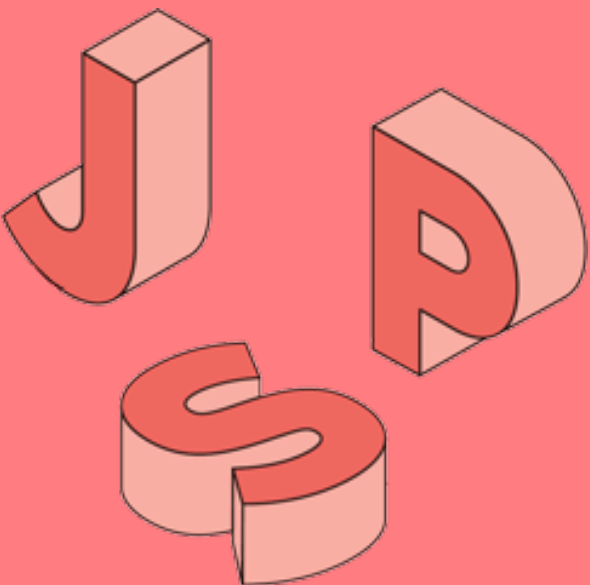


Chapter 06

JSP의 기초 다지기



목차

1. JSP의 개요
2. JSP 지시어
3. 템플릿 데이터와 스크립트 요소
4. [실습 6-1] JSP 기초 종합 예제
5. [실습 6-2] JSP 프로그래밍 : 계산기 구현

학습목표

- JSP의 개념을 이해하고 특징과 구성요소를 익힌다.
- JSP 지시어의 개념과 종류를 이해한다.
- 템플릿 데이터와 스크립트 요소의 개념을 이해한다.
- 실습을 통해 JSP의 기본 구성요소와 동작 구조를 이해한다

Section 01

JSP의 개요

JSP의 등장배경

문제점

- 웹 프로그램의 화면 기능이 복잡해지므로 서블릿의 자바 기반으로 화면 기능 구현 시 어려움이 발생함
- 디자이너 입장에서 화면 구현 시 자바 코드로 인해 작업이 어려워함
- 서블릿에 비즈니스 로직과 화면 기능이 같이 있다 보니 개발 후 유지관리가 불편함

해결책

- 서블릿의 비즈니스 로직과 결과를 보여주는 화면 기능을 분리하자!
- 비즈니스 로직과 화면을 분리함으로써 개발자는 비즈니스 로직 구현에 집중하고, 디자이너는 화면 기능 구현에만 집중하자!
- 개발 후 재사용성과 유지관리가 훨씬 수월해진다!

JSP의 특징과 구성요소

■ JSP의 특징

- HTML 페이지에 자바 코드를 직접 사용함
- 서블릿 컨테이너에 의해 관리되는 내장객체의 생명 주기를 이용하여 페이지 간 속성을 관리함
- 커스텀 태그 기술을 사용하여 코드를 태그화(action, JSTL 등)함
- EL Expression Language을 통해 데이터를 표현함

JSP의 특징과 구성요소

■ JSP의 구성요소

- 지시어(Standard Directives)
- 액션(Standard Action)
- 템플릿 데이터(Template Data)
- 스크립트 요소(Script Element)
- 커스텀 태그(Custom Tag)와 EL(Expression Language)

- HTML 태그, CSS 그리고 자바스크립트 코드
- JSP 기본 태그
- JSP 액션 태그
- 개발자가 직접 만들거나 프레임워크에서 제공하는 커스텀(custom) 태그

JSP의 동작 과정

■ JSP가 서블릿으로 컴파일되고 실행되는 과정

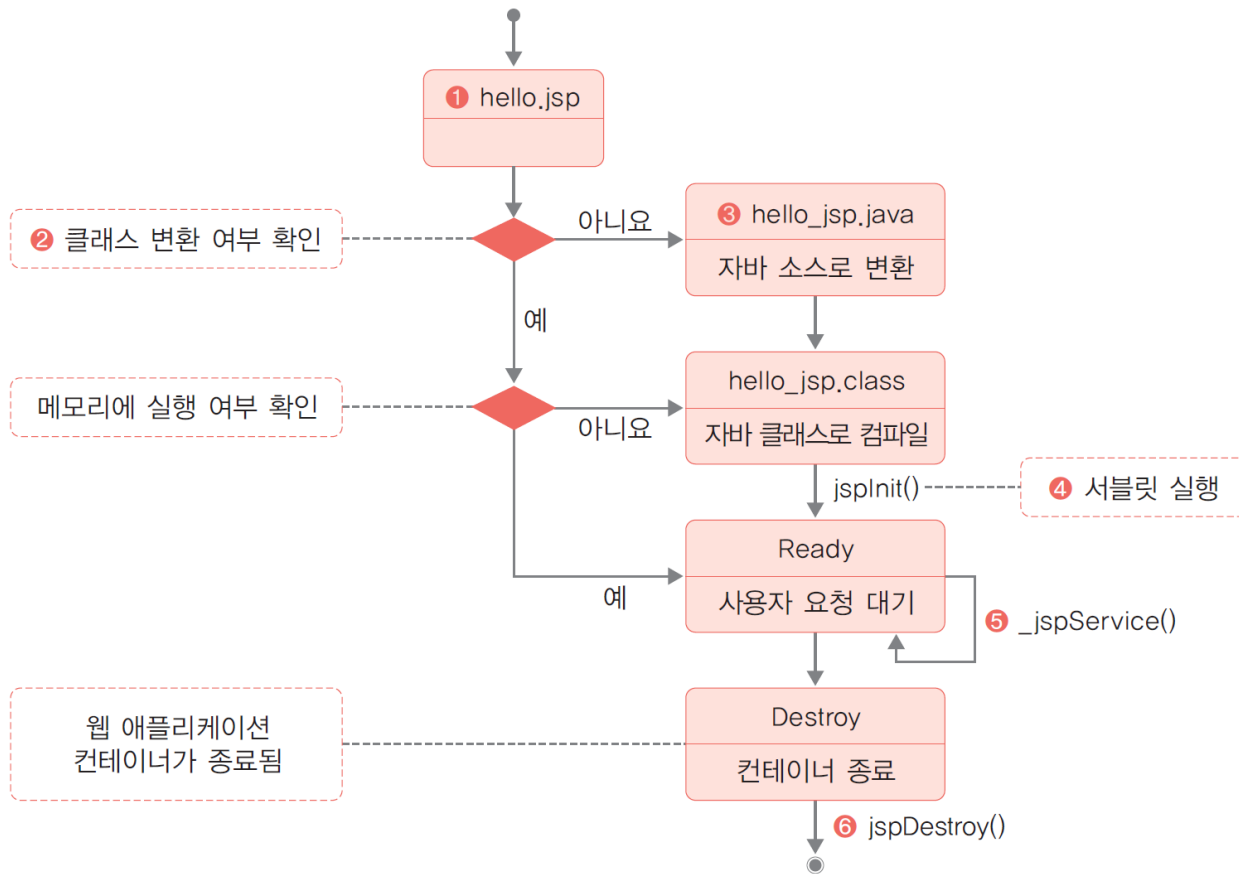


그림 6-1 JSP 개발과 동작 과정

1. 변환 단계(Translation Step): 컨테이너는 JSP 파일을 자바 파일로 변환
2. 컴파일 단계(Compile Step): 컨테이너는 변환된 자바(java) 파일을 클래스(class) 파일로 컴파일
3. 실행 단계(Interpret Step): 컨테이너는 class 파일을 실행하여 그 결과(HTML, CSS와 자바스크립트 코드)를 브라우저로 전송해 출력

JSP의 장단점

■ JSP의 장점

- HTML 파일에 자바 기술을 거의 무한대로 사용할 수 있으며, 비교적 쉽게 프로그래밍할 수 있음
- 커스텀 태그 라이브러리 등 확장 태그 구조를 사용할 수 있음
- 서블릿으로 변환되어 실행되므로 서블릿 기술의 장점을 모두 가짐
- MVC 패턴, 스프링 프레임워크 등 잘 설계된 구조를 적용할 수 있어 개발 생산성이 향상되고 성능이 보장됨
- 모든 개발이 서버에서 이루어지므로 개발의 집중화를 통한 효율이 오름

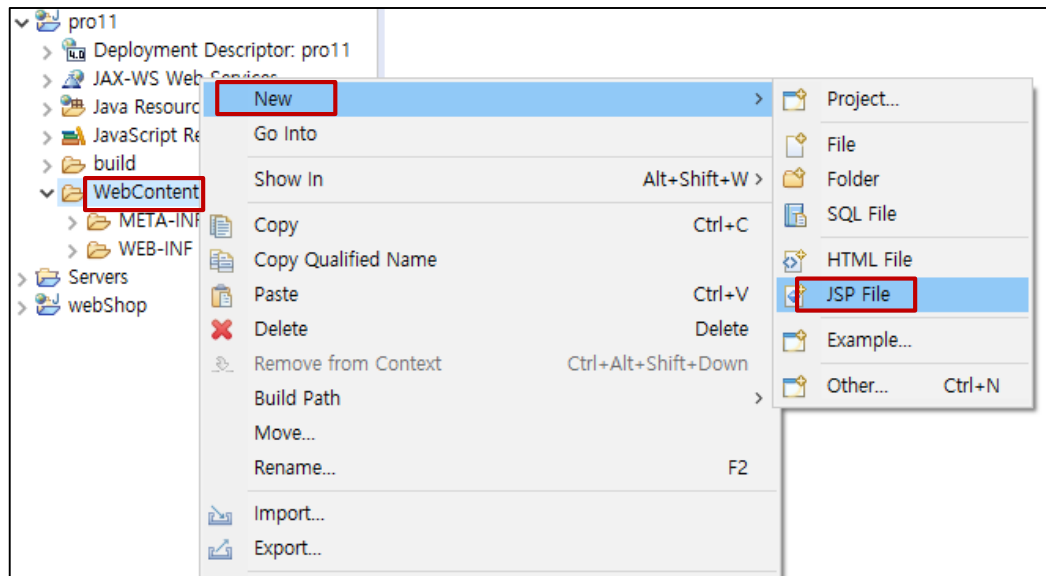
JSP의 장단점

■ JSP의 단점

- 화면 구성요소를 변경하면 JSP → 자바 → 클래스 → 서블릿 실행 과정을 거치므로 개발 과정에서 사소한 UI 변경일지라도 매번 확인하는 데 시간이 소요됨
- 개발자와 디자이너 간 역할 분담에 제약이 있음
- JSP 파일의 화면 디자인 확인에도 반드시 서블릿 컨테이너의 실행이 필요함

이클립스에서 JSP 변환 과정

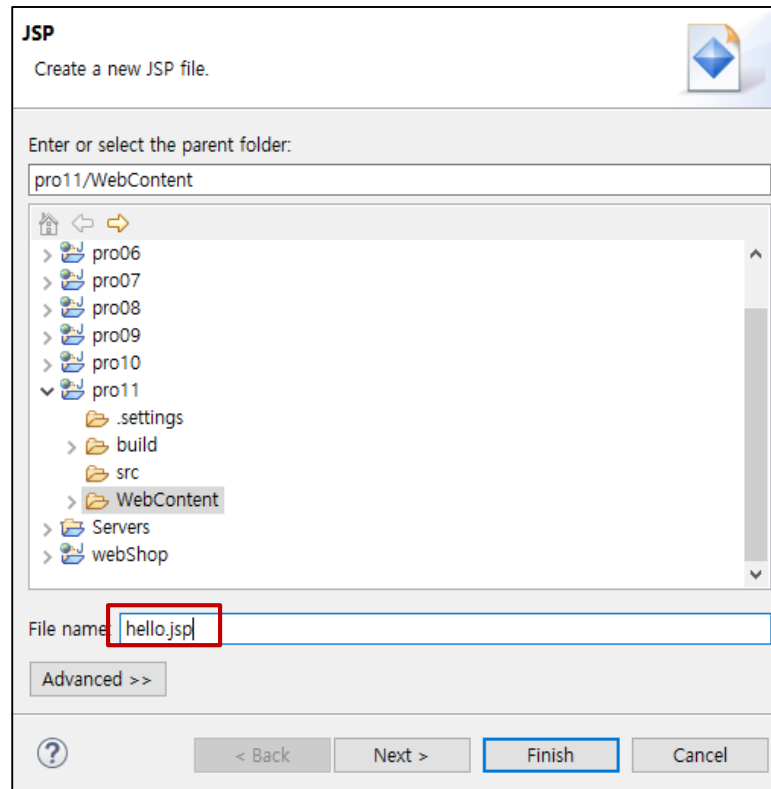
1. 이클립스에서 새 프로젝트 pro11을 만들고 WebContent 폴더에서 마우스 오른쪽 버튼을 클릭한 후 New > JSP File을 선택합니다.



❖ 반드시 servlet_api.jar을 설정해 줍니다(5.4절 참고).

이클립스에서 JSP 변환 과정

2. 파일 이름으로 hello.jsp를 입력한 후 Finish를 클릭합니다.



이클립스에서 JSP 변환 과정

3. 생성된 JSP 파일에 간단한 HTML 태그와 메시지를 작성합니다.

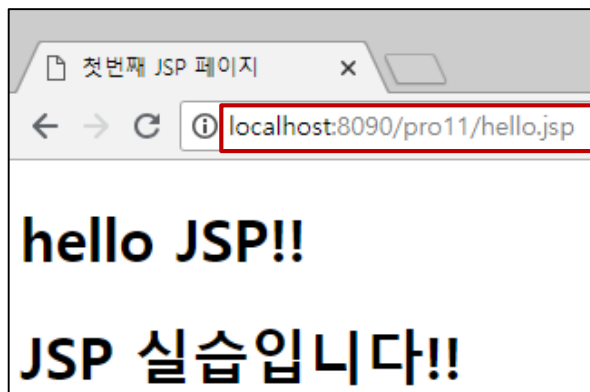
코드 11-2 pro11/WebContent/hello.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>첫 번째 JSP 페이지</title>
</head>
<body>
    <h1>hello JSP!!</h1>
    <h1>JSP 실습입니다!!</h1>
</body>
</html>
```

이클립스에서 JSP 변환 과정

4. 톰캣 컨테이너에 프로젝트를 추가합니다. 톰캣을 실행한 후 브라우저에서 HTML 파일을요청 하듯이 JSP 파일을 요청합니다.

• **http://ip주소:포트번호/프로젝트이름/JSP파일이름**



이클립스에서 JSP 변환 과정

hello.jsp 출력 과정

브라우저에서 hello.jsp 요청



톰캣 컨테이너는 hello.jsp를 읽어와 hello_jsp.java로 변환



톰캣 컨테이너는 hello_jsp.java를 hello_jsp.class로 컴파일



컨테이너는 hello_jsp.class를 실행해서 브라우저로 HTML 전송

Section 02

JSP 지시어

지시어란?

■ 지시어(Directives)

- JSP 파일의 속성을 기술하는 요소
- JSP 컨테이너에 해당 페이지를 어떻게 처리해야 하는지를 전달하는 내용을 담음
- 지시어는 크게 page, include, taglib으로 나눌 수 있으며, 각각의 속성이 다름
- 지시어의 기본 형식

```
<%@ 지시어 속성="값" %>
```

page 지시어

■ page 지시어

- 현재 JSP 페이지를 컨테이너에서 처리(서블릿으로 변환)하는 데 필요한 각종 속성을 기술함
- 소스코드 맨 앞에 위치하며 이클립스에서 JSP 파일을 생성할 때 자동으로 생성됨
- page 지시어의 구문과 사용 형식

```
<%@ page language="java" contentType="text/html; charset=UTF-8"  
    pageEncoding="UTF-8" import="java.util.*" errorPage="error.jsp"%>
```

■ page 지시어

- **language**: 현재 페이지의 스크립트 언어를 지정하는 속성
- **contentType**: 현재 페이지의 파일 형식을 지정하는 속성
 - 클라이언트 요청에 응답할 때 전달하는 HTTP 헤더 정보가 됨
- **pageEncoding**: JSP 파일을 컨테이너가 처리할 때 사용하는 캐릭터 인코딩을 지정하는 속성
 - 올바른 한글 처리를 위해서는 'UTF-8'로 지정해야 함
- **import**: JSP 파일 내에서 자바 코드(스크립트릿)를 직접 사용하는 경우 일반 자바 코드와 마찬가지로 클래스에 대한 패키지 import가 필요함
- **errorPage**: 현재 JSP 요청 처리 중에 에러가 발생하는 경우 서버 에러를 클라이언트에 전달하지 않고 별도의 페이지에서 처리하기 위한 속성
 - JSP에서 에러 페이지 설정을 넣는 것보다는 서버 설정을 사용하는 것을 권장함

page 지시어

■ page 지시어

이클립스에서 자동으로 생성된 페이지 디렉티브 태그

```
hello.jsp
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6   <meta charset="UTF-8">
7   <title>첫번째 JSP 페이지</title>
8 </head>
9 <body>
10   <h1>hello JSP!!</h1>
11   <h1>JSP 실습입니다!!</h1>
12 </body>
13 </html>
```

■ 페이지 디렉티브 태그 사용 예제

코드 11-2 pro11/WebContent/hello2.jsp

```
<%@ page contentType="text/html; charset=utf-8"
    import="java.util.*"
    language="java"
    session="true"
    buffer="8kb"
    autoFlush="true"
    isThreadSafe="true"
    info="(ShoppingMall.....)"
    isErrorPage="false"
    errorPage="" %>
```

import 속성을 제외한 다른 속성은
한 번만 선언해야 합니다.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
<title>페이지 디렉티브 연습</title>
```

```
</head>
```

```
<body>
```

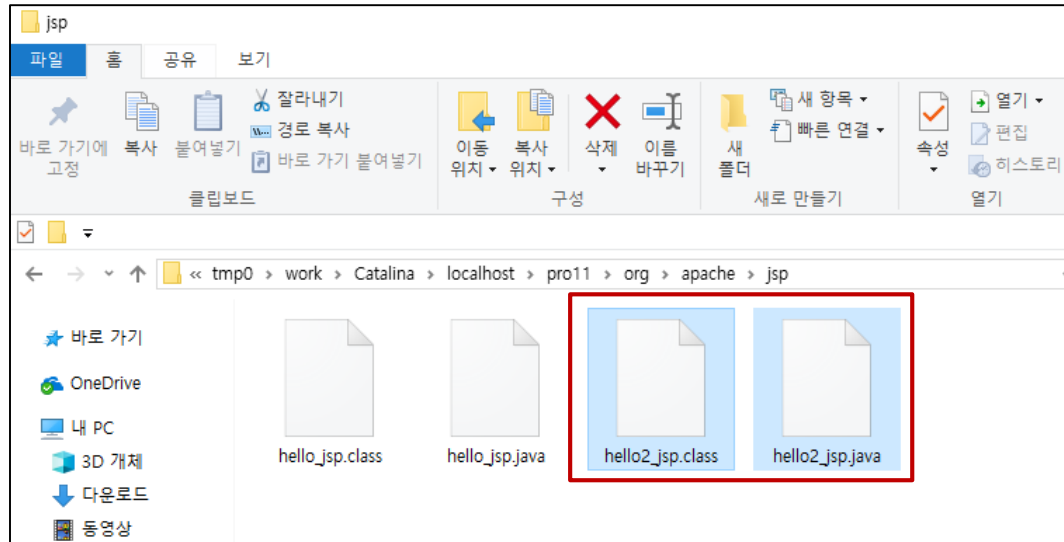
```
<h1>쇼핑몰 구현 중심 JSP입니다.!!!</h1>
```

```
</body>
```

```
</html>
```

■ 페이지 디렉티브 태그 사용 예제

JSP 파일이 변환되어서 생성된 java 파일



■ 페이지 디렉티브 태그 사용 예제

- 페이지 디렉티브 태그의 속성은 브라우저에서 요청 시 모두 자바 코드로 변환됨

```
9  package org.apache.jsp;  
10  
11  import javax.servlet.*;  
12  import javax.servlet.http.*;  
13  import javax.servlet.jsp.*;  
14  import java.util.*;
```

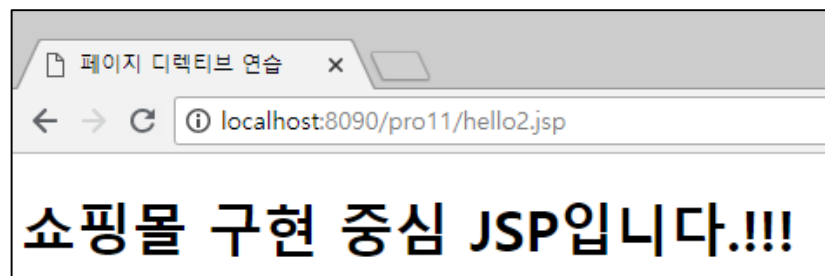
```
16  public final class hello2_jsp extends org.apache.jasper.runtime.HttpJspBase  
17      implements org.apache.jasper.runtime.JspSourceDependent,  
18      | | | | | org.apache.jasper.runtime.JspSourceImports {  
19  
20      public java.lang.String getServletInfo() {  
21          return "(ShoppingMall.....)";  
22      }  
23  
24      private static final javax.servlet.jsp.JspFactory _jspxFactory =  
25      | | | javax.servlet.jsp.JspFactory.getDefaultFactory();  
26  
27      private static java.util.Map<java.lang.String,java.lang.Long> _jspx_dependants;  
28  
29      private static final java.util.Set<java.lang.String> _jspx_imports_packages;  
30  
31      private static final java.util.Set<java.lang.String> _jspx_imports_classes;
```

■ 페이지 디렉티브 태그 사용 예제

contentType 속성이 변환된 자바 코드

```
111     try {  
112         response.setContentType("text/html; charset=utf-8");  
113         pageContext = _jspxFactory.getPageContext(this, request, response,  
114             "", true, 8192, true);  
115         _jspx_page_context = pageContext;  
116         application = pageContext.getServletContext();  
117         config = pageContext.getServletConfig();  
118         session = pageContext.getSession();  
119         out = pageContext.getOut();  
120         _jspx_out = out;
```

실행 결과



page 지시어

■ 페이지 디렉티브 태그 사용 예제

❖ 페이지 디렉티브 속성을 설정할 때는 대소문자에 유의하세요!

```
hello2.jsp
1 <%@ page contentType="text/html; charset=utf-8"
2     import="java.util.*"
3     language="java"
4     session="true"
5     buffer="8kb"
6     autoflush="true"
7     isThreadSafe="true"
8     info="(ShoppingMall.....)"
9     isErrorPage="false"
10    errorPage="" %>
11 <!DOCTYPE html>
```

'autoFlush' 속성이름이 잘못 되었습니다.

페이지 디렉티브 속성을 잘못 설정한 한 브라우저 요청 결과

HTTP Status 500 – Internal Server Error

localhost:8090/pro11/hello2.jsp

HTTP Status 500 – Internal Server Error

Type Exception Report

Message /hello2.jsp (line: [1], column: [2]) [Page directive] has invalid attribute: [autoflush]

Description The server encountered an unexpected condition that prevented it from fulfilling the request.

Exception

```
org.apache.jasper.JasperException: /hello2.jsp (line: [1], column: [2]) [Page directive] has invalid attribute: [autoflush]
    org.apache.jasper.compiler.DefaultErrorHandler.jspError(DefaultErrorHandler.java:42)
    org.apache.jasper.compiler.ErrorDispatcher.dispatch(ErrorDispatcher.java:292)
    org.apache.jasper.compiler.ErrorDispatcher.jspError(ErrorDispatcher.java:98)
    org.apache.jasper.compiler.JspUtil.checkAttributes(JspUtil.java:211)
    org.apache.jasper.compiler.Validator$DirectiveVisitor.visit(Validator.java:111)
    org.apache.jasper.compiler.Node$PageDirective.accept(Node.java:579)
    org.apache.jasper.compiler.Node$Nodes.visit(Node.java:2389)
    org.apache.jasper.compiler.Node$Visitor.visitBody(Node.java:2441)
    org.apache.jasper.compiler.Node$Visitor.visit(Node.java:2447)
    org.apache.jasper.compiler.Node$Root.accept(Node.java:470)
    org.apache.jasper.compiler.Node$Nodes.visit(Node.java:2389)
```

include 지시어

■ include 지시어

- 다른 파일을 포함하기 위한 지시어
- 사용된 위치에 특정 파일(html, jsp)을 불러옴
- 컨테이너에서는 포함된 파일을 하나로 처리하며 자바 소스를 생성한 뒤 서블릿으로 컴파일함
 - include에 사용된 파일의 내용을 모두 포함한 하나의 서블릿 코드로 생성되어 컴파일됨
- 포함되는 파일의 경우 해당 파일을 직접 요청해서 실행하는 것이 아니라면 개별 구성요소를 갖출 필요는 없음(page 지시어, HTML 기본 태그 구성요소 등)
- include 지시어는 원하는 위치에 자유롭게 사용할 수 있음
- include 지시어 사용 형식

```
<%@ include file="파일 위치" %>
```

include 지시어

- 하나의 JSP 파일(화면)이 실제로는 여러 파일의 조합으로 구성되는 경우

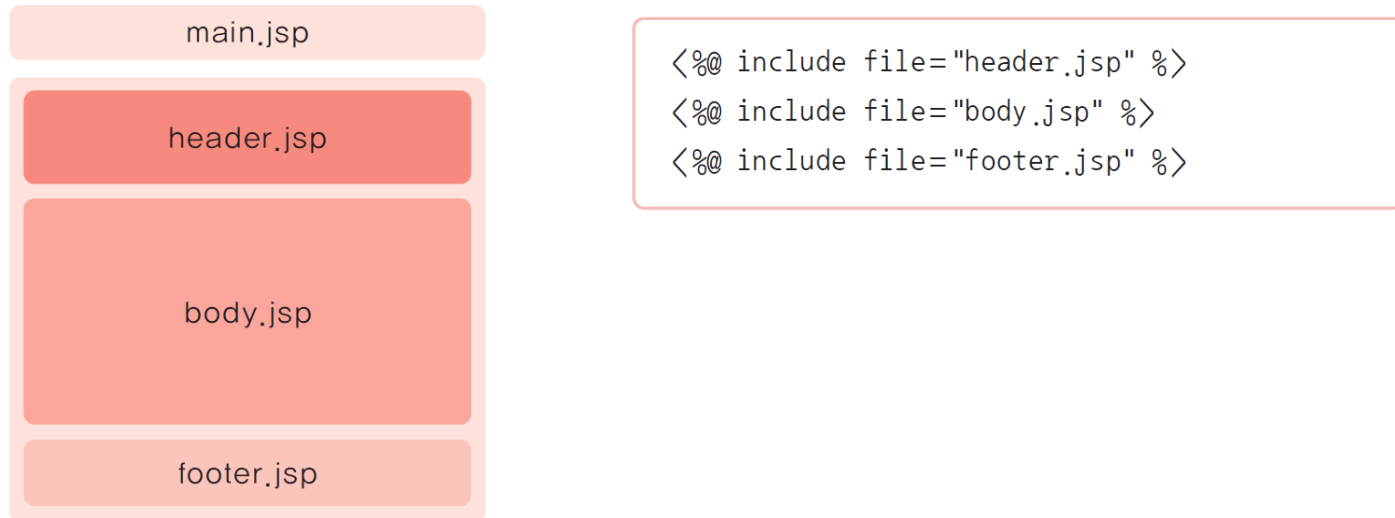


그림 6-2 여러 파일을 포함해 구성한 화면

여기서  **잠깐!** include 액션

include 지시어와 비슷한 목적으로 사용할 수 있는 include 액션도 있다. 액션의 경우 지시어와 달리 각각의 파일을 실행한 결과를 포함하여 보여주는 구조다. 자세한 내용은 7장에서 살펴본다.

■ 인클루드 디렉티브 태그 정의와 사용법

쇼핑몰 메인 화면

Book Shopping Mall

로그인 | 회원가입 | 고객센터 |

BookTopia

검색

국내외 도서

IT/인터넷
경제/경영
대학교재
자기개발
자연과학/공학
역사/인문학

음반

가요
록
클래식
뉴에이지
재즈
기타

Pay 11월 이벤트

11월에도 포인트 더블!!
무조건 2% 적립!!
간편결제 이용시
최대 4천원 적립!!

공지사항

공지사항입니다.1
공지사항입니다.2
공지사항입니다.3

하루 15분, 누구나 프로그래밍을 할 수 있다!

Try! helloworld 파이썬

베스트 셀러

모두의 답라닝 24,000원

마인크래프트 게임 제작 14,000원

핀테크 가계부 30,000원

부동산 상식 사전 20,000원

Try! helloworld 자바스크립트 25,000원

파이썬을 위한 실용 25,000원

시나공 워드 프로세서 실기 25,000원

컴퓨터 활용능력 2급 실기 25,000원

페이스북
트위터
RSS 피드

최근 본 상품

Java Programming

쇼핑몰 상품 상세 화면

Book Shopping Mall

로그인 | 회원가입 | 고객센터 |

BookTopia

검색

국내외 도서

IT/인터넷
경제/경영
대학교재
자기개발
자연과학/공학
역사/인문학

음반

가요
록
클래식
뉴에이지
재즈
기타

Pay 11월 이벤트

11월에도 포인트 더블!!
무조건 2% 적립!!
간편결제 이용시
최대 4천원 적립!!

공지사항

공지사항입니다.1

컴퓨터와 인터넷

국내외 도서 > 컴퓨터와 인터넷 > 웹 개발

초보자를 위한 자바 프로그래밍

이병승 저 | 인포북

Java Programming

정가 30,000원

판매가 27,000원(10%할인)

포인트적립 2000P(10%적립)

포인트 추가적립 만원이상 구매시 1,000P, 5만원이상 구매시 2,000P추가적립 권역점 배송 이용시 300P 추가적립

발행일 2018-10-02

페이지 수 996쪽

ISBN 2323454566778

배송료 무료

배송안내 [당일배송] 당일배송 서비스 시점! [휴일배송] 휴일에도 배송받는 Bookshop

도착예정일 지금 주문 시 내일 도착 예정

수량 1

구매하기

장바구니

위시리스트

책소개

저자소개

책목차

출판사서평

추천사

리뷰

페이스북
트위터
RSS 피드

최근 본 상품

Java Programming

- 여러 웹 페이지에서 공통으로 사용되는 JSP 페이지를 미리 만들어 놓고 요청 시 부모 웹페이지에 추가해서 사용하는 방법

include 지시어

■ 인클루드 디렉티브 태그 정의와 사용법

인클루드 디렉티브 태그의 특징

- 재사용성이 높다.
- JSP 페이지의 유지관리가 쉽다.

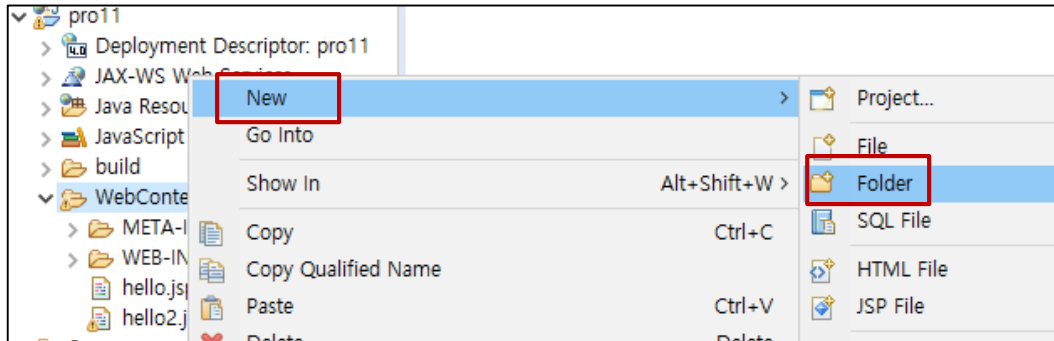
인클루드 디렉티브 태그 형식

```
<%@ include file="공통기능.jsp" %>
```

include 지시어

■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

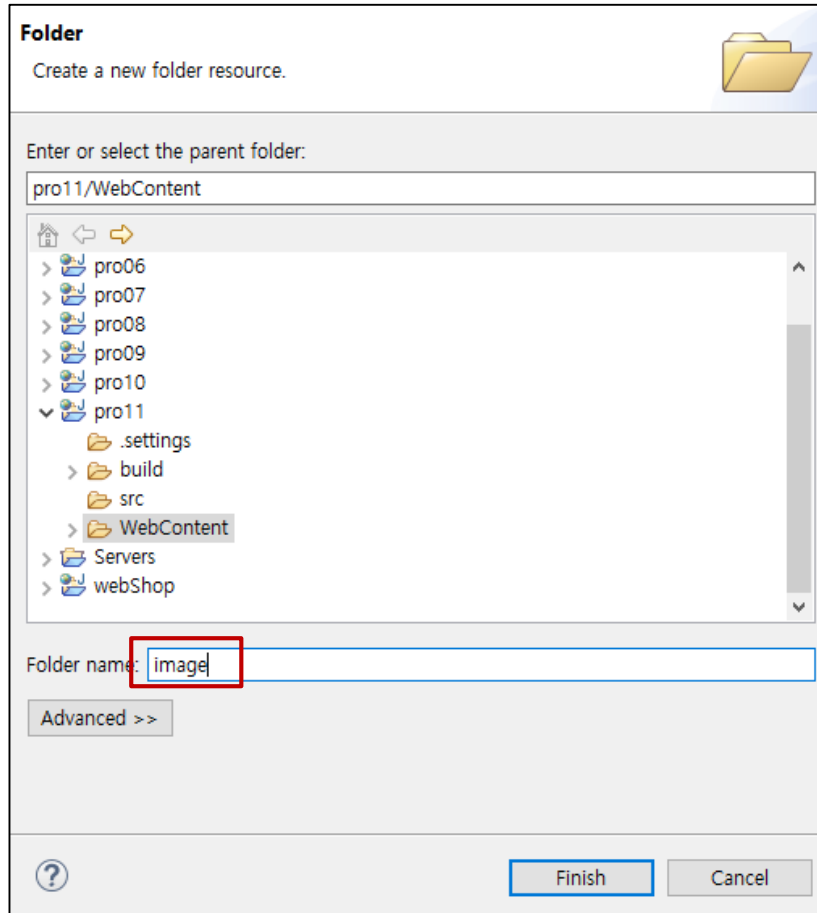
1. 프로젝트의 WebContent에서 마우스 오른쪽 버튼을 클릭한 후 New > Folder를 선택합니다.



include 지시어

■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

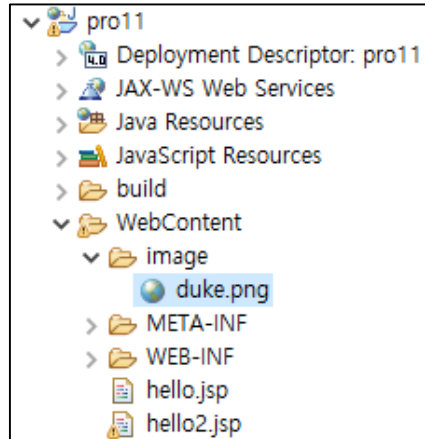
2. 폴더 이름으로 image를 입력한 후 Finish를 클릭합니다.



include 지시어

■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

3. 이미지를 복사한 후 image 폴더에 붙여 넣습니다.



■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

4. 다음과 같이 인클루드 디렉티브 태그를 이용해 두 개의 jsp 파일을 작성합니다.

코드 11-3 pro11/WebContent/duke_image.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>duke_image</title>
</head>
<body>
    
</body>
</html>
```

image 폴더의 duke.png를 표시합니다.

include 지시어

■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

코드 11-4 pro11/WebContent/include.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>인클루드 디렉티브</title>
</head>
<body>
    <h1>안녕하세요. 쇼핑몰 중심 JSP 시작입니다!!! </h1><br>
    <%@ include file="duke_image.jsp" %> <br>
    <h1>안녕하세요. 쇼핑몰 중심 JSP 끝 부분입니다.!!!</h1>
</body>
</html>
```

인클루드 디렉티브 태그를 이용해 duke_image.jsp를 포함합니다.

include 지시어

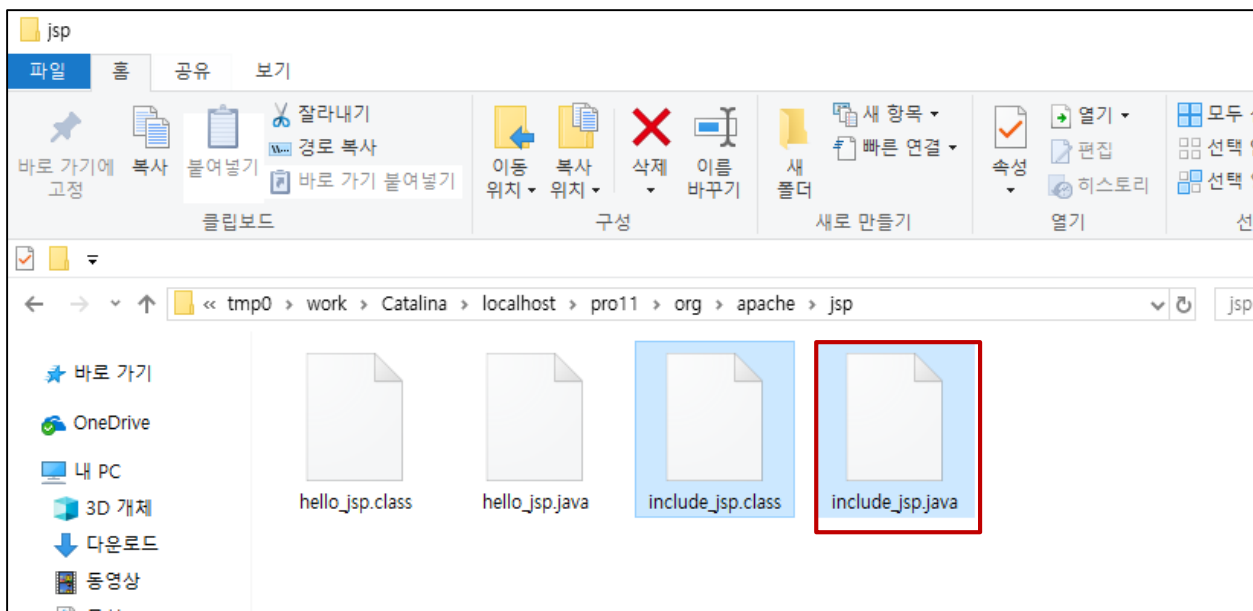
■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

5. 브라우저에서 요청하면 include.jsp 안에 duke_image.jsp가 포함되어 표시됩니다.



■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

6. 윈도 탐색기에서 다음 경로에 들어가면 브라우저에서 include.jsp를 요청할 때 변환된 자바 파일이 생성된 것을 볼 수 있습니다. 자바 파일을 열어보면 인클루드 디렉티브 태그로 포함된 duke_image.jsp의 HTML 태그가 합쳐져 있습니다.



include 지시어

■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

인클루드 디렉티브 태그에 의해 합쳐진 HTML 태그

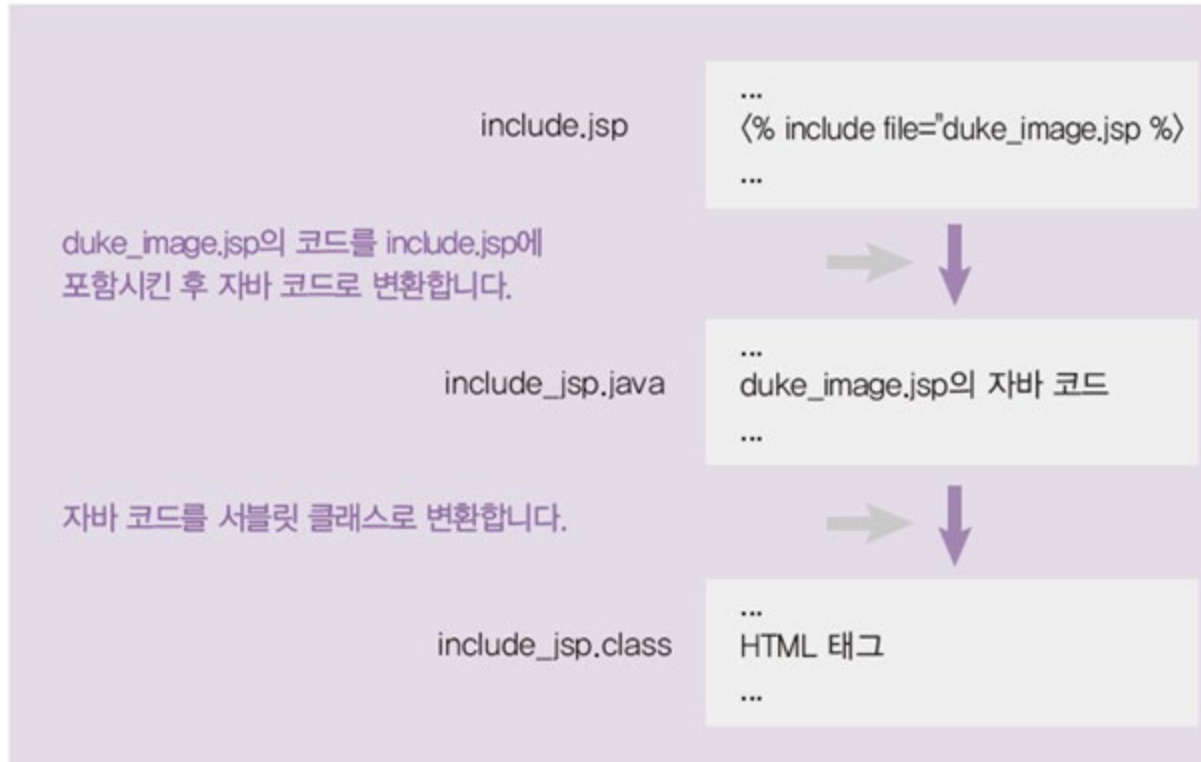
```
122 out.write("\r\n");
123 out.write("<!DOCTYPE html>\r\n");
124 out.write("<html>\r\n");
125 out.write("<head>\r\n");
126 out.write("  <meta charset=\"UTF-8\">\r\n");
127 out.write("  <title>인클루드 디렉티브</title>\r\n");
128 out.write("</head>\r\n");
129 out.write("<body>\r\n");
130 out.write("  <h1>안녕하세요. 쇼핑몰 중심 JSP 시작입니다!!! </h1><br>\r\n");
131 out.write("  ");
132 out.write("\r\n");
133 out.write("\r\n");
134 out.write("<!DOCTYPE html>\r\n");
135 out.write("<html>\r\n");
136 out.write("<head>\r\n");
137 out.write("  <meta charset=\"UTF-8\">\r\n");
138 out.write("  <title>duke_image</title>\r\n");
139 out.write("</head>\r\n");
140 out.write("<body>\r\n");
141 out.write("  <img src=\"../image/duke.png\" />\r\n");
142 out.write("</body>\r\n");
143 out.write("</html>\r\n");
144 out.write("<br>\r\n");
145 out.write("  <h1>안녕하세요. 쇼핑몰 중심 JSP 끝 부분입니다.!!!</h1>\r\n");
146 out.write("</body>\r\n");
147 out.write("</html>\r\n");
148 } catch (java.lang.Throwable t) {
```

duke_image.jsp 페이지가 포함됨

include 지시어

■ 인클루드 디렉티브 태그 이용해 이미지 삽입하기

인클루드 디렉티브 태그 실행 과정



❖ 인클루드 디렉티브 태그를 이용해 JSP 페이지를 요청하면 요청하는 JSP 페이지에 대해 실행하는 자바 파일은 단 한 개만 생성됨

taglib 지시어

■ taglib 지시어

- JSP의 태그 확장 메커니즘인 커스텀 태그를 사용하기 위한 지시어
- taglib 지시어의 구문과 사용 형식

```
<%@ taglib ( uri="태그 라이브러리 경로" 혹은 tagdir="태그 파일 경로" )  
prefix="태그 접두어" %>
```

- **uri**: 태그 라이브러리 위치로 태그를 정의하고 있는 .tld 파일 경로를 나타냄
- **tagdir**: 태그 파일로 태그를 구현한 경우 태그 파일 경로를 나타냄
- **prefix**: 해당 태그를 구분해서 사용하기 위한 접두어

taglib 지시어

- 태그 파일로 커스텀 태그를 구현한 예시
 - [WEB-INF/tags] 폴더에 있는 'printData.tag' 파일에 태그에서 처리할 내용이 작성되어 있음

```
<%@ taglib tagdir="/WEB-INF/tags" prefix="m" %>
...
<h2><m:printData /></h2>
```


Section 03

템플릿 데이터와 스크립트 요소

템플릿 데이터란?

■ 템플릿 데이터

- JSP의 화면 구성요소를 의미함
- 시작 부분의 page 지시어를 제외하면 JSP 파일의 전반적인 구조는 HTML의 문서 구조를 따름. 따라서 일반적인 HTML 파일처럼 CSS, 자바스크립트도 사용할 수 있음
 - 기본적으로 HTML5를 사용하며 문서 구조 중심으로 간략하게 작성하고 데이터 표현은 뒤에서 배울 JSTL과 EL을 사용함
 - 화면 디자인을 위해 자체적인 CSS 정의 이외에도 Bootstrap과 같은 라이브러리를 사용할 수 있음
 - REST API 호출을 위해 Axios 같은 자바스크립트 라이브러리 역시 사용할 수 있음
- 주의할 점) React, Vue와 같은 자바스크립트 요소는 프론트엔드 개발 기술로 JSP와 함께 사용하지 않음

스크립트 요소

■ 스크립트 요소

- JSP는 HTML과 자바 코드를 섞어 사용할 수 있는데, 이때 사용되는 자바 코드를 의미함
- 자주 쓰이는 스크립트 요소 : `<%! %>`, `<%= %>`, `<% %>`

■ `<%! %>`

- 선언(Declaration) 태그
- JSP가 서블릿 코드로 변환될 때 `_jspService()` 메서드 안에 들어가게 되므로 JSP에서는 일반 자바 코드와 달리 멤버 변수나 메서드 선언은 기본적으로 불가능함
- 멤버 변수나 메서드 선언이 필요하다면 사용할 수는 있으나 권장하지 않음

스크립트 요소

- `<%= %>`
 - 표현(Expression) 태그
 - 웹 브라우저를 통해 클라이언트에 전달될(HTML 응답에 포함될) 자바 표현식을 포함
 - `out.println()`의 인자로 적합한 모든 자바 코드가 가능함
 - 사칙연산, 메서드 호출, 변수값 출력 등에 사용됨
 - EL로 대체할 수 있음

```
<h2><%= member.getUserName() %></h2>
```

```
현재날짜와 시간: <%= java.time.LocalDateTime.now() %>
```

스크립트 요소

- `<% %>`
 - 스크립트릿(Scriptlet) 태그
 - 모든 자바 코드의 사용이 가능함
 - 단, `_jspService()` 메서드 내에 포함되는 것을 고려해야 함
 - 서블릿 코드로 변환될 때 모든 HTML은 `out.write()` 형태로 변경됨
 - HTML과 스크립트릿을 중간에 섞어 사용하는 것도 가능함
 - MVC 패턴 적용과 JSTL + EL로 대체할 수 있음

스크립트 요소

선언문(Declaration Tag)

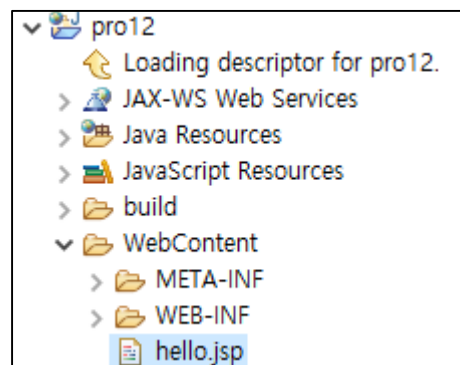
- JSP 페이지에서 사용하는 멤버 변수나 멤버 메서드를 선언할 때 사용
- 선언문 안의 멤버는 서블릿 변환 시 서블릿 클래스의 멤버로 변환됨

선언문 형식

```
<%! 멤버 변수 or 멤버 메서드  
%>
```

- 12.2.1 JSP에서 선언문 실습

1. 새 프로젝트 pro12를 만들고 hello.jsp 파일을 생성합니다.



스크립트 요소

2. 선언문을 사용한 hello.jsp를 다음과 같이 작성합니다. 선언문은 일반적으로 JSP 페이지의 상단에서 주로 사용됩니다.

코드 12-1 pro12/WebContent/hello.jsp

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

```
<%!
    String name = "듀크";
    public String getName(){ return name;}
%>
```

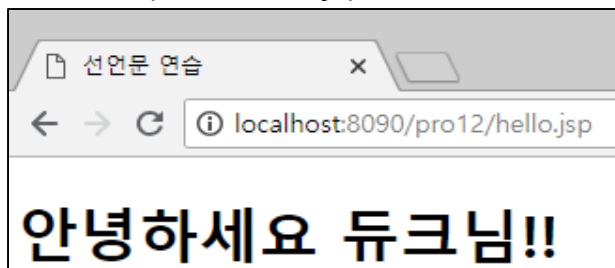
선언문을 이용해 멤버 변수 `name`과 멤버 메서드 `getName()`을 선언합니다.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>선언문 연습</title>
</head>
<body>
    <h1>안녕하세요 <%=name %> 님!!</h1>
</body>
</html>
```

표현식을 이용해 선언문에서 선언한 `name`의 값을 출력합니다.

스크립트 요소

3. 브라우저에서 `http://localhost:8090/pro12/hello.jsp`로 요청합니다.



4. 변환된 자바 코드를 보면 선언문에서 선언된 변수와 메서드는 서블릿 클래스의 멤버 변수와 멤버 메서드로 변환된 것을 알 수 있습니다. 따라서 선언문에서 선언된 변수는 JSP(서블릿 클래스) 안에서 자유롭게 접근할 수 있습니다.

```
9 package org.apache.jsp;  
10  
11 import javax.servlet.*;  
12 import javax.servlet.http.*;  
13 import javax.servlet.jsp.*;  
14  
15 public final class hello_jsp extends org.apache.jasper.runtime.HttpJspBase  
16     implements org.apache.jasper.runtime.JspSourceDependent,  
17         org.apache.jasper.runtime.JspSourceImports {  
18  
19     String name = "듀크";  
20     public String getName(){ return name;}  
21  
22  
23     private static final javax.servlet.jsp.JspFactory _jspxFactory =  
24         javax.servlet.jsp.JspFactory.getDefaultFactory();  
25  
26     private static java.util.Map<java.lang.String,java.lang.Long> _jspx_dependants;  
27  
28     private static final java.util.Set<java.lang.String> _jspx_imports_packages;  
29  
30     private static final java.util.Set<java.lang.String> _jspx_imports_classes;  
31
```

서블릿 클래스의 멤버 변수
와 멤버 메서드로 변환됩니
다.

스크립트 요소

스크립트릿(Scriptlet)

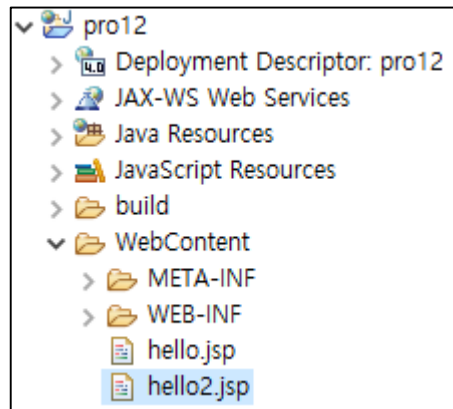
주로 초기 웹페이지에서 동적인 기능을 구현하기 위해서 사용됨

스크립트릿 형식

`<% 자바 코드 %>`

- 12.3.1 JSP에서 스크립트릿 실습

1. JSP에서 스크립트릿 실습을 위해 hello2.jsp 파일을 준비합니다.



스크립트 요소

2. 브라우저에서 JSP로 전송된 값을 얻기 위해 <% %> 안에 자바 코드를 사용하여 age 값을 가져옵니다.

코드 12-2 pro12/WebContent/hello2.jsp

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

```
<%!
```

```
    String name = "이순신";
    public String getName(){ return name;}
```

```
%>
```

```
<% String age=request.getParameter("age"); %>
```

스크립트릿을 이용해 자바 코드를 작성합니다.

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
<meta charset="UTF-8">
```

```
    <title>스크립트릿 연습</title>
```

```
</head>
```

```
<body>
```

```
    <h1>안녕하세요 <%=name %>님!!</h1>
```

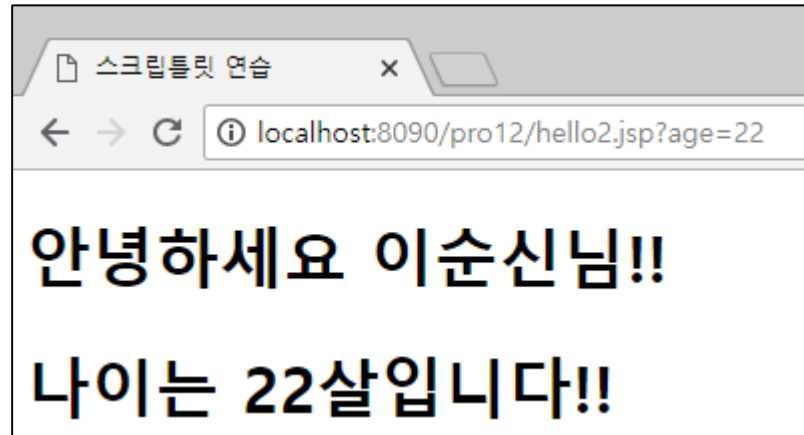
```
    <h1>나이는 <%=age %>살입니다!!</h1>'
```

표현식을 이용해 전송된 나이를 출력합니다.

```
</body>
```

```
</html>
```

3. `http:localhost:8090/hello2.jsp?age=22`로 요청합니다.



스크립트 요소

서블릿으로 변경된 상태

```
109 try {
110     response.setContentType("text/html; charset=UTF-8");
111     pageContext = _jspxFactory.getPageContext(this, request, response,
112         null, true, 8192, true);
113     _jspx_page_context = pageContext;
114     application = pageContext.getServletContext();
115     config = pageContext.getServletConfig();
116     session = pageContext.getSession();
117     out = pageContext.getOut();
118     _jspx_out = out;
119
120     out.write('\r');
121     out.write('\n');
122     out.write(" \r\n");
123     String age=request.getParameter("age");
124     out.write(" \r\n");
125     out.write("\r\n");
126     out.write("<!DOCTYPE html>\r\n");
127     out.write("<html>\r\n");
128     out.write("<head>\r\n");
129     out.write("<meta charset=\"UTF-8\">\r\n");
130     out.write("    <title>스크립틀릿 연습</title>\r\n");
131     out.write("</head>\r\n");
132     out.write("<body>\r\n");
133     out.write("    <h1>안녕하세요 ");
134     out.print(name );
135     out.write("</h1>\r\n");
136     out.write("    <h1>나이는 ");
137     out.print(age );
138     out.write("</h1>\r\n");
```

서블릿의 `_jspService()` 메서드
안의 자바 코드로 변환됩니다.

`name`과 `age`의 값이 `print()`로 브라우
저로 전송됩니다.

브라우저로 전송된 HTML 태그

```
5 <!DOCTYPE html>
6 <html>
7 <head>
8   <meta charset="UTF-8">
9     <title>스크립틀릿 연습</title>
10 </head>
11 <body>
12   <h1>안녕하세요 미순신님!!</h1>
13   <h1>나이는 22살입니다!!</h1>
14 </body>
15 </html>
```

- ❖ JSP의 스크립트 요소는 브라우저로 전송되지 않고 브라우저로 전송되기 전에 컨테이너에서 자바 코드로 변환됨

스크립트 요소

표현식(Expression Tag)

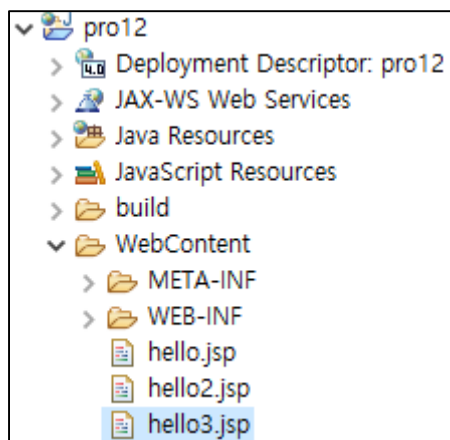
➤ JSP 페이지에서 원하는 위치에 값을 출력하는 기능

표현식 형식

<%=값 or 자바 변수 or 자바 식 %>

- 12.4.1 JSP에서 표현식 실습

1. 다음과 같이 hello3.jsp 파일을 준비합니다.



스크립트 요소

2. 다음과 같이 hello3.jsp를 작성합니다.

코드 12-3 pro12/WebContent/hello3.jsp

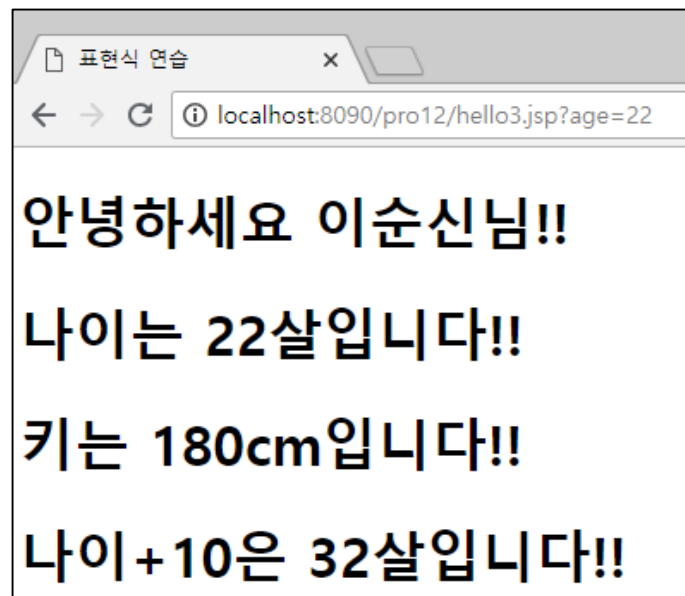
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%!
    String name = "이순신";
    public String getName(){ return name;}
%>
<% String age=request.getParameter("age"); %>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>표현식 연습</title>
</head>
<body>
    <h1>안녕하세요 <%=name %>님!!</h1>
    <h1>나이는 <%=age %>살입니다!!</h1>
    <h1>키는 <%=180 %>cm입니다!!</h1>
    <h1>나이+10은 <%=Integer.parseInt(age)+10 %>살입니다!!</h1>
</body>
</html>
```

<%= %>를 이용해 값을 출력합니다.

age의 값에 10을 더한 값을 출력합니다.

3. `http://localhost:8090/pro12/hello3.jsp?age=22`로 요청하여 결과를 확인합니다.



스크립트 요소

```
120 out.write('\r');
121 out.write('\n');
122 out.write(" \r\n");
123 String age=request.getParameter("age");
124 out.write(" \r\n");
125 out.write("\r\n");
126 out.write("<!DOCTYPE html>\r\n");
127 out.write("<html>\r\n");
128 out.write("<head>\r\n");
129 out.write("    <meta charset=\"UTF-8\">\r\n");
130 out.write("    <title>표현식 연습</title>\r\n");
131 out.write("</head>\r\n");
132 out.write("<body>\r\n");
133 out.write("    <h1>안녕하세요!</h1>\r\n");
134 out.print(name);
135 out.write("님!!</h1>\r\n");
136 out.write("    <h1>나이는 ");
137 out.print(age);
138 out.write("살입니다!!</h1>\r\n");
139 out.write("    <h1>키는 ");
140 out.print(180);
141 out.write("cm입니다!!</h1>\r\n");
142 out.write("    <h1>나이+10은 ");
143 out.print(Integer.parseInt(age)+10);
144 out.write("살입니다!!</h1>\r\n");
145 out.write("</body>\r\n");
146 out.write("</html>\r\n");
147 } catch (java.lang.Throwable t) {
```

표현식의 위치에서 print()를
이용해서 브라우저에 출력하
입니다.

❖ 표현식 안의 값은 print()를 이용해 브라우저에 출력됩니다.

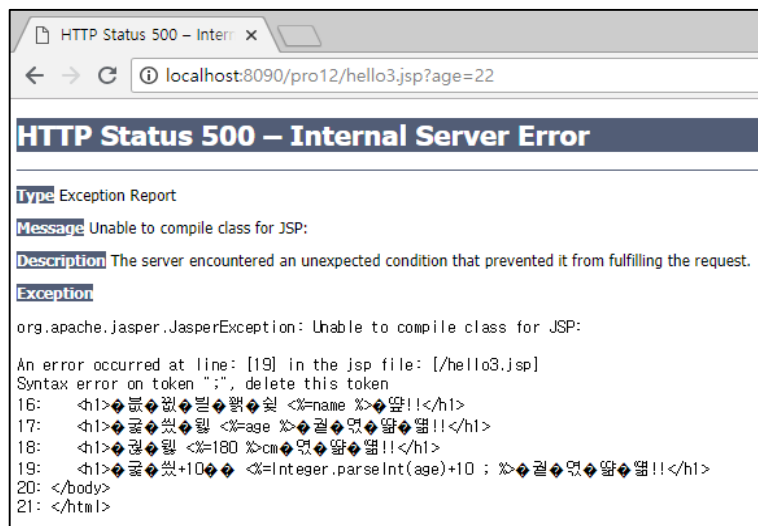
스크립트 요소

선언문에 세미콜론(;) 추가

```
9 <!DOCTYPE html>
10 <html>
11 <head>
12   <meta charset="UTF-8">
13   <title>표현식 연습</title>
14 </head>
15 <body>
16   <h1>안녕하세요 <%=name %>님!!</h1>
17   <h1>나이는 <%=age %>살입니다!!</h1>
18   <h1>키는 <%=180 %>cm입니다!!</h1>
19   <h1>나이+10은 <%=Integer.parseInt(age)+10; %>살입니다!!</h1>
20 </body>
21 </html>
```



❖ <%= %> 안의 자바 변수나 자바 식에는 세미콜론(;)이 있으면 안 된다는 것 꼭 기억하세요!



JSP 주석문 사용하기

JSP에 사용되는 주석문

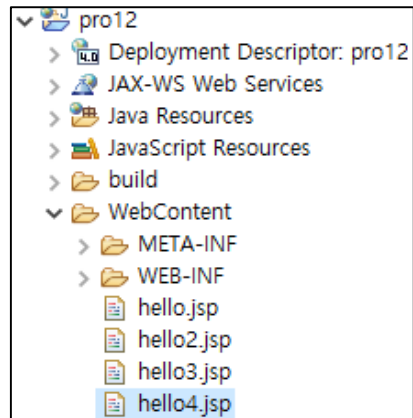
- HTML 주석
- 자바 주석
- JSP 주석

JSP 주석문 형식

`<%-- 내용 --%>`

• 12.5.1 JSP 페이지에서 주석문 사용하기

1. 다음과 같이 hello4.jsp 파일을 준비합니다.



스크립트 요소

2. hello4.jsp를 다음과 같이 작성합니다. JSP 페이지에서 사용되는 여러 가지 주석문이 포함되어 있습니다.

코드 12-4 pro12/WebContent/hello4.jsp

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

```
<%
```

```
/*
```

```
String age=request.getParameter("age");
```

```
*/
```

```
%>
```

```
<!DOCTYPE html>
```

```
<!-- HTML 주석문입니다. -->
```

```
<html>
```

```
<head>
```

```
    <title>주석문 연습</title>
```

```
</head>
```

```
<body>
```

```
    <h1>주석문 예제입니다!!</h1>
```

```
    <%-- <%=Integer.parseInt(age)+10 %> --%>
```

```
</body>
```

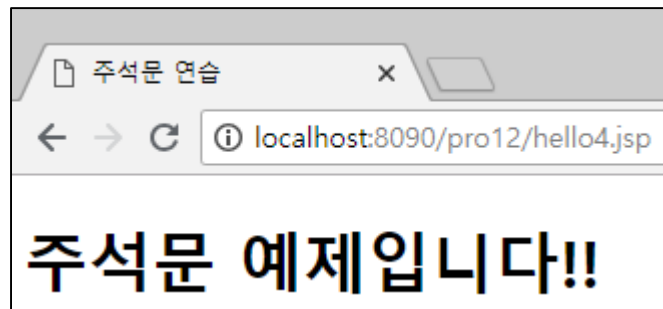
```
</html>
```

〈% %〉 안의 자바 코드에 대한 주석문

HTML 태그에 대한 주석문

JSP 페이지에 대한 주석문

3. . http://localhost:8090/pro12/hello4.jsp로 요청합니다.



```
4 <!DOCTYPE html>
5 <!-- HTML 주석문입니다. -->
6 <html>
7 <head>
8   <title>주석문 연습</title>
9 </head>
10 <body>
11   <h1>주석문 예제입니다!!</h1>
12
13 </body>
14 </html>
15
```

HTML 주석문은 브라우저로 전달됩니다.

스크립트 요소

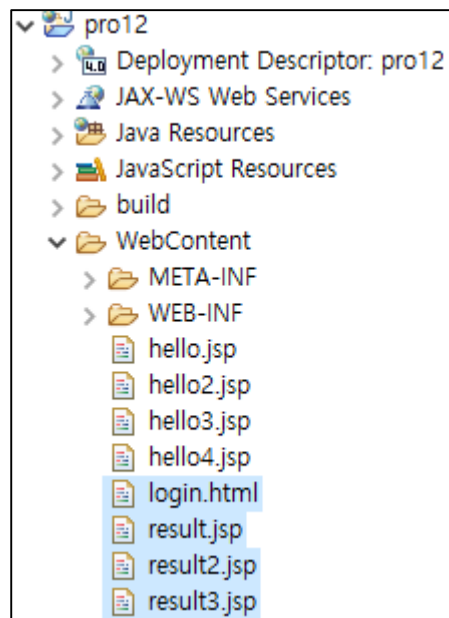
자바 주석문은 서블릿으로 변환 시 자바 주석문으로 표시됩니다.

```
113      out = pageContext.getOut();
114      _jspx_out = out;
115
116      out.write("\r\n");
117      out.write("\r\n");
118
119      /*
120      String age=request.getParameter("age");
121      */
122
123      out.write(" \r\n");
124      out.write("<!DOCTYPE html>\r\n");
125      out.write("<!-- HTML 주석문입니다. -->\r\n");
126      out.write("<html>\r\n");
127      out.write("<head>\r\n");
128      out.write("    <title>주석문 연습</title>\r\n");
129      out.write("</head>\r\n");
130      out.write("<body>\r\n");
131      out.write("    <h1>주석문 예제입니다!!</h1>\r\n");
```

서블릿에 자바 주석문을 표시
됩니다.

■ 로그인 예제

1. 로그인창에서 ID와 비밀번호를 입력한 후 JSP로 전송하여 출력하는 예제입니다.
다음과 같이 실습 파일 login.html, result.jsp, result2.jsp, result3.jsp를 준비합니다.



■ 로그인 예제

2. login.html을 다음과 같이 작성합니다. 로그인창에서 ID와 비밀번호를 입력한 후 action의 result.jsp로 전송합니다.

코드 12-5 pro12/WebContent/login.html

```
<!DOCTYPE html>
<html>...</head>
<body>
  <form name="frmLogin" method="post" action="result.jsp" enctype="utf-8">
    아이디 :<input type="text" name="user_id"><br>
    비밀번호:<input type="password" name="user_pw"><br>
    <input type="submit" value="로그인">
    <input type="reset" value="다시 입력">
  </form>
</body>
</html>
```

입력한 ID와 비밀번호를 result.jsp로 전송합니다.

로그인 예제

3. result.jsp를 다음과 같이 작성합니다. 스크립트릿을 이용해 전송된 ID와 비밀번호를 가져온 후 표현식을 이용해 변수의 값을 출력합니다.

코드 12-6 pro12/WebContent/result.jsp

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

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
    <meta charset="UTF-8">
```

```
    <title>결과출력창</title>
```

```
</head>
```

```
<body>
```

```
    <h1>결과 출력</h1>
```

```
<%
```

```
    request.setCharacterEncoding("utf-8");
```

```
    String user_id=request.getParameter("user_id");
```

```
    String user_pw=request.getParameter("user_pw");
```

```
%>
```

```
    <h1>아이디 : <%= user_id %></h1>
```

```
    <h1>비밀번호: <%= user_pw %></h1>
```

```
</body>
```

```
</html>
```

getParameter() 메서드를 이용해
입력 정보를 가져옵니다.

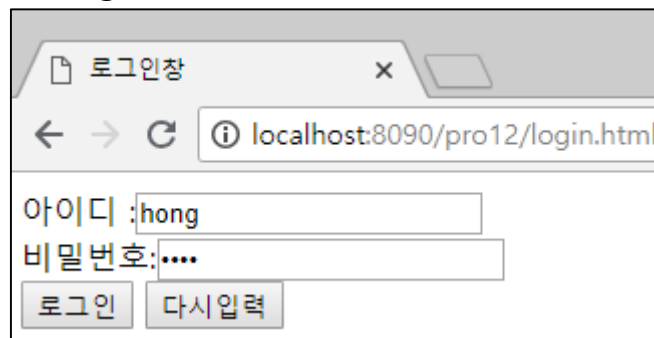
ID를 표현식으로 출력합니다.

비밀번호를 표현식으로 출력합니다.

스크립트 요소

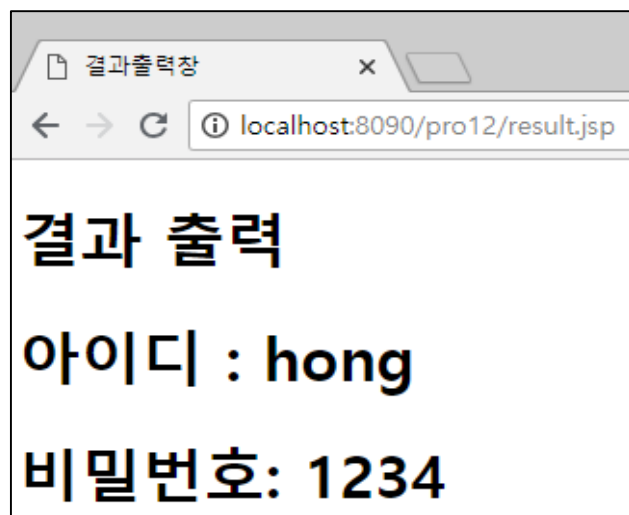
로그인 예제

4. `http://localhost:8090/pro12/login.html`로 요청한 후 ID와 비밀번호를 입력하여 로그인합니다.



A screenshot of a web browser window titled '로그인창' (Login Window). The address bar shows 'localhost:8090/pro12/login.html'. The form contains two input fields: '아이디 :hong' (ID) and '비밀번호:....' (Password). Below the fields are two buttons: '로그인' (Login) and '다시입력' (Re-enter).

5. 로그인 정보가 출력됩니다.



A screenshot of a web browser window titled '결과출력창' (Result Output Window). The address bar shows 'localhost:8090/pro12/result.jsp'. The page displays the following text: '결과 출력' (Result Output), '아이디 : hong' (ID : hong), and '비밀번호: 1234' (Password: 1234).

로그인 예제

6. 이번에는 한 걸음 더 나아가 스크립트릿 안에 자바 코드를 사용해 ID가 정상적으로 입력되었는지 체크한 후 정상 입력 여부에 따라 동적으로 다른 결과를 출력하도록 구현해 보겠습니다.
result2.jsp를 다음과 같이 작성합니다.

코드 12-7 pro112/WebContent/result2.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    request.setCharacterEncoding( "utf-8" );
    String user_id = request.getParameter("user_id");
    String user_pw = request.getParameter("user_pw");
%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>결과출력창</title>
</head>
<body>
<%
    if(user_id==null || user_id.length()==0){
        아이디를 입력하세요.<br>
        <a href="/pro12/login.html">로그인하기</a>
    }else{
        <h1> 환영합니다. <%=user_id %> 님!!!</h1>
    }
%>
</body>
</html>
```

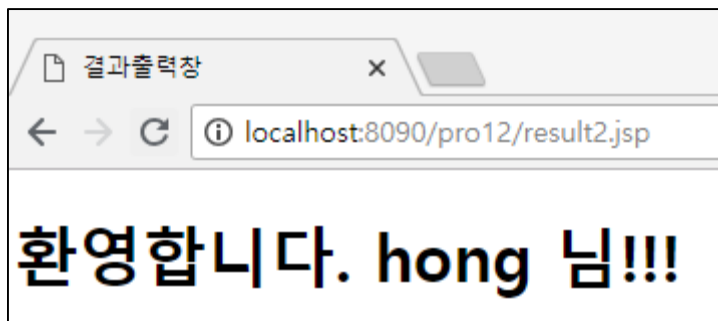
ID가 정상적으로 입력되었는지 체크합니다.

ID를 입력하지 않았을 경우 다시 로그인창으로 이동합니다.

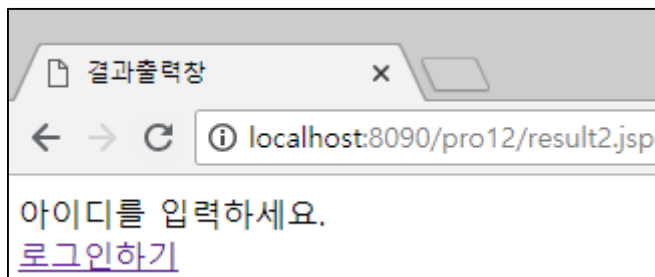
ID를 정상적으로 입력했을 경우 메시지를 표시합니다.

■ 로그인 예제

7. login.html의 action 속성을 result2.jsp로 수정 후, 로그인 창에서 먼저 ID를 정상적으로 입력한 후 전송했을 때의 결과를 확인합니다.



8. 다음은 ID를 입력하지 않고 전송한 경우입니다.



■ 로그인 예제

9. 로그인 예제를 조금 더 응용해 보겠습니다. 다음과 같이 result3.jsp를 작성합니다.

첫 번째 if문에서 먼저 ID가 입력되었는지 체크한 후 정상적으로 입력되었으면 다시 내부 if문을 수행하여 ID가 admin인지 체크합니다.

코드 12-8 pro12/WebContent/result3.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    request.setCharacterEncoding( "utf-8" );
    String user_id = request.getParameter("user_id");
    String user_pw = request.getParameter("user_pw");
%>
<!DOCTYPE html>
<html>
<head>
    <title>결과출력창</title>
    <meta charset="UTF-8">
</head>
<body>
<%
    if(user_id == null || user_id.length()==0){
        아이디를 입력하세요.<br>
        <a href="/pro12 /login.html">로그인하기</a>
    }
%>
```

ID가 정상적으로 입력되었는지 체크합니다.

스크립트 요소

로그인 예제

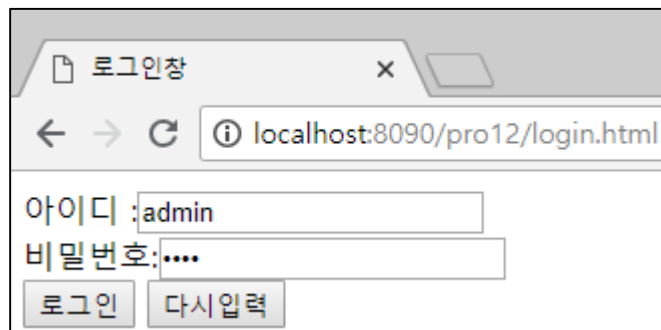
```
}else{
  if(user_id.equals("admin")){
  %>
    <h1>관리자로 로그인 했습니다.</h1>
    <form>
      <input type=button value="회원정보 삭제하기" />
      <input type=button value="회원정보 수정하기" />
    </form>
  <%
  }else{
  %>
    <h1> 환영합니다. <%=user_id %> 님!!!</h1>
  <%
  }
  }
  %>
</body>
</html>
```

ID를 입력한 경우 ID가 admin인지 다시 체크합니다.

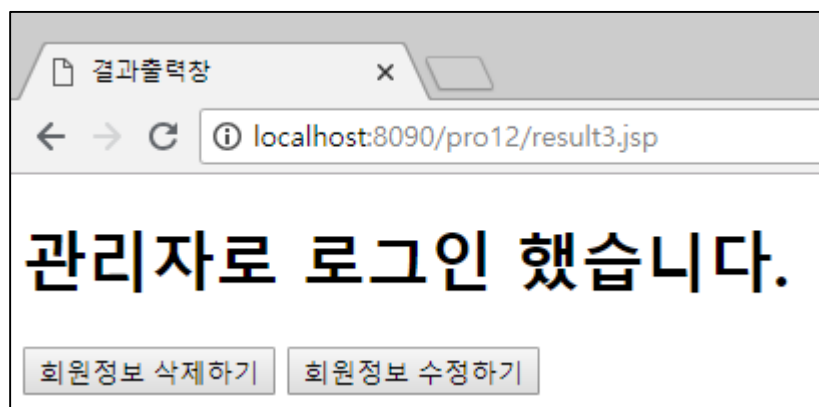
ID가 admin이면 관리자창을 나타냅니다.

로그인 예제

10. 다음은 admin으로 로그인했을 때의 실행 결과입니다.

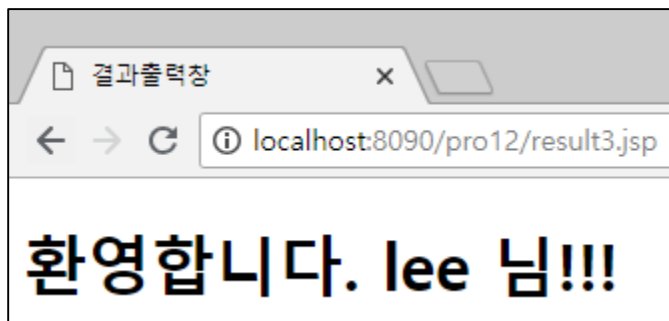


11. 관리자창이 나타납니다.



■ 로그인 예제

12. 다른 ID로 로그인 시 "환영합니다. lee 님!!!"이라는 메시지가 나타납니다.



스크립트 요소

로그인 예제

주의

❖ JSP 페이지의 화면 기능이 복잡해질수록 스크립트릿의 자바 코드와 HTML 태그가 같이 표시되므로 코드가 복잡해질 수 있습니다. 따라서 들여쓰기를 습관화해서 스크립트릿의 여닫는 부분이나 자바 코드의 괄호 여닫는 부분이 틀리지 않도록 주의해서 작성해야 합니다.

23행의 %>가 누락된 경우

```
14=<body>
15=<%
16  if(user_id == null || user_id.length()==0){
17  %>
18    아이디를 입력하세요.<br>
19    <a href="/pro12/login.html">로그인하기</a>
20=<%
21  }else{
22    if(user_id.equals("admin")){
23    <h1>관리자로 로그인 했습니다.</h1>
24=<form>
25=<input type=button value="회원정보 삭제하기" />
26=<input type=button value="회원정보 수정하기" />
27=</form>
28=<%
29=<%
30  }else{
31  %>
32  <h1> 환영합니다. <%=user_id %> 님!!!</h1>
33=<%
34  }
35  }
36=<%
37</body>
```

JSP 실행 시 스크립트릿 오류 출력

HTTP Status 500 – Internal Server Error

Type: Exception Report

Message: Unable to compile class for JSP:

Description: The server encountered an unexpected condition that prevented it from fulfilling the request.

Exception:

org.apache.jasper.JasperException: Unable to compile class for JSP:

An error occurred at line: [24] in the jsp file: [/result3.jsp]

h1 cannot be resolved to a type

```
21:     }else{
22:         if(user_id.equals("admin")){
23:
24:         <h1>관리자로 로그인 했습니다.</h1>
25:         <form>
26:         <input type=button value="회원정보 삭제하기" />
27:         <input type=button value="회원정보 수정하기" />
```

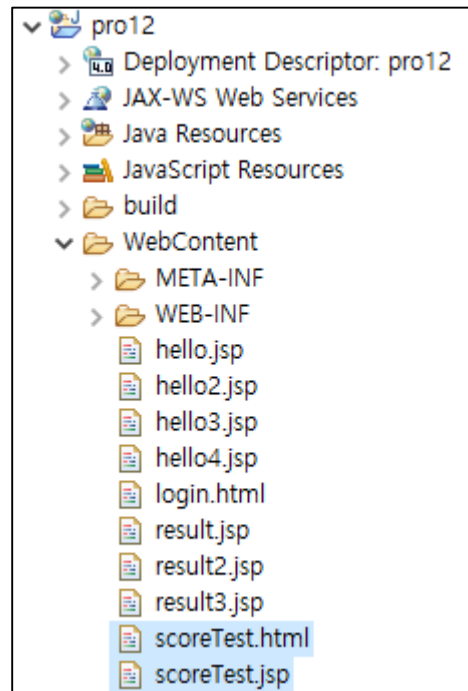
An error occurred at line: [24] in the jsp file: [/result3.jsp]

Syntax error, insert "super () ;" to complete BlockStatements

```
21:     }else{
22:         if(user_id.equals("admin")){
```

■ 학점 변환 예제

1. 다음과 같이 scoreTest.html, scoreTest.jsp 파일을 준비합니다.



■ 학점 변환 예제

2. scoreTest.html을 다음과 같이 작성합니다. 사용자로부터 시험 점수를 입력 받아 scoreTest.jsp로 전송합니다.

코드 12-9 pro12/WebContent/scoreTest.html

```
<!DOCTYPE html>
<html>
<head>...</head>
<body>
  <h1>시험 점수를 입력해 주세요</h1>
  <form method=get action="scoreTest.jsp">
    시험점수 :<input type=text name="score" /> <br>
    <input type="submit" value="변환하기">
  </form>
</body>
</html>
```

입력한 시험 점수를 scoreTest.jsp로 전송합니다.

■ 학점 변환 예제

3. scoreTest.jsp를 다음과 같이 작성합니다. scoreTest.html로부터 받은 점수를 다중 if~else if문을 이용해 학점으로 변환합니다.

코드 12-10 pro12/WebContent/scoreTest.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    request.setCharacterEncoding("utf-8");
    int score=Integer.parseInt(request.getParameter("score"));
%>
<!DOCTYPE html>
<html>
<head>
    <title>점수 출력창</title>
    <meta charset="UTF-8">
</head>
<body>
    <h1>시험점수 <%=score %>점</h1><br>
    <%
        if(score>=90){
```

전송된 시험 점수를 가져옵니다.

90점 이상이면 A를 출력합니다.

■ 학점 변환 예제

```
<h1>A학점입니다.</h1>
<%
}else if(score>=80 && score<90){
%>
<h1> B학점입니다.</h1>
<%
}else if(score>=70 && score<80){
%>
<h1> C학점입니다.</h1>
<%
}else if(score>=60 && score<70){
%>
<h1> D학점입니다.</h1>
<%
}else{
%>
<h1> F학점입니다.</h1>
%>
}
%>
<br>
<a href="scoreTest.html">시험점수입력</a>
</body>
</html>
```

80~90점 사이면 B를 출력합니다.

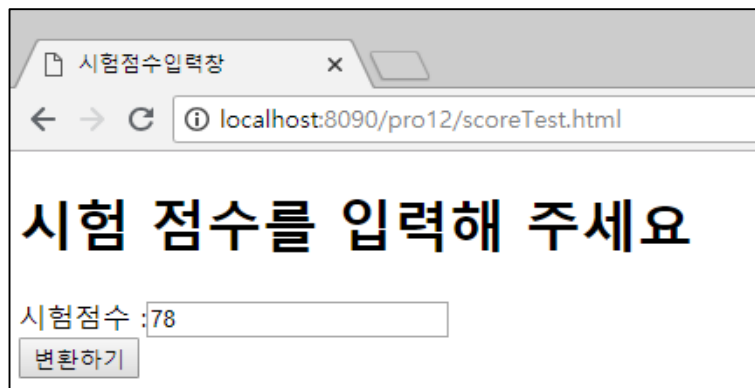
70~80점 사이면 C를 출력합니다.

60~70점 사이면 D를 출력합니다.

그 외 점수는 F를 출력합니다.

■ 학점 변환 예제

4. `http://localhost:8090/pro12/scoreTest.html`로 요청하여 시험점수 입력창에 시험 점수를 입력한 후 변환하기를 클릭합니다.



시험점수입력창

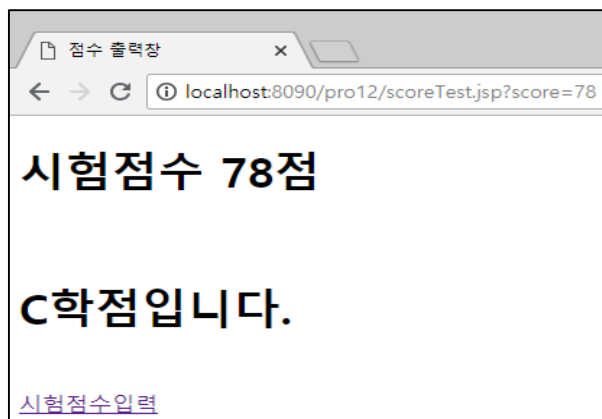
localhost:8090/pro12/scoreTest.html

시험 점수를 입력해 주세요

시험점수 :78

변환하기

5. 시험 점수를 학점으로 변환하여 출력합니다.



점수 출력창

localhost:8090/pro12/scoreTest.jsp?score=78

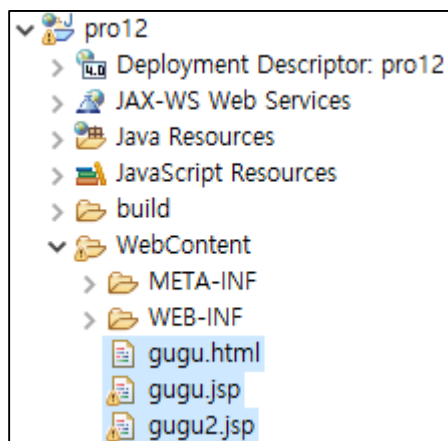
시험점수 78점

C학점입니다.

[시험점수입력](#)

■ 구구단 출력 예제

1. 구구단 예제 실습 파일인 gugu.html, gugu.jsp, gugu2.jsp를 준비합니다.



■ 구구단 출력 예제

2. gugu.html을 다음과 같이 작성합니다. 출력할 구구단의 단수를 입력 받아 gugu.jsp로 전송합니다.

코드 12-11 pro12/WebContent/gugu.html

```
<!DOCTYPE html>
<html>
<head>...</head>
<body>
  <h1> 구구단의 단수를 입력하세요.</h1>
  <form method=get action="gugu.jsp">
    출력할 구구단 : <input type='text' name='dan' /> <br>
                   <input type='submit' value='출력하기'>
  </form>
</body>
</html>
```

구구단의 단수를 gugu.jsp로 ~~포워딩~~ **전송합니다.**

스크립트 요소

■ 구구단 출력 예제

3. gugu.jsp를 다음과 같이 작성합니다. 스크립트릿 안에서 자바 for문을 이용해 <table> 태그의 행을 나타내는 <tr> 태그를 연속해서 브라우저로 출력합니다.

코드 12-12 pro12/WebContent/gugu.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    request.setCharacterEncoding("utf-8");
    int dan=Integer.parseInt(request.getParameter("dan"));
%>
<!DOCTYPE html>
<html>
<head>...</head>
<body>
    <table border='1' width='800' >
        <tr align='center' bgcolor='#FFFF66'>
```

전송된 단수를 구합니다.

스크립트 요소

■ 구구단 출력 예제

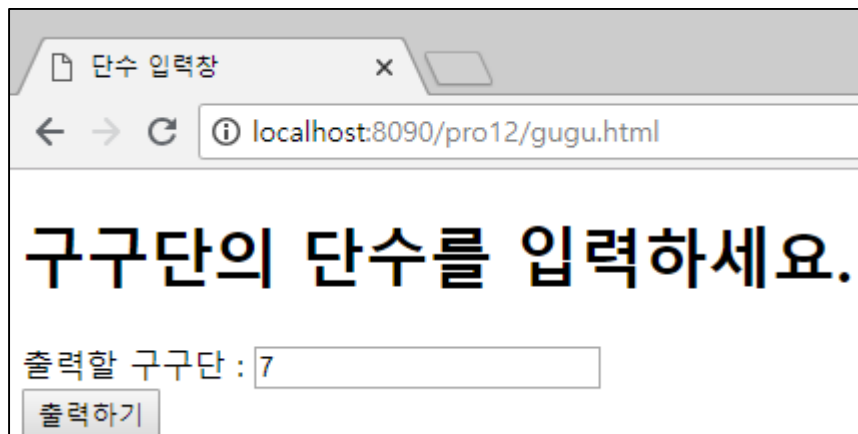
```
<td colspan='2'><%= dan %>단 출력 </td>
</tr>
<%
  for(int i=1; i<10;i++){
  %>
    <tr align='center'>
      <td width='400'>
        <%=dan %> * <%=i %>
      </td>
      <td width='400'>
        <%=i*dan %>
      </td>
    </tr>
  <%
    }
  %>
</table>
</body>
</html>
```

전송된 단수를 출력합니다.

for 반복문을 이용해 테이블의 각 행에 연속해서 구구단을 출력합니다.

■ 구구단 출력 예제

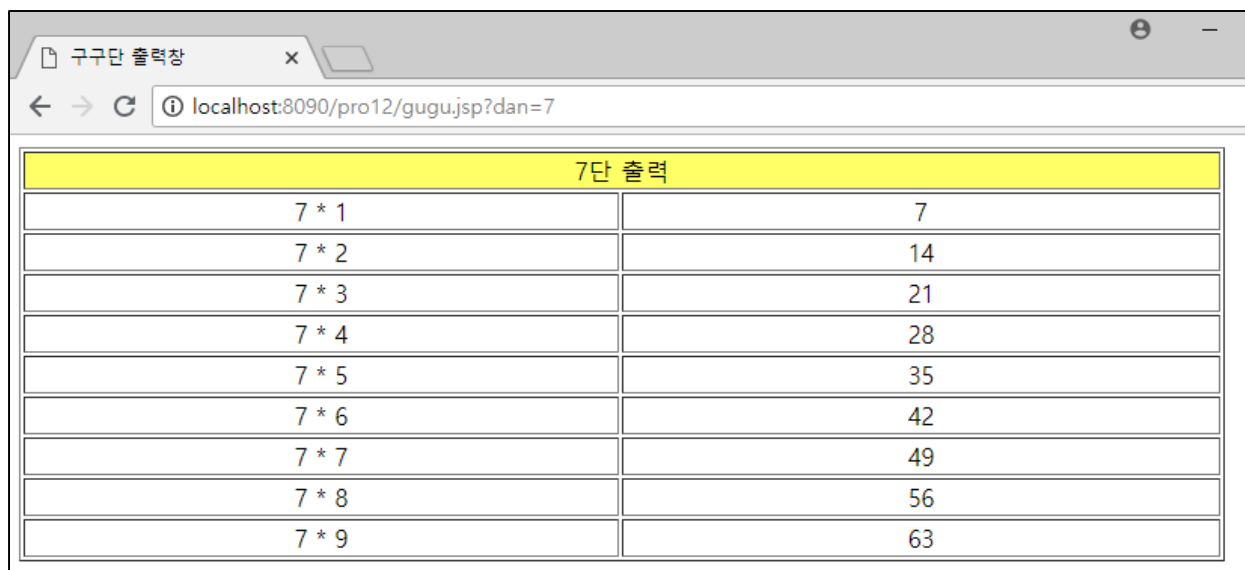
4. <http://localhost:8090/pro12/gugu.html>로 요청하여 입력창에서 단수를 입력한 후 전송합니다.



The screenshot shows a web browser window with the title '단수 입력창' (Single Number Input) and the address bar displaying 'localhost:8090/pro12/gugu.html'. The main content area contains the text '구구단의 단수를 입력하세요.' (Enter the number for the multiplication table). Below this text is a label '출력할 구구단 : ' followed by a text input field containing the number '7'. At the bottom left of the form is a button labeled '출력하기' (Output).

■ 구구단 출력 예제

5. for문을 이용해 구구단을 리스트로 출력합니다.



The screenshot shows a web browser window with the title '구구단 출력창' and the URL 'localhost:8090/pro12/gugu.jsp?dan=7'. The main content is a table with a yellow header row labeled '7단 출력'. The table contains 9 rows of multiplication results for the number 7, from 7 * 1 to 7 * 9.

7단 출력	
7 * 1	7
7 * 2	14
7 * 3	21
7 * 4	28
7 * 5	35
7 * 6	42
7 * 7	49
7 * 8	56
7 * 9	63

■ 구구단 출력 예제

6. 다음과 같이 gugu2.jsp를 작성합니다. if문에서 for 반복문의 반복 변수 i를 사용해 홀수인지 짝수인지를 체크합니다. 그런 다음 <tr> 태그의 bgcolor 속성 값을 다르게 설정하여 브라우저로 출력합니다.

코드 12-13 pro12/WebContent/gugu2.jsp

```
...
<%
    for(int i=1; i<10;i++){
%>
<%
    if(i%2==1){
%>
        <tr align=center bgcolor="#CCFF66">
<%
    }else{
%>
        <tr align=center bgcolor="#CCCCFF">
<%
    }
%>
...

```

테이블의 홀수 행과 짝수 행의 색깔을
다르게 표시합니다.

■ 구구단 출력 예제

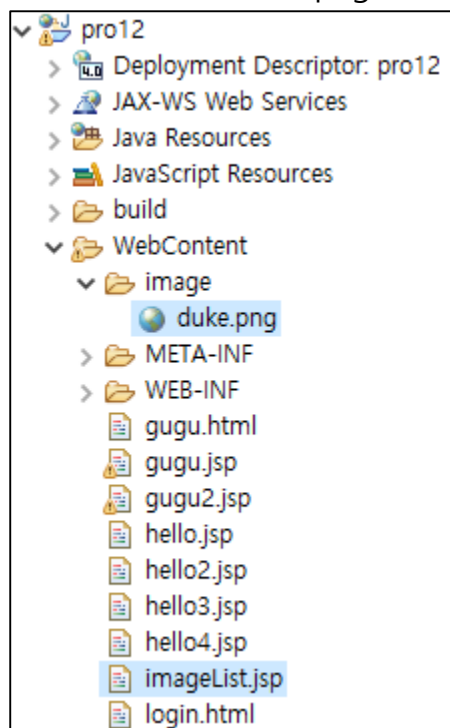
7. 브라우저에서 실행하면 홀수 행과 짝수 행의 배경색이 다르게 출력됩니다.



7 단 출력	
7 * 1	7
7 * 2	14
7 * 3	21
7 * 4	28
7 * 5	35
7 * 6	42
7 * 7	49
7 * 8	56
7 * 9	63

■ 이미지 리스트 출력 예제

1. imageList.jsp를 생성하고 실습 이미지인 duke.png를 추가합니다.



■ 이미지 리스트 출력 예제

2. imageList.jsp를 다음과 같이 작성합니다. for 반복문을 이용해 태그 안에 태그를 연속적으로 출력해서 이미지를 나타냅니다.

```
<body>
<ul class="lst_type">
  <li>
    <span style='margin-left:50px' >이미지 </span>
    <span >이미지 이름</span>
    <span >선택하기</span>
  </li>
  <%
    for(int i=0 ; i<10; i++){
      <li>
        <a href='#' style='margin-left:50px' >
          <img src='image/duke.png' width='90' height='90' alt='' /></a>
          <a href='#' ><strong>이미지 이름: 듀크<%=i %> </strong></a>
          <a href='#' > <input name='chk<%=i %>' type='checkbox' /></a>
        </li>
      <%
        }
      <%
    }
  </ul>
```

리스트의 헤더를 표시합니다.

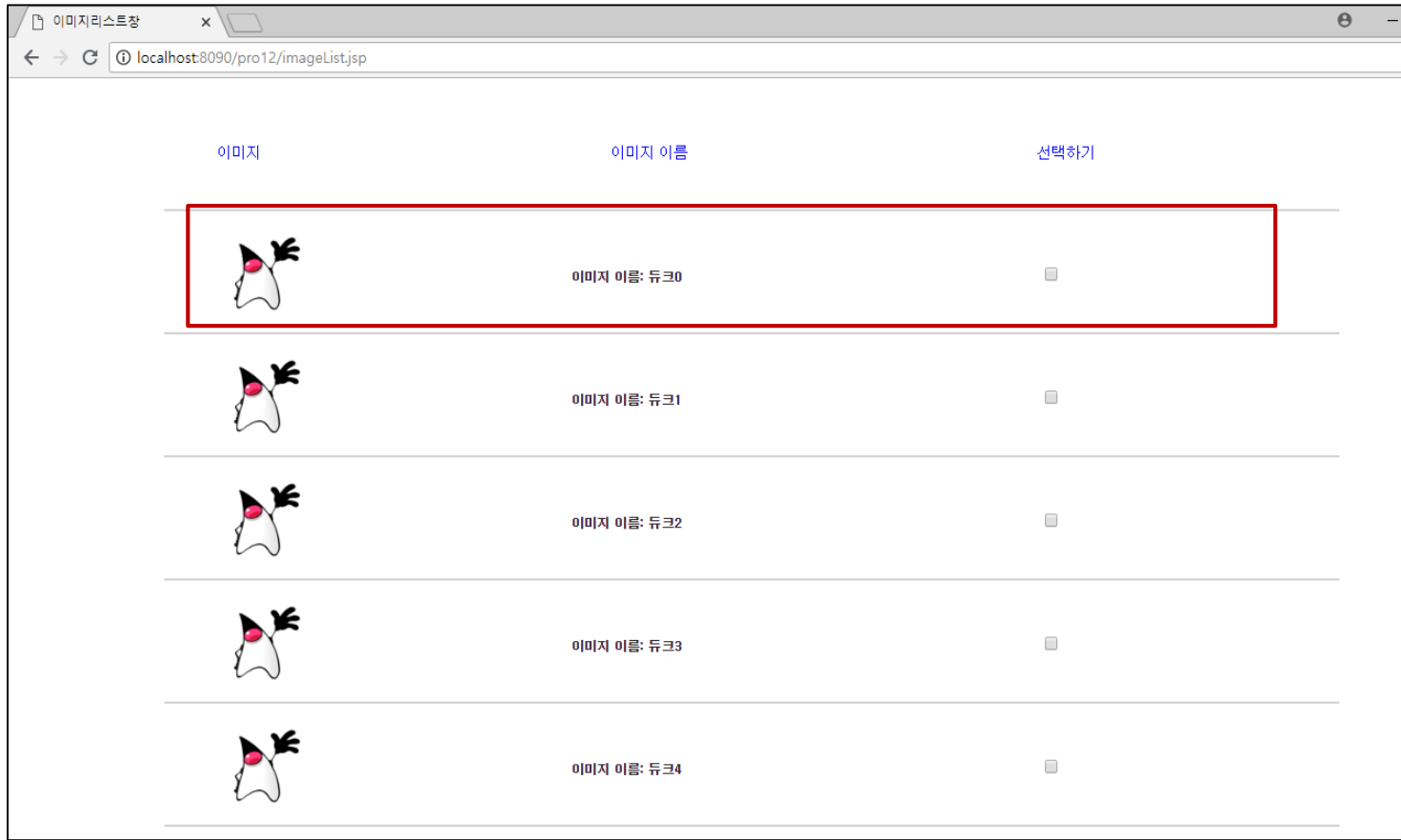
for 반복문을 이용해 태그를 연속해서 출력합니다.

 태그를 이용해 한 행에 <a> 태그의 이미지와 텍스트를 나타냅니다.

image 폴더의 이미지를 나타냅니다.

■ 이미지 리스트 출력 예제
















3. <http://localhost:8090/pro12/imageList.jsp>로 요청하면 다음과 같이 출력됩니다.



스크립트 요소

■ 이미지 리스트 출력 예제

검색 상품 리스트 출력

플러스 상품 <small>광고</small>			
	[삼성전자] 삼성컴퓨터 SSD장착 최대80% 할인	199,000원 <small>400,000원 50% ↓</small> <small>무료배송</small>	 
삼성 컴퓨터 모음전		상품평 	
	[삼성전자] 삼성 사무용 컴퓨터 i3 i5 듀얼 쿼드코어 PC 모음전	155,000원 <small>무료배송</small>	 
삼성 사무용 컴퓨터		상품평 	
	[LG전자] LG데스크탑 Z71EV-AX7P26 G4560 4G SSD 128 원도우선택	419,000원 <small>445,740원 5% ↓</small> <small>무료배송</small>	 
<small>원도우G4560 / 4G / SSD 128G / 유선네트워크 / 윈도우선택</small> LG 데스크탑 PC Z71EV-AX7P26		상품평 	
	<small>중고</small> 빠른부팅신속SSD넉넉한저장공간 DB-Z600 슬림PC	299,000원 <small>무료배송</small>	 

Section 04

[실습 6-1] JSP 기초 종합 예제

- 이번 실습의 개요
 - JSP의 기본 구성요소인 지시어와 주석, 스크립트 요소를 실습
 - 주석
 - 선언과 참조
 - include 지시어
 - 스크립트릿

- 1) 이클립스에서 [jwbook] → [src] → [main] → [webapp] 폴더에 [ch06] 폴더를 생성한 다음 <New> → <JSP File> 메뉴를 클릭하여 JSP 파일을 생성
 - 파일 이름: 'jspTotal.jsp'
- 주석 작성
- 2) 'jspTotal.jsp'의 <body> 내부에 다음 내용을 작성하여 HTML 주석과 JSP 주석 확인

```
<h3>
  1. JSP 주석
  <!-- HTML 주석: 화면에서는 안 보이고 소스 보기에는 보임 -->
  <%-- JSP 주석: 화면과 소스 보기에서 보이지 않음 --%>
</h3>
```

- 선언과 참조

- 3) 일반 변수의 선언은 스크립트릿에서 가능하지만 멤버 변수 혹은 함수 선언은 스크립트릿에서 불가능함. 따라서 이번 실습에서는 선언문을 이용해 배열과 함수를 선언해두고 참조하도록 코드를 구성함

```
<%!  
    String[] members = { "김길동", "홍길동", "김사랑", "박사랑" };  
    int num1 = 10;  
  
    int calc(int num2) {  
        return num1 + num2;  
    }  
%>
```

4) 스크립트릿을 활용하여 calc() 메서드를 호출

- 선언된 코드는 본문의 표현식 혹은 스크립트릿 등에서 자유롭게 사용할 수 있음

```
<h3>  
    2. calc(10) 메서드 실행 결과:  
    <%=calc(10)%></h3>  
<hr>
```

- **include** 지시어

5) 2장에서 작성했던 'hello.jsp' 파일을 포함하기.

- 파일의 위치와 경로를 작성할 때 주의해야 함
- 현재 코드의 위치는 [ch06] 폴더에 있고, 'hello.jsp' 파일은 [webapp] 폴더 바로 아래에 있기 때문에 상대 경로로 위치를 표현해주어야 함

```
<h3>3. include: hello.jsp</h3>  
<%@ include file="../hello.jsp"%>
```

- 스크립트릿

6) 자바의 for 문으로 앞에서 선언했던 members 배열의 값을 모두 출력하기

- 이름을 출력하는 부분에서 out.println()을 사용하지 않기 위해 for 문을 중간에 닫고 스크립트 요소를 사용한 다음 다시 for 문을 닫는 코드 형식으로 구현

```
<h3>4. 스크립트(배열 데이터 출력)</h3>
<ul>
  <%
    for (String name : members) {
  %>
    <li><%=name%></li>
  <%
    }
  %>
</ul>
```


7) 완성된 코드와 실행

- ① 서블릿 클래스를 선택한 다음 실행 버튼을 누르기
- ② 마우스 오른쪽 버튼을 클릭하고 [Run as] → [Run on Server]를 클릭하여 실행하기
 - JSP와 마찬가지로 서버 런타임 선택 화면에서 프로젝트 생성 시 등록한 'Tomcat v9.0'을 선택하고 실행

예제 6-1

jspTotal.jsp

```
1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3  <!DOCTYPE html>
4  <html>
5  <head>
6  <meta charset="UTF-8">
7  <title>JSP 종합 예제</title>
8  </head>
9  <body>
10     <h2>JSP 종합 예제</h2>
11     <hr>
12
13     <%!
14         String[] members = { "김길동", "홍길동", "김사랑", "박사랑" };
15         int num1 = 10;
16
17         int calc(int num2) {
18             return num1 + num2;
19         }
20     %>
21
```

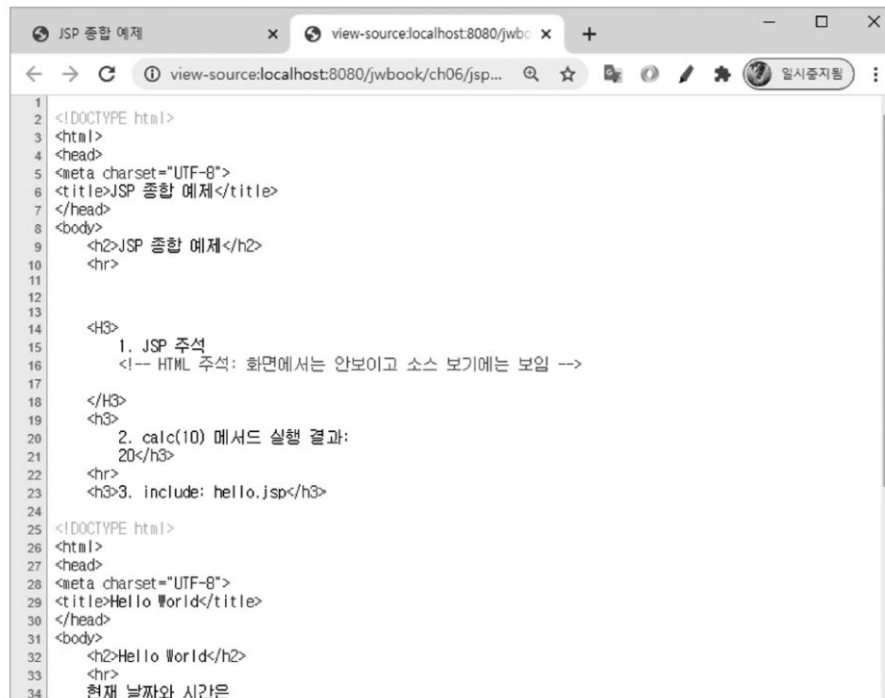
```

22 <h3>
23     1. JSP 주석
24     <!-- HTML 주석: 화면에서는 안보이고 소스 보기에는 보임 -->
25     <%-- JSP 주석: 화면과 소스 보기에서 보이지 않음 --%>
26 </h3>
27 <h3>
28     2. calc(10) 메서드 실행 결과:
29     <%=calc(10)%></h3>
30 <hr>
31 <h3>3. include: hello.jsp</h3>
32     <%@ include file="../hello.jsp"%>
33 <hr>
34 <h3>4. 스크립트(배열 데이터 출력)</h3>
35     <ul>
36     <%
37         for (String name : members) {
38         %>
39         <li><%=name%></li>
40     <%
41         }
42     %>
43     </ul>
44 </body>
45 </html>

```



- 8) 크롬 웹 브라우저에서 마우스 오른쪽 버튼을 클릭하고 <페이지 소스 보기>를 클릭하여 출력 여부를 확인
- HTML 주석은 <소스 보기> 화면에서 확인이 됨
 - JSP 주석은 확인이 안 됨



```
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title>JSP 종합 예제</title>
6 </head>
7 <body>
8 <h2>JSP 종합 예제</h2>
9 <hr>
10
11
12
13 <h3>
14 1. JSP 주석
15 <!-- HTML 주석: 화면에서는 안보이고 소스 보기에는 보임 -->
16
17 </h3>
18 <h3>
19 2. calc(10) 메서드 실행 결과:
20 20</h3>
21 <hr>
22 <%>3. include: hello.jsp<%>
23
24 <!DOCTYPE html>
25 <html>
26 <head>
27 <meta charset="UTF-8">
28 <title>Hello World</title>
29 </head>
30 <body>
31 <h2>Hello World</h2>
32 <hr>
33 현재 날짜와 시간은
34
```

Section 05

[실습 6-2] JSP 프로그래밍 : 계산기 구현

- 이번 실습의 개요
 - 5장에서 만들었던 계산기 서블릿을 JSP 버전으로 구현
 - 2개의 숫자와 연산자를 선택하고 계산 버튼을 누르면 입력값을 JSP로 전달함
 - HTML 폼을 통해 숫자, 연산자를 입력하고 입력값을 JSP로 전달
 - JSP는 브라우저로부터 전달된 입력값을 가져와 계산 후 결과를 포함한 화면을 출력

- 입력 파라미터

- **n1, n2:** 숫자 입력. HTML `<input>`으로 구현함
- **op:** 연산자 선택 드롭다운 리스트로 HTML `<select>`로 구현함
 - 연산자의 종류 : +, -, *, /

1) [ch05] 폴더의 'calcForm.html' 파일을 복사해 [ch06]에 붙여 넣고 코드 수정

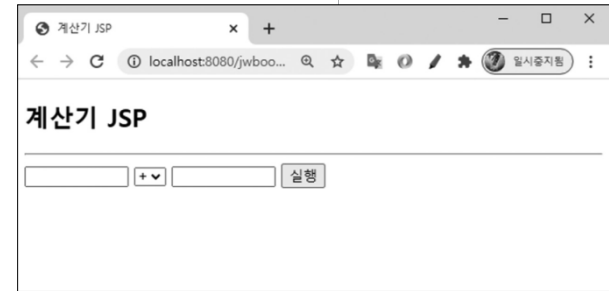
예제 6-2

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title>계산기 JSP</title>
6 </head>
7 <body>
8   <h2>계산기 JSP</h2>
9   <hr>
10   <form method="post" action="calc.jsp">
11     <input type="text" name="n1" size="10"> <select name="op">
12       <option>+</option>
13       <option>-</option>
14       <option>*</option>
15       <option>/</option>
16     </select> <input type="text" name="n2" size="10">
17     <input type="submit" value="실행">
18   </form>
19 </body>
20 </html>

```

calcForm.html



- 2) [ch06] 폴더에 'calc.jsp' 이름으로 계산기 JSP를 생성하고 page 지시어 바로 아래에 다음 내용을 코딩
- 계산기 기본 코드는 서블릿과 동일함
 - 따라서 기본 생성된 JSP 코드에 5장에서 작성한 'CalcServlet.java'의 서블릿 계산기 구현 부분인 doGet() 메서드의 내용을 스크립트릿 영역에 넣어주면 됨
 - 주의할 점) HTML 부분에서 결과를 출력해야 하므로 코드는 앞쪽에 배치해야 함

```
<%
    int n1 = Integer.parseInt(request.getParameter("n1"));
    int n2 = Integer.parseInt(request.getParameter("n2"));

    long result = 0;

    switch(request.getParameter("op")) {
        case "+" : result = n1+n2; break;
        case "-" : result = n1-n2; break;
        case "/" : result = n1/n2; break;
        case "*" : result = n1*n2; break;
    }
%>
```

3) 결과 화면은 <body> 부분에 표현식을 이용하거나 EL을 사용할 수도 있음

```
<body>  
  <h2>계산 결과-JSP</h2>  
  <hr>  
  결과: <%=result %>  
</body>
```


4) 계산기 JSP의 전체 코드

예제 6-3

calc.html

```

1  <%@ page language="java" contentType="text/html; charset=UTF-8"
2      pageEncoding="UTF-8"%>
3
4  <%
5      int n1 = Integer.parseInt(request.getParameter("n1"));
6      int n2 = Integer.parseInt(request.getParameter("n2"));
7
8      long result = 0;
9
10     switch(request.getParameter("op")) {
11         case "+": result = n1+n2; break;
12         case "-": result = n1-n2; break;
13         case "/": result = n1/n2; break;
14         case "*": result = n1*n2; break;
15     }
16 %>
17 <!DOCTYPE html>
18 <html>
19 <head>
20 <meta charset="UTF-8">
21 <title>계산기 JSP</title>
22 </head>

```

```

23 <body>
24     <h2>계산 결과-JSP</h2>
25     <hr>
26     결과: <%=result %>
27 </body>
28 </html>

```

- 5) 'calcForm.html'을 먼저 실행하고 숫자 입력과 연산자를 선택한 다음 실행 버튼을 눌러 계산기 JSP가 정상적으로 호출되고 결과가 출력되는지 확인

