductAdminController

```
//선택한 상품 삭제
```

```
logger.info("관리자 - 선택한 상품 삭제, 파라미터 pdListVo={}", pdListVo);
```

```
List<ProductVO> pdList = pdListVo.getPdItems();
```

```
logger.info("pdList.size()={}", pdList.size());
```

```
int cnt = productService.deleteProduct(pdList);
```

```
logger.info("선택한 상품 삭제 처리 결과, cnt={}", cnt);
```

```
//파일 삭제 처리
```

```
String msg="", url="/admin/product/productList.do";
```

```
if(cnt>0){
```

```
    for(int i=0;i<pdList.size();i++){
```

```
        ProductVO pdVo = pdList.get(i);
```

```
        int productNo = pdVo.getProductNo();
```

```
        String imageURL = pdVo.getImageURL();
```

```
        logger.info("i={}", i);
```

```
        logger.info("productNo={}, imageURL={}", productNo, imageURL);
```

```

        //체크한 상품만 파일 삭제
        if(productNo!=0){
            //업로드 경로
            String upPath
            =fileUtil.getUploadPath(request, fileUtil.IMAGE_UPLOAD);

            //File객체 생성 후 파일 삭제
            File delFile = new File(upPath, imageURL);
            if(delFile.exists()){
                boolean bool = delFile.delete();
                logger.info("파일 삭제 결과={}", bool);
            }
        } //if
    } //for
    msg="선택한 상품을 삭제하였습니다";
}else{
    msg="선택한 상품을 삭제하지 못했습니다";
} //if

//3.
model.addAttribute("msg", msg);
model.addAttribute("url", url);

return "common/message";
}

```

```
@RequestMapping("/productEventMulti.do")
```

```
public String pdEventMulti(ProductListVO pdListVo, String eventSel){
```

```
    //선택한 상품들을 이벤트 상품으로 등록
```

```
    //1. 파라미터
```

ProductAdminController

```
    logger.debug("파라미터 : eventSel={}, pdListVo.getPdlItems().size()={}",  
                eventSel, pdListVo.getPdlItems().size());
```

```
    //2. db작업 - insert
```

```
    List<ProductVO> alist = pdListVo.getPdlItems();
```

```
    int cnt=productService.insertEventProduct(alist);
```

```
    logger.debug("이벤트 상품 등록 결과, cnt={}", cnt);
```

```
    //3. 결과 저장, 리턴
```

```
    return "redirect:/admin/product/productList.do";
```

```
}
```

```
}
```



ProductServiceImpl

@Transactional

```
public int deleteProduct(List<ProductVO> pdList) {  
    int cnt = 0;  
    try{  
        for(ProductVO vo : pdList){  
            int productNo = vo.getProductNo();  
            if(productNo!=0){  
                cnt = productDao.deleteProduct(vo.getProductNo());  
            }  
        } //for  
    }catch(RuntimeException e){  
        e.printStackTrace();  
        cnt=-1;  
    }  
  
    return cnt;  
}
```

- 선언적 트랜잭션에서는 런타임 예외가 발생하면 롤백한다. 반면에 예외가 전혀 발생하지 않거나 체크 예외가 발생하면 커밋한다.
- 스프링에서는 데이터 액세스 기술의 예외는 런타임 예외로 전환돼서 던져지므로 런타임 예외만 롤백 대상으로 삼은 것이다.

@Override

```
public int insertEventProduct(List<ProductVO> pdList, String eventName) {  
    int cnt=0;  
    try{  
        for(ProductVO vo : pdList){  
            //체크한 상품만 등록  
            int productNo=vo.getProductNo();  
            if(productNo!=0){  
                EventProductVO evPdVo = new EventProductVO();  
                evPdVo.setProductNo(productNo);  
                evPdVo.setEventName(eventName);  
                int count= productDao.selectEventProductCount(evPdVo);  
  
                //이미 해당 이벤트로 등록된 상품은 insert 불가  
                //-> skip  
                if(count>0){  
                    //이미 등록된 경우  
                    cnt=1; //성공으로 세팅  
                }else{  
                    //등록되어 있지 않은 경우  
                    cnt=productDao.insertEventProduct(evPdVo);  
                }  
            } //if  
        } //for  
    }catch(RuntimeException e){    e.printStackTrace(); }  
    return cnt;  
}
```



ProductListVO

```
public class ProductListVO {  
    private List<ProductVO> pdItems;  
  
    public List<ProductVO> getPdItems() {  
        return pdItems;  
    }  
    public void setPdItems(List<ProductVO> pdItems) {  
        this.pdItems = pdItems;  
    }  
}
```

```

<script type="text/javascript">
    $(document).ready(function() {
        $('#wr_submit').click(function () {
            if($("#productName").val().length<1){
                alert("상품명을 입력하세요");
                $("#productName").focus();
                return false;
            }else if($("#categoryNo").val()==0){
                alert("카테고리를 입력하세요");
                $("#categoryNo").focus();
                return false;
            }else if($("#imageUpload").val().length<1){
                alert("상품이미지를 입력하세요");
                $("#imageUpload").focus();
                return false;
            }
            if($("#sellPrice").val()==""){
                $("#sellPrice").val("0");
            }
            if($("#mileage").val()==""){
                $("#mileage").val("0");
            }
        });
    });
</script>

```

```

<div class="divForm">
<form name="frm1" method="post" action='<c:url value="/admin/product/productWrite.do"/>'
    enctype="multipart/form-data">
    <fieldset>
        <legend>상품 등록</legend>

```

```

<div>
    <label for="productName">상품명</label>
    <input type ="text" name="productName" id="productName" size="50">
</div>
<div>
    <label for="categoryNo">카테고리</label>
    <select name="categoryNo" id="categoryNo" title="카테고리">
        <option value="0">선택하세요</option>
        <!-- 반복문 시작 -->
        <c:forEach var="cgVo" items="${cgList }">
            <option value="${cgVo.categoryNo}">${cgVo.categoryName }</option>

        </c:forEach>
        <!-- 반복문 끝 -->
    </select>
</div>
<div>
    <label for="imageUpload">상품이미지</label>
    <input type ="file" name="imageUpload" id="imageUpload">
</div>
<div>
    <label for="sellPrice">가격</label>
    <input type ="text" name="sellPrice" id="sellPrice" >
    <br/><form:errors path="sellPrice" />
</div>
.....
<div class="center">
    <input type ="submit" id="wr_submit" value="등록">
    <input type ="reset" value="취소" >
</div>
</fieldset>

```

productWrite.jsp


```
<script type="text/javascript">
$(document).ready(function() {
    $('#btnSearch').click(function () {
        frmList.action="<c:url value='/admin/product/productList.do' />";
        frmList.submit();
    });
    $('#btnDel').click(function () {
        if(confirm("선택한 상품을 삭제하시겠습니까?")){
            var count=$('#tbody input[type=checkbox]:checked').length;
            if(count>0){
                //선택한 상품들 삭제
                frmList.action="<c:url value='/admin/product/productDeleteMulti.do' />";
                frmList.submit();

            }else{
                alert("처리하고 싶은 상품을 먼저 체크하세요");
            }
        }
    });
    $('#btnEvent').click(function () {
        var count=$('#tbody input[type=checkbox]:checked').length;
        if(count>0){
            //선택한 상품 이벤트 상품으로 등록
            frmList.action="<c:url value='/admin/product/productEventMulti.do' />";
            frmList.submit();
        }else{
            alert("처리하고 싶은 상품을 먼저 체크하세요");
        }
    });
});
```

```

$('#chkAll').click( function() {
    $('tbody input[type=checkbox]').prop( 'checked', this.checked );
} );
});
function boardList(curPage){    //페이지 번호를 클릭했을 때 처리
    frmPage.currentPage.value=curPage;
    frmPage.action="<c:url value='/admin/product/productList.do' />";
    frmPage.submit();
}

```

```

</script>
<h2>상품 목록</h2>
<c:if test="${empty param.eventName}">
    전체 상품 ${pagingInfo.totalRecord } 건 입니다.
</c:if>
<c:if test="${ !empty eventProductVO.eventName}">
    ${eventProductVO.eventName} 상품  ${pagingInfo.totalRecord }건입니다.
</c:if>
<!-- 페이징 처리를 위한 form 시작-->
<form name="frmPage" method="post">
    <input type="hidden" name="eventName" value="${eventProductVO.eventName }">
    <input type="hidden" name="currentPage">
</form>
<!-- 페이징 처리 form 끝 -->
<form name="frmList" method="post" >
<div class="divRight">이벤트 상품 조회
    <select name="eventName">
        <option value=""></option>
        <option value="NEW" <c:if test="${eventProductVO.eventName=='NEW' }">selected</c:if>>신상품</option>
        <option value="BEST"<c:if test="${eventProductVO.eventName=='BEST' }">selected</c:if>>인기상품</option>
        <option value="MD"<c:if test="${eventProductVO.eventName=='MD' }">selected</c:if>>MD추천상품</option>
    </select>

```

```

<input type="image" src="<c:url value='/images/bt_search3.png'/>" align="absmiddle" id="btnSearch">
</div>
<div class="divList">
<table width="700" class="box2"
summary="상품목록에 관한 표로써, 번호, 제목, 작성자, 작성일, 조회수에 대한 정보를 제공합니다.">
<caption>상품 목록</caption>
<colgroup><col style="width:5%" /><col style="width:15%" /><col style="width:32%" />
    <col style="width:15%" /><col style="width:15%" /><col style="width:9%" /><col style="width:9%" />
</colgroup>
<thead>
    <tr>
        <th><input type="checkbox" name="chkAll" id="chkAll"></th>
        <th scope="col">상품이미지</th><th scope="col">상품명</th>
        <th scope="col">가격</th><th scope="col">등록일</th><th scope="col">수정</th><th scope="col">삭제</th>
    </tr>
</thead>
<tbody>
<!-- 반복 시작 -->
<!-- 페이징 처리 위해 한 페이지에 5개씩만 보여주기 -->
<c:set var="idx" value="0" />
<c:forEach var="productVo" items="${pdList }">
    <tr>
        <td align="center">
            <input type="checkbox" id="chk_${idx}" name="pdItems[${idx}].productNo"
                value="${productVo.productNo}">
            <input type="hidden" name="pdItems[${idx}].imageUrl" value="${productVo.imageUrl}">
        </td>
        <td>
            <a href='<c:url value="/admin/product/productDetail.do?pdNo=${productVo.productNo}"/>'
                >
${productVo.productName}</td>
        <td align="right"><fmt:formatNumber value="${productVo.sellPrice}" pattern="#,###" /> 원</td>
        <td align="center">
        <fmt:formatDate value="${productVo.regDate}" pattern="yyyy-MM-dd" /> </td>
        <td align="center">
            <a href='<c:url value="/admin/product/productUpdate.do?pdNo=${productVo.productNo }"/>'>
                수정</a></td>
        <td align="center">
            <a href='<c:url value="/admin/product/productDelete.do?pdNo=${productVo.productNo }"/>'>
                삭제</a></td>
    </tr>
    <c:set var="idx" value="${idx+1 }" />
</c:forEach>
<!-- 반복 끝 -->
</tbody>
</table>
</div>
<div class="divPage">
    <!-- 페이지 번호 추가 -->
    <c:if test="${pagingInfo.firstPageNo>1 }">
        <a href="#" onclick="boardList(${pagingInfo.firstPageNo-1})">
            <img src='<c:url value="/images/first.JPG" />' border="0"> </a>
    </c:if>
    <!-- [1][2][3][4][5][6][7][8][9][10] -->
    <c:forEach var="i" begin="${pagingInfo.firstPageNo }" end="${pagingInfo.lastPageNo }">
        <c:if test="${i==pagingInfo.currentPageNo }">
            <span style="color:blue;font-weight:bold">${i }</span>
        </c:if>
        <c:if test="${i!=pagingInfo.currentPageNo }">
            <a href="#" onclick="boardList(${i})">[ ${i }]</a>
        </c:if>
    </c:forEach>

```

```
</c:forEach>
<c:if test="${pagingInfo.lastPageNo<pagingInfo.totalPageCount }">
    <a href="#" onclick="boardList(${pagingInfo.lastPageNo+1})">
        " border="0">
    </a>
</c:if>
<!-- 페이지 번호 끝 -->
</div>
```

```
<div class="divRight">
    <input type="button" value="선택한 상품 삭제" id="btnDel"><br><br>
    선택한 상품을
    <select name="eventSel">
        <option value=""></option>
        <option value="NEW">신상품으로</option>
        <option value="BEST">인기상품으로</option>
        <option value="MD">MD추천상품으로</option>
    </select>
    <input type="button" value="등록" id="btnEvent">
</div>
</form>
```