

11장 JSP 정의와 구성 요소

11.1 JSP 등장 배경

11.2 JSP의 3단계 작업 과정

11.3 JSP 페이지 구성 요소

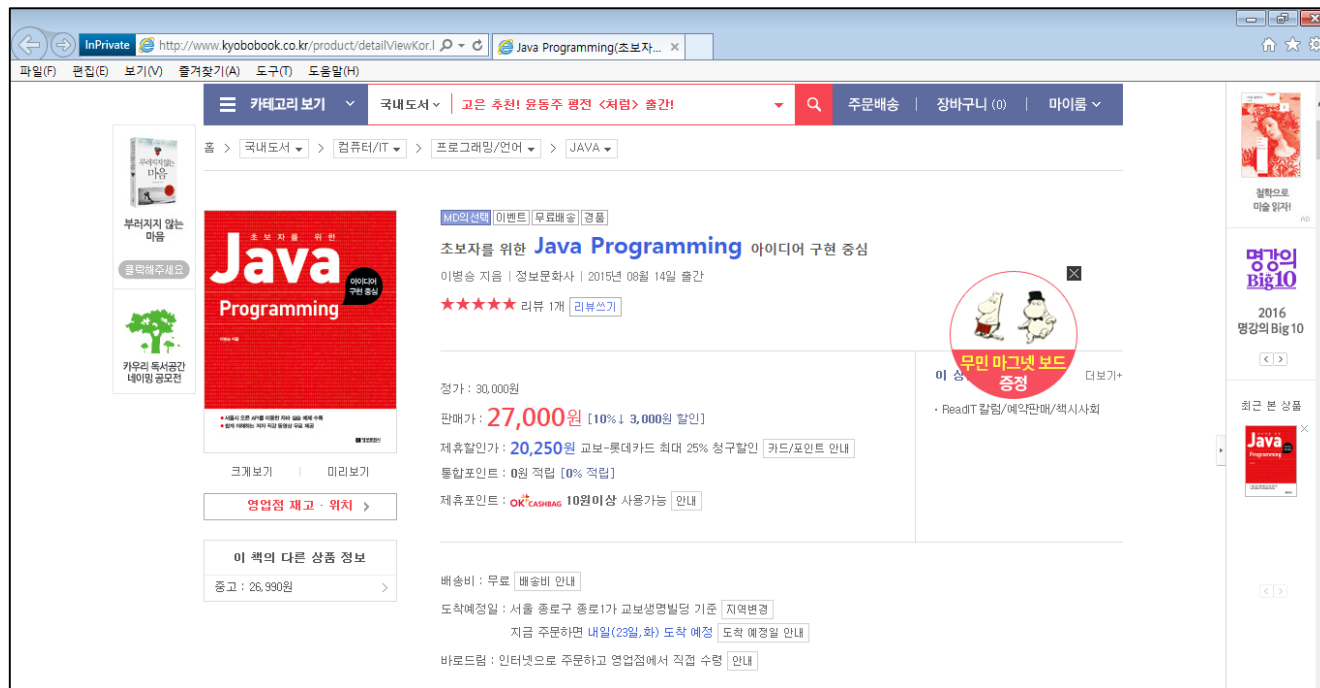
11.4 디렉티브 태그

11.1 JSP 등장 배경

• 11.1.1 서블릿으로 화면 구현 시 문제점

- 기존 서블릿에서는 자바 코드를 기반으로 문자열을 사용해 HTML과 자바스크립트로 화면을 구현했음
- JSP는 이와 반대로 HTML, CSS와 자바스크립트를 기반으로 JSP 요소들을 사용해 화면을 구현함

온라인 서점 화면



11.1 JSP 등장 배경

온라인 서점 구현 HTML과 자바스크립트 코드

```

771 function getFileExtension( filePath ) { //파일의 확장자를 가져옴
772     var lastIndex = -1;
773     var extension = "";
774
775     lastIndex = filePath.lastIndexOf(".");
776     if (lastIndex != -1) {
777         extension = filePath.substring(lastIndex + 1, filePath.length);
778     }
779
780     return extension.toLowerCase();
781 }
782
783 function regReview(){
784     var forms = document.reviewFrm;
785
786
787     alert("로그인 후 작성 가능합니다.");
788     fn_openLogin(location.href);
789     return;
790
791
792     if(jQuery("#reviewFrm input[name='imageFile']").val() != null && jQuery("input[name='imageFile']").val() == ""){
793         var forms = document.reviewFrm;
794         var extension = getFileExtension(forms.imageFile.value);
795         if (extension != ".jpg" && extension != ".jpeg") {
796             alert(".jpg, jpeg파일만 사용할 수 있습니다.");
797             return;
798         }
799     }
800
801     if(jQuery(".book_review textarea").val() == "" || jQuery(".book_review textarea").val() == "내용을 입력해주세요. 주제와 무관한 댓글, 악플은 삭제될 수 있습니다."){
802         alert("내용을 입력하세요.");
803         return;
804     }
805
806     if(jQuery("input[name='rating']").val() == "0"){
807         alert("컨텐츠 평가를 해주세요.");
808         return;
809     }
810
811     if(!confirm("댓글을 등록하시겠습니까?"))

```

11.1 JSP 등장 배경

```
1473 function divThisShow(divName){
1474     document.getElementById(divName).style.display = 'block';
1475     for (i=0;i<intSelect.length;i++){intSelect[i].style.display = "none";}
1476 }
1477 function divThisHidden(divName){
1478     document.getElementById(divName).style.display = 'none';
1479     for (i=0;i<intSelect.length;i++){intSelect[i].style.display = "";}
1480 }
1481 function selThisHidden(){
1482     for (i=0;i<intSelect.length;i++){intSelect[i].style.display = "none";}
1483 }
1484 function selThisShow(){
1485     for (i=0;i<intSelect.length;i++){intSelect[i].style.display = "";}
1486 }
1487
1488 ///## [사이트 북마크 관련] ##
1489 function bookmarksiteFav() {
1490     var formname = document.FormFav;
1491     formname.target = "iframefav";
1492     formname.action = "http://www.kyobobook.co.kr/indexfav1.jsp?Kc=GNHHNDbookmark&orderClick=c23";
1493     formname.method = "post";
1494     formname.submit();
1495 }
1496 function bookmarksite(title,url) {
1497     if (window.sidebar) // firefox
1498         window.sidebar.addPanel(title, url, "");
1499     else if(window.opera && window.print)
1500     { // opera
1501         var elem = document.createElement('a');
1502         elem.setAttribute('href',url);
1503         elem.setAttribute('title',title);
1504         elem.setAttribute('rel','sidebar');
1505         elem.click();
1506     }
1507     else if(document.all) // ie
1508         window.external.AddFavorite(url, title);
1509 }
1510 </script>
1511
1512
1513
1514 <iframe name="iframefav" class="hidden" frameborder="0" width="0" height="0" title="빈프레임" ></iframe>
```

11.1 JSP 등장 배경

문제점

- 웹 프로그램의 화면 기능이 복잡해지므로 서블릿의 자바 기반으로 화면 기능 구현 시 어려움이 발생함
- 디자이너 입장에서 화면 구현 시 자바 코드로 인해 작업이 어려움
- 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!

11.1 JSP 등장 배경

- 11.1.2 JSP의 구성 요소

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

11.2 JSP의 3단계 작업 과정

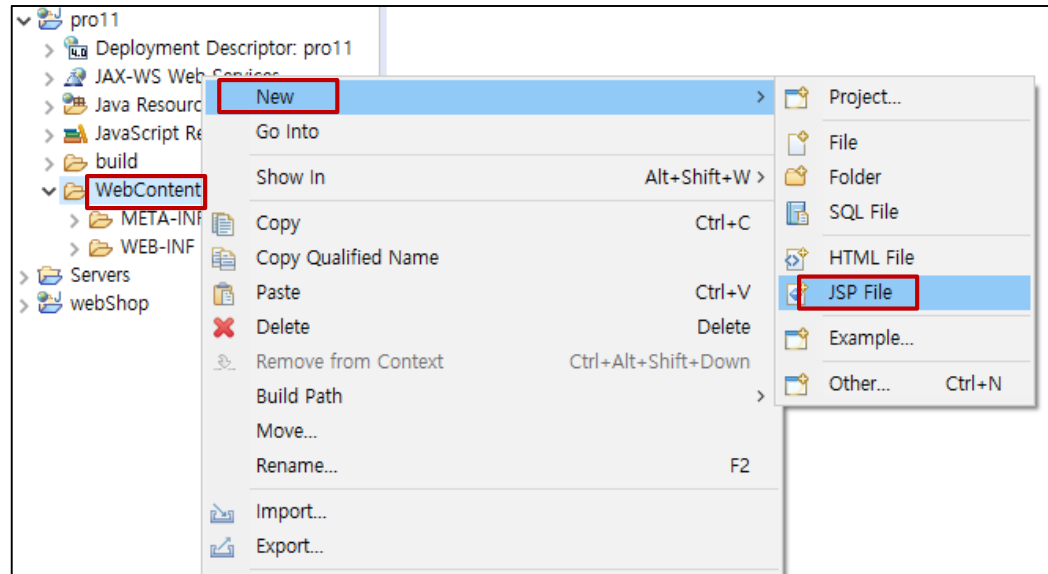
- 11.2.1 톰캣 컨테이너에서 JSP 변환 과정

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

11.2 JSP의 3단계 작업 과정

- 11.2.2 이클립스에서 JSP 변환 과정

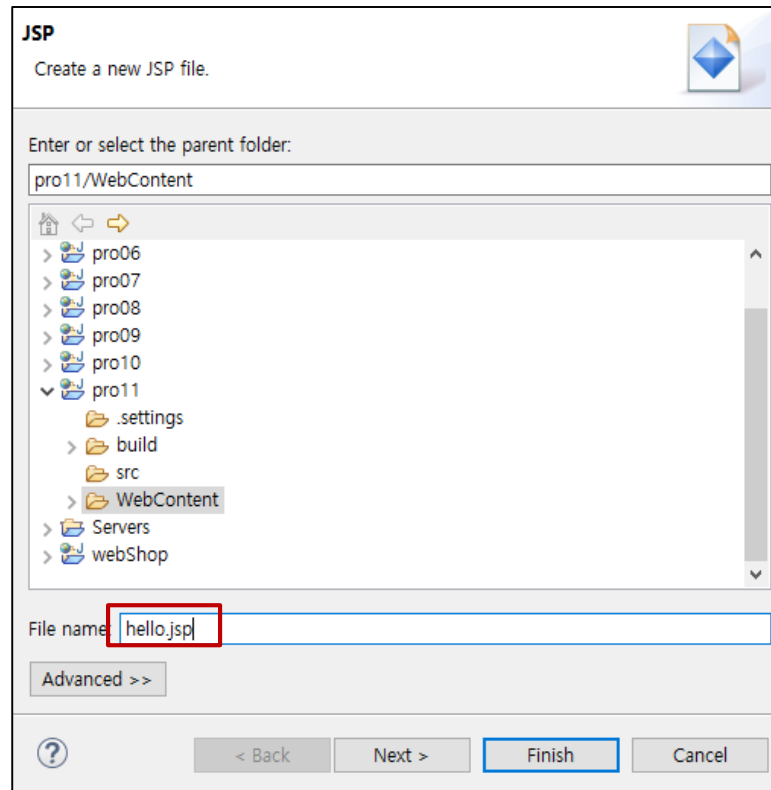
1. 이클립스에서 새 프로젝트 pro11을 만들고 WebContent 폴더에서 마우스 오른쪽 버튼을 클릭한 후 New > JSP File을 선택합니다.



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

11.2 JSP의 3단계 작업 과정

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



11.2 JSP의 3단계 작업 과정

3. 생성된 JSP 파일에 간단한 HTML 태그와 메시지를 작성합니다.

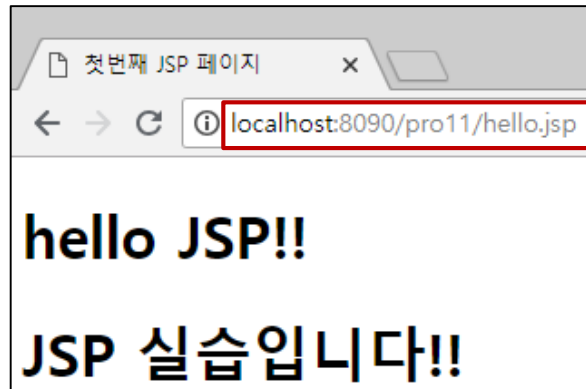
코드 11-2 pro11/WebContent/hello.jsp

```
<%@ 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>
```

11.2 JSP의 3단계 작업 과정

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

- **http://ip주소:포트번호/프로젝트이름/JSP파일이름**



11.2 JSP의 3단계 작업 과정

hello.jsp 출력 과정

브라우저에서 hello.jsp 요청



톰캣 컨테이너는 hello.jsp를 읽어와 hello_jsp.java로 변환



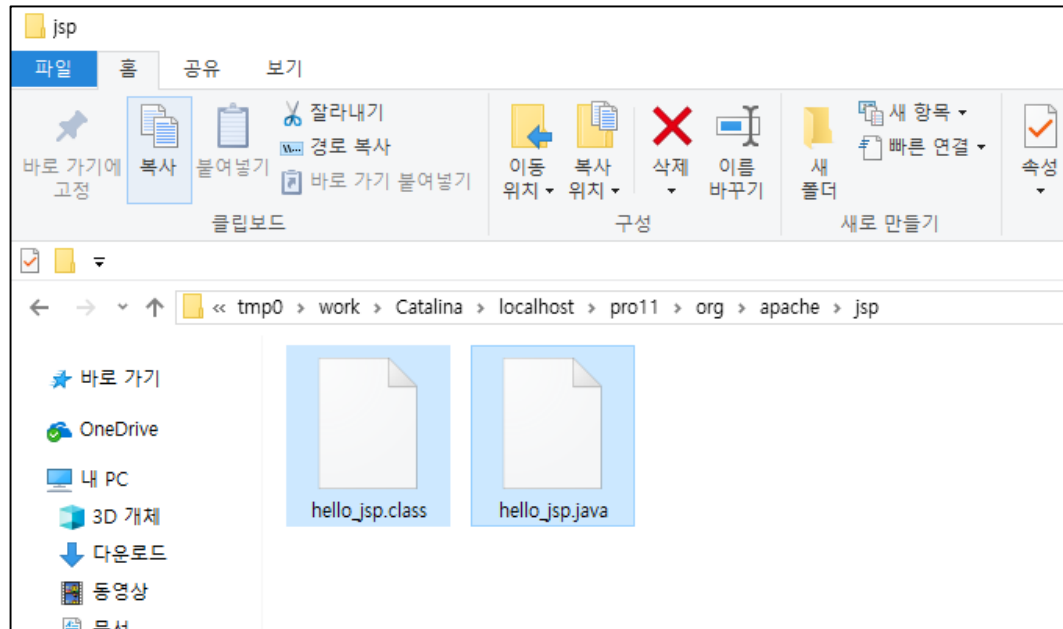
톰캣 컨테이너는 hello_jsp.java를 hello_jsp.class로 컴파일



컨테이너는 hello_jsp.class를 실행해서 브라우저로 HTML 전송

11.2 JSP의 3단계 작업 과정

hello_jsp.java 파일로 변환된 상태



❖ 이클립스에서 실행 시 자바 파일과 클래스 파일 생성 위치

```
%이클립스_workspace%\ .metadata\ .plugins\  
org.eclipse.wst.server.core\tmp0 \work\Catalina\localhost\pro11\org\apache\jsp
```

11.2 JSP의 3단계 작업 과정

hello_jsp.java로 변환한 후 브라우저로 전송한 HTML 태그

```
105 try {
106     response.setContentType("text/html; charset=UTF-8");
107     pageContext = _jspxFactory.getPageContext(this, request, response,
108         null, true, 8192, true);
109     _jspx_page_context = pageContext;
110     application = pageContext.getServletContext();
111     config = pageContext.getServletConfig();
112     session = pageContext.getSession();
113     out = pageContext.getOut();
114     _jspx_out = out;
115
116     out.write("\r\n");
117     out.write("<!DOCTYPE html>\r\n");
118     out.write("<html>\r\n");
119     out.write("<head>\r\n");
120     out.write("<meta charset=\"UTF-8\">\r\n");
121     out.write("<title>첫번째 JSP 페이지</title>\r\n");
122     out.write("</head>\r\n");
123     out.write("<body>\r\n");
124     out.write("    <h1>hello JSP!!</h1>\r\n");
125     out.write("    <h1>JSP 실습입니다!!</h1>\r\n");
126     out.write("</body>\r\n");
127     out.write("</html>\r\n");
128     out.write("\r\n");
129     out.write("\r\n");
130     out.write("\r\n");
131     out.write("\r\n");
132     out.write("\r\n");
133     out.write("\r\n");
134 } catch (java.lang.Throwable t) {
```

브라우저로 전송된 HTML 태그



```
< view-source:localhost:8090/pro11/hello.jsp
1 <!DOCTYPE html>
2 <html>
3 <head>
4 <meta charset="UTF-8">
5 <title>첫번째 JSP 페이지</title>
6 </head>
7 <body>
8     <h1>hello JSP!!</h1>
9     <h1>JSP 실습입니다!!</h1>
10 </body>
11 </html>
```

클래스 파일에서 전송된 HTML 태그와 일치

11.3 JSP 페이지 구성 요소

JSP 페이지 구성 요소

- 디렉티브 태그(Directive Tag)
- 스크립트 요소(Scripting Element): 주석문, 스크립트릿(Scriptlet), 표현식, 선언식
- 표현 언어(Expression Language)
- 내장 객체(내장 변수)
- 액션 태그(Action Tag)
- 커스텀 태그(Custom Tag)

11.4 디렉티브 태그

디렉티브 태그의 종류

- 페이지 디렉티브 태그(Page Directive Tag): JSP 페이지의 전반적인 정보를 설정할 때 사용
- 인클루드 디렉티브 태그(Include Directive Tag): 공통으로 사용하는 JSP 페이지를 다른 JSP 페이지에 추가할 때 사용
- 태그라이브 디렉티브 태그(Taglib Directive Tag): 개발자나 프레임워크에서 제공하는 태그를 사용할 때 사용

• 11.4.1 페이지 디렉티브 태그 정의와 사용법

- JSP 페이지의 여러가지 속성을 설정하는데 사용

페이지 디렉티브 태그로 설정하는 여러가지 JSP 속성

| 속성 | 기본값 | 설명 |
|-------------|-------------|---------------------------------------------------|
| info | 없음 | 페이지를 설명해 주는 문자열을 지정합니다. |
| language | "java" | JSP 페이지에서 사용할 언어를 지정합니다. |
| contentType | "text/html" | JSP 페이지 출력 형식을 지정합니다. |
| import | 없음 | JSP 페이지에서 다른 패키지의 클래스를 임포트할 때 지정합니다. |
| session | "true" | JSP 페이지에서 HttpSession 객체의 사용 여부를 지정합니다. |
| buffer | "8kb" | JSP 페이지 출력 시 사용할 버퍼 크기를 지정합니다. |
| autoFlush | "true" | JSP 페이지의 내용이 출력되기 전 버퍼가 다 채워질 경우 동작을 지정합니다. |
| errorPage | "false" | JSP 페이지 처리 도중 예외가 발생할 경우 예외 처리 담당 JSP 페이지를 지정합니다. |

11.4 디렉티브 태그

| 속성 | 기본값 | 설명 |
|--------------|--------------|----------------------------------------|
| isErrorPage | "false" | 현재 JSP 페이지가 예외 처리 담당 JSP 페이지인지를 지정합니다. |
| pageEncoding | "ISO-8859-1" | JSP 페이지에서 사용하는 문자열 인코딩을 지정합니다. |
| isELIgnored | "true" | JSP 2.0 버전에서 추가된 기능으로 EL 사용 유무를 지정합니다. |

페이지 디렉티브 태그 사용 형식

```
<%@ page 속성1="값1" 속성2="값2" 속성3="값3".... %>
```

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>
```

11.4 디렉티브 태그

- 11.4.2 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>
```

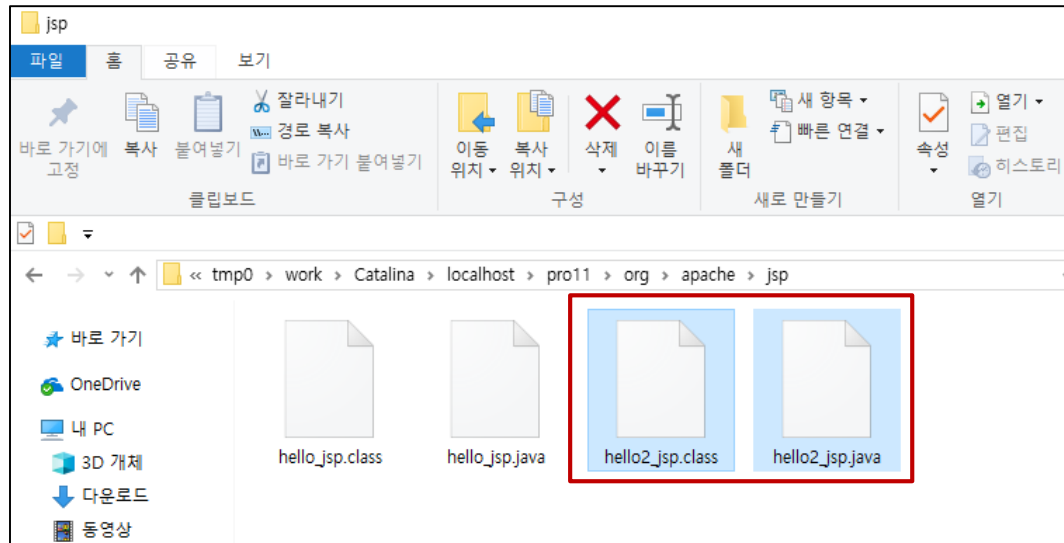
```
<h1>쇼핑몰 구현 중심 JSP입니다.!!!</h1>
```

```
</body>
```

```
</html>
```

11.4 디렉티브 태그

JSP 파일이 변환되어서 생성된 java 파일



11.4 디렉티브 태그

- 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.*;
```

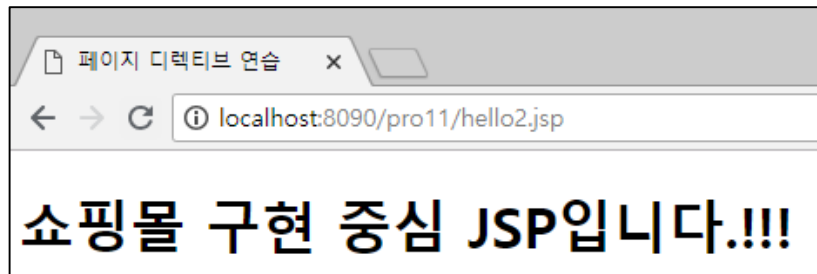
```
16  public final class hello2_jsp extends org.apache.jasper.runtime.HttpJspBase  
17      implements org.apache.jasper.runtime.JspSourceDependent,  
18      | | | | | org.apache.jasper.runtime.JspSourceImports {  
19  
20      public java.lang.String getServletInfo() {  
21          return "(ShoppingMall.....)";  
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;
```

11.4 디렉티브 태그

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



11.4 디렉티브 태그

❖ 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)
  
```

11.4 디렉티브 태그

• 11.4.3 인클루드 디렉티브 태그 정의와 사용법

쇼핑몰 메인 화면

쇼핑몰 상품 상세 화면

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

11.4 디렉티브 태그

인클루드 디렉티브 태그의 특징

- 재사용성이 높다.
- JSP 페이지의 유지관리가 쉽다.

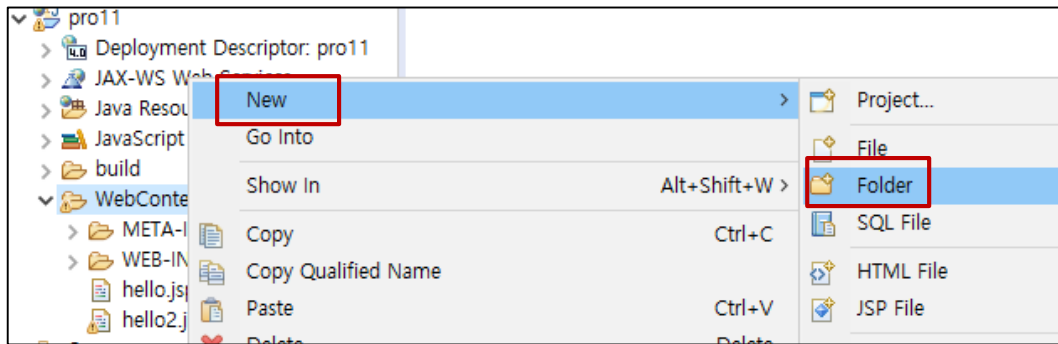
인클루드 디렉티브 태그 형식

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


11.4 디렉티브 태그

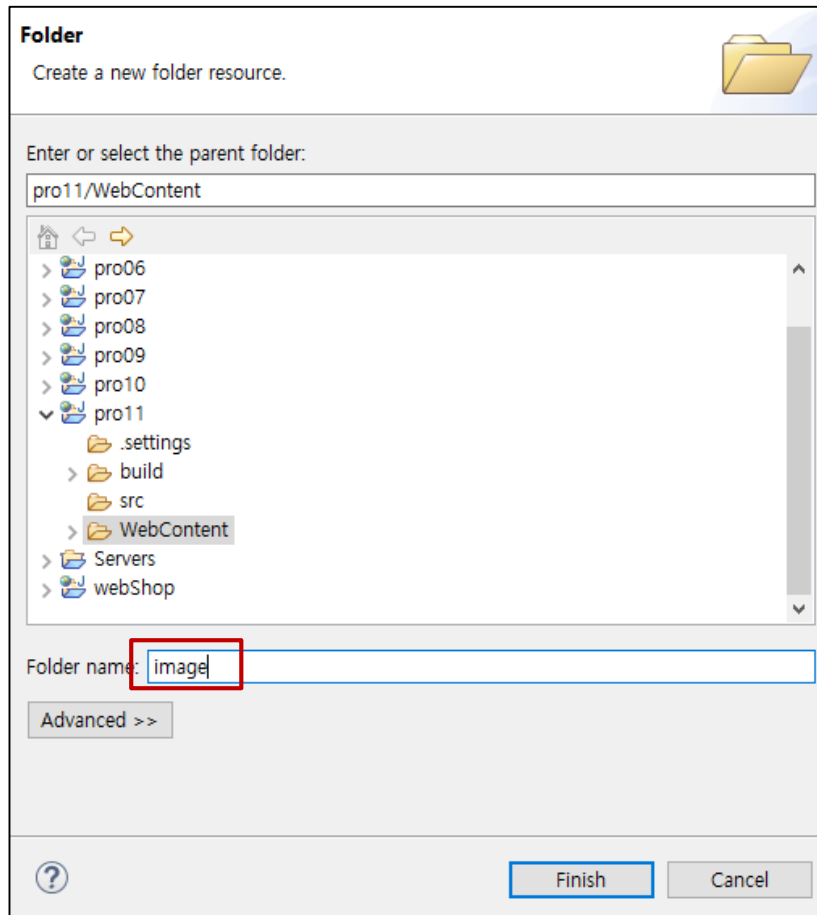
- 11.4.4 인클루드 디렉티브 태그 이용해 이미지 삽입하기

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



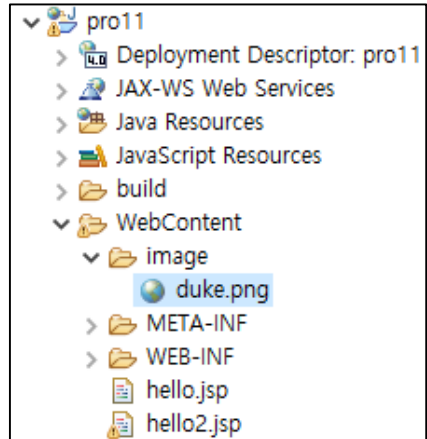
11.4 디렉티브 태그

2. 폴더 이름으로 image를 입력한 후 Finish를 클릭합니다.



11.4 디렉티브 태그

3. 이미지를 복사한 후 image 폴더에 붙여 넣습니다.



11.4 디렉티브 태그

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.

11.4 디렉티브 태그

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.

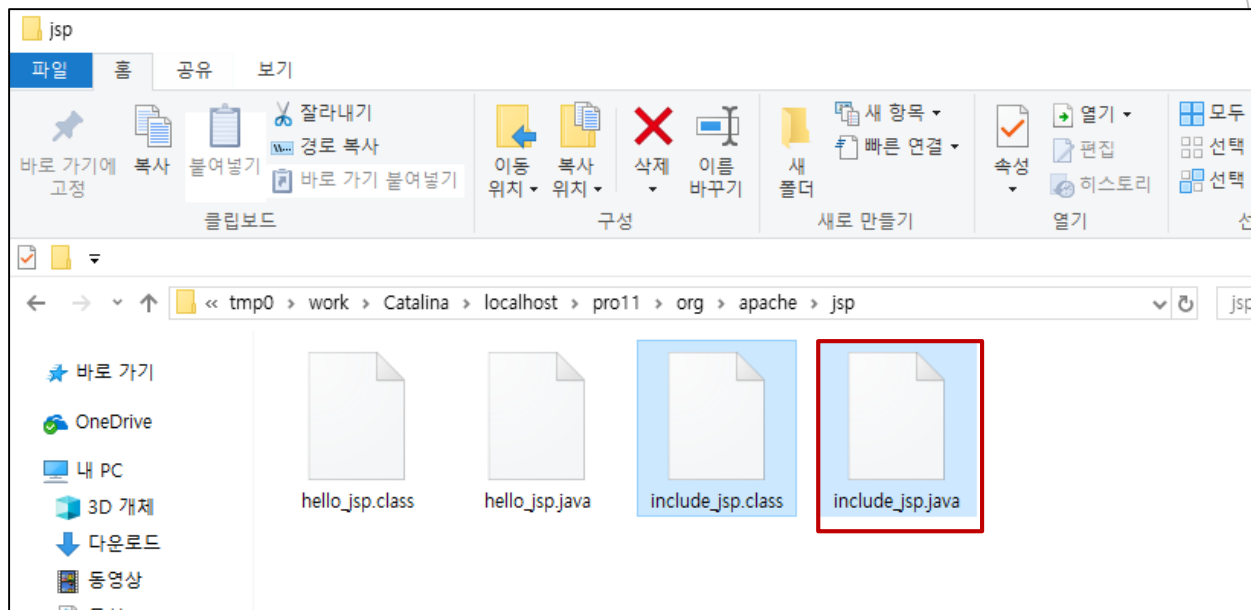
11.4 디렉티브 태그

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



11.4 디렉티브 태그

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



11.4 디렉티브 태그

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

11.4 디렉티브 태그

인클루드 디렉티브 태그 실행 과정



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