



프로젝트로 배우는 자바 웹 프로그래밍

Servlet, JSP, JDBC

황희정 지음

Chapter 03. JSP와의 첫 만남

목차

1. JSP 개요
2. JSP 처리 과정의 이해
3. JSP 프로그램 기술 변천
4. [기본실습] JSP프로그래밍 : Hello World JSP

01. JSP 개요

1. 서블릿(Servlet)과 JSP(Java Server Page)

- 서블릿은 자바를 이용한 서버 프로그래밍 기술이다.
- 초기 웹 프로그래밍 기술인 [CGI\(Common Gateway Interface\)](#)를 대체하기 위해 개발되었으나, 느린 처리 속도, 많은 메모리 요구, 불편한 화면 제어 등의 한계로 PHP, ASP 등 서버 스크립트 언어 등장
- JSP는 PHP와 유사한 형태로 HTML을 중심으로 자바 프로그램과의 유기적인 연결을 지원

■ 서블릿 구현코드 예

```
01 public class HelloWorldServlet extends HttpServlet {
02     public void doGet(HttpServletRequest request,
03                         HttpServletResponse response)
04     throws ServletException, IOException {
05         response.setContentType("text/html; charset=EUC_KR");
06         PrintWriter out = response.getWriter();
07         out.println("<HTML> <HEAD> <TITLE>로그인
                                </TITLE> </HEAD>");
08         out.println("<BODY> <H2>Hello World : 헬로월드 </H2>");
09         out.println("오늘의 날짜와 시간은 : "+new
10             java.util.Date());
11         out.println("</BODY> </HTML>");
12     }
13 }
```

■ JSP 구현코드 예

```
01 <%@ page contentType="text/html; charset=utf-8" %>
02 <HTML>
03 <HEAD> <TITLE>Hello World</TITLE> </HEAD>
04 <BODY> <H2>Hello World : 헬로월드 </H2>
05   오늘의 날짜와 시간은 : <%= new java.util.Date() %>
06 </BODY>
07 </HTML>
```

01. JSP 개요

■ JSP의 특징

- ❶ 자바의 모든 기능을 사용할 수 있어 발전 가능성이 무한하다.
- ❷ 서블릿으로 컴파일된 후 메모리에서 처리되기 때문에 많은 사용자의 접속도 원활하게 처리할 수 있다.
- ❸ JSP 또는 다른 서블릿 간의 데이터를 쉽게 공유 할 수 있다.
- ❹ 빈즈(Beans)라고 하는 자바 컴포넌트를 사용할 수 있다.
- ❺ 커스텀 태그를 만들어 사용할 수 있으며, JSTL(JSP Standard Tag Library)과 같은 태그 라이브러리를 이용할 수 있다.
- ❻ 스트러츠, 스프링 @MVC 등 다양한 프레임워크와 결합하여 개발할 수 있다.

2. JSP 학습에 필요한 기술

- JSP는 웹 프로그래밍 기술로 HTML, 자바스크립트, CSS와 같은 기본 웹 프로그래밍 경험이 요구됨.

[표 3-1] 웹 프로그래밍을 위한 기본 기술 경험의 요구 수준

기술	설명	요구 수준
HTML	클라이언트 기술로서, 웹 프로그램의 기본이 되며 시각적인 부분을 담당한다.	• HTML 문서 구조와 기본 태그 • FORM 관련 태그 • HTML5 기본 구조
자바스크립트	웹 화면과 사용자와의 상호작용 및 동적 웹 페이지를 구현할 때 필요한 기술이다.	• 기본 문법 • 객체와 메서드 • 내장 객체 • 이벤트 핸들링
CSS	웹 화면의 레이아웃과 디자인 요소를 구현할 때 필요한 기술이다.	• 스타일시트 정의 및 셀렉터 이해 • DOM 연동에 의한 동적 스타일 제어

01. JSP 개요

- JSP는 자바언어 기반이며 개발 시 순수 자바 코드가 50% 이상으로 탄탄한 자바 기본기가 요구됨

[표 3-2] 자바 관련 기본 기술 경험의 요구 수준

기술	설명	요구 수준
자바	소스코드를 작성하기 위한 프로그래밍 기본 언어로서, Java SE를 기준으로 한다.	<ul style="list-style-type: none">• 자바 기본• 상속, 오버로딩, 오버라이딩• java.util, java.io 패키지• 예외 핸들링• 객체지향 개념• 인터페이스 구현• 스레드
JDBC	Java DataBase Connectivity의 약자로서, 자바에서 데이터베이스 프로그래밍을 하기 위한 기술이다.	<ul style="list-style-type: none">• JDBC 드라이버 세팅• PreparedStatement• 기초 SQL문• ResultSet• 데이터 핸들링
서블릿	JSP의 기본이 되는 자바 기반의 웹 프로그래밍 핵심 기술이다.	<ul style="list-style-type: none">• 서블릿 구조 이해• request, response 처리• 간단한 서블릿 프로그래밍• GET/POST 처리

01. JSP 개요

- 이외 추가적으로 다음 기술들에 대한 경험이 있다면 고급 웹 프로그래밍 학습에 도움이 됨

[표 3-3] 고급 웹 프로그래밍을 위한 주변 기술 경험의 요구 수준

기술	필요성	요구 수준
데이터베이스	프로그램의 데이터를 처리하려고 할 때 반드시 필요하다.	<ul style="list-style-type: none">다양한 SQL문의 사용데이터베이스 연계 프로그래밍 경험데이터베이스 함수 및 내장 프로시저
XML	eXtensible Markup Language의 약자로서, 확장 가능한 구조적 문서 표현을 제공한다. 많은 프로그램에서 데이터 구조를 XML 기반으로 처리한다.	<ul style="list-style-type: none">XML 스키마 및 DTD 이해XML DOM 개요
모바일 프로그래밍	최근에는 스마트폰을 중심으로 하는 모바일 기반의 개발이 증가하고 있는 추세다.	<ul style="list-style-type: none">안드로이드 혹은 아이폰 앱 개발 경험하이브리드 앱 개발 경험
프레임워크	개발자로 하여금 더욱 좋은 프로그램을 만들 수 있도록 미리 제공되는 틀을 말한다.	<ul style="list-style-type: none">소프트웨어 아키텍처 이해스프링 프레임워크스프링3 @MVC

01. 네트워크, 인터넷, 웹 (p. 31)

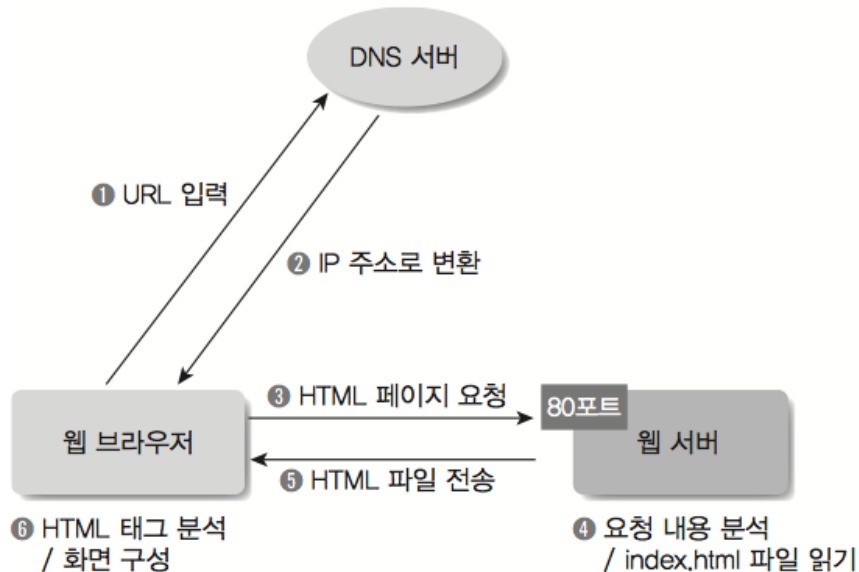
■ 웹 서비스의 동작 과정

■ 웹 서버 소프트웨어

- 서버에서 웹 서비스를 제공하는 소프트웨어
- 아파치(Apache), 마이크로소프트 IIS(Internet Information Server) 가 대표적임.

■ 클라이언트 소프트웨어

- 웹 서비스를 이용하기 위한 클라이언트 소프트웨어 → 웹 브라우저(Web Browser)
- 인터넷 익스플로러(Internet Explorer), 크롬(Chrome), 파이어폭스(Firefox), 애플 사파리(Safari) 등



- 1 웹 브라우저에서 `http://www.xxx.com/index.html`을 입력
- 2 `www.xxx.com` 도메인의 IP 주소를 DNS 서버로부터 받음
- 3 IP 주소의 해당 서버 80번 포트에 접속을 시도
- 4 웹 서버는 요청 내용을 분석하고 요청된 `index.html` 파일을 디스크에서 읽음.
- 5 웹 서버는 파일 내용을 텍스트 그대로 요청한 클라이언트에 전송.
- 6 웹 브라우저는 웹 서버에서 보내는 텍스트 내용 중 HTML 태그를 분석해 적절히 변환하여 화면을 구성.

[그림 1-7] 클라이언트와 서버간 동작 과정

01. 네트워크, 인터넷, 웹 (p. 30)

■ 웹 서버와 HTTP

■ 서버(Server)

- 네트워크에서 서비스를 제공하는 컴퓨터
- 웹 서버, FTP 서버, 파일 서버, 프린트 서버

■ 클라이언트(Client)

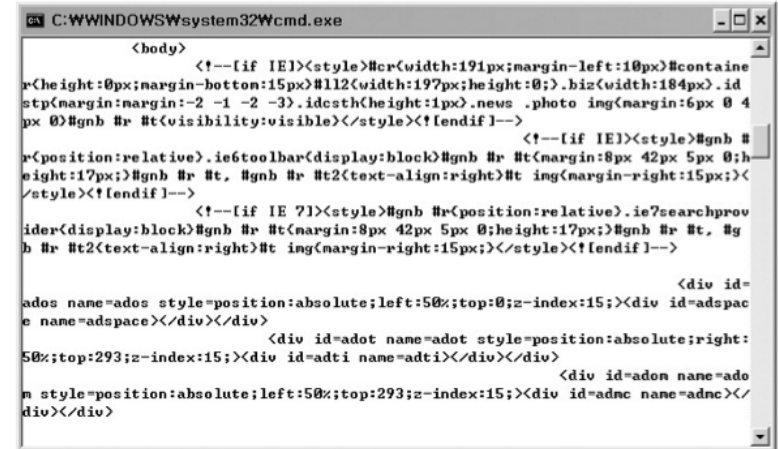
- 서비스를 이용하는 컴퓨터
- PC, 스마트폰, 태블릿 등

■ HTTP(Hyper Text Transfer Protocol)

- 웹 서비스에 사용되는 통신 규격
- 간단한 명령어와 헤더 규격으로 되어 있음

■ HTTP 프로토콜 체험

- 명령창 에서 telnet www.naver.com 80 입력
- GET /index.html HTTP/1.0 입력 후 엔터
- 네이버 서버의 index.html 파일을 보내 달라는 요청임.
- 실제 index.html은 서버 설정에 따라 실제 파일이 아닐 수도 있음.



```
C:\WINDOWS\system32\cmd.exe

<body>
<!--[if IE]><style>#cr{width:191px;margin-left:10px}#containe
r{height:0px;margin-bottom:15px}#l12{width:197px;height:0}#biz{width:184px}.id
st{margin;margin:-2 -1 -2 -3}.idcsth{height:1px}.news .photo img{margin:6px 0 4
px 0}#gnb #r #t{visibility:visible}</style><!--endif-->
<!--[if IE]><style>#gnb #
r{position:relative}.ie6toolbar{display:block}#gnb #r #t{margin:8px 42px 5px 0;h
eight:17px}#gnb #r #t, #gnb #r #t2{text-align:right}#t ing{margin-right:15px}<
/style><!--endif-->
<!--[if IE ?]><style>#gnb #r{position:relative}.ie7searchprov
ider{display:block}#gnb #r #t{margin:8px 42px 5px 0;height:17px}#gnb #r #t, #g
b #r #t2{text-align:right}#t ing{margin-right:15px}</style><!--endif-->

<div id=
ados name=ados style="position:absolute;left:50%;top:0;z-index:15;"><div id=adspace
e name=adspace></div></div>
<div id=adot name=adot style="position:absolute:right:
50%;top:293;z-index:15;"><div id=adti name=adti></div></div>
<div id=adom name=ado
n style="position:absolute;left:50%;top:293;z-index:15;"><div id=admc name=admc></
div></div>
```

[그림 1-5] HTTP GET 명령 결과

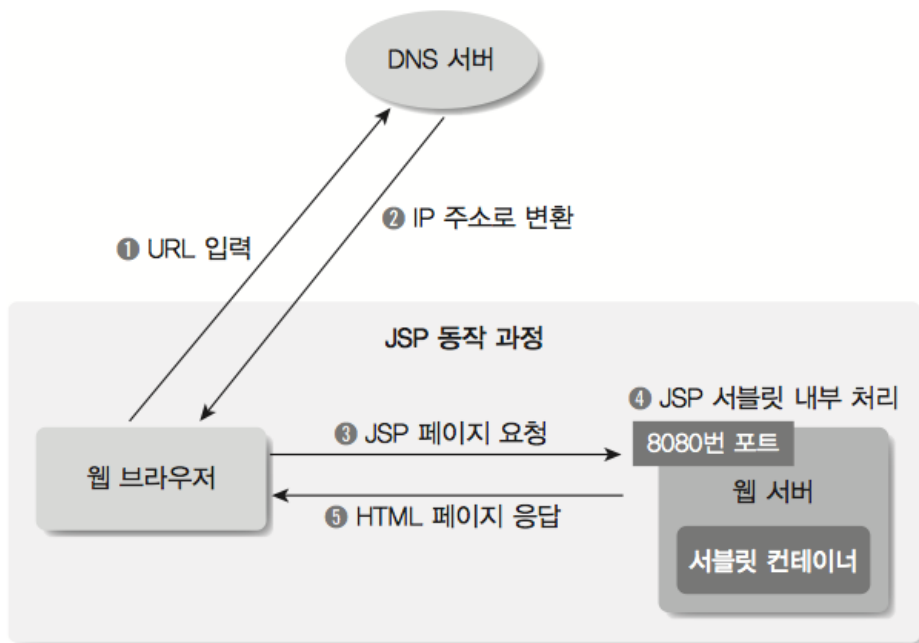


[그림 1-6] 웹 브라우저로 해석된 HTTP GET 명령 결과

02. JSP 처리 과정의 이해

1. JSP 전체 동작 과정

- JSP 는 HTML 과 유사한 처리 과정을 거치나 HTML이 단순 서버 파일을 브라우저로 보내주는 것에 비해 JSP는 서버에서 프로그램이 실행된 결과를 웹 브라우저로 전달하는 차이가 있음.



- 1 웹 브라우저에서 URL을 입력한다.
- 2 DNS 서버로부터 입력한 URL을 변환한 IP 주소를 받는다.
- 3 받은 IP 주소의 웹 서버 8080번 포트에 JSP 페이지를 요청한다.
- 4 웹 서버가 요청 내용을 분석하고 서블릿 컨테이너에 요청을 넘겨 처리한다.
- 5 화면에 보일 내용을 HTML 문서 형태로 웹 브라우저에 전송한다.

[그림 3-1] JSP 전체 동작 과정

02. JSP 처리 과정의 이해

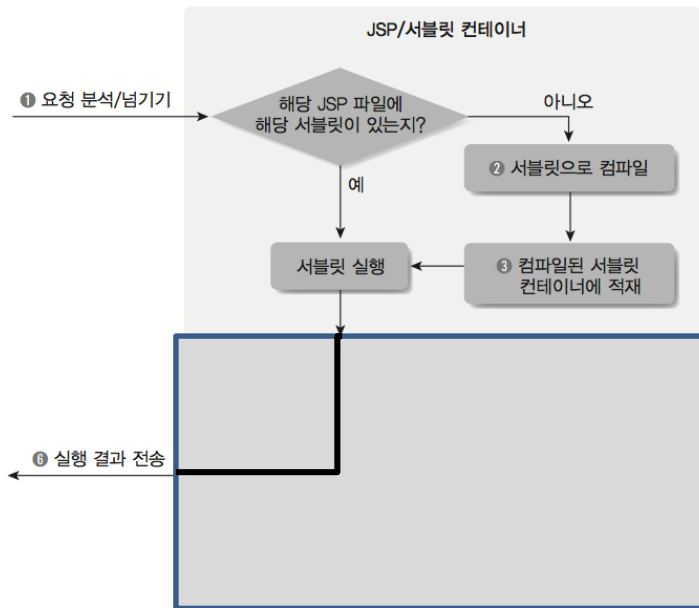
2. 서블릿 컨테이너 내부 과정

▪ JSP와 서블릿 차이

- JSP는 HTML과 같은 일반적인 텍스트 파일에 저장된 프로그램
- 서블릿은 자바 프로그램
- JSP는 서블릿 컨테이너에 의해 서블릿 형태의 자바 코드로 변환되어 클래스로 컴파일 됨

▪ 서블릿 컨테이너

- 서블릿 컨테이너는 서블릿을 실행하고 JSP를 서블릿 코드로 변환하는 기능을 수행함
- 변환된 JSP의 서블릿 클래스를 실행하고 웹 서버의 메모리에 적재하고 사용자 요청에 따라 실행



- 1 웹 서버로부터 JSP에 대한 사용자 요청이 컨테이너로 전달된다.
- 2 요청 JSP에 대한 서블릿이 존재하면 다음 단계로 진행하고, 존재하지 않을 경우 **JSP를 .java 파일로 변환한 다음 .class 파일로 컴파일한다.**
- 3 컴파일된 서블릿 클래스를 컨테이너의 메모리에 적재하고 실행한다.
- 4~5 데이터베이스 처리 혹은 별도의 기능을 위한 클래스 호출 등이 있다면 실행하고 결과를 취합해 HTML 형태로 구성한다.
- 6 HTML 형태의 결과를 웹 서버를 경유해 사용자 브라우저에 전달한다.

[그림 3-2] JSP 서블릿 컴파일과 처리 과정

02. JSP 처리 과정의 이해

■ JSP에 관해서 이것만은 알고 있자.

- ❶ JSP는 일반 텍스트 파일로 되어 있다(텍스트 파일은 컴퓨터가 이해할 수 없다. 즉 실행 가능한 프로그램이 아니며 특정 동작을 할 수 없다).
- ❷ JSP는 HTML 코드와 JSP 태그, 그리고 자바 코드가 섞여 있다.
- ❸ 사용자가 요청할 경우 JSP는 컨테이너(e.g., 톰캣)에 의해 서블릿 형태의 .java 소스로 변환되고 컴파일된다.
- ❹ 컴파일된 .class는 컴퓨터에서 실행할 수 있는 형태로 특정한 기능을 수행할 수 있게 된다. 이후 소스 변경 전까지 해당 파일은 메모리에 상주하면서 다시 컴파일 되지 않고 서비스된다.

* 컴파일된 클래스 파일을 실행시키면 결과물로 HTML 문서가 만들어진다.

01. JSP 개발환경 개요 (p. 52)

- JSP 개발환경을 구축하려면 여러 개발 툴을 상호 연동하여 설치해야 한다.
- JSP는 자바로 구현되므로 자바 개발환경이 필요하다.
- 또한 작성한 코드를 웹 서버에서 실행하려면 JSP 운영환경이 필요하고, 자바를 이용한 편리한 개발을 위해 통합 개발환경도 필요하다.
- 이 책에서 구축하는 개발환경은 다음과 같다.

[표 2-1] JSP 개발환경

항목	필요 프로그램
자바 개발환경	JDK
JSP 운영환경(서블릿 컨테이너)	아파치 톰캣
통합 개발환경	이클립스

- 개발환경을 구축할 때는 안정적인 하드웨어와 검증된 운영체제를 사용하는 것이 좋다. 문제가 생기면 개발 중인 중요한 소스를 날릴 수도 있고, 개발이 지연될 경우 전체 프로젝트에 막대한 지장을 초래.
- 이 책에서는 기본적인 컴퓨터 운영체제로 윈도우 7을 사용한다. 그러나 [표 2-1]에서 소개한 자바 개발환경은 리눅스, 매킨토시 등 다른 운영체제에서도 동일한 버전을 설치할 수 있으므로 해당 운영체제에 익숙하다면 굳이 윈도우를 사용할 필요는 없다.

First JSP Program: Hello World JSP

■ HelloWorld.jsp (교재 p. 101)

- **page 지시어** : 모든 jsp 파일에 기술되어야 하는 요소로 현재 jsp 문서와 관련된 처리 정보를 기술함.

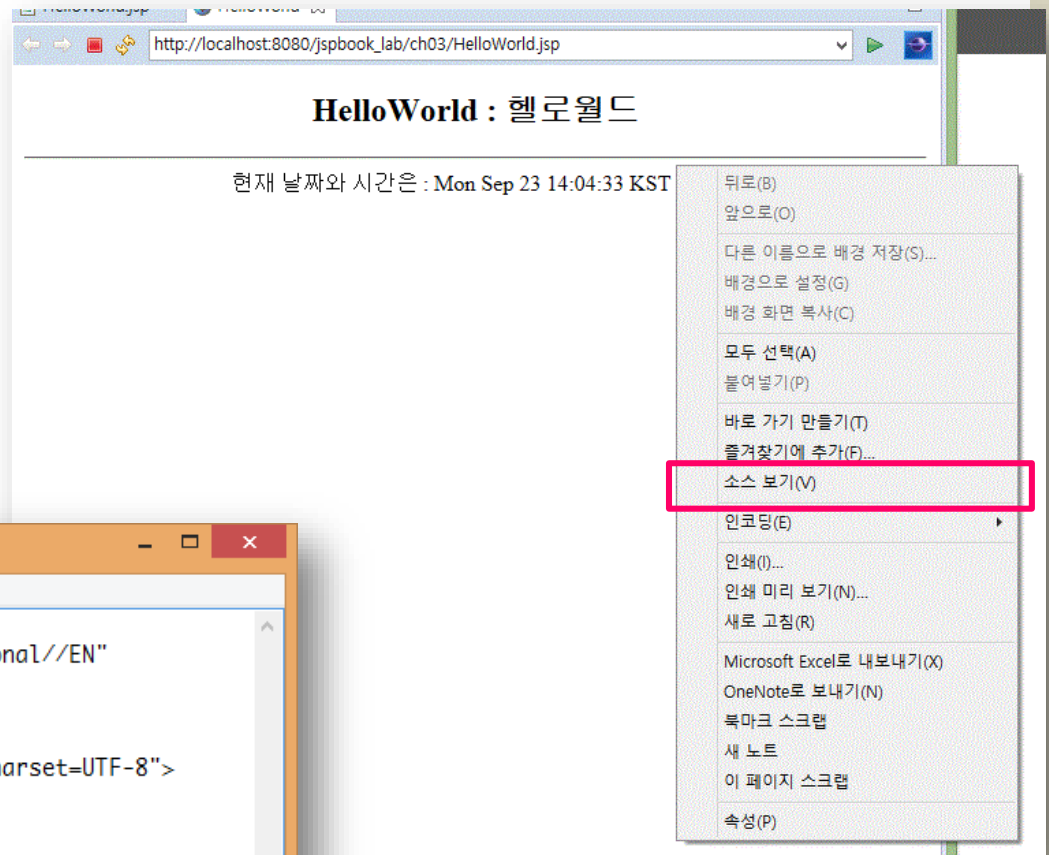
```
01 <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
```

- **HTML 선언부 및 기본 태그**

```
02 <!DOCTYPE html PUBLIC "-//W3c//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
03 <HTML>
04 <HEAD>
05 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
06 <TITLE>HelloWorld</TITLE>
07 </HEAD>
08 <BODY>
09 <center>
10 <H2> HelloWorld : 헬로월드 </H2>
11 <HR>
12 현재 날짜와 시간은 : <%= new java.util.Date() %>
13 </center>
14 </BODY>
15 </HTML>
```




HelloWorld.jsp (실행 결과)





JSP 표현식

- 교재 p. 188
- 현재 날짜와 시간은 : `<%= new java.util.Date() %>`

- 
- `<%= %>` 는 JSP 문법에서 표현식이라고 함
 - 현재 날짜와 시간을 HTML에 출력하는 코드



JSP 동작원리

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<HTML>
<HEAD><TITLE>HelloWorld</TITLE></HEAD>
<BODY><H2>Hello World : 헬로 월드</H2>
<HR>
현재 날짜와 시간은 : <%= new java.util.Date() %>
</BODY>
</HTML>
```

```
01 public class HelloWorldServlet extends HttpServlet {
02     public void doGet(HttpServletRequest request,
03                         HttpServletResponse response)
04     throws ServletException, IOException {
05         response.setContentType("text/html; charset=EUC_KR");
06         PrintWriter out = response.getWriter();
07         out.println("<HTML><HEAD><TITLE>HelloWorld
                                </TITLE></HEAD>");
08         out.println("<BODY><H2>Hello World : 헬로 월드</H2>");
09         out.println("<HR>");
10         out.println("현재 날짜와 시간은 : " + new java.util.Date());
11         out.println("</BODY></HTML>");
12     }
13 }
```




JSP 동작원리

HelloWorld.jsp

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

<HTML>

<HEAD><TITLE>HelloWorld</TITLE>

</HEAD>

<BODY>

<H2>Hello World : 헬로 월드</H2>

<HR>

현재의 시간과 날짜는 : <%= ... %>

</HTML>
```

실행 결과물 (HTML 파일)

```
<HTML>

<HEAD><TITLE>HelloWorld</TITLE>

</HEAD>

<BODY>

<H2>Hello World : 헬로 월드</H2>

<HR>

현재의 시간과 날짜는 : Tue Sep 18 14:43:51 KST 2018

</BODY>

</HTML>
```



프로젝트로 배우는 자바 웹 프로그래밍

Servlet, JSP, JDBC

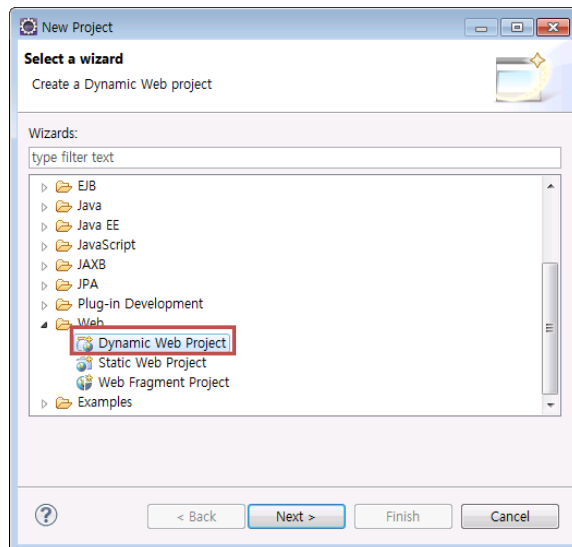
황희정 지음

04. [기본실습]JSP 프로그래밍: Hello World JSP

이클립스를 통한 JSP 개발 과정과 JSP 기본 구조를 간단한 프로그램 개발을 통해 알아본다.

1. 이클립스 프로젝트 생성

- 이클립스를 실행하고 [File] → [New] → [Project]를 선택하면 미리 정의된 특정 유형의 프로젝트 템플릿을 이용하여 프로젝트를 생성할 수 있다.
- 다이나믹 웹 프로젝트 생성하기
 - 트리 메뉴 중 [Web]을 선택하고 [Dynamic Web Project]를 선택하면 JSP 개발을 위한 프로젝트가 생성된다.

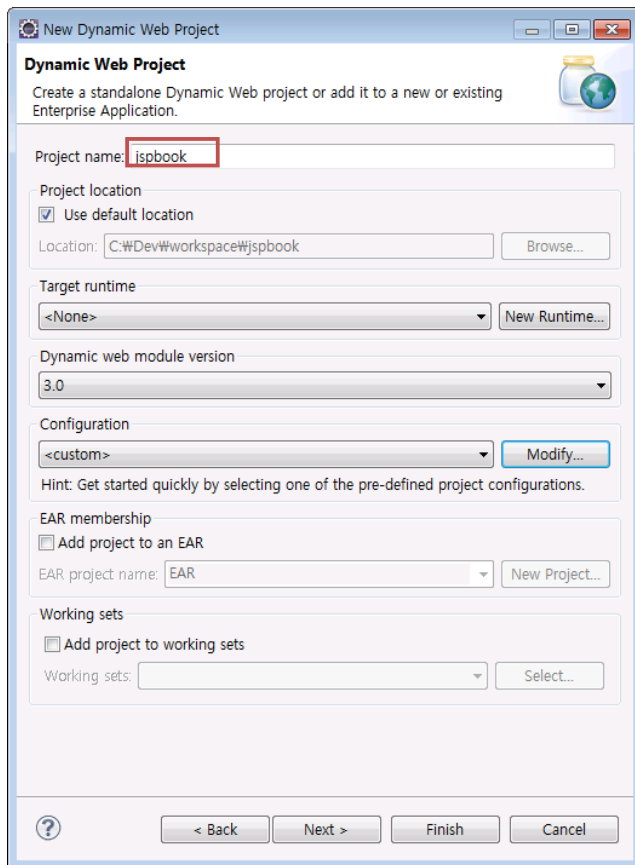


[그림 3-6] 다이나믹 웹 프로젝트 생성

04. [기본실습]JSP 프로그래밍: Hello World JSP

■ 기본 정보 설정하기

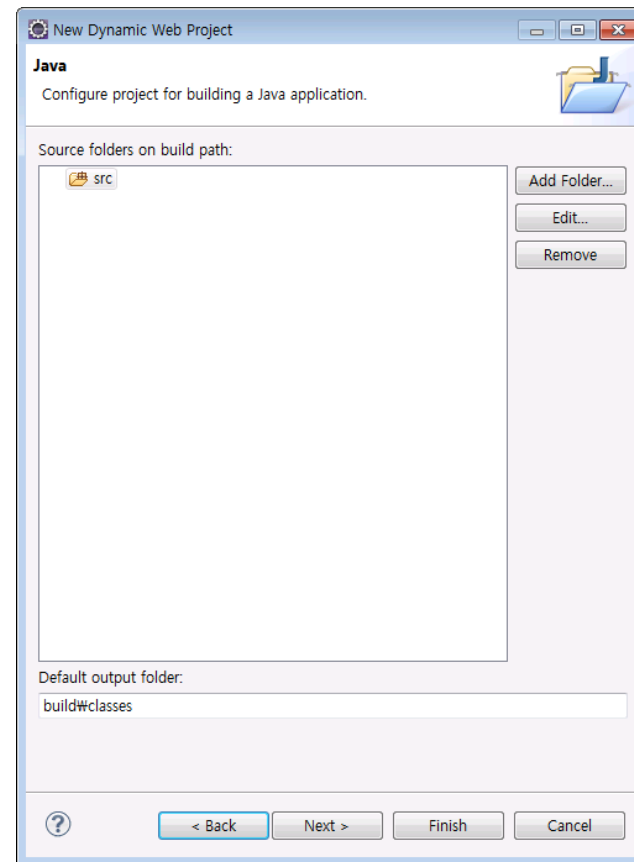
- 프로젝트 이름, 웹 모듈 버전 등 프로젝트 기본 정보를 설정한다.



[그림 3-7] 프로젝트 기본 정보 설정

■ 소스 폴더 설정하기

- 자바 클래스 소스 폴더를 지정한다.

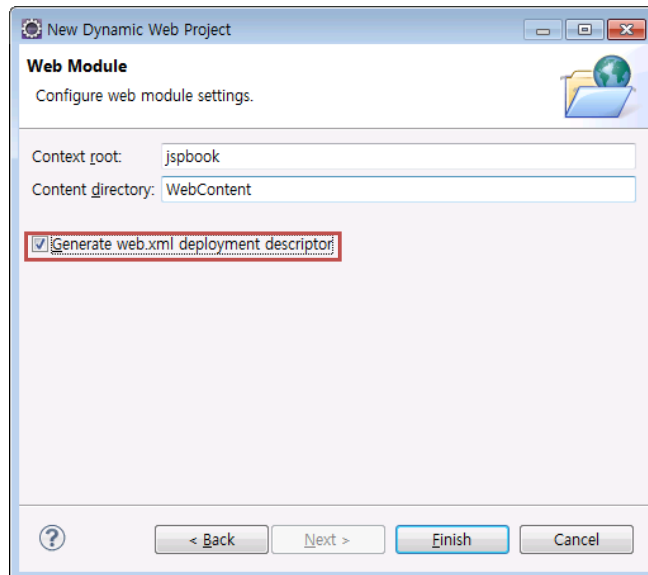


[그림 3-8] 소스 폴더 설정

04. [기본실습]JSP 프로그래밍: Hello World JSP

■ 웹 모듈 설정하기

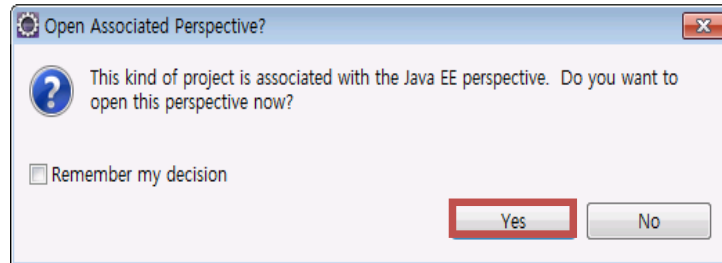
- Context Root
 - 웹 애플리케이션의 메인 접속 경로를 말함.
 - <http://localhost:8080/jspbook> 과 같이 JSP 실행을 위한 기본 URL에 적용됨.
- Content Directory
 - JSP, HTML, 이미지 등 기본 웹 콘텐츠가 위치하는 디렉터리.
 - 이클립스 프로젝트 구조에서는 [WebContent] 폴더가 기본 값으로 사용됨.



[그림 3-9] 웹 모듈 설정

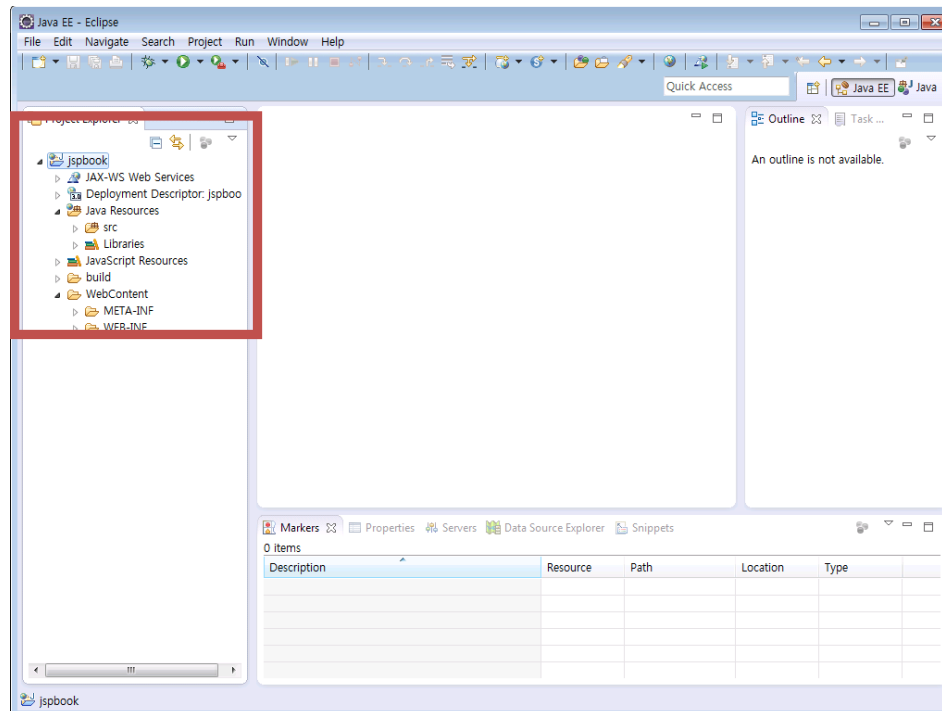
04. [기본실습]JSP 프로그래밍: Hello World JSP

■ 퍼스펙티브 변경하기



[그림 3-10] 퍼스펙티브 변경

■ 생성된 프로젝트 확인

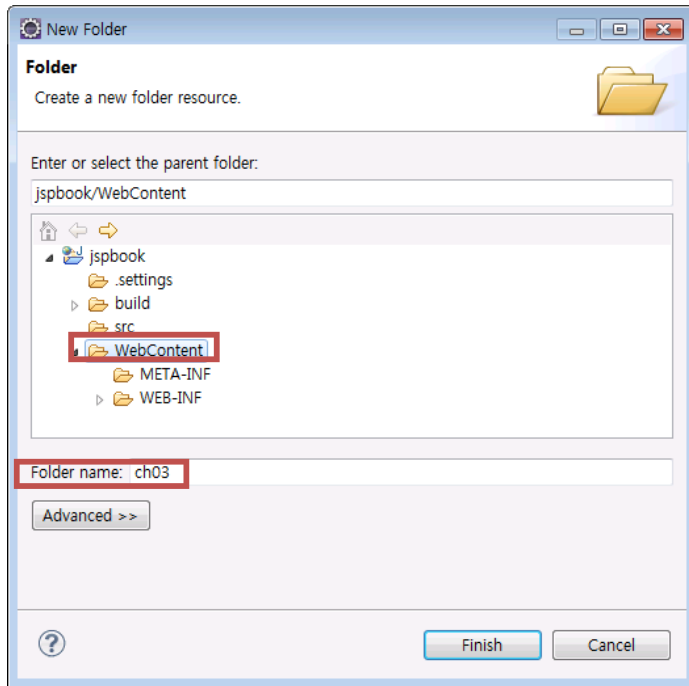


[그림 3-11] 생성된 프로젝트 확인

04. [기본실습]JSP 프로그래밍: Hello World JSP

2. Hello World 프로그램 소스 작성

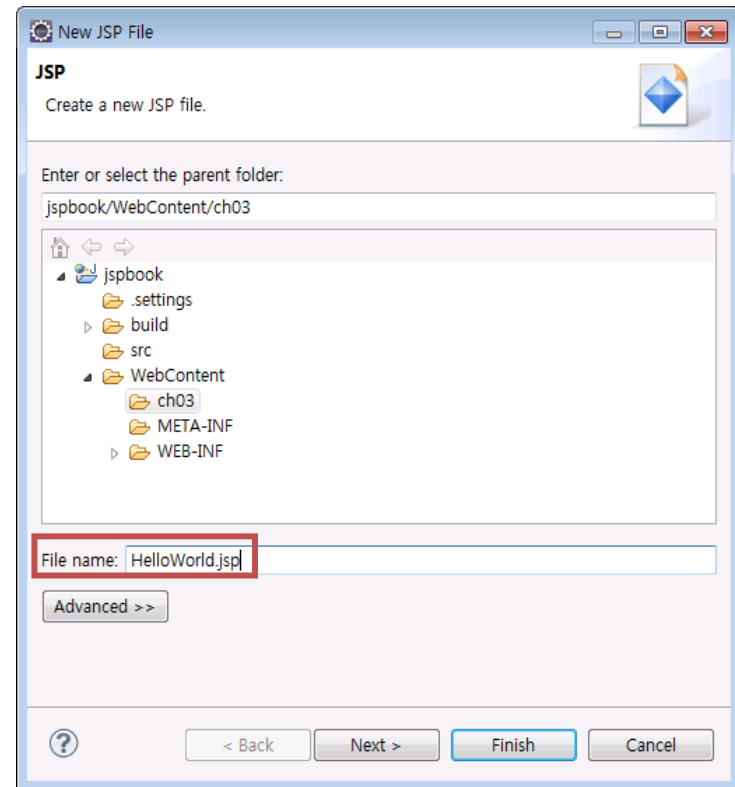
폴더 생성



[그림 3-12] ch03 폴더 생성

JSP 생성

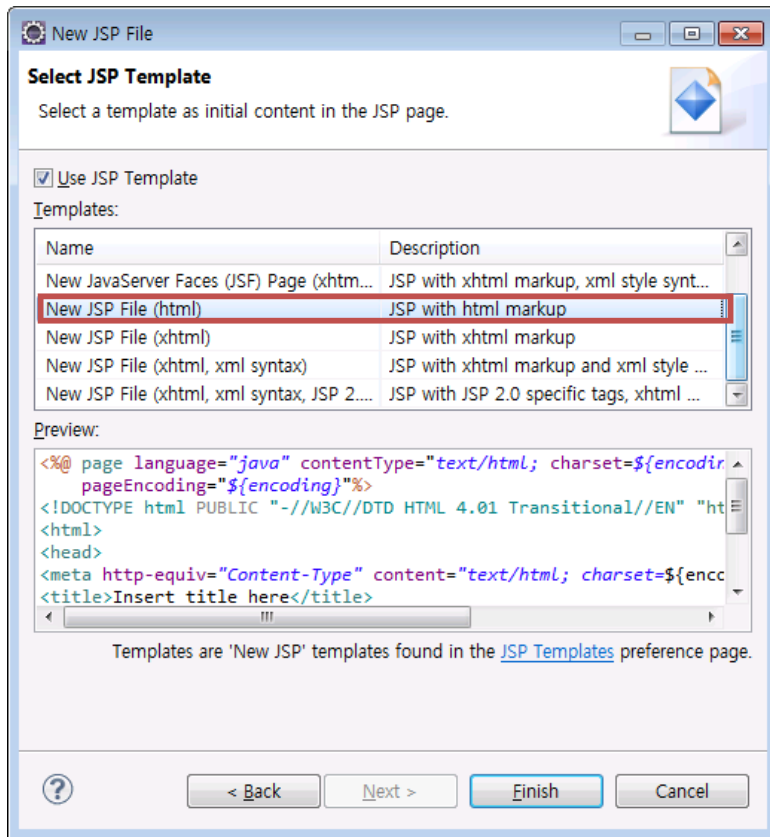
① JSP 파일 이름 지정하기



[그림 3-13] JSP 파일 이름 지정

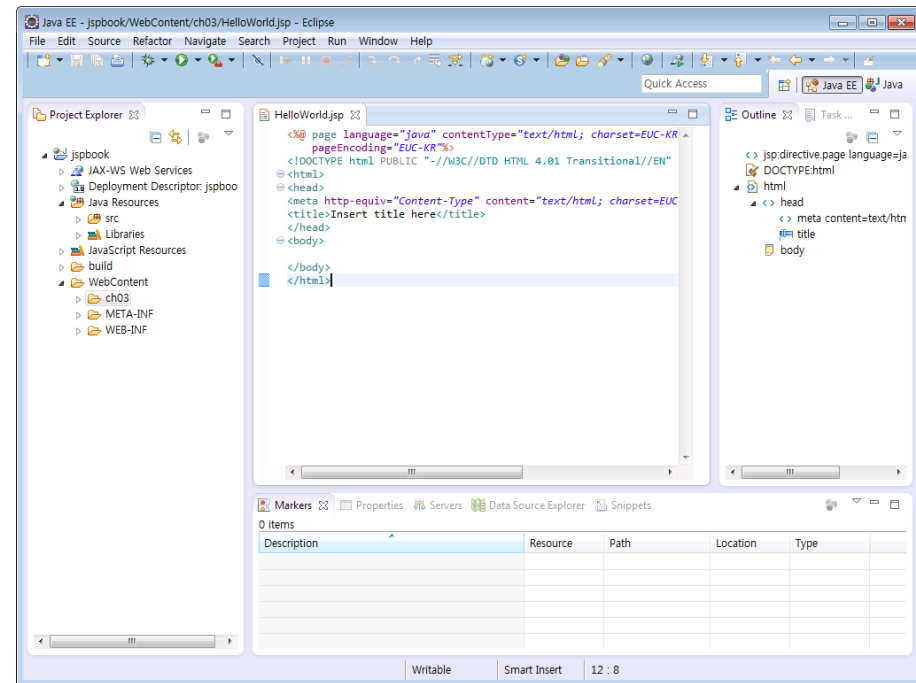
04. [기본실습]JSP 프로그래밍: Hello World JSP

② 템플릿 코드 지정하기



[그림 3-14] 템플릿 선택

③ 생성된 코드 확인하기



[그림 3-15] 생성된 기본 코드

04. [기본실습] JSP 프로그래밍: Hello World JSP

■ 소스코드 작성 : 헬로월드(HelloWorld.jsp)

- **page 지시어** : 모든 jsp 파일에 기술되어야 하는 요소로 현재 jsp 문서와 관련된 처리 정보를 기술함.

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

- **HTML 선언부 및 기본 태그**

```
02 <!DOCTYPE html PUBLIC "-//W3c//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
03 <HTML>
04 <HEAD>
05 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
06 <TITLE>HelloWorld</TITLE>
07 </HEAD>
08 <BODY>
```

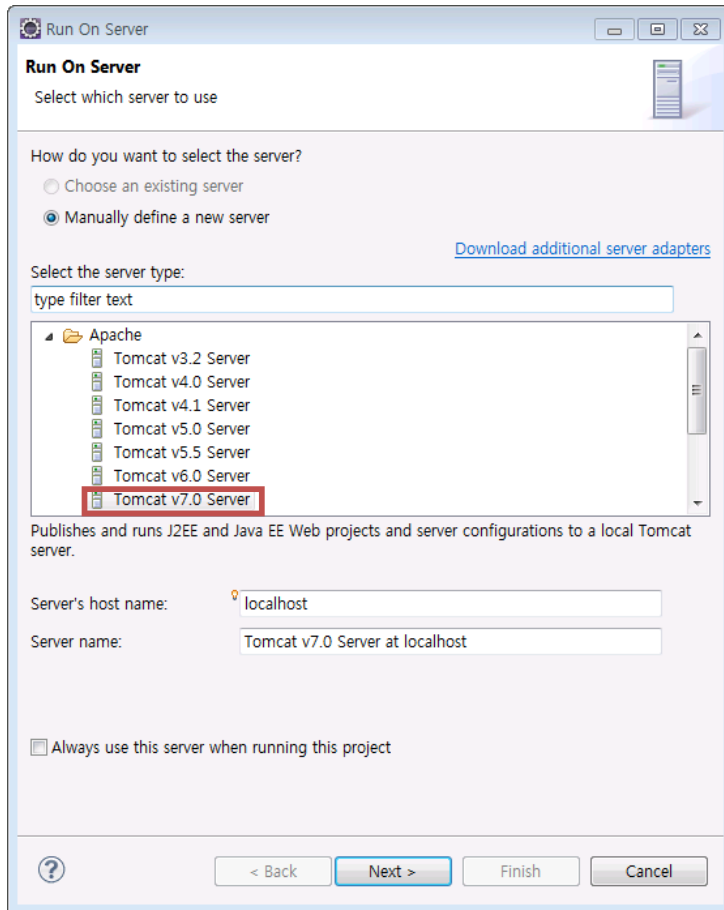
- **JSP 표현식** : 자바 코드를 이용해 간단한 출력을 위해 사용. 여기서는 현재 날짜와 시간 정보가 출력됨.

```
12 현재 날짜와 시간은 : <%=new java.util.Date() %>
```

04. [기본실습]JSP 프로그래밍: Hello World JSP

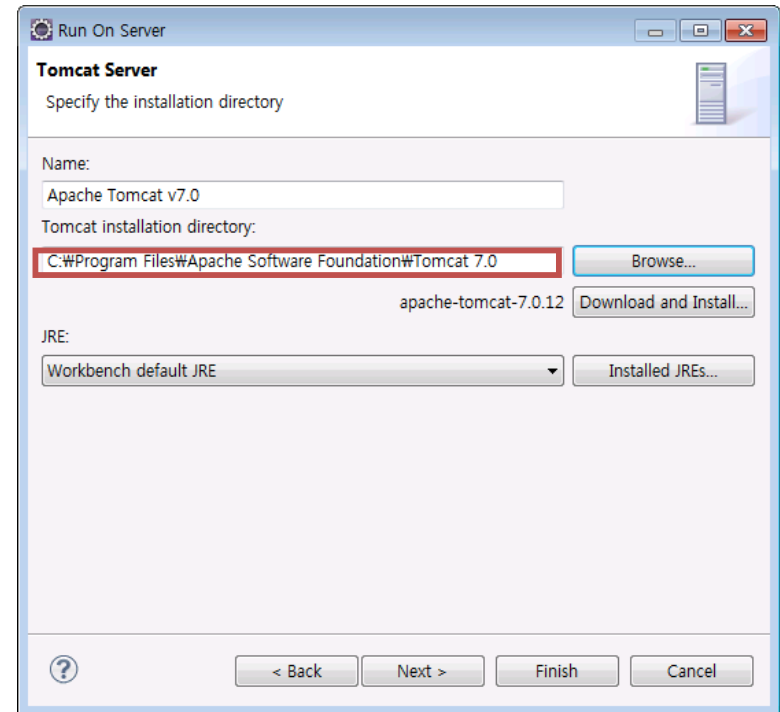
3. 서버 설정 및 실행

■ 서버 설정하기



[그림 3-16] 서버 설정

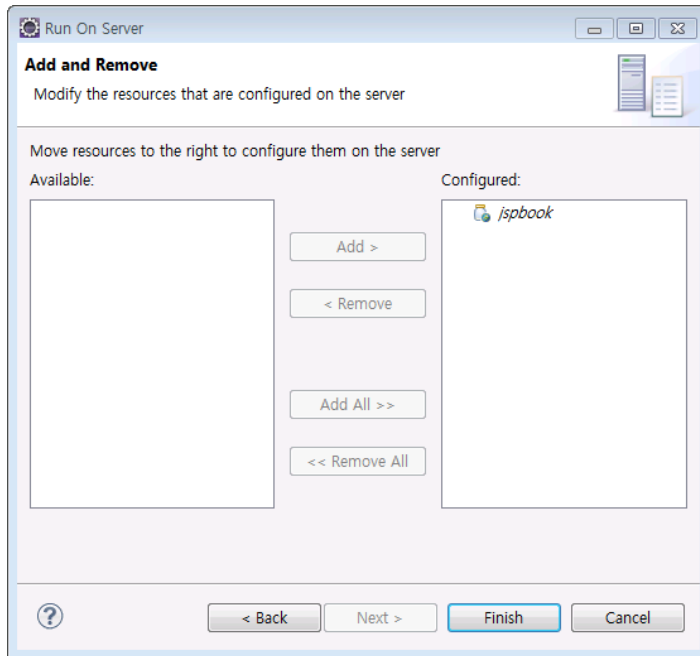
■ 톰캣 폴더 지정하기



[그림 3-17] 톰캣 폴더 지정

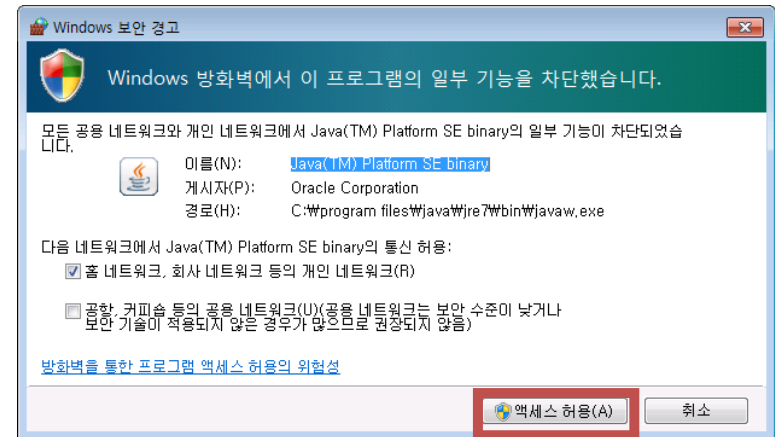
04. [기본실습]JSP 프로그래밍: Hello World JSP

■ 실행할 프로젝트 선택하기



[그림 3-18] 실행 프로젝트 선택

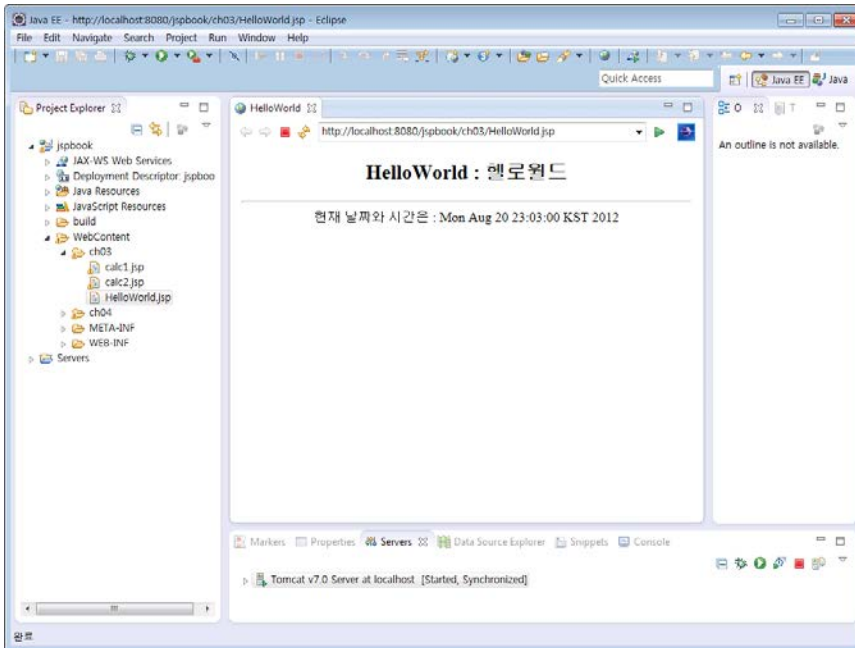
■ 보안 경고 해제하기



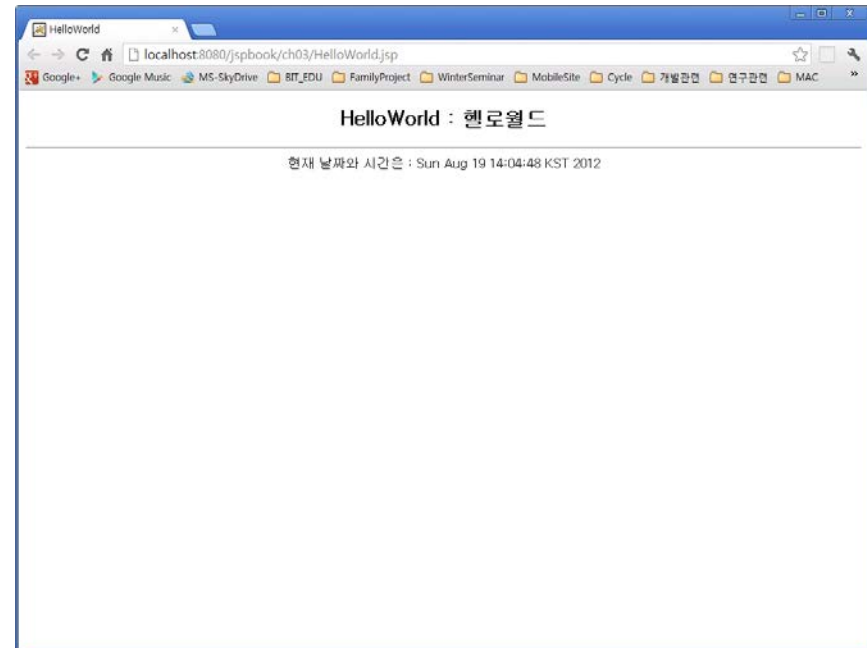
[그림 3-19] 윈도우 보안 경고

04. [기본실습]JSP 프로그래밍: Hello World JSP

■ 실행 결과 확인하기



[그림 3-20] 실행 결과 확인



[그림 3-22] 외부 브라우저(크롬)를 이용한 실행 결과



웹 브라우저에서 실행하기

- <http://localhost:8080/jspbook/Ch03/HelloWorld.jsp>

- 웹 브라우저 화면 > 오른쪽 마우스키 > 소스 보기

- <html> <head>

- <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">

- <title>Insert title here</title>

- </head>

- <body>

- <center>

- <h2> HelloWorld: 헬로월드 </h2>

- <hr>

- 현재 날짜와 시간은: Tue Mar 29 09:51:14 KST 2011

- </center>

- </body>

- </html>

JSP를 실행시킨
순간의 시간이
텍스트로 직접 입력됨



Java Servlet 클래스의 위치

- 교재 107 페이지
- `$workspace\.metadata\.plugins\`
- `org.eclipse.wst.server.core\tmp0\`
- `work\Catalina\localhost\jspbook\`
- `org\apache\jsp\Ch03`

- `package org.apache.jsp.Ch03;`
- `import javax.servlet.*;`
- `import javax.servlet.http.*;`
- `import javax.servlet.jsp.*;`

- `public final class HelloWorld_jsp extends org.apache.jasper.runtime.HttpJspBase`
- `implements org.apache.jasper.runtime.JspSourceDependent {`

- `private static java.util.List _jspx_dependants;`

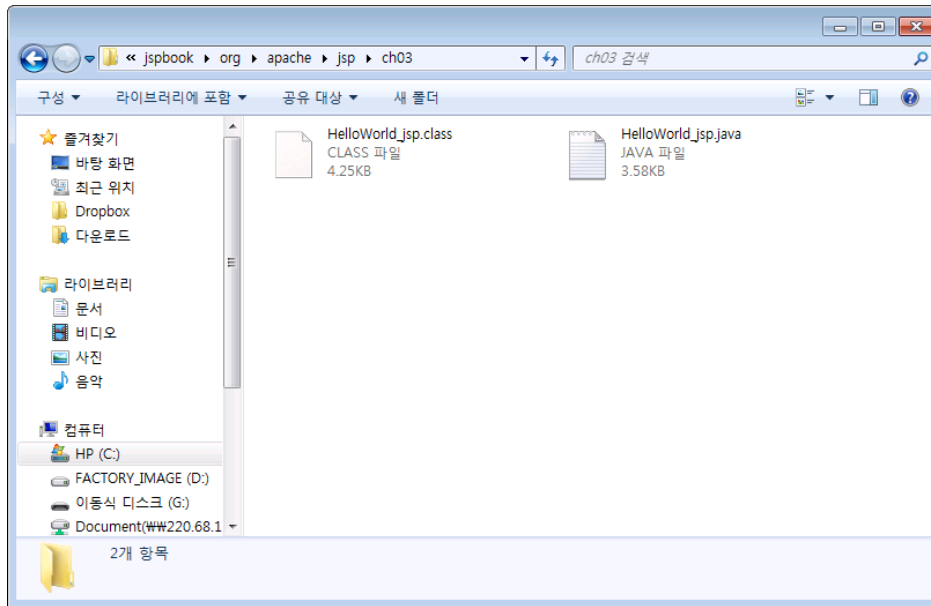
- `public void _jspService(HttpServletRequest request, HttpServletResponse response)`
- `throws java.io.IOException, ServletException {`

- `JspFactory _jspxFactory = null;`
- `...`
- `}`

04. [기본실습]JSP 프로그래밍: Hello World JSP

4. 서블릿으로 변환된 소스 확인

- [c:\wdev\workspace\metadata\plugins\org.eclipse.wst.server.core\wtmp0\work\Catalina\localhost\jspbook\org\apache\jsp\ch0] 폴더에 위치



[그림 3-23] JSP가 변환된 자바 파일이 있는 폴더

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
/*
 * Generated by the Jasper component of Apache Tomcat
 * Version: Apache Tomcat/7.0.68
 * Generated at: 2018-09-18 05:43:50 UTC
 * Note: The last modified time of this file was set to
 *       the last modified time of the source file after
 *       generation to assist with modification tracking.
 */
package org.apache.jsp.ch03;

import javax.servlet.*;
import javax.servlet.http.*;
import javax.servlet.jsp.*;

public final class HelloWorld_jsp extends org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent {

    private static final javax.servlet.jsp.JspFactory _jspxFactory =
        javax.servlet.jsp.JspFactory.getDefaultFactory();

    private static java.util.Map<java.lang.String,java.lang.Long> _jspx_dependants;

    private volatile javax.el.ExpressionFactory _el_expressionfactory;
    private volatile org.apache.tomcat.InstanceManager _jsp_instancemanager;

    public java.util.Map<java.lang.String,java.lang.Long> getDependants() {
        return _jspx_dependants;
    }

    public javax.el.ExpressionFactory _jsp_getExpressionFactory() {
        if (_el_expressionfactory == null) {
            synchronized (this) {
                if (_el_expressionfactory == null) {
                    _el_expressionfactory = _jspxFactory.getJspApplicationContext(getServletConfig().getServletContext()).getExpressionFactory();
                }
            }
        }
        return _el_expressionfactory;
    }

    public org.apache.tomcat.InstanceManager _jsp_getInstanceManager() {
        if (_jsp_instancemanager == null) {
            synchronized (this) {
                if (_jsp_instancemanager == null) {
                    _jsp_instancemanager = org.apache.jasper.runtime.InstanceManagerFactory.getInstanceManager(getServletConfig());
                }
            }
        }
    }
}
```



```
public void _jspInit() {  
}
```

```
public void _jspDestroy() {  
}
```

```
public void _jspService(final javax.servlet.http.HttpServletRequest request, final javax.servlet.http.HttpServletResponse response)  
    throws java.io.IOException, javax.servlet.ServletException {
```

```
    final javax.servlet.jsp.PageContext pageContext;  
    javax.servlet.http.HttpSession session = null;  
    final javax.servlet.ServletContext application;  
    final javax.servlet.ServletConfig config;  
    javax.servlet.jsp.JspWriter out = null;  
    final java.lang.Object page = this;  
    javax.servlet.jsp.JspWriter _jspx_out = null;  
    javax.servlet.jsp.PageContext _jspx_page_context = null;
```

```
    try {  
        response.setContentType("text/html; charset=UTF-8");  
        pageContext = _jspxFactory.getPageContext(this, request, response,  
            null, true, 8192, true);  
        _jspx_page_context = pageContext;  
        application = pageContext.getServletContext();  
        config = pageContext.getServletConfig();  
        session = pageContext.getSession();  
        out = pageContext.getOut();  
        _jspx_out = out;  
  
        out.write("\r\n");  
        out.write("<!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 4.01 Transitional//EN\" \"http://www.w3.org/TR/html4/loose.dtd\">\r\n");  
        out.write("<html>\r\n");  
        out.write("<head>\r\n");  
        out.write("<meta http-equiv=\"Content-Type\" content=\"text/html; charset=UTF-8\">\r\n");  
        out.write("<title>Insert title here</title>\r\n");  
        out.write("</head>\r\n");  
        out.write("<body>\r\n");  
        out.write("<h2>Hello, World!! </h2>\r\n");  
        out.write("오늘의 날짜와 시간은 : ");  
        out.print( new java.util.Date() );  
        out.write("\r\n");  
        out.write("</body>\r\n");  
        out.write("</html>");  
    } catch (java.lang.Throwable t) {  
        if (!(t instanceof javax.servlet.jsp.SkipPageException)){  
            out = _jspx_out;
```



프로젝트로 배우는 자바 웹 프로그래밍

Servlet, JSP, JDBC

황희정 지음