특별부록.

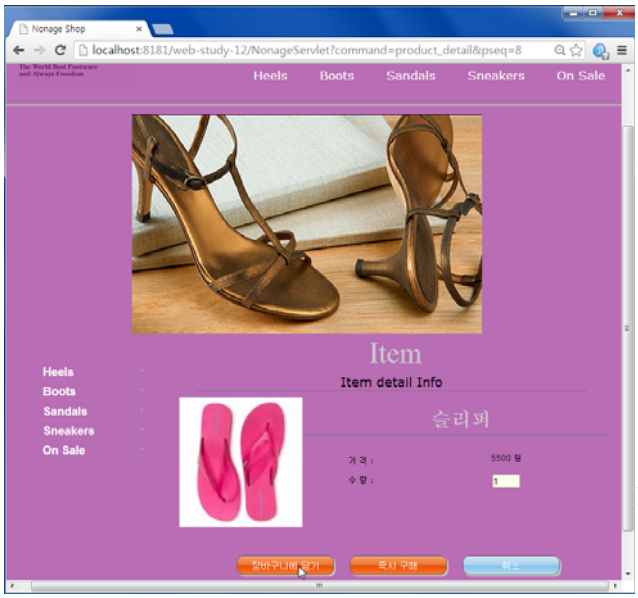
온라인 쇼핑몰 구축

_1st Update: 2014년 7월 17일

12장.

쇼핑몰-상품과 회원관리

이번 장에서는 앞장에서 배운 모든 예제와 그에 따른 응용부분을 포함해서 온라인 쇼핑몰 구축을 통해 정리하고자 합니다. 쇼핑몰의 종합적인 개발 과정을 진행하면서 웹 애플리케이션에 대한 이해와 빠른 개발 능력을 키워보도록 합시다.



12-012

12-1 온라인 쇼핑몰 애플리케이션 구성
12-2 데이터베이스 구축
12-3 메인 화면 만들기
12-4 상품 상세보기 페이지 만들기
12-5 메뉴별 페이지 만들기
12-6 회원 가입 및 로그인

12-1 온라인 쇼핑몰 애플리케이션 구성

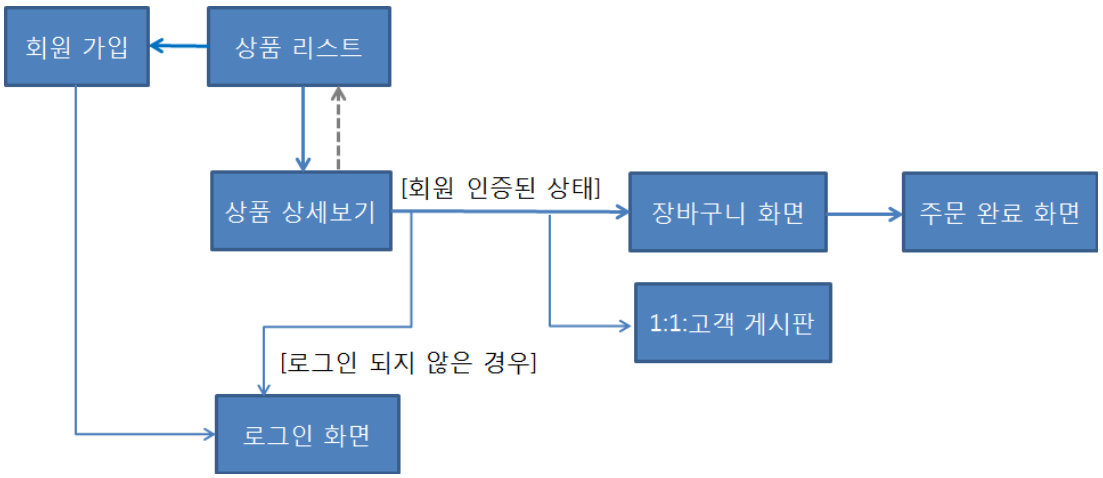
개발할 프로젝트는 상품(구두)을 판매하는 쇼핑몰로 이름은 Nonage Shop이라고 합시다. Nonage Shop을 만들기 위해서는 어떠한 형태로 쇼핑몰을 운영할 것인가를 생각해야합니다. Nonage Shop은 종합적인 쇼핑몰이 아닌 전문적인 쇼핑몰 개념을 가지고 개발에 들어갑니다. 즉 하나의 상품을 전문적으로 다루는 전문 쇼핑몰입니다. 많은 상품 중에서 Nonage Shop은 신발만 전문적으로 취급하는 사이트를 개발해 달라는 의뢰가 들어 왔다는 가정 하에서 진행하도록 하겠습니다. 그럼 먼저 무엇을 생각해야 할까요?

일단 디자인을 생각해야 할 것입니다. 특화된 상품을 어떻게 멋있게 진열해서 소비자로 하여금 구매 욕구를 느끼게 할 것인가? 어떤 사용자 인터페이스를 제공할 것인가?에 대해서 생각해야 합니다. 디자인을 어떻게 하느냐에 따라서 고객의 눈을 잡을 수 있는지 아니면 3초간의 눈요기로 가져갈 것인지가 정해지기 때문입니다. 그러나 이런 부분은 웹 마케팅을 담당하는 사람들의 입장이라고 생각하고 개발자 입장에서는 어떤 작업들이 주로 일어나며 이러한 작업이 일어나기 위해서는 데이터베이스를 어떻게 구축해야 할 것인가를 먼저 생각하게 됩니다. 따라서 우리는 개발자자 입장에서 테이블 설계를 정확하게 하기 위해서 업무 분석부터 하도록 하겠습니다.

Nonage Shop 시스템이 할 수 있는 기능을 정리하면 다음과 같이 사용자 측면과 관리자 측면으로 나눌 수 있습니다.

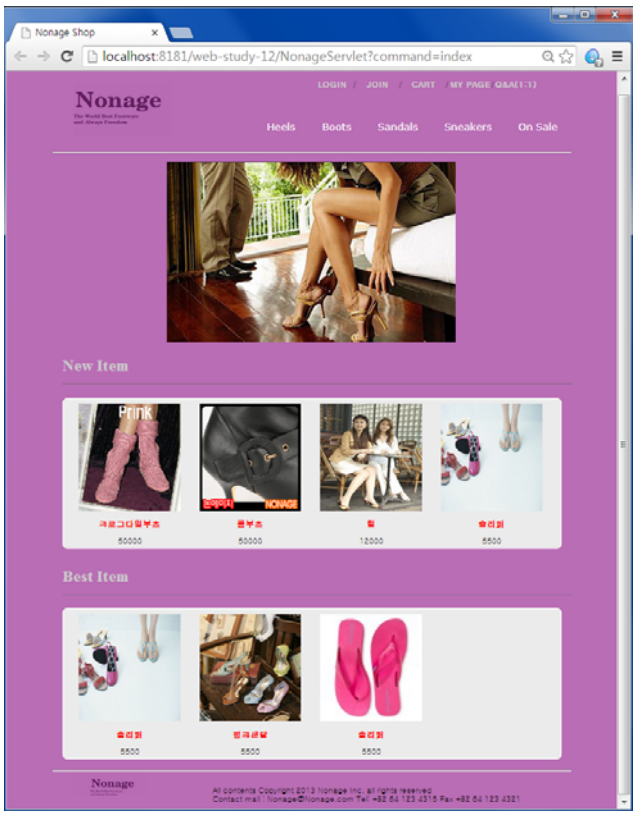
사용자	관리자
1. 회원 가입하기	1. 상품 관리하기
2. 로그인하기	2. 주문 내역 조회하기
3. 상품 검색하기	3. Q&N 게시판에 답변 달기
4. 상품 상세 보기	
5. 장바구니에 상품 담기	
6. 장바구니 확인하기	
7. 주문하기	
8. 주문 내역보기	
9. Q&N 게시판 이용하기	

Nonage Shop 시스템을 위해 구현해야 하는 화면을 소개하겠습니다. 우선 사용자에게 제공되는 쇼핑몰의 기능부터 살펴보겠습니다. 쇼핑몰의 주된 기능은 상품 리스트, 회원 가입, 주문과 1:1 고객 게시판입니다.



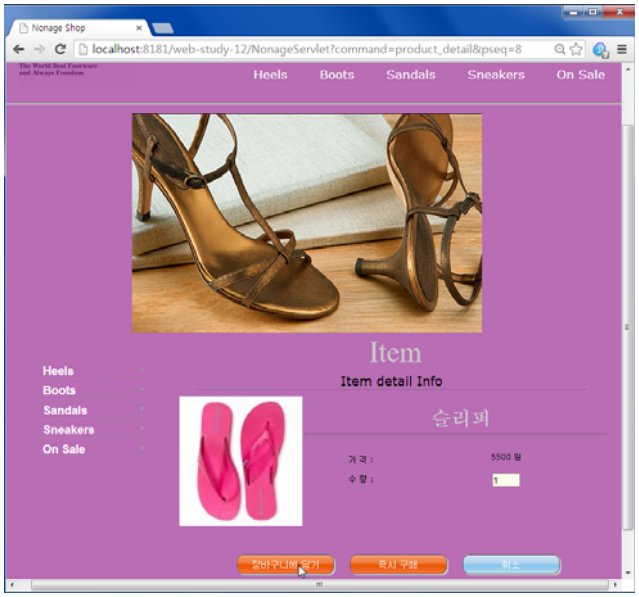
▼ 상품 리스트 화면

쇼핑몰의 생명은 메인 화면입니다. 메인 화면에는 상품들을 진열해 둘 것입니다. 특정 상품을 클릭하면 상품 상세보기 화면으로 이동합니다.



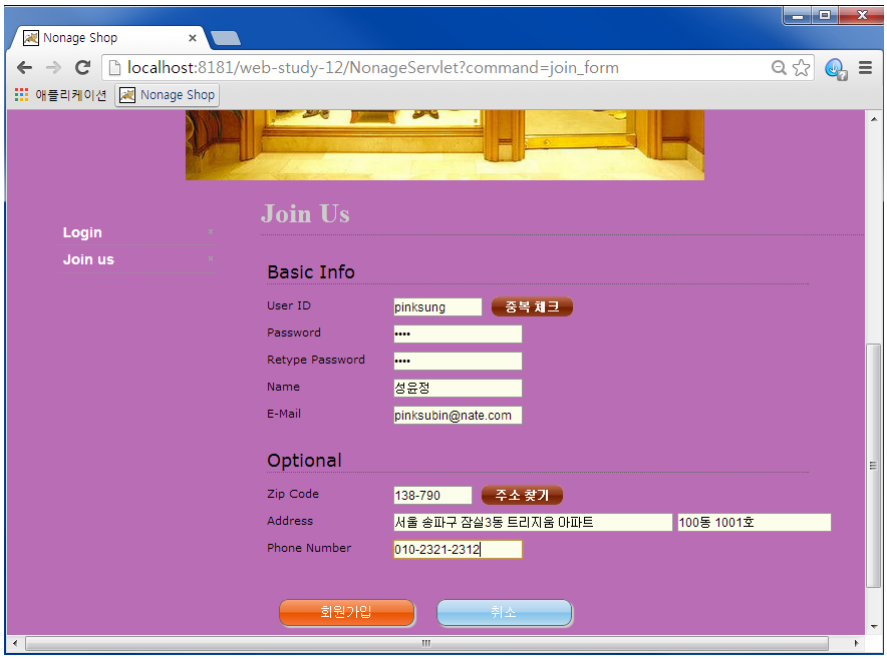
▼ 상품 상세 화면

상품 상세 화면에는 장바구니에 넣으면 로그인 되어 있는 경우에는 장바구니 확인 화면으로 이동하지만 로그인하지 않은 사용자는 로그인 과정을 거쳐야만 장바구니에 물건을 담을 수 있습니다. 로그인은 회원가입이 되어야만 할 수 있습니다.



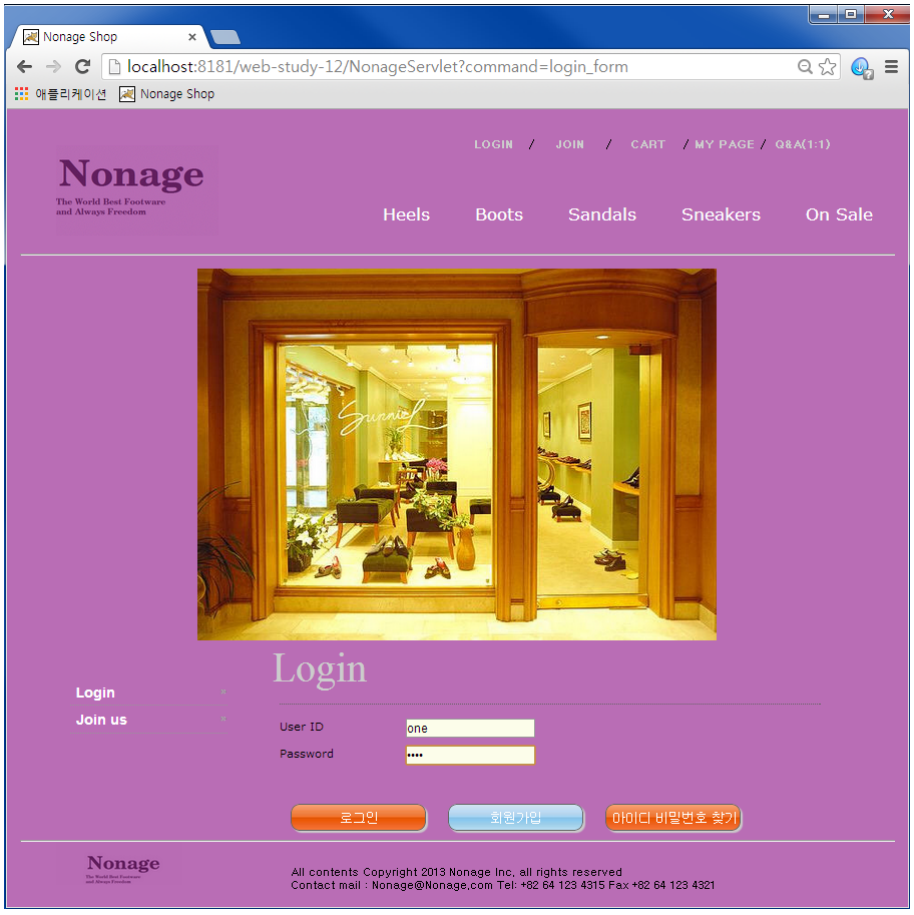
▼ 회원 가입 화면

다음은 회원 가입 페이지입니다.



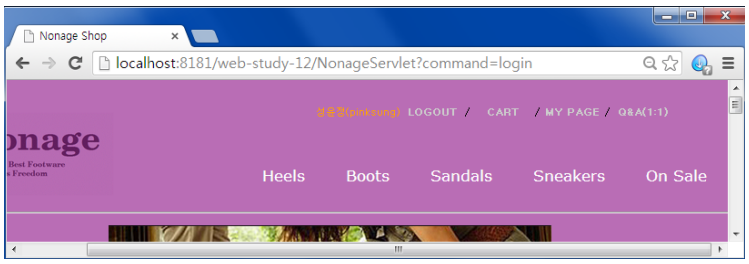
▼ 로그인 화면

회원 가입이 완료되면 로그인 화면이 나타나서 로그인 작업을 진행합니다.



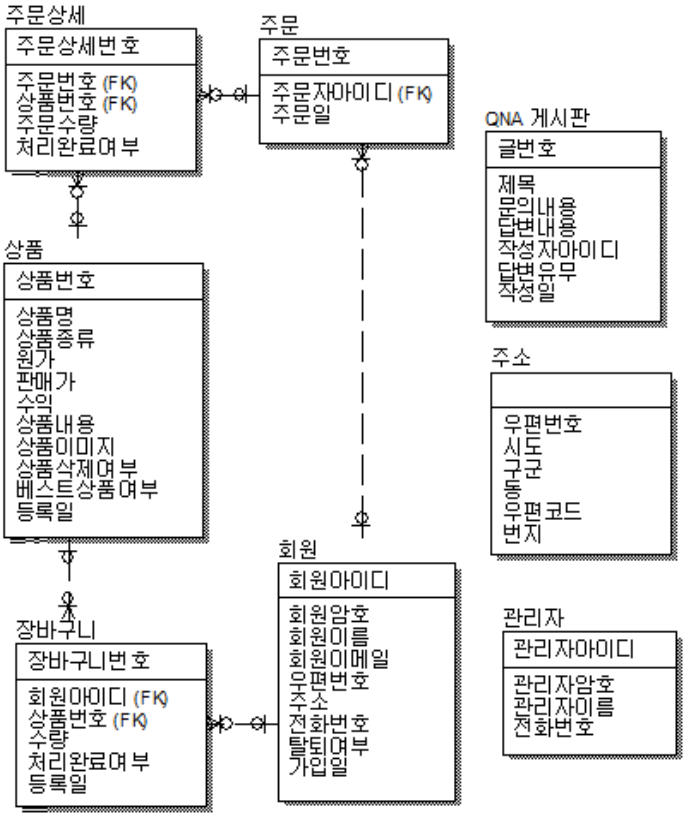
▼ 로그인 완료 화면

로그인 완료하고 나면 세션 상에는 회원 정보를 저장하기 때문에 헤더 부분에 로그인한 사용자의 이름과 아이디를 표시할 수 있게 됩니다.



12-2 데이터베이스 구축

Nonage Shop에서 어떤 작업들이 주로 일어나는지를 업무 분석을 하였다면 웹 사이트를 구축하기에 앞서 데이터베이스를 어떻게 구축해야 할 것인가를 먼저 생각해 봅시다. 데이터베이스 구축을 위해서 테이블을 설계해야 합니다. 어떠한 테이블이 필요하고 어떤 관계를 가지고 있는지를 생각해야 합니다. 일단 필요한 테이블을 살펴보면 상품 테이블, 회원 테이블, 장바구니 테이블, 주문 테이블, Q&A 테이블, 관리자 테이블 등이 필요합니다. 이들 테이블을 한눈에 분석하기 쉽게 하기 위해서는 이들 관계를 그림으로 표현하면 좋습니다. 다음은 Nonage Shop 시스템에서 사용할 테이블을 그림으로 표현한 것입니다.



Nonage Shop 시스템에서 사용할 테이블이 정해졌으면 데이터베이스로 구현될 수 있도록 구체적인 설계하는 과정이 필요합니다. 이 과정에서는 개체 간의 관계는 테이블의 외래키로 변환해야 하고 각 속성에 대해서는 데이터 형식과 각종 제약 조건, 인덱스 등을 설정해야 합니다. 다음에 나오는 테이블 명세표는 이를 정리하여 명세서로 정리해 둔 것입니다.

<엔티티 명 : 관리자　　테이블 명 : worker>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	아이디	id	varchar2	20	N	PK		
2	암호	pwd	varchar2	20				
3	이름	name	varchar2	40				
4	전화번호	phone	varchar2	20				

<엔티티 명 : 주소 테이블 명 : address>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	우편번호	zip_num	varchar2	7				
2	시도	sido	varchar2	30				
3	구군	gugun	varchar2	30				
4	동	dong	varchar2	100				
5	우편코드	zip_code	varchar2	30				
6	번지	bunji	varchar2	30				

<엔티티 명 : 회원 테이블 명 : member>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	회원아이디	id	varchar2	20	N	PK		
2	회원암호	pwd	varchar2	20				
3	회원이름	name	varchar2	40				
4	회원이메일	email	varchar2	40				
5	우편번호	zip_num	varchar2	7				
6	주소	address	varchar2	100				
7	전화번호	phone	varchar2	20				
8	탈퇴여부	useyn	char	1			y	회원의 탈퇴 여부 체크 y:사용가능, n:탈퇴
9	가입일	indate	date				SYSDATE	

<엔티티 명 : 상품 테이블 명 : product>

NO	속성명	칼럼명	자료형	크기	NULL 허용	제약조건	디폴트값	비고
1	상품번호	pseq	number	5	N	PK		product_seq 시퀀스 객체로 자동 일련 번호 부여
2	상품명	name	varchar2	100			0	
3	상품종류	kind	char	1				1:힐, 2:부츠, 3:샌달, 4:슬리퍼, 5:스니커즈
4	원가	price1	number	7			0	
5	판매가	price2	number	7			0	
6	판매가-원가	price3	number	7			'0'	
7	상품내용	content	varchar2	1000			NULL	
8	상품이미지	image	varchar2	50			default.jpg	
9	상품삭제여부	useyn	char	1			y	상품 사용유무 체크 y:사용가능, n:사용불가능
10	베스트상품여부	bestyn	char	1			n	베스트 상품인지 여부 체크 y:베스트 상품, n:베스트 상품 아님
11	등록일	indate	date				SYSDATE	

<엔티티 명 : 장바구니 테이블 명 : cart>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	장바구니번호	cseq	number	10	N	PK		cart_seq 시퀀스 객체로 자동 일련 번호 부여
2	회원아이디	id	varchar2	20		FK		member 테이블의 기본키인 id 컬럼
3	상품번호	pseq	number	5		FK		product테이블의 기본키인

								pseq 컬럼
4	수량	quantity	number	5			1	
5	처리완료여부	result	char	1			1	1:미처리 2:처리
6	등록일	indate	date				SYSDATE	

<엔티티 명 : 주문 테이블 명 : orders>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	주문번호	oseq	number	10	N	PK		orders_seq 시퀀스 객체로 자동 일련 번호 부여
2	주문자 아이디	id	varchar2	20		FK		member 테이블의 기본키인 id 컬럼
3	주문일	indate	date				SYSDATE	

<엔티티 명 : 주문 상세 테이블 명 : order_detail>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	주문상세번호	odseq	number	10	N	PK		order_detail_seq 시퀀스 객체로 자동 일련 번호 부여
2	주문번호	oseq	number	10		FK		orders 테이블의 기본 키인 oseq 컬럼
3	상품번호	pseq	number	5		FK		product 테이블의 기본 키인 pseq 컬럼
4	주문수량	quantity	number	5				
5	처리여부	result	char	1			1	1:미처리 2: 처리

<엔티티 명 : QNA 게시판 테이블 명 : qna>

NO	속성명	칼럼명	자료형	크기	NULL 허용	키	디폴트값	비고
1	글번호	qseq	number	5	N	PK		qna_seq 시퀀스 객체로 자동 일련 번호 부여
2	제목	subject	varchar2	30				
3	문의내용	content	varchar2	1000				
4	답변내용	reply	varchar2	1000				
5	작성자아이디	id	varchar2	20		FK		member 테이블의 기본키인 id 컬럼
6	답변유무	rep	char	1			1	1:답변 무 2:답변 유
7	작성일	indate	date				SYSDATE	

데이터베이스를 설계하고 구축하는 과정은 서블릿 JSP를 학습하는 것만큼 어렵고 복잡한 과정이기 때문에 저자가 이미 Nonage Shop 웹 애플리케이션을 위한 프로젝트를 진행하면서 테이블이나 시퀀스와 같은 이미 구축된 데이터베이스를 백업받아 놓았습니다. 백업 받는 과정은 저자가 하는 작업이지만 이론적으로 알아둘 필요가 있기 때문에 언급하기로 하겠습니다. 오라클에서는 데이터베이스의 백업을 위해서 exp 명령어를 제공합니다.

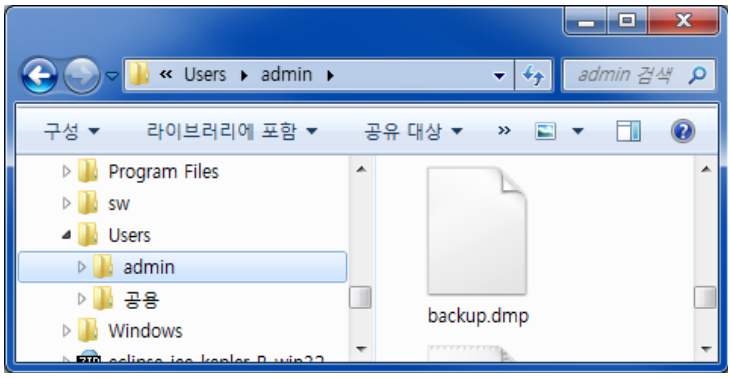
형식

```
exp userId/userPw@SID file=backup.dmp
```

다음은 저자의 PC에 설치된 오라클의 nonage 계정에 구축된 데이터베이스를 백업받는 예입니다. 이 내용은 여러분이 따라하며 실습할 내용은 아닙니다. 저자가 이렇게 백업을 받아 놓은 파일을 여러분이 복원해서 사용해야 하기 때문에 백업을 하는 과정을 알려드리는 것입니다.

```
exp nonage/tiger file=backup.dmp
```

윈도 탐색기에서 백업 파일이 성공적으로 생성되었음을 확인할 수 있습니다.



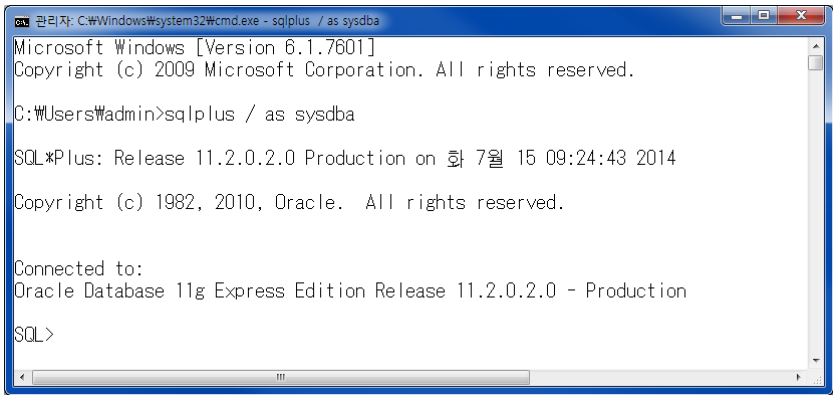
여러분의 PC에 설치된 오라클에 백업 파일을 복원하면 Nonage Shop 시스템에서 사용할 테이블이 데이터베이스에 구축됩니다. 백업 파일은 예제소스의 sql(WebContent 하위) 디렉토리에서 확인할 수 있습니다.

Nonage Shop 시스템을 위한 테이블은 새로운 사용자 계정을 생성한 후 만들기로 합시다.

[직접해보세요] Nonage Shop 데이터베이스 구축을 위한 사용자 계정 생성하기

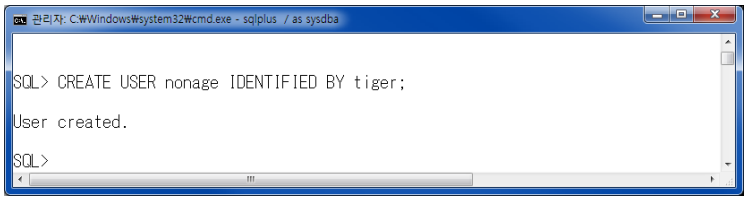
1. 명령 프롬프트 창을 띄웁니다. 그런 후에 DBA (Database Administrator)인 시스템 계정으로 로그인합니다.

```
sqlplus / as sysdba
```



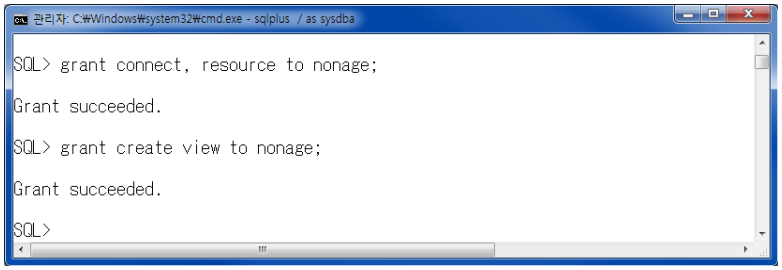
2. 오라클을 설치할 경우 제공되는 DBA(데이터베이스 administrator) 계정과는 달리 JDBC에서 사용할 테이블을 생성해 놓을 사용자 계정을 생성합니다. 사용자 이름은 오라클에 접속할 때 사용할 계정이고 접속을 성공리에 하려면 사용자 생성할 때 지정한 암호를 입력해야 합니다. 사용자 이름과 암호는 사용자 이름은 독자가 원하는 이름을 주면 됩니다.

```
CREATE USER nonage IDENTIFIED BY tiger;
```



3. 생성된 사용자 계정에 권한을 부여합니다. 권한은 데이터베이스 생성 및 테이블 생성과 같은 기능을 사용하기 위한 것입니다.

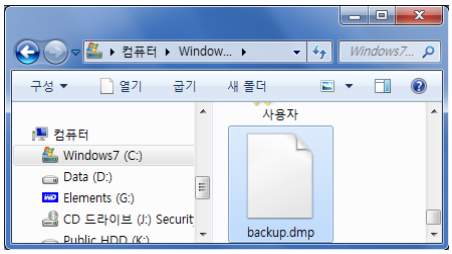
```
grant connect, resource to nonage;
grant create view to nonage;
```



새로 생성한 nonage 사용자에 Nonage Shop 시스템에서 사용할 데이터베이스를 구축해 봅시다.

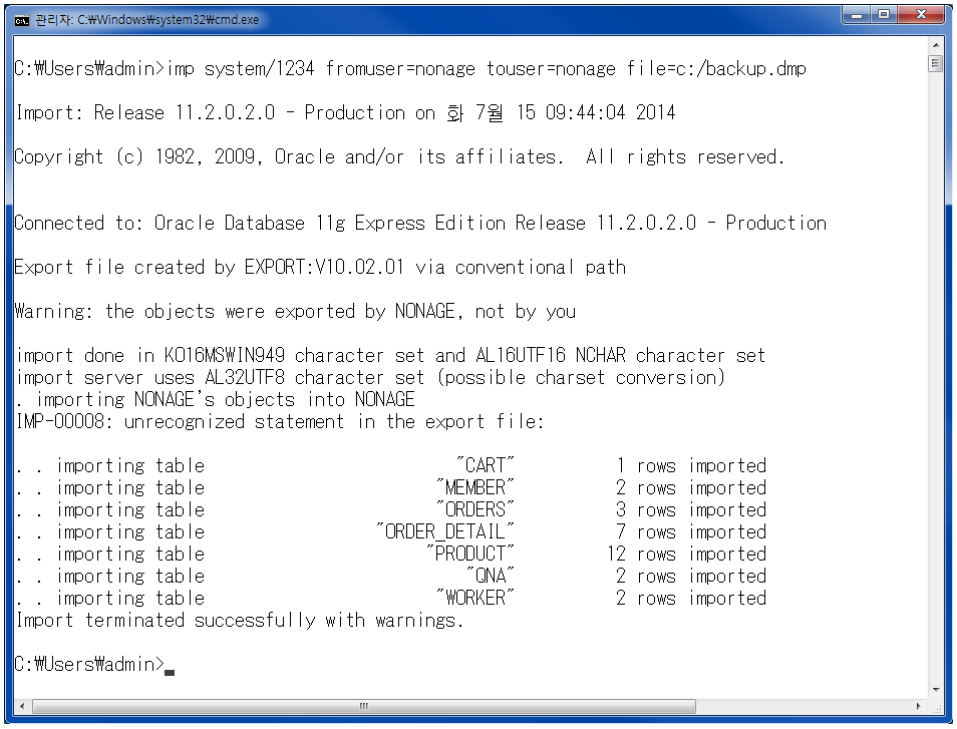
[직접해보세요] Nonage Shop 시스템을 위한 데이터베이스 구축하기

1. backup.dmp 파일을 C:₩ 아래에 복사해 둡니다.



2. 다음은 덤프 파일을 임포트하여 데이터베이스를 구축합시다.

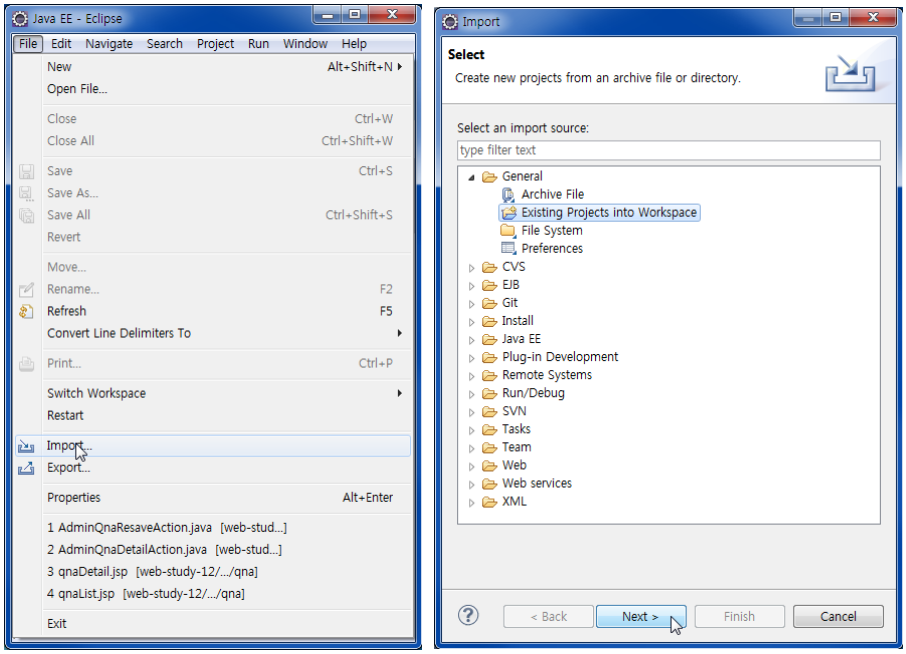
```
imp system/1234 fromuser=nonage touser=nonage file=c:/backup.dmp
```



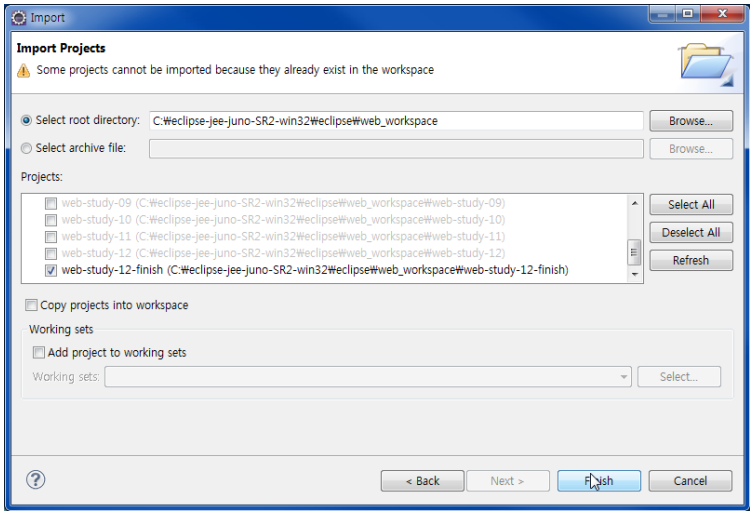
지금부터 학습할 프로젝트는 소스 코드가 방대합니다. 이 방대한 코드를 일일이 입력하고 오류를 수정하면서 직접 실습하다보면 다소 지칠 수 있기 때문에 이미 완성된 프로젝트를 실행해 보고 소스를 분석해 보는 것도 나쁘지 않을 것 같아서 미리 완성된 프로젝트를 이클립스에 추가(import) 하여 실행시키는 방법을 학습하도록 합시다.

<실습> 프로젝트 불러오기

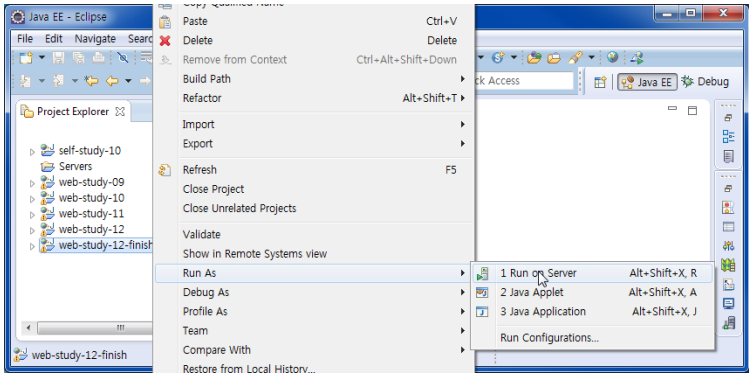
1. 로드북 사이트에서 특별 부록 자료를 다운 로드하여 web-study-12-finish.zip 파일을 압축 푼 후에 프로젝트를 불러와야 합니다. 불러오는 방법은 다음과 같습니다. [File]→ [Import...] 메뉴를 선택하여 [Import] 창이 나타나면 [General → Existing Projects into Workspace]를 선택한 후 [Next] 버튼을 클릭합니다.



2. 이미 완성된 프로젝트 소스가 있는 폴더에서 프로젝트 폴더를 선택한 후 [확인] 버튼을 누른 다음 해당 프로젝트가 선택된 것을 확인한 후 [Finish] 버튼을 누른다.

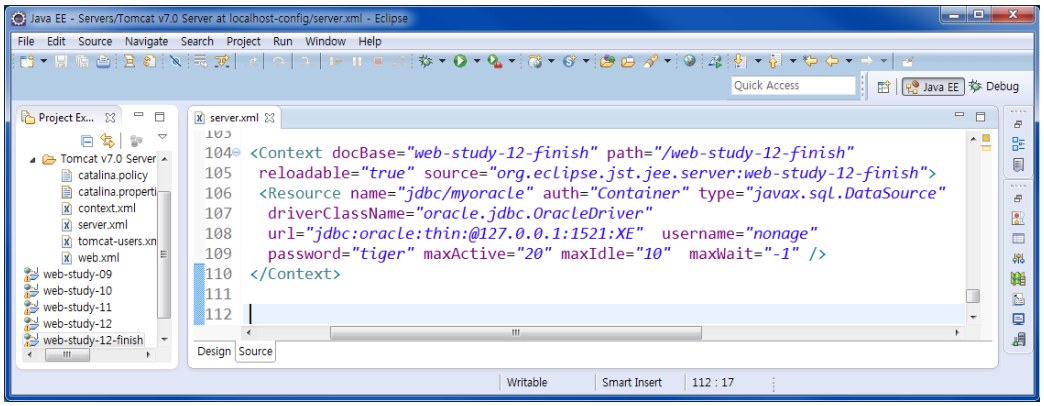


3. import한 프로젝트를 실행시키면 커넥션 풀을 위한 환경설정을 하지 않아 정상으로 동작하지 않을 것입니다. 이제 server.xml에 <Context> 태그에 커넥션 풀을 위한 환경설정을 하여 제대로 프로젝트가 실행되도록 합시다.

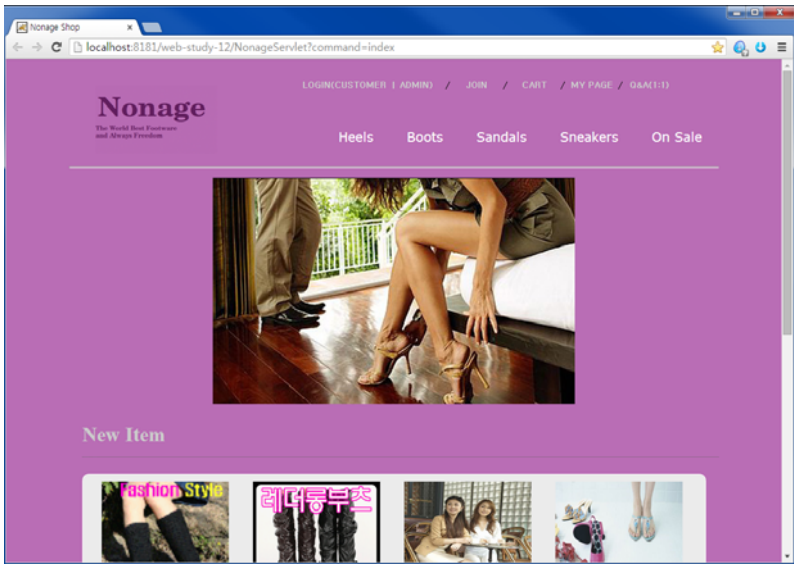


4. server.xml에서 프로젝트를 실행하여 생긴 <Context> 태그를 찾아 <Resource> 태그를 추가한 후에 다시 실행하면 데이터베이스와 연동되어 상품 정보는 물론 사용자나 관리자 모드로 로그인해서 웹 사이트를 사용할 수 있게 됩니다.

```
<Context docBase="web-study-12-finish" path="/web-study-12"
  reloadable="true" source="org.eclipse.jst.jee.server:web-study-12-finish">
  <Resource name="jdbc/myoracle" auth="Container"  type="javax.sql.DataSource"
    driverClassName="oracle.jdbc.OracleDriver"
    url="jdbc:oracle:thin:@127.0.0.1:1521:XE" username="nonage"
    password="tiger" maxActive="20" maxIdle="10" maxWait="-1"/>
</Context>
```



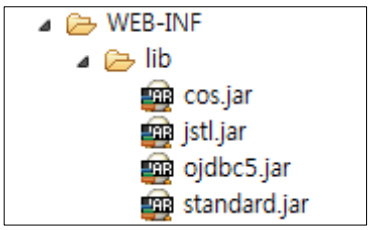
5. 완성된 소스가 있는 프로젝트를 임포트하는 작업이 완료되었다면 이제 이를 실행해 보도록 합시다. 실행하실 경우 스타일시트가 제대로 적용되도록 하려면 브라우저를 크롬으로 선택하시던지 익스플로러인 경우 HTML5가 적용되는 10버전 이후에서 실행하셔야 합니다.



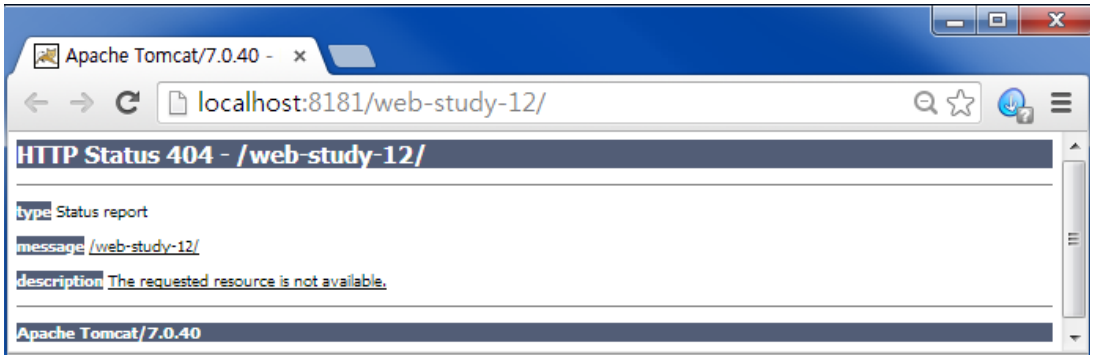
예제를 실행한 후 사용해 보았다면 이제 새롭게 프로젝트를 생성하여 프로그램을 일일이 구현해 보도록 하겠습니다.

12-3 메인 페이지 만들기

이클립스에서 Dynamic Web Project로 12장에서 사용할 web-study-12란 이름으로 프로젝트를 만듭니다. 오라클 드라이버를 위한 ojdbcX.jar와 JSTL을 위한 jstl.jar와 standard.jar와 상품 이미지 파일을 서버에 올리기 위해서 cos.jar 파일을 "web_workspace₩web-study-12₩WebContent₩WEB-INF₩lib" 폴더에 복사합니다.



DBCP를 이용해서 데이터베이스에 연결 가능한 Connection 객체를 얻어오기 위해서 server.xml 파일의 <Context> 태그에 <Resource> 태그를 추가해야 합니다.
<Context> 태그는 프로젝트를 실행하면 이클립스에서 자동으로 생성해주는 태그이므로 현재는 jsp 파일을 만들지 않아 프로젝트를 실행하면 오류가 발생하지만 오류가 발생하더라도 프로젝트를 실행시켜 <Context> 태그가 자동 생성되도록 합시다.



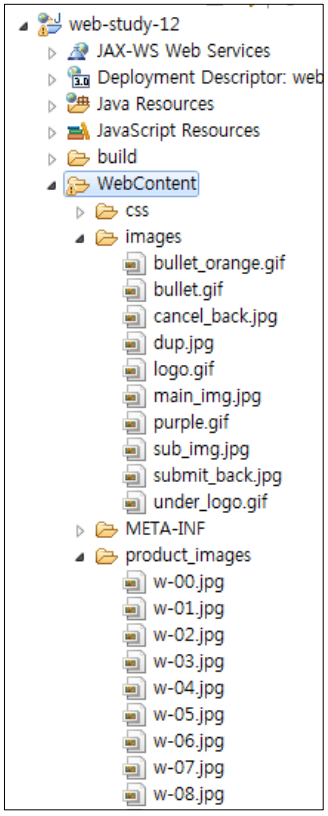
프로젝트를 실행하면 위 그림과 같이 오류가 발생합니다. 당황하지 말고 server.xml 파일을 열어 web-study-12 프로젝트에 관련된 <Context> 태그를 찾습니다.

server.xml 파일의 <Context>와 </Context> 사이에 <Resource> 태그를 붙여 넣습니다.

```
<Context docBase="web-study-12" path="/web-study-12"
    reloadable="true" source="org.eclipse.jst.jee.server:web-study-12">
    <Resource auth="Container" driverClassName="oracle.jdbc.OracleDriver"
        maxActive="20" maxIdle="10" name="jdbc/myoracle" password="tiger"
        type="javax.sql.DataSource" url="jdbc:oracle:thin:@127.0.0.1:1521:XE"
        username="nonage" />
</Context>
```

JSP 파일은 화면의 레이아웃을 조정하기 위해서 사용되는 shopping.css(내용은 생략:출판사 홈페이지에서 다운받으세요)를 WebContent의 하위 폴더인 css에 복사해 둡니다.

또한 프로젝트에서 사용할 이미지 파일을 저장한 폴더(images와 product_images)도 출판사 홈페이지에서 다운받아 WebContent의 하위 폴더에 복사해둡니다.



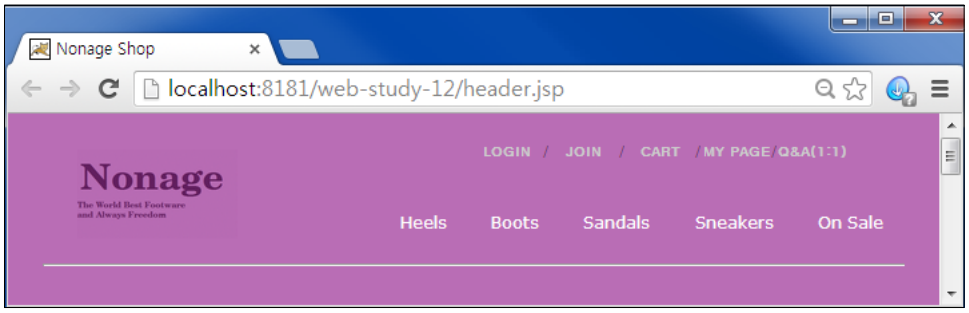
웹 서핑을 하다보면 웹 애플리케이션의 화면 중 상단과 좌측 메뉴는 동일한 형태로 제공되고 메뉴에 따라 메인 페이지만 변경되는 것을 확인할 수 있습니다. 모든 웹 페이지에서 공통된 화면을 제공받을 수 있도록 하기 위해서 파일을 분리하여 관리하는 것이 합리적입니다. 모든 웹 페이지에 공통적으로 들어가는 파일들을 살펴보도록 합시다.

[공통적으로 적용되는 파일]

파일이름	설명
header.jsp	상단 메뉴를 구성하는 파일
footer.jsp	사이트 정보를 출력하는 파일

각 화면에는 공통의 헤더와 풋터를 include하고 있기 때문에 일관성 있는 화면이 제공됩니다.

▼ 공통으로 사용하는 헤더



[직접해보세요] 각 화면에서 공통으로 사용되는 헤더 파일을 작성하기

[파일이름 : WebContent/header.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2    pageEncoding="UTF-8"%>
3  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
4  <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
5  <!DOCTYPE html>
6  <html>
7  <head>
8    <meta charset="UTF-8">
9    <title>Nonage Shop</title>
10   <link href="css/shopping.css" rel="stylesheet">
11   <script type="text/javascript" src="member/member.js"></script>
12   <script type="text/javascript" src="mypage/mypage.js"></script>
13 </head>
14
15 <body>
16 <div id="wrap">
17   <!--헤더 들어가는 곳 시작 -->
18   <header>
19     <!--로고 들어가는 곳 시작--->
20     <div id="logo">
21       <a href="NonageServlet?command=index">
22         <img src="images/logo.gif" width="180" height="100" alt="nonageshop">
23       </a>
24     </div>
25     <!--로고 들어가는 곳 끝-->
26     <nav id="catagory_menu">
27      <ul>
28        <c:choose>
29        <c:when test="${empty sessionScope.loginUser}">
30        <li><a href="NonageServlet?command=login_form">LOGIN</a></li>
31        <li>/</li>
32        <li><a href="NonageServlet?command=contract">JOIN</a></li>
33        </c:when>
```

```
34          <c:otherwise>
35          <li style="color:orange">
36            ${sessionScope.loginUser.name}(${sessionScope.loginUser.id})
37          </li>
38          <li><a href="NonageServlet?command=logout">LOGOUT</a></li>
39          </c:otherwise>
40          </c:choose>
41          <li>/</li>
42          <li>
43            <a href="NonageServlet?command=cart_list">CART</a>
44          </li><li>/</li>
45          <li>
46            <a href="NonageServlet?command=mypage">MY PAGE</a>
47          </li><li>/</li>
48          <li>
49            <a href="NonageServlet?command=qna_list">Q&amp;A(1:1)</a>
50          </li>
51        </ul>
52      </nav>
53
54      <nav id="top_menu">
55        <ul>
56          <li>
57            <a href="NonageServlet?command=catagory&kind=1">Heels</a>
58          </li>
59          <li>
60            <a href="NonageServlet?command=catagory&kind=2">Boots</a>
61          </li>
62          <li>
63            <a href="NonageServlet?command=catagory&kind=3">Sandals</a>
64          </li>
65          <li>
66            <a href="NonageServlet?command=catagory&kind=4">Sneakers</a>
67          </li>
68          <li>
69            <a href="NonageServlet?command=catagory&kind=5">On Sale</a>
70          </li>
71        </ul>
72      </nav>
73      <div class="clear"></div>
74      <hr>
75    </header>
76    <!--헤더 들어가는 곳 끝 -->
```

[직접해보세요] 각 화면에서 공통으로 사용되는 풋터 파일을 작성하기

[파일이름 : WebContent/footer.jsp]

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
  <div class="clear"></div>
  <!--푸터 들어가는 곳 시작 -->
  <footer>
    <hr>
    <div id="copy">
     All contents Copyright 2013 Nonage Inc. all rights reserved<br>
     Contact mail : Nonage@Nonage.com Tel: +82 64 123 4315
     Fax +82 64 123 4321
    </div>
  </footer>
  <!--푸터 들어가는 곳 끝 -->
</div>
</body>
</html>
```

[직접해보세요] 헤더와 풋터 파일 포함(include)하여 메인 화면 작성하기

[파일이름 : WebContent/index.jsp]

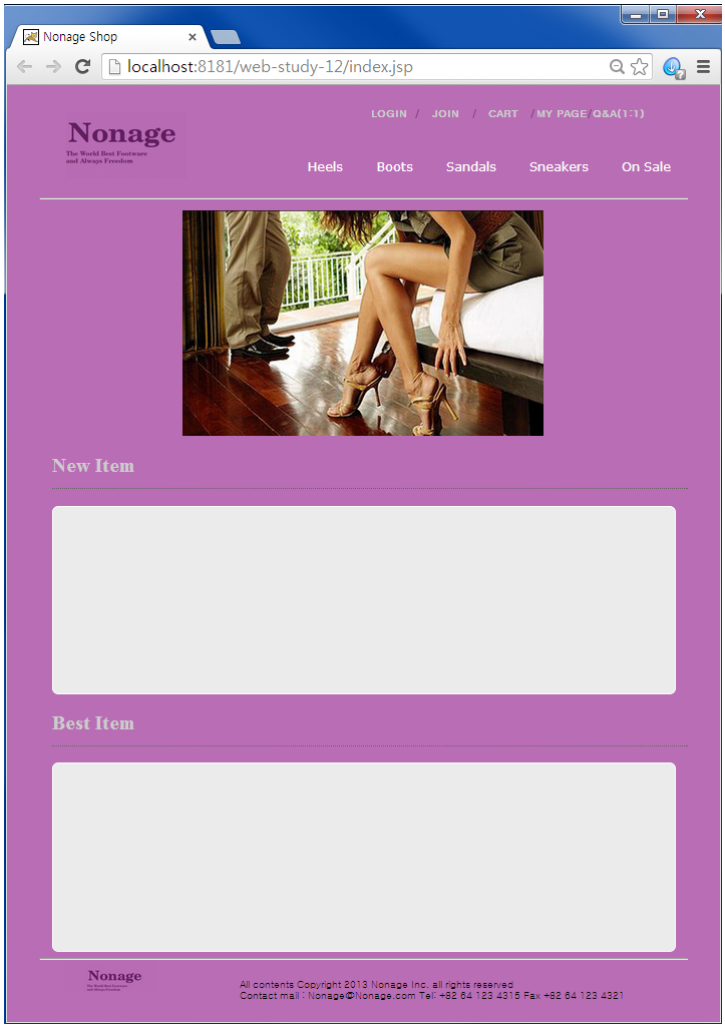
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
  pageEncoding="UTF-8"%>
 <!--헤더 파일 포함하기--->
 <%@ include file="../header.jsp" %>

  <!--메인 이미지 들어가는 곳 시작 --->
  <div class="clear"></div>
  <div id="main_img">
    <img src="images/main_img.jpg" >
  </div>
  <!--메인 이미지 들어가는 곳 끝--->

  <div class="clear"></div>

  <div id="front">
    <h2> New Item</h2>
    <div id="bestProduct">
      <c:forEach items="${newProductList }"  var="productVO">
        <div id="item">
          <a href=
"NonageServlet?command=product_detail&pseq=${productVO.pseq}">
            <img src="product_images/${productVO.image}" />
            <h3> ${productVO.name} </h3>
            <p>${productVO.price2} </p>
          </a>
        </div>
      </c:forEach>
```

```
28      </div>
29     <div class="clear"></div>
30
31      <h2> Best Item</h2>
32        <div id="bestProduct">
33          <c:forEach items="${bestProductList}"   var="productVO">
34            <div id="item">
35             <a href=
36 "NonageServlet?command=product_detail&pseq=${productVO.pseq}">
37                <img src="product_images/${productVO.image}" />
38              <h3> ${productVO.name} </h3>
39              <p>${productVO.price2} </p>
40          </a>
41        </div>
42      </c:forEach>
43    </div>
44    <div class="clear"></div>
45    </div>
46    <!--푸터 파일 포함하기--->
47    <%@ include file="../footer.jsp" %>
```



아직은 데이터베이스와 연동하는 부분은 구현하지 않았기 때문에 신상품과 베스트 상품을 출력할 수 있는 메인 화면에 대한 레이아웃만 출력됩니다.

이제 메인 화면에는 신상품과 베스트 상품을 출력해기 위해서 데이터베이스와 연동하도록 합시다. 오라클 덤프 파일에는 뷰(오라클에서 사용되는 가상 테이블)를 만들어 두었기 때문에 이 뷰를 통해서 신상품과 베스트 상품 리스트를 출력하는 DAO를 작성해 보도록 합니다.

뷰를 조회해서 상품 리스트를 위한 DAO를 만들기 위해서 우선 데이터베이스에서 얻어온 상품 정보를 담을 VO부터 만듭시다.

[직접해보세요] 이클립스에서 상품 정보를 저장하는 VO 클래스 만들기
이클립스 화면 왼쪽에서 프로젝트를 선택한 후에 마우스 오른쪽 버튼을 클릭하여 나타난 바로가기 메뉴에서 [New → Class]를 선택합니다. [Package:] 입력란에 패키지 이름(com.nonage.dto)을 [Name:] 입력란에 클래스 이름(ProductVO)을 입력한 후 [Finish] 버튼을 클릭합니다. 게시글 정보를 저장할 필드를 선언합니다. ProductVO 클래스 내부에서 마우스 오른쪽 버튼을 클릭하여 나타난 바로가기 메뉴에서 [Source]-[Generate Getters and Setters]를 선택하여 getter, setter를 일괄적으로 생성합니다.

```
1  package com.nonage.dto;
2
3  import java.sql.Timestamp;
4
5  public class ProductVO {
6    private int pseq;
7    private String name;
8    private String kind;
9    private int price1;
10   private int price2;
11   private int price3;
12   private String content;
13   private String image;
14   private String useyn;
15   private String bestyn;
16   private Timestamp indate;
17
18   public int getPseq() {
19     return pseq;
20   }
21
22   public void setPseq(int pseq) {
23     this.pseq = pseq;
24   }
25   public String getName() {
26     return name;
27   }
```

```
28    public void setName(String name) {
29      this.name = name;
30    }
31    public String getKind() {
32      return kind;
33    }
34    public void setKind(String kind) {
35      this.kind = kind;
36    }
37    public int getPrice1() {
38      return price1;
39    }
40    public void setPrice1(int price1) {
41      this.price1 = price1;
42    }
43    public int getPrice2() {
44      return price2;
45    }
46    public void setPrice2(int price2) {
47      this.price2 = price2;
48    }
49    public int getPrice3() {
50      return price3;
51    }
52    public void setPrice3(int price3) {
53      this.price3 = price3;
54    }
55    public String getContent() {
56      return content;
57    }
58    public void setContent(String content) {
59      this.content = content;
60    }
61    public String getImage() {
62      return image;
63    }
64    public void setImage(String image) {
65      this.image = image;
66    }
67    public String getUseyn() {
68      return useyn;
69    }
70    public void setUseyn(String useyn) {
71      this.useyn = useyn;
72    }
73    public String getBestyn() {
74      return bestyn;
75    }
```

```
76    public void setBestyn(String bestyn) {
77      this.bestyn = bestyn;
78    }
79    public Timestamp getIndate() {
80      return indate;
81    }
82    public void setIndate(Timestamp indate) {
83      this.indate = indate;
84    }
85  }
```

Connenction 객체를 얻어오고 사용이 끝난 리소스를 해제하는 DBManager 클래스 만듭니다.

[직접해보세요] Connenction 객체를 얻어오고 사용이 끝난 리소스를 해제하는 클래스 만들기[파일이름 : util₩DBManager.java]

```
1   package util;
2
3   import java.sql.Connection;
4   import java.sql.DriverManager;
5   import java.sql.ResultSet;
6   import java.sql.Statement;
7
8   import javax.naming.Context;
9   import javax.naming.InitialContext;
10  import javax.sql.DataSource;
11
12  public class DBManager {
13    public static Connection getConnection() {
14      Connection conn = null;
15      try {
16        Context initContext = new InitialContext();
17        Context envContext = (Context) initContext.lookup("java:/comp/env");
18        // jdbc/myoracle이란 이름을 객체를 찾아서 DataSource가 받는다.
19        DataSource ds = (DataSource) envContext.lookup("jdbc/myoracle");
20        // ds가 생성되었으므로 Connection을 구합니다.
21        conn = ds.getConnection();
22      } catch (Exception e) {
23        e.printStackTrace();
24      }
25      return conn;
26    }
27    // select을 수행한 후 리소스 해제를 위한 메소드
28    public static void close(Connection conn, Statement stmt, ResultSet rs) {
29      try {
30        rs.close();
31        stmt.close();
```

```
32          conn.close();
33        } catch (Exception e) {
34          e.printStackTrace();
35        }
36      }
37
38      // DML(insert, update, delete)을 수행한 후 리소스 해제를 위한 메소드
39      public static void close(Connection conn, Statement stmt) {
40        try {
41          stmt.close();
42          conn.close();
43        } catch (Exception e) {
44          e.printStackTrace();
45        }
46      }
47  }
```

상품 정보를 얻어오기 위한 ProductDAO를 만듭시다.

[직접해보세요] 상품 테이블을 액세스하는 DAO 클래스 만들기

[파일 이름 :src₩com₩nonage₩dao₩ProductDAO.java]

```
1   package com.nonage.dao;
2
3   import java.sql.Connection;
4   import java.sql.PreparedStatement;
5   import java.sql.ResultSet;
6   import java.util.ArrayList;
7
8   import util.DBManager;
9
10  import com.nonage.dto.ProductVO;
11
12  public class ProductDAO {
13    private ProductDAO() {
14    }
15
16    private static ProductDAO instance = new ProductDAO();
17
18    public static ProductDAO getInstance() {
19      return instance;
20    }
21
22    // 신상품 리스트 얻어오기
23    public ArrayList<ProductVO> listNewProduct() {
24      ArrayList<ProductVO> productList = new ArrayList<ProductVO>();
25      String sql = "select *  from new_pro_view";
26
27      Connection conn = null;
```

```
28      PreparedStatement pstmt = null;
29      ResultSet rs = null;
30
31      try {
32        conn = DBManager.getConnection();
33        pstmt = conn.prepareStatement(sql);
34        rs = pstmt.executeQuery();
35        while (rs.next()) {
36          ProductVO product = new ProductVO();
37          product.setPseq(rs.getInt("pseq"));
38          product.setName(rs.getString("name"));
39          product.setPrice2(rs.getInt("price2"));
40          product.setImage(rs.getString("image"));
41          productList.add(product);
42        }
43      } catch (Exception e) {
44        e.printStackTrace();
45      } finally {
46        DBManager.close(conn, pstmt, rs);
47      }
48      return productList;
49    }
50
51    // 베스트 상품 리스트 얻어오기
52    public ArrayList<ProductVO> listBestProduct() {
53      ArrayList<ProductVO> productList = new ArrayList<ProductVO>();
54      String sql = "select *  from best_pro_view";
55
56      Connection conn = null;
57      PreparedStatement pstmt = null;
58      ResultSet rs = null;
59
60      try {
61        conn = DBManager.getConnection();
62        pstmt = conn.prepareStatement(sql);
63        rs = pstmt.executeQuery();
64        while (rs.next()) {
65          ProductVO product = new ProductVO();
66          product.setPseq(rs.getInt("pseq"));
67          product.setName(rs.getString("name"));
68          product.setPrice2(rs.getInt("price2"));
69          product.setImage(rs.getString("image"));
70          productList.add(product);
71        }
72      } catch (Exception e) {
73        e.printStackTrace();
74      } finally {
75        DBManager.close(conn, pstmt, rs);
```

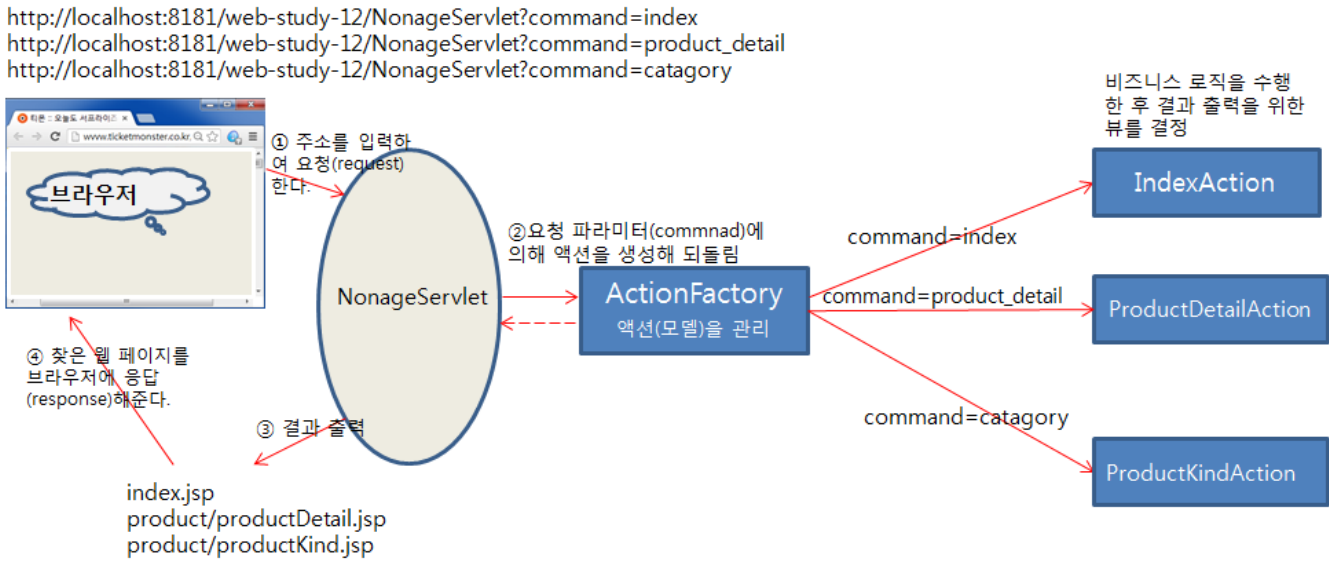
```
76      }
77      return productList;
78    }
79
80    //  상품번호로 상품 정보 한개 얻어오기
81    public ProductVO getProduct(String pseq) {
82      ProductVO product = null;
83      String sql = "select * from product where pseq=?";
84
85      Connection con = null;
86      PreparedStatement pstmt = null;
87      ResultSet rs = null;
88
89      try {
90        con = DBManager.getConnection();
91        pstmt = con.prepareStatement(sql);
92        pstmt.setString(1, pseq);
93        rs = pstmt.executeQuery();
94        if (rs.next()) {
95          product = new ProductVO();
96          product.setPseq(rs.getInt("pseq"));
97          product.setName(rs.getString("name"));
98          product.setKind(rs.getString("kind"));
99          product.setPrice1(rs.getInt("price1"));
100         product.setPrice2(rs.getInt("price2"));
101         product.setPrice3(rs.getInt("price3"));
102         product.setContent(rs.getString("content"));
103         product.setImage(rs.getString("image"));
104         product.setUseyn(rs.getString("useyn"));
105         product.setBestyn(rs.getString("bestyn"));
106         product.setIndate(rs.getTimestamp("indate"));
107       }
108     } catch (Exception e) {
109       e.printStackTrace();
110     } finally {
111       DBManager.close(con, pstmt, rs);
112     }
113     return product;
114   }
115
116   // 상품종류별 상품 리스트 얻어오기
117   public ArrayList<ProductVO> listKindProduct(String kind) {
118     ArrayList<ProductVO> productList = new ArrayList<ProductVO>();
119     String sql= "select * from product where kind=?";
120
121     Connection conn = null;
122     PreparedStatement pstmt = null;
123     ResultSet rs = null;
```

```
124
125       try {
126         conn = DBManager.getConnection();
127         pstmt = conn.prepareStatement(sql);
128         pstmt.setString(1, kind);
129         rs = pstmt.executeQuery();
130
131         while (rs.next()) {
132           ProductVO product = new ProductVO();
133           product.setPseq(rs.getInt("pseq"));
134           product.setName(rs.getString("name"));
135           product.setPrice2(rs.getInt("price2"));
136           product.setImage(rs.getString("image"));
137           productList.add(product);
138         }
139       } catch (Exception e) {
140         e.printStackTrace();
141       } finally {
142         DBManager.close(conn, pstmt, rs);
143       }
144       return productList;
145     }
```

MVC 패턴에서는 컨트롤러를 통해서 모델과 뷰를 결정하기 때문에 Nonage Shop의 메인 화면을 확인하기 위해서는 다음과 같이 요청할 것입니다.

```
http://localhost:8181/web-study-12/NonageServlet?command=index
```

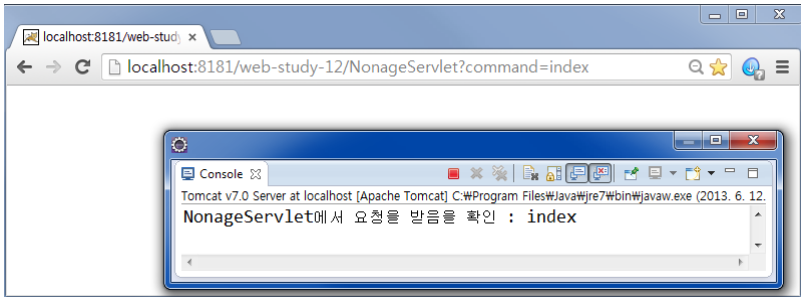
위와 같은 형태로 요청을 하기 위해서는 11장에서와 마찬가지로 MVC 패턴에서 컨트롤러(Controller)로서의 역할을 하는 서블릿을 만들어야 합니다.



[직접해보세요] MVC 패턴의 Controller로서의 역할을 하는 서블릿 만들기

[파일이름 : src₩com₩nonage₩controller₩NonageServlet.java]

```
1  package com.nonage.controller;
2
3  import java.io.IOException;
4  import javax.servlet.ServletException;
5  import javax.servlet.annotation.WebServlet;
6  import javax.servlet.http.HttpServlet;
7  import javax.servlet.http.HttpServletRequest;
8  import javax.servlet.http.HttpServletResponse;
9
10 @WebServlet("/NonageServlet")
11 public class NonageServlet extends HttpServlet {
12   private static final long serialVersionUID = 1L;
13   protected void doGet(HttpServletRequest request,
14         HttpServletResponse response) throws ServletException, IOException {
15     //요청시 보낸 파라미터 command 값을 얻어온다.
16     String command = request.getParameter("command");
17     //파라미터 command 값이 제대로 전달되었는지 확인차 출력
18     System.out.println(command);
19   }
20
21   protected void doPost(HttpServletRequest request,
22         HttpServletResponse response) throws ServletException, IOException {
23     //post 방식으로 요청시 한글 깨짐을 방지하기 위한 코드 추가
24     request.setCharacterEncoding("UTF-8");
25     //post 방식으로 요청되어도 doGet()를 호출하도록 하여
26     //doGet()에서만 요청에 대한 처리를 한다.
27     doGet(request, response);
28   }
29 }
```



요청이 들어오면 액션 객체에서 모든 비즈니스 로직을 수행하고 뷰를 결정합니다. 액션 클래스들은 요청에 대해서 동일한 메소드(execute)로 처리하도록 하기 위해서 인터페이스(Action)의 상속을 받도록 합니다.

다음은 요청에 대한 처리를 담당할 추상 메소드를 갖는 인터페이스를 만듭니다. 이렇게 만든 인터페이스를 모든 액션 클래스가 상속 받도록 할 것입니다.

[직접해보세요] 요청 처리를 추상 메소드를 위한 인터페이스 만들기

[파일이름 : src₩com₩nonage₩controller₩action₩Action.java]

```
1  package com.nonage.controller.action;
2
3  import java.io.IOException;
4
5  import javax.servlet.ServletException;
6  import javax.servlet.http.HttpServletRequest;
7  import javax.servlet.http.HttpServletResponse;
8
9  public interface Action {
10   public void execute(HttpServletRequest request, HttpServletResponse response)
11         throws ServletException, IOException;
12 }
```

요청이 들어왔을 때 어떤 액션 객체가 동작해야 하는지 판단하여 액션 객체를 생성해 주는 일을 담당하는 클래스인 ActionFactory를 만듭시다.

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 만들기

[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
1  package com.nonage.controller;
2
3  import com.nonage.controller.action.Action;
4
5  public class ActionFactory {
6    private static ActionFactory instance = new ActionFactory();
7    public ActionFactory() {
8      super();
9    }
10   public static ActionFactory getInstance() {
11     return instance;
12   }
13
14   public Action getAction(String command) {
15     Action action = null;
16     System.out.println("ActionFactory  :" + command);
17     return action;
18   }
19 }
```

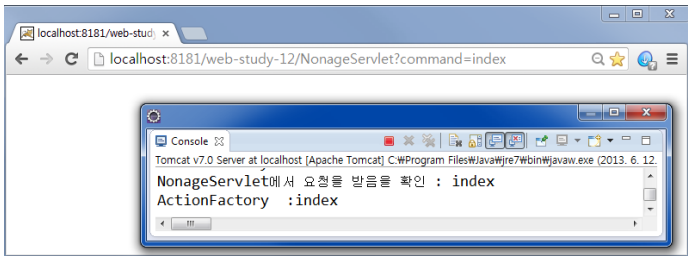
요청은 서블릿에서 1차적으로 받지만 받은 요청에 따라 어떤 액션 객체를 생성할 것인지는 ActionFactory 객체의 getAction() 메소드가 요청 파라미터에 의해서 결정하여 알려줍니다.

이미 만들어 놓은 서블릿에서 ActionFactory의 getAction()에 요청 파라미터 값을 넘겨주어 요청에 따른 액션 객체를 얻어온 후에 execute() 메소드를 호출하여 요청에 대한 처리가 일어나도록 합니다.

[직접해보세요] MVC 패턴의 Controller로서의 역할을 하는 서블릿 수정하기

[파일이름 : src₩com₩nonage₩controller₩NonageServlet.java]

```
1  package com.nonage.controller;
2
3  import java.io.IOException;
4
5  import javax.servlet.ServletException;
6  import javax.servlet.annotation.WebServlet;
7  import javax.servlet.http.HttpServlet;
8  import javax.servlet.http.HttpServletRequest;
9  import javax.servlet.http.HttpServletResponse;
10
11 import com.nonage.controller.action.Action;
12
13 @WebServlet("/NonageServlet")
14 public class NonageServlet extends HttpServlet {
15    private static final long serialVersionUID = 1L;
16
17    protected void doGet(HttpServletRequest request,
18          HttpServletResponse response) throws ServletException, IOException {
19       String command = request.getParameter("command");
20       System.out.println("NonageServlet에서 요청을 받음을 확인 : " + command);
21
22       ActionFactory af = ActionFactory.getInstance();
23       Action action = af.getAction(command);
24
25       if (action != null) {
26          action.execute(request, response);
27       }
28    }
29
30    protected void doPost(HttpServletRequest request,
31          HttpServletResponse response) throws ServletException, IOException {
32       request.setCharacterEncoding("UTF-8");
33       doGet(request, response);
34    }
35 }
```



ActionFactory의 getAction() 메소드에서 요청에 따라 액션 객체가 생성되어야 하기 때문에 이를 위한 액션 클래스를 설계합시다. index란 요청에 대해서는 메인 화면이 출력되어야 합니다. 메인 화면에서는 신상품과 베스트 상품이 진열되어야 하기 때문에 데이터베이스에서 상품 정보 얻어오는 비즈니스 로직을 수행한 후에 메인 페이지인 index.jsp로 이동하도록 해야 합니다. 다음은 메인 페이지를 위한 요청입니다.

```
http://localhost:8181/web-study-12/NonageServlet?command=index
```

NonageServlet?command=index라는 요청에 의해서 신상품과 베스트 상품을 메인 페이지에 게시하도록 하는 액션 클래스를 만듭시다.

[직접해보세요] 메인 화면을 출력하기 위한 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩IndexActoin.java]

```
1  package com.nonage.controller.action;
2
3  import java.io.IOException;
4
5  import javax.servlet.RequestDispatcher;
6  import javax.servlet.ServletException;
7  import javax.servlet.http.HttpServletRequest;
8  import javax.servlet.http.HttpServletResponse;
9
10 public class IndexAction implements Action {
11
12   @Override
13   public void execute(HttpServletRequest request, HttpServletResponse response)
14         throws ServletException, IOException {
15      String url = "/index.jsp";
16      /* 데이터베이스에서 상품 정보 얻어오는 비즈니스 로직*/
17      ProductDAO productDAO=ProductDAO.getInstance();
18      ArrayList<ProductVO> newProductList = productDAO.listNewProduct();
19      ArrayList<ProductVO> bestProductList = productDAO.listBestProduct();
20
21      request.setAttribute("newProductList", newProductList);
22      request.setAttribute("bestProductList", bestProductList);
23
24      RequestDispatcher dispatcher = request.getRequestDispatcher(url);
25      dispatcher.forward(request, response);
```

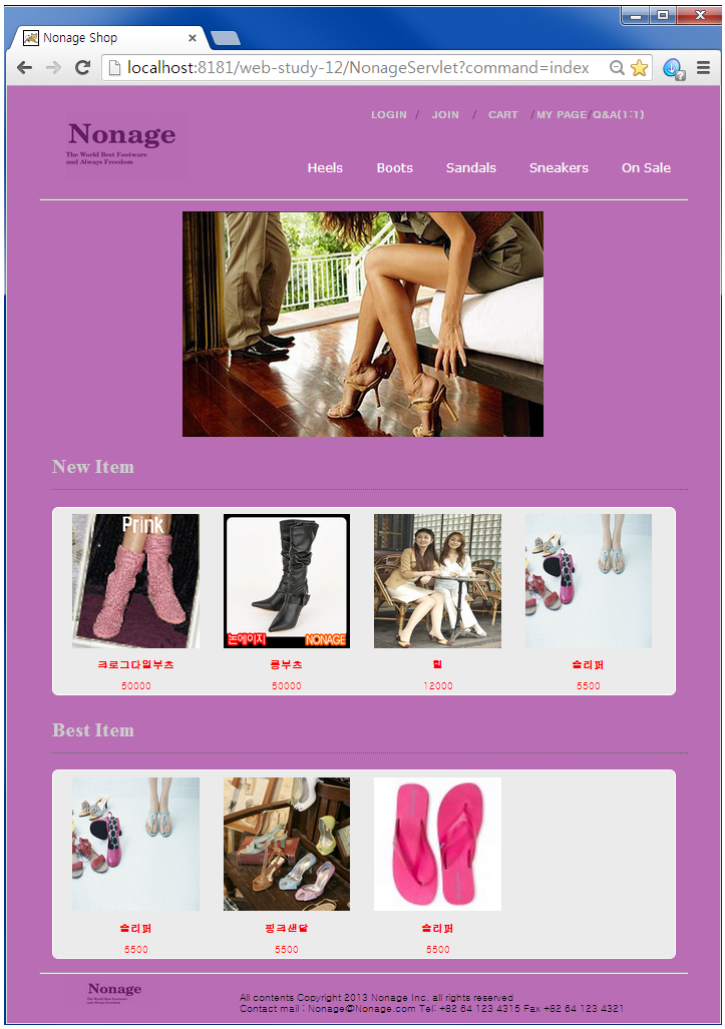
```
26    }
27  }
```

index란 커맨드(command)에 대해 알맞은 작업을 처리하기 위한 액션 객체를 ActionFactory의 getAction()에서 생성될 수 있도록 다음과 같이 코드를 추가합시다.

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 Action Factory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
1  package com.nonage.controller;
2
3  import com.nonage.controller.action.*;
4
5  public class ActionFactory {
6    private static ActionFactory instance = new ActionFactory();
7    private ActionFactory() {
8      super();
9    }
10   public static ActionFactory getInstance() {
11     return instance;
12   }
13
14   public Action getAction(String command) {
15     Action action = null;
16     System.out.println("ActionFactory  :" + command);
17     /* 추가된 부분 */
18     if (command.equals("index")) {
19       action = new IndexAction();
20     }
21     return action;
22   }
23 }
```

MVC 패턴에서는 요청을 서블릿에 보내서 적당한 액션 객체가 생성되어 execute() 메소드가 실행되어 데이터베이스에서 상품 정보를 얻어온 값을 뷰(jsp)에 전달하여 데이터베이스에 저장된 상품 정보를 사용자가 볼 수 있도록 합니다.


Nonage Shop
localhost:8181/web-study-12/NonageServlet?command=index
Nonage
The World Best Footware and Always Freedom
LOGIN / JOIN / CART / MY PAGE/ Q&A(1:1)
Heels Boots Sandals Sneakers On Sale
New Item
Prink
크로그다일부츠
50000
롱부츠
50000
힐
12000
슬리퍼
5500
Best Item
슬리퍼
5500
핑크샌달
5500
슬리퍼
5500
Nonage
All contents Copyright 2013 Nonage Inc. all rights reserved
Contact mail : Nonage@Nonage.com Tel: +82 64 123 4315 Fax +82 64 123 4321

12-4 상품 상세보기 페이지 만들기

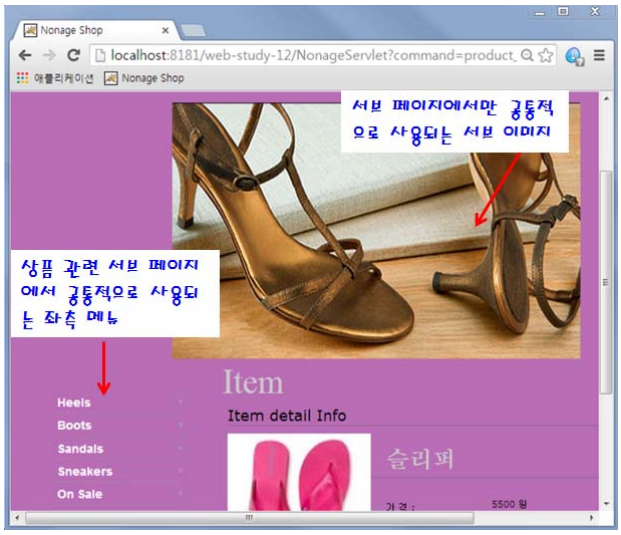
특정 상품을 클릭하면 상품 상세보기 화면으로 이동합니다. 상품을 클릭하면 "NonageServlet?command=product_detail&pseq=${productVO.pseq}"가 요청되어 상품 상세 보기 페이지로 이동해야 하기 때문에 액션 클래스를 만들고 이 요청을 처리할 코드를 ActionFactory에 추가합니다. 액션 클래스에서는 상품 번호로 상품 정보 하나를 얻어오기 위한 메소드를 호출해야 합니다.

[직접해보세요] 상품 상세 보기 페이지로 이동하도록 하는 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩ProductDetailAction.java]

```
 1  package com.nonage.controller.action;
 2
 3  import java.io.IOException;
 4
 5  import javax.servlet.RequestDispatcher;
 6  import javax.servlet.ServletException;
 7  import javax.servlet.http.HttpServletRequest;
 8  import javax.servlet.http.HttpServletResponse;
 9
10  import com.nonage.dao.ProductDAO;
11  import com.nonage.dto.ProductVO;
12
13  public class ProductDetailAction implements Action {
14
15    @Override
16    public void execute(HttpServletRequest request, HttpServletResponse response)
17          throws ServletException, IOException {
18      String url="product/productDetail.jsp";
19
20      String pseq=request.getParameter("pseq").trim();
21
22      ProductDAO productDAO=ProductDAO.getInstance();
23      ProductVO productVO= productDAO.getProduct(pseq);
24
25      request.setAttribute("productVO", productVO);
26
27      RequestDispatcher dispatcher = request.getRequestDispatcher(url);
28      dispatcher.forward(request, response);
29    }
30  }
```

화면 상단의 주 메뉴에 따라 화면 좌측의 서브 메뉴는 달라집니다. 하지만 동일한 주 메뉴 내에서는 공통된 서브 페이지가 제공됩니다. 다음은 상품 관련 페이지에서만 공통적으로 사용되는 서브 이미지와 서브 페이지입니다.



서브 페이지에서만 공통적으로 사용되는 파일은 다음과 같습니다.

[서브 페이지에서만 공통적으로 적용되는 파일]

파일이름	설명
sub_img.html	서브 페이지에서 이미지를 출력하는 파일
sub_menu.html	서브 페이지에서 좌측 메뉴를 구성하는 파일

우선 상품 상세 보기를 작성하기에 앞서 상품 정보를 위한 페이지에서 공통적으로 사용되는 페이지부터 만들어 봅시다. 상품과 관련된 작업을 위해서 사용하는 모든 페이지는 product 폴더에서 관리되기 때문에 product 폴더를 생성하고 여기에 sub_img.html과 sub_menu.html 파일을 추가합니다.

[직접해보세요] 서브 페이지에서 이미지를 출력하는 파일

[파일이름 : WebContent/product/sub_img.html]

```
1  <div id="sub_img">
2    <img src="images/product/sub_img.jpg" >
3  </div>
4  <div class="clear"></div>
```

[직접해보세요] 서브 페이지에서 좌측 메뉴를 구성하는 파일

[파일이름 : WebContent/product/sub_menu.html]

```
1  <nav id="sub_menu">
2    <ul>
3      <li><a href="NonageServlet?command=catagory&kind=1">Heels</a></li>
4      <li><a href="NonageServlet?command=catagory&kind=2">Boots</a></li>
5      <li><a href="NonageServlet?command=catagory&kind=3">Sandals</a></li>
6      <li><a href="NonageServlet?command=catagory&kind=4">Sneakers</a></li>
7      <li><a href="NonageServlet?command=catagory&kind=5"> On Sale</a></li>
8    </ul>
```

```
9  </nav>
```

[직접해보세요] 상품 상세 보기를 위한 JSP 페이지

[파일이름 : WebContent/product/productDetail.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2    pageEncoding="UTF-8"%>
3  <%@ include file="../header.jsp" %>
4  <%@ include file="sub_img.html"%>
5  <%@ include file="sub_menu.html" %>
6    <article>
7      <h1> Item </h1>
8      <div id="itemdetail" >
9        <form  method="post" name="formm">
10         <fieldset>
11           <legend> Item detail Info</legend>
12           <a href=
13 "NonageServlet?command=product_detail&pseq=${productVO.pseq}">
14             <span style="float: left;">
15               <img  src="product_images/${productVO.image}"  />
16             </span>
17             <h2> ${productVO.name} </h2>
18           </a>
19           <label> 가 격 :  </label>
20           <p> ${productVO.price2} 원</p>
21           <label> 수 량 : </label>
22           <input  type="text"       name="quantity"  size="2"       value="1"><br>
23           <input  type="hidden"     name="pseq"
24         value="${productVO.pseq}"><br>
25         </fieldset>
26         <div class="clear"></div>
27         <div id="buttons">
28           <input type="button" value="장바구니에 담기"   class="submit"
29     onclick="go_cart()">
30           <input type="button" value="즉시 구매"         class="submit"
31 onclick="go_order()">
32           <input type="reset"  value="취소"              class="cancel">
33         </div>
34       </form>
35     </div>
36   </article>
37 <%@ include file="../footer.jsp" %>
```

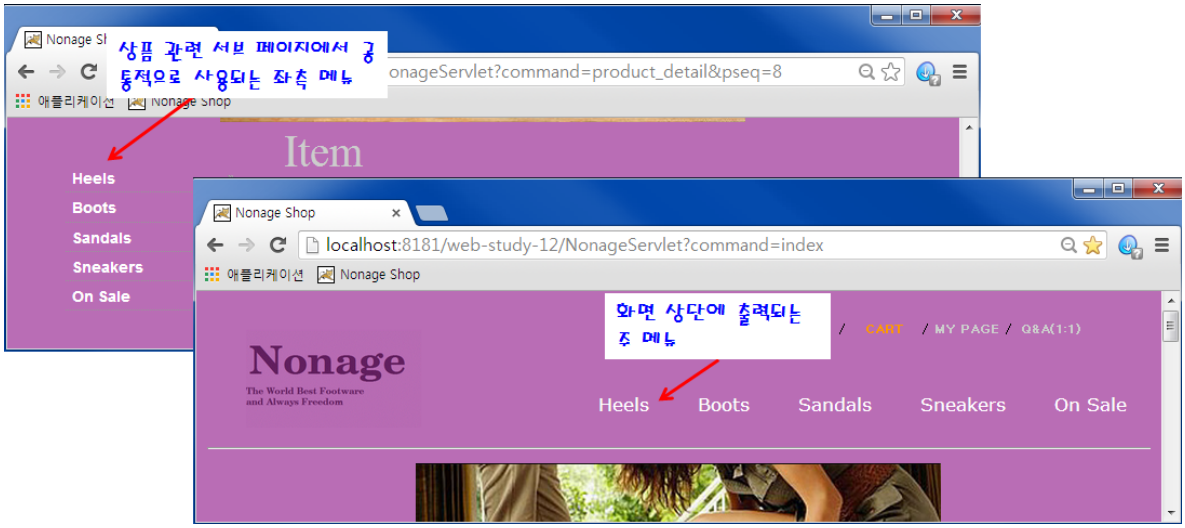
[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
```

```
20      } else if (command.equals("product_detail")) {
21          action = new ProductDetailAction();
22      }
..  //중복되는 내용 생략
```

12-5 메뉴별 페이지 만들기

화면 상단에 출력되는 주 메뉴와 특정 상품을 클릭하면 나타나는 상품 상세보기 페이지의 좌측의 서브 메뉴에서는 카테고리가 나타납니다. 여기서 특정 카테고리를 선택하면 상품 분류별 상품 리스트를 출력해주는 화면으로 이동할 수 있습니다.



카테고리를 클릭하면 "NonageServlet?command=catagory&kind=1"가 요청되어 분류별 상품 리스트를 출력해주는 화면으로 이동해야 하기 때문에 액션 클래스를 만들고 이 요청을 처리할 코드를 ActionFactory에 추가합니다. 액션 클래스에서는 카테고리 번호로 상품 리스트를 얻어오기 위한 메소드를 호출해야 합니다.

[직접해보세요] 상품 분류별 상품 리스트 페이지로 이동하도록 하는 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩ProductKindAction.java]

```
 1  package com.nonage.controller.action;
 2
 3  import java.io.IOException;
 4  import java.util.ArrayList;
 5
 6  import javax.servlet.RequestDispatcher;
 7  import javax.servlet.ServletException;
 8  import javax.servlet.http.HttpServletRequest;
 9  import javax.servlet.http.HttpServletResponse;
10
11  import com.nonage.dao.ProductDAO;
12  import com.nonage.dto.ProductVO;
13
14  public class ProductKindAction implements Action {
15
16    @Override
17    public void execute(HttpServletRequest request, HttpServletResponse response)
18        throws ServletException, IOException {
```

```
19        String url="product/productKind.jsp";
20
21        String kind=request.getParameter("kind").trim();
22
23        ProductDAO productDAO=ProductDAO.getInstance();
24        ArrayList<ProductVO>productKindList= productDAO.listKindProduct(kind);
25
26        request.setAttribute("productKindList", productKindList);
27        RequestDispatcher dispatcher = request
28              .getRequestDispatcher(url);
29        dispatcher.forward(request, response);
30     }
31  }
```

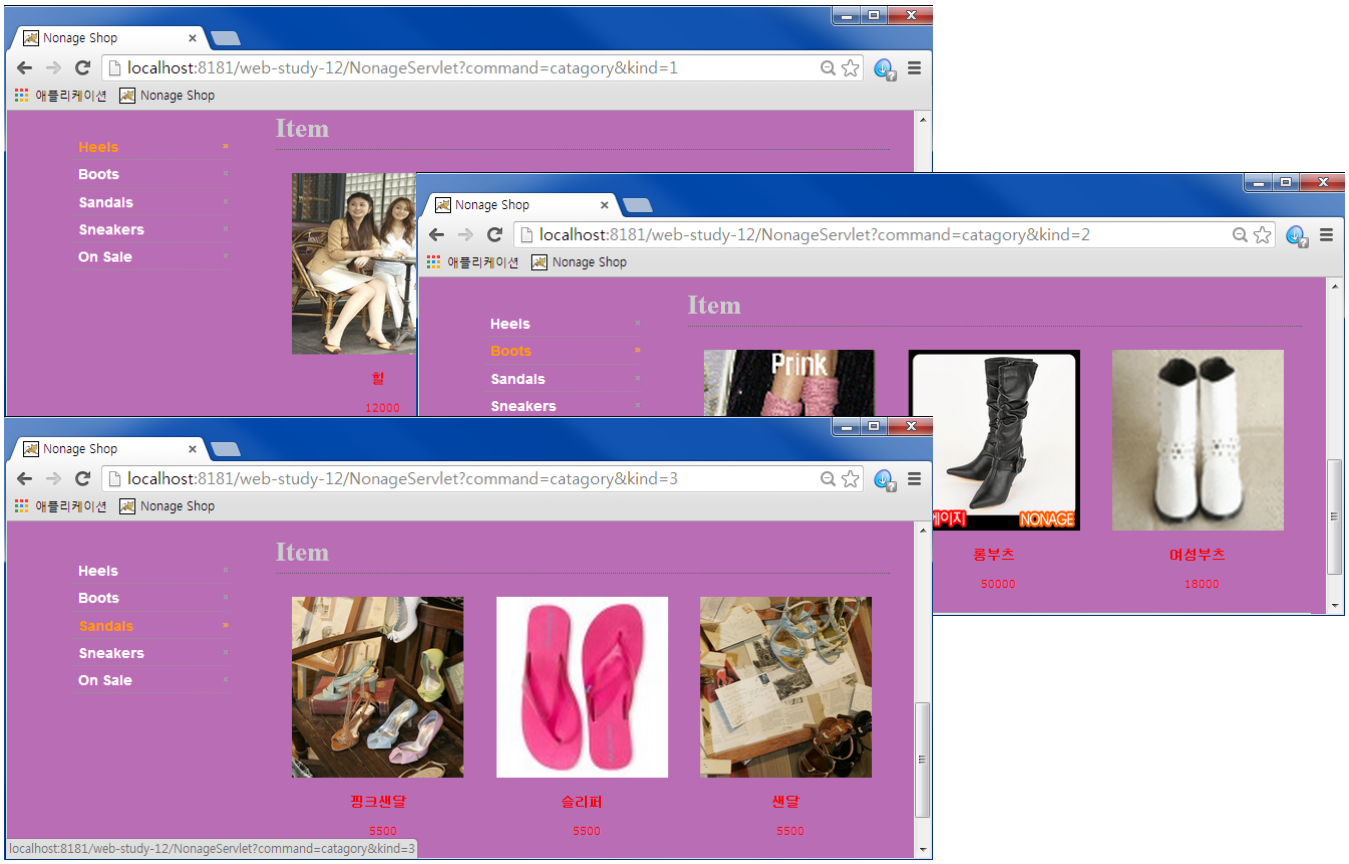
[직접해보세요] 상품 분류별 상품 리스트를 위한 JSP 페이지

[파일이름 : WebContent/product/productKind.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3  <%@ include file="../header.jsp" %>
4  <%@ include file="sub_img.html"%>
5  <%@ include file="sub_menu.html" %>
6     <article>
7        <h2> Item</h2>
8        <c:forEach items="${productKindList }"  var="productVO">
9           <div id="item">
10             <a href=
11 "NonageServlet?command=product_detail&pseq=${productVO.pseq}">
12                <img src="product_images/${productVO.image}" />
13                <h3>${productVO.name} </h3>
14                <p>${productVO.price2} </p>
15             </a>
16          </div>
17       </c:forEach>
18       <div class="clear"></div>
19    </article>
20 <%@ include file="../footer.jsp" %>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
22       } else if (command.equals("catagory")) {
23             action = new ProductKindAction();
24       }
..  //중복되는 내용 생략
```



위 그림은 상품 분류별 상품 리스트 페이지로 이동한 결과입니다.

12-6 회원 가입 및 로그인

각 메뉴별 상품 진열과 상품 상세보기 등을 완성하였습니다. 이제 회원에 관련된 부분을 진행해 보도록 하겠습니다.

회원 가입을 위해서는 데이터베이스에 저장된 member 테이블에 회원 정보를 저장해야 하고 로그인 처리를 위해서는 데이터베이스에서 가입한 회원인지를 조회해야 합니다. 이를 member 테이블을 처리하기 위한 DAO를 작성해야 합니다. 회원 관련 처리를 위한 DAO를 만들기 위해서 우선 데이터베이스에서 얻어온 회원 정보를 담을 VO부터 만듭시다.

[직접해보세요] 이클립스에서 회원 정보를 저장하는 VO 클래스 만들기
[파일 이름 : com₩nonage₩dto₩MemberVO.java]

```
1  package com.nonage.dto;
2
3  import java.sql.Timestamp;
4
5  public class MemberVO {
6    private String id;
7    private String pwd;
8    private String name;
9    private String email;
10   private String zipNum;
11   private String address;
12   private String phone;
13   private String useyn;
14   private Timestamp indate;
15
16   public String getId() {
17     return id;
18   }
19   public void setId(String id) {
20     this.id = id;
21   }
22   public String getPwd() {
23     return pwd;
24   }
25   public void setPwd(String pwd) {
26     this.pwd = pwd;
27   }
28   public String getName() {
29     return name;
30   }
31   public void setName(String name) {
32     this.name = name;
33   }
34   public String getEmail() {
```

```
35      return email;
36    }
37    public void setEmail(String email) {
38      this.email = email;
39    }
40    public String getZipNum() {
41      return zipNum;
42    }
43    public void setZipNum(String zipNum) {
44      this.zipNum = zipNum;
45    }
46
47    public String getAddress() {
48      return address;
49    }
50    public void setAddress(String address) {
51      this.address = address;
52    }
53    public String getPhone() {
54      return phone;
55    }
56    public void setPhone(String phone) {
57      this.phone = phone;
58    }
59    public String getUseyn() {
60      return useyn;
61    }
62    public void setUseyn(String useyn) {
63      this.useyn = useyn;
64    }
65    public Timestamp getIndate() {
66      return indate;
67    }
68    public void setIndate(Timestamp indate) {
69      this.indate = indate;
70    }
71  }
```

회원 관리를 위한 MemberDAO를 만듭시다.

[직접해보세요] 회원 테이블을 액세스하는 DAO 클래스 만들기

[파일 이름 :src₩com₩nonage₩dao₩MemberDAO.java]

```
1   package com.nonage.dao;
2
3   import java.sql.Connection;
4   import java.sql.PreparedStatement;
5   import java.sql.ResultSet;
6
```

```
 7  import util.DBManager;
 8
 9  import com.nonage.dto.MemberVO;
10
11  public class MemberDAO {
12
13    private MemberDAO() {
14    }
15
16    private static MemberDAO instance = new MemberDAO();
17
18    public static MemberDAO getInstance() {
19      return instance;
20    }
21
22    public int confirmID(String userid) {
23            int result = -1;
24      String sql = "select * from member where id=?";
25
26      Connection connn = null;
27      PreparedStatement pstmt = null;
28      ResultSet rs = null;
29
30      try {
31        connn = DBManager.getConnection();
32        pstmt = connn.prepareStatement(sql);
33        pstmt.setString(1, userid);
34        rs = pstmt.executeQuery();
35        if (rs.next()) {
36          result = 1;
37        } else {
38          result = -1;
39        }
40      } catch (Exception e) {
41        e.printStackTrace();
42      } finally {
43        DBManager.close(connn, pstmt, rs);
44      }
45      return result;
46    }
47
48    public MemberVO getMember(String id) {
49      MemberVO memberVO= null;
50      String sql = "select * from member where id=?";
51
52      Connection connn = null;
53      PreparedStatement pstmt = null;
54      ResultSet rs = null;
```

```
55
56      try {
57        connn = DBManager.getConnection();
58        pstmt = connn.prepareStatement(sql);
59        pstmt.setString(1, id);
60        rs = pstmt.executeQuery();
61        if(rs.next()){
62          memberVO = new MemberVO();
63          memberVO.setId(rs.getString("id"));
64          memberVO.setPwd(rs.getString("pwd"));
65          memberVO.setName(rs.getString("name"));
66          memberVO.setEmail(rs.getString("email"));
67          memberVO.setZipNum(rs.getString("zip_num"));
68          memberVO.setAddress(rs.getString("address"));
69          memberVO.setPhone(rs.getString("phone"));
70          memberVO.setUseyn(rs.getString("useyn"));
71          memberVO.setIndate(rs.getTimestamp("indate"));
72        }
73      } catch (Exception e) {
74        e.printStackTrace();
75      } finally {
76        DBManager.close(connn, pstmt, rs);
77      }
78      return memberVO;
79    }
80
81    public int insertMember(MemberVO memberVO) {
82      int result = 0;
83      String sql = "insert into member(id, pwd, name, zip_num,";
84      sql += " address, phone) values(?, ?, ?, ?, ?, ?)";
85
86      Connection conn = null;
87      PreparedStatement pstmt = null;
88
89      try {
90        conn = DBManager.getConnection();
91        pstmt = conn.prepareStatement(sql);
92        pstmt.setString(1, memberVO.getId());
93        pstmt.setString(2, memberVO.getPwd());
94        pstmt.setString(3, memberVO.getName());
95        pstmt.setString(4, memberVO.getZipNum());
96        pstmt.setString(5, memberVO.getAddress());
97        pstmt.setString(6, memberVO.getPhone());
98        result = pstmt.executeUpdate();
99      } catch (Exception e) {
100       e.printStackTrace();
101     } finally {
102       DBManager.close(conn, pstmt);
```

```
    }
    return result;
  }
}
```

DAO 클래스를 작성하였으면 회원 가입 처리를 위한 페이지에서 공통적으로 사용하는 페이지부터 만들어 봅시다. 화면 상단의 주 메뉴에 따라 화면 좌측의 서브 메뉴는 달라집니다. 상품과 관련된 작업에서 공통적으로 사용되는 서브 이미지와 서브 메뉴를 위한 페이지는 product 폴더에 저장해 두었지만 회원과 관련된 작업을 위해서 사용하는 모든 페이지는 member 폴더에서 관리되기 때문에 이번에 작성할 회원 관련 공통 페이지인 sub_img.html과 sub_menu.html 파일은 member 폴더를 생성하고 여기에 추가합니다.

[직접해보세요] 서브 페이지에서 이미지를 출력하는 파일

[파일이름 : WebContent/member/sub_img.html]

```
<div id="sub_img">
  <img src="images/member/sub_img.jpg" >
</div>
<div class="clear"></div>
```

[직접해보세요] 서브 페이지에서 좌측 메뉴를 구성하는 파일

[파일이름 : WebContent/member/sub_menu.html]

```
<nav id="sub_menu">
  <ul>
    <li><a href="NonageServlet?command=login_form">Login</a></li>
    <li><a href="NonageServlet?command=contract">Join us</a></li>
  </ul>
</nav>
```

회원 가입을 할 때 폼에 입력된 정보가 올바른지 판단하는 자바스크립트 파일을 WebContent의 하위 폴더인 member에 member.js란 이름으로 작성합니다.

자바스크립트 파일은 이미 header.jsp 페이지에서 포함하고 있습니다.

header.jsp

```
<head>
  <meta charset="UTF-8">
  <title>Nonage Shop</title>
  <link href="css/shopping.css" rel="stylesheet">
  <script type="text/javascript" src="member/member.js"></script>
  <script type="text/javascript" src="mypage/mypage.js"></script>
</head>
```

[직접해보세요] 폼에 입력된 정보가 올바른지 판단하는 자바스크립트 파일

[파일이름 : WebContent/member/member.js]

```
1  function go_save() {
2    if (document.formm.id.value == "") {
3      alert("아이디를 입력하여 주세요.");
4      document.formm.id.focus();
5    } else if (document.formm.id.value != document.formm.reid.value) {
6      alert("중복확인을 클릭하여 주세요.");
7      document.formm.id.focus();
8    } else if (document.formm.pwd.value == "") {
9      alert("비밀번호를 입력해 주세요.");
10     document.formm.pwd.focus();
11   } else if ((document.formm.pwd.value != document.formm.pwdCheck.value)) {
12     alert("비밀번호가 일치하지 않습니다.");
13     document.formm.pwd.focus();
14   } else if (document.formm.name.value == "") {
15     alert("이름을 입력해 주세요.");
16     document.formm.name.focus();
17   } else if (document.formm.email.value == "") {
18     alert("이메일을 입력해 주세요.");
19     document.formm.email.focus();
20   } else {
21     document.formm.action = "NonageServlet?command=join";
22     document.formm.submit();
23   }
24 }
25
26 function idcheck() {
27   if (document.formm.id.value == "") {
28     alert('아이디를 입력하여 주십시오.');
29     document.formm.id.focus();
30     return;
31   }
32   var url = "NonageServlet?command=id_check_form&id="
33 + document.formm.id.value;
34   window.open( url, "_blank_1",
35 "toolbar=no, menubar=no, scrollbars=yes, resizable=no, width=330, height=200");
36 }
37
38 function post_zip() {
39   var url = "NonageServlet?command=find_zip_num";
40   window.open( url, "_blank_1",
41 "toolbar=no, menubar=no, scrollbars=yes, resizable=no, width=550, height=300,
42 top=300, left=300, ");
43 }
44
45 function go_next() {
46   if (document.formm.okon1[0].checked == true) {
47     document.formm.action = "NonageServlet?command=join_form";
```

```
48        document.formm.submit();
49     } else if (document.formm.okon1[1].checked == true) {
50        alert('약관에 동의하셔야만 합니다.');
51     }
52  }
```

회원 가입을 위해서 화면 상단의 주 메뉴에서 join을 클릭하면 이용약관이 출력되는 페이지로 이동해야 하기 위해서 “NonageServlet?command=contract” 요청이 발생합니다. 이 요청을 받으면 약관 동의를 위한 페이지로 이동하는 액션을 만들어 봅시다.

[직접해보세요] 약관 동의를 위한 페이지로 이동하는 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩ContractAction.java]

```
1  package com.nonage.controller.action;
2
3  import java.io.IOException;
4
5  import javax.servlet.RequestDispatcher;
6  import javax.servlet.ServletException;
7  import javax.servlet.http.HttpServletRequest;
8  import javax.servlet.http.HttpServletResponse;
9
10 public class ContractAction implements Action {
11
12    @Override
13    public void execute(HttpServletRequest request, HttpServletResponse response)
14          throws ServletException, IOException {
15       String url = "/member/contract.jsp";
16
17       RequestDispatcher dispatcher=request.getRequestDispatcher(url);
18       dispatcher.forward(request, response);
19    }
20 }
```

[직접해보세요] 약관 동의를 위한 JSP 페이지

[파일이름 : WebContent/member/contract.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2       pageEncoding="UTF-8"%>
3  <%@ include file="../header.jsp" %>
4  <%@ include file="sub_img.html"%>
5  <%@ include file="sub_menu.html" %>
6    <article>
7       <h2>Join Us</h2>
8       <form id="join" action="NonageServlet?command=join_form"
9  method="post" name="formm">
10         언제나 새로운 즐거움이 가득한 Nonage의 회원가입 페이지 입니다. <br>
```

```
11          Nonage의 회원가입은 무료이며, 회원님의 개인신상에 관한 정보는
12          '정보통신망이용촉진 및 정보보호등에관한법률'에 의해 회원님의 동의없이
13          제 3자에게 제공되지 않으며, 철저히 보호되고 있사오니 안심하고 이용하시기 바랍
14  니다.<br><br>
15          <textarea rows="15" cols="100">
16          제 1 장 총 칙
17
18  제 1 조 (목적)
19  이 이용약관(이하 '약관')은 주식회사 Nonage(이하 회사라 합니다)
20  와 이용 고객(이하 '회원')간에 회사가 제공하는 jjang0u.com서비스
21  의 가입조건 및 이용에 관한 다음의 제반 사항과 기타 기본적인 사
22  항을 구체적으로 규정함을 목적으로 합니다.
23  //중략
..
188         </textarea>
189         <br><br>
190         <div style="text-align: center;">
191         <input type="radio" name="okon1" checked> 동의함      
192         <input type="radio" name="okon1" checked> 동의안함
193         </div>
194         <input type="button" value="Next" class="submit"
195 onclick="go_next()" style="float: right;">
196       </form>
197     </article>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
26      } else if (command.equals("contract")) {
27        action = new ContractAction();
28      }
..  //중복되는 내용 생략
```

약관을 동의하고 [Next] 버튼을 클릭하면 "NonageServlet?command=join_form" 요청이 발생합니다. 이 요청이 발생하면 회원 가입을 위한 폼으로 이동합니다. 회원 정보를 입력 받을 폼을 작성합시다.

[직접해보세요] 회원 가입을 위한 폼으로 이동하도록 하는 액션 클래스 [파일이름 : src₩com₩nonage₩controller₩action₩JoinFormAction.java]

```
1   package com.nonage.controller.action;
2
3   import java.io.IOException;
4
5   import javax.servlet.RequestDispatcher;
6   import javax.servlet.ServletException;
7   import javax.servlet.http.HttpServletRequest;
```

```
8  import javax.servlet.http.HttpServletResponse;
9
10
11 public class JoinFormAction implements Action {
12
13   @Override
14   public void execute(HttpServletRequest request, HttpServletResponse response)
15       throws ServletException, IOException {
16     String url = "/member/join.jsp";
17
18     RequestDispatcher dispatcher=request.getRequestDispatcher(url);
19     dispatcher.forward(request, response);
20   }
21 }
```

[직접해보세요] 회원 가입을 위한 폼을 출력하는 JSP 파일

[파일이름 : WebContent/member/join.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <%@ include file="../header.jsp" %>
4  <%@ include file="sub_img.html"%>
5  <%@ include file="sub_menu.html" %>
6    <article>
7      <h2>Join Us</h2>
8      <form id="join" action="NonageServlet?command=join" method="post"
9  name="formm">
10       <fieldset>
11         <legend>Basic Info</legend>
12         <label>User ID</label>
13         <input type="text" name="id" size="12" >
14         <input type="hidden" name="reid">
15         <input type="button" value="중복 체크" class="dup"
16 onclick="idcheck()"><br>
17         <label>Password</label>
18         <input type="password" name="pwd"><br>
19         <label>Retype Password</label>
20         <input type="password" name="pwdCheck"><br>
21         <label>Name</label>
22         <input type="text" name="name"><br>
23         <label>E-Mail</label>
24         <input type="text" name="email"><br>
25
26       </fieldset>
27       <fieldset>
28         <legend>Optional</legend>
29         <label>Zip Code</label>
30         <input type="text" name="zipNum" size="10" >
```

```
31            <input type="button"       value="주소 찾기" class="dup"
32  onclick="post_zip()"><br>
33            <label>Address</label>
34            <input type="text"          name="addr1"    size="50">
35            <input type="text"          name="addr2"    size="25"> <br>
36            <label>Phone Number</label>
37            <input   type="text"        name="phone"><br>
38          </fieldset>
39          <div class="clear"></div>
40          <div id="buttons">
41            <input type="button"     value="회원가입"    class="submit"
42  onclick="go_save()">
43            <input type="reset"       value="취소"       class="cancel">
44          </div>
45        </form>
46    </article>
47  <%@ include file="../footer.jsp" %>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
24      } else if (command.equals("join_form")) {
25          action = new JoinFormAction();
26      }
..  //중복되는 내용 생략
```

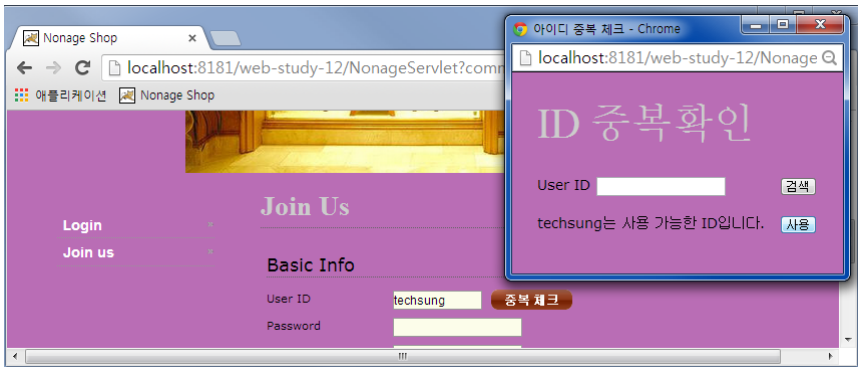
회원 가입이 제대로 이루어지려면 아이디 중복 체크를 해야 합니다. 아이디를 입력하고 나서 [중복 체크] 버튼을 클릭하면 “NonageServlet?command=id_check_form” 요청이 발생합니다. 이 요청에 대한 처리를 하기 위한 액션 클래스를 작성합시다.

[직접해보세요] 아이디 중복 체크를 위한 액션 클래스
[파일이름 : src₩com₩nonage₩controller₩action₩IdCheckFormAction.java]

```
1   package com.nonage.controller.action;
2
3   import java.io.IOException;
4
5   import javax.servlet.RequestDispatcher;
6   import javax.servlet.ServletException;
7   import javax.servlet.http.HttpServletRequest;
8   import javax.servlet.http.HttpServletResponse;
9
10  import com.nonage.dao.MemberDAO;
11
12  public class IdCheckFormAction implements Action {
13
```

```
14    @Override
15    public void execute(HttpServletRequest request, HttpServletResponse response)
16          throws ServletException, IOException {
17       String url = "/member/idcheck.jsp";
18
19       String id = request.getParameter("id").trim();
20
21       MemberDAO memberDAO=MemberDAO.getInstance();
22       int message = memberDAO.confirmID(id);
23
24       request.setAttribute("message", message);
25       request.setAttribute("id", id);
26       RequestDispatcher dispatcher = request.getRequestDispatcher(url);
27       dispatcher.forward(request, response);
28    }
29 }
```

다음은 아이디 중복 체를 위한 과정입니다.



[직접해보세요] 아이디 중복 체크를 위한 폼을 출력하는 JSP 파일

[파일이름 : WebContent/member/idcheck.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3  <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
4  <!DOCTYPE html>
5  <html>
6  <head>
7  <meta charset="UTF-8">
8  <title>아이디 중복 체크</title>
9  <link href="CSS/subpage.css" rel="stylesheet">
10 <style type="text/css">
11 body{
12    background-color:#B96DB5;
13    font-family: Verdana;
14 }
15 #wrap{
```

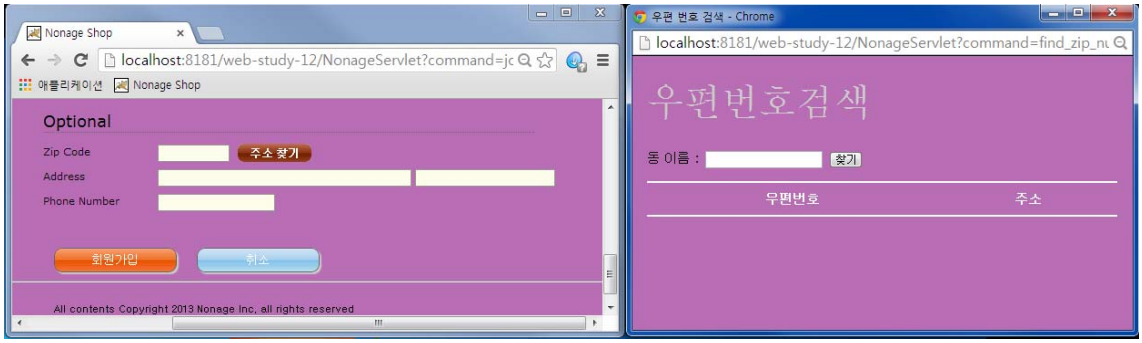
```
16    margin: 0 20px;
17  }
18  h1 {
19    font-family: "Times New Roman", Times, serif;
20    font-size: 45px;
21    color: #CCC;
22    font-weight: normal;
23  }
24  input[type=button], input[type=submit] {
25    float: right;
26  }
27  </style>
28  <script type="text/javascript">
29  function idok(){
30    opener.formm.id.value="${id}";
31    opener.formm.reid.value="${id}";
32    self.close();
33  }
34  </script>
35  </head>
36  <body>
37  <div id="wrap">
38    <h1>ID 중복확인</h1>
39    <form method=post name=formm style="margin-right:0 "
40  action="NonageServlet?command=id_check_form" >
41        User ID <input type=text name="id" value="">
42                <input type=submit value="검색" class="submit"><br>
43      <div style="margin-top: 20px">
44        <c:if test="${message == 1}">
45          <script type="text/javascript">
46            opener.document.formm.id.value="";
47          </script>
48          ${id}는 이미 사용중인 아이디입니다.
49        </c:if>
50        <c:if test="${message==-1}">
51          ${id}는 사용 가능한 ID입니다.
52          <input type="button" value="사용" class="cancel" onclick="idok()">
53        </c:if>
54      </div>
55    </form>
56  </div>
57  </body>
58  </html>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
```

```
28      } else if (command.equals("id_check_form")) {
29          action = new IdCheckFormAction();
30      }
..  //중복되는 내용 생략
```

회원가입을 위해서는 주소를 입력해야 합니다. 주소를 직접 입력하기 보다는 주소 테이블에서 우편 번호로 주소를 찾아 자동 입력하도록 합시다.



주소 검색을 위한 주소 테이블은 데이터베이스 구축을 하면서 이미 구축되어 있는 상태입니다. 이 테이블에서 우편 번호를 검색하기 위한 DAO를 만듭시다. 우선 주소 정보를 담을 VO부터 만듭시다.

[직접해보세요] 이클립스에서 주소 정보를 저장하는 VO 클래스 만들기

[파일 이름 : com₩nonage₩dto₩AddressVO.java]

```
1   package com.nonage.dto;
2
3   public class AddressVO {
4     private String zipNum;
5     private String sido;
6     private String gugun;
7     private String dong;
8     private String zipCode;
9     private String bunji;
10
11    public String getzipNum() {
12      return zipNum;
13    }
14    public void setzipNum(String zipNum) {
15      this.zipNum = zipNum;
16    }
17    public String getSido() {
18      return sido;
19    }
20    public void setSido(String sido) {
21      this.sido = sido;
22    }
23    public String getGugun() {
```

```
24      return gugun;
25    }
26    public void setGugun(String gugun) {
27      this.gugun = gugun;
28    }
29    public String getDong() {
30      return dong;
31    }
32    public void setDong(String dong) {
33      this.dong = dong;
34    }
35    public String getzipCode() {
36      return zipCode;
37    }
38    public void setzipCode(String zipCode) {
39      this.zipCode = zipCode;
40    }
41    public String getBunji() {
42      return bunji;
43    }
44    public void setBunji(String bunji) {
45      this.bunji = bunji;
46    }
47  }
```

주소 찾기를 위한 회원 AddressDAO를 만듭시다.

[직접해보세요] 주소 테이블을 액세스하는 DAO 클래스 만들기

[파일 이름 :src₩com₩nonage₩dao₩AddressDAO.java]

```
1   package com.nonage.dao;
2
3   import java.sql.Connection;
4   import java.sql.PreparedStatement;
5   import java.sql.ResultSet;
6   import java.util.ArrayList;
7
8   import util.DBManager;
9
10  import com.nonage.dto.AddressVO;
11
12  public class AddressDAO {
13    private AddressDAO() {
14    }
15
16    private static AddressDAO instance = new AddressDAO();
17
18    public static AddressDAO getInstance() {
```

```
19      return instance;
20    }
21
22    public ArrayList<AddressVO> selectAddressByDong(String dong) {
23      ArrayList<AddressVO> list = new ArrayList<AddressVO>();
24
25      String sql = "select * from address where dong like '%'||?||'%'";
26
27      Connection conn = null;
28      PreparedStatement pstmt = null;
29      ResultSet rs = null;
30
31      try {
32        conn = DBManager.getConnection();
33        pstmt=conn.prepareStatement(sql);
34        pstmt.setString(1, dong);
35        rs = pstmt.executeQuery();
36
37        while (rs.next()) {
38          AddressVO addressVO = new AddressVO();
39          addressVO.setzipNum(rs.getString("zip_num"));
40          addressVO.setSido(rs.getString("sido"));
41          addressVO.setGugun(rs.getString("gugun"));
42          addressVO.setDong(rs.getString("dong"));
43          addressVO.setzipCode(rs.getString("zip_code"));
44          addressVO.setBunji(rs.getString("bunji"));
45
46          list.add(addressVO);
47        }
48      } catch (Exception e) {
49        e.printStackTrace();
50      }
51      return list;
52    }
53  }
```

[주소 찾기] 버튼이 눌리면 “NonageServlet?command=find_zip_num”이 발생합니다. 이 요청을 받아 주소 정보를 찾아오는 액션을 만들어 봅시다.

[직접해보세요] 주소 찾기를 위한 액션 클래스
[파일이름 : src₩com₩nonage₩controller₩action₩FindZipNumAction.java]

```
1   package com.nonage.controller.action;
2
3   import java.io.IOException;
4   import java.util.ArrayList;
5
```

```
 6 import javax.servlet.RequestDispatcher;
 7 import javax.servlet.ServletException;
 8 import javax.servlet.http.HttpServletRequest;
 9 import javax.servlet.http.HttpServletResponse;
10
11 import com.nonage.dao.AddressDAO;
12 import com.nonage.dto.AddressVO;
13
14 public class FindZipNumAction implements Action{
15
16   @Override
17   public void execute(HttpServletRequest request, HttpServletResponse response)
18         throws ServletException, IOException {
19      String url="member/findZipNum.jsp";
20
21      String dong=request.getParameter("dong");
22
23      if(dong!=null && dong.trim().equals("")==false){
24         AddressDAO addressDAO=AddressDAO.getInstance();
25         ArrayList<AddressVO> addressList =
26 addressDAO.selectAddressByDong(dong.trim());
27         request.setAttribute("addressList", addressList);
28      }
29      RequestDispatcher dispatcher=request.getRequestDispatcher(url);
30      dispatcher.forward(request, response);
31   }
32 }
```

[직접해보세요] 주소 찾기를 위한 JSP 페이지

[파일이름 : WebContent/member/findZipNum.jsp]

```
 1 <%@ page language="java" contentType="text/html; charset=UTF-8"
 2     pageEncoding="UTF-8"%>
 3 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
 4 <!DOCTYPE html>
 5 <html>
 6 <head>
 7 <meta charset="UTF-8">
 8 <title>우편 번호 검색</title>
 9 <link href="CSS/subpage.css" rel="stylesheet">
10 <style type="text/css">
11 body{
12    background-color:#B96DB5;
13    font-family: Verdana;
14 }
15 #popup{
16    padding: 0 10px;
17 }
```

```
#popup h1 {
    font-family: "Times New Roman", Times, serif;
    font-size: 45px;
    color: #CCC;
    font-weight: normal;
}
table#zipcode {
     border-collapse:collapse;     /* border 사이의 간격 없앰 */
     border-top: 3px solid  #fff;
     border-bottom: 3px solid  #fff;
     width: 100%;
     margin-top: 15px;
}
table#zipcode th, table#zipcode td{
    text-align: center;
    border-bottom: 1px dotted  #fff;
    color:#FFF;
}
table th, td{
  padding: 10px;
}
table#zipcode  a{
    display:block;
     height:20px;
     text-decoration:none;
     color:#fff;
     padding: 10px;
}
table#zipcode a:hover{
     color: #F90;
     font-weight: bold;
}
</style>
<script type="text/javascript">
function result(zipNum,sido,gugun,dong) {
    opener.document.formm.zipNum.value=zipNum;
    opener.document.formm.addr1.value=sido+" "+gugun+" "+dong;
    self.close();
};
</script>
</head>
<body>
<div id="popup">
  <h1>우편번호검색</h1>
  <form method=post name=formm action="NonageServlet?command=find_zip_num">
    동 이름 : <input name="dong" type="text">
    <input type="submit" value="찾기"  class="submit">
  </form>
```

```
66    <table id="zipcode">
67      <tr>
68        <th>우편번호</th>
69        <th>주소</th>
70      </tr>
71      <c:forEach items="${addressList}" var="addressVO">
72      <tr>
73        <td>${addressVO.zipNum}</td>
74          <td>
75            <a href="#" onclick="return result('${addressVO.zipNum}'
76 ,'${addressVO.sido}', '${addressVO.gugun}','${addressVO.dong}')">
77              ${addressVO.sido} ${addressVO.gugun} ${addressVO.dong}
78            </a>
79          </td>
80      </tr>
81      </c:forEach>
82    </table>
83 </div>
84 </body>
85 </html>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
30      } else if (command.equals("find_zip_num")) {
31          action = new FindZipNumAction();
32      }
..  //중복되는 내용 생략
```

회원 정보를 입력한 후 [회원 가입] 버튼을 클릭하면 “NonageServlet?command=join” 요청이 발생합니다. 이 요청이 발생하면 회원을 위한 데이터베이스 처리를 해야 하는 액션이 호출되어야 합니다.

[직접해보세요] 회원 가입을 위한 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩JoinAction.java]

```
1   package com.nonage.controller.action;
2
3   import java.io.IOException;
4
5   import javax.servlet.RequestDispatcher;
6   import javax.servlet.ServletException;
7   import javax.servlet.http.HttpServletRequest;
8   import javax.servlet.http.HttpServletResponse;
9   import javax.servlet.http.HttpSession;
10
11  import com.nonage.dao.MemberDAO;
```

```
12  import com.nonage.dto.MemberVO;
13
14  public class JoinAction implements Action {
15
16    @Override
17    public void execute(HttpServletRequest request, HttpServletResponse response)
18          throws ServletException, IOException {
19       String url = "member/login.jsp";
20
21       HttpSession session = request.getSession();
22
23       MemberVO memberVO = new MemberVO();
24
25       memberVO.setId(request.getParameter("id"));
26       memberVO.setPwd(request.getParameter("pwd"));
27       memberVO.setName(request.getParameter("name"));
28       memberVO.setEmail(request.getParameter("email"));
29       memberVO.setZipNum(request.getParameter("zipNum"));
30       memberVO.setAddress(request.getParameter("addr1") +
31  request.getParameter("addr2"));
32       memberVO.setPhone(request.getParameter("phone"));
33
34       session.setAttribute("id", request.getParameter("id"));
35
36       MemberDAO memberDAO = MemberDAO.getInstance();
37       memberDAO.insertMember(memberVO);
38
39
40       RequestDispatcher dispatcher = request.getRequestDispatcher(url);
41       dispatcher.forward(request, response);
42    }
43  }
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
32     } else if (command.equals("join")) {
33        action = new JoinAction();
34      }
..  //중복되는 내용 생략
```

회원 가입이 완료되면 로그인 화면으로 이동합니다. 로그인 작업은 화면 상단의 주 메뉴에서 login(로그인)을 클릭해도 일어납니다. login을 클릭하면 “NonageServlet?command=login_form” 요청이 발생합니다. 이 요청을 받으면 로그인을 위한 화면을 표시하는 액션을 만들어 봅시다.

[직접해보세요] 로그인 화면을 출력하기 위한 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩LoginFormActoin.java]

```
1  package com.nonage.controller.action;
2
3  import java.io.IOException;
4
5  import javax.servlet.RequestDispatcher;
6  import javax.servlet.ServletException;
7  import javax.servlet.http.HttpServletRequest;
8  import javax.servlet.http.HttpServletResponse;
9
10 public class LoginFormAction implements Action {
11
12   @Override
13   public void execute(HttpServletRequest request, HttpServletResponse response)
14       throws ServletException, IOException {
15     String url = "member/login.jsp";
16
17     RequestDispatcher dispatcher=request.getRequestDispatcher(url);
18     dispatcher.forward(request, response);
19   }
20 }
```

[직접해보세요] 로그인 폼을 위한 JSP 페이지[파일이름 : WebContent/member/login.jsp]

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2    pageEncoding="UTF-8"%>
3  <%@ include file="../header.jsp" %>
4  <%@ include file="sub_img.html"%>
5  <%@ include file="sub_menu.html" %>
6    <article>
7      <h1>Login</h1>
8      <form method="post" action="NonageServlet?command=login">
9        <fieldset>
10         <legend></legend>
11           <label>User ID</label>
12           <input name="id" type="text" value="${id}" value="one"><br>
13           <label>Password</label>
14           <input name="pwd" type="password" value="1111"><br>
15       </fieldset>
16       <div class="clear"></div>
17       <div id="buttons">
18         <input type="submit" value="로그인" class="submit">
19         <input type="button" value="회원가입" class="cancel"
20             onclick="location='NonageServlet?command=join_form'">
21         <input type="button" value="아이디 비밀번호 찾기" class="submit"
22             onclick="location='NonageServlet?command=find_id_form'">
23       </div>
24     </form>
25   </article>
```

```
26  <%@ include file="../footer.jsp" %>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
34      } else if (command.equals("login_form")) {
35          action = new LoginFormAction();
36      }
..  //중복되는 내용 생략
```

로그인 폼에서 아이디와 비밀번호를 입력한 후 [로그인] 버튼을 클릭하면 “NonageServlet?command=login” 요청이 발생합니다. 이 요청을 받으면 로그인 처리를 위한 액션이 호출됩니다.

[직접해보세요] 로그인 처리를 위한 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩LoginActoin.java]

```
1   package com.nonage.controller.action;
2
3   import java.io.IOException;
4
5   import javax.servlet.ServletException;
6   import javax.servlet.http.HttpServletRequest;
7   import javax.servlet.http.HttpServletResponse;
8   import javax.servlet.http.HttpSession;
9
10  import com.nonage.dao.MemberDAO;
11  import com.nonage.dto.MemberVO;
12
13  public class LoginAction implements Action {
14
15    @Override
16    public void execute(HttpServletRequest request, HttpServletResponse response)
17          throws ServletException, IOException {
18      String url="member/login_fail.jsp";
19      HttpSession session=request.getSession();
20
21      String id=request.getParameter("id");
22      String pwd=request.getParameter("pwd");
23
24      MemberDAO memberDAO=MemberDAO.getInstance();
25      MemberVO memberVO=memberDAO.getMember(id);
26
27      if(memberVO!=null){
28        if(memberVO.getPwd().equals(pwd)){
29          session.removeAttribute("id");
30          session.setAttribute("loginUser", memberVO);
```

```
31              url="NonageServlet?command=index";
32            }
33          }
34
35          request.getRequestDispatcher(url).forward(request, response);
36        }
37    }
```

로그인 성공했을 때에는 메인 화면으로 이동하지만 실패했을 때에는 login_fail.jsp 페이지로 이동합니다.

[직접해보세요] 로그인 실패시 호출되는 JSP 페이지

[파일이름 : WebContent/member/login_fail.jsp]

```
1   <%@ page language="java" contentType="text/html; charset=UTF-8"
2       pageEncoding="UTF-8"%>
3   <script type="text/javascript">
4     alert("로그인 실패");
5     history.go(-1);
6   </script>
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
38      } else if (command.equals("login")) {
39        action = new LoginAction();
40      }
..  //중복되는 내용 생략
```

이번에는 로그아웃 처리를 하겠습니다.

[직접해보세요] 로그아웃 처리를 위한 액션 클래스

[파일이름 : src₩com₩nonage₩controller₩action₩LogoutActoin.java]

```
1   package com.nonage.controller.action;
2
3   import java.io.IOException;
4
5   import javax.servlet.ServletException;
6   import javax.servlet.http.HttpServletRequest;
7   import javax.servlet.http.HttpServletResponse;
8   import javax.servlet.http.HttpSession;
9
10  public class LogoutAction implements Action {
11
12    @Override
13    public void execute(HttpServletRequest request, HttpServletResponse response)
14        throws ServletException, IOException {
15      String url="NonageServlet?command=index";
16
```

```
17        HttpSession session=request.getSession(false);
18        if(session!=null){
19          session.invalidate();
20        }
21        request.getRequestDispatcher(url).forward(request, response);
22      }
23  }
```

[직접해보세요] 커맨드(command) 패턴으로 작업 처리를 위한 명령 처리 클래스 ActionFactory 수정하기[파일이름 : src₩com₩nonage₩controller₩ActionFactory.java]

```
..  //중복되는 내용 생략
36        } else if (command.equals("cart_insert")) {
37          action = new CartInsertAction();
38        }
..  //중복되는 내용 생략
```

이제 쇼핑몰에 대한 전반적인 준비가 끝이 났습니다. 본격적으로 상품에 대한 고객의 주문 과정은 다음 장에서 공부해 봅시다.