

6. JSP 개발

6.1 JSP 란?

6.2 JSP Elements

6.3 내장 객체

6.4 JSP 지시자 태그

6.5 JSP Exception 처리

6.6 JSP 디버깅

6.1 JSP 란?

6.2 JSP Elements


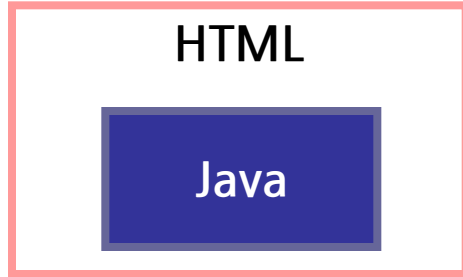
6.3 내장 객체

6.4 JSP 지시자 태그

6.5 JSP Exception 처리

6.6 JSP 디버깅

❖ Servlet 과 JSP 비교

	Servlet	JSP
형태	<p>Java 코드에 HTML 코드가 삽입</p> 	<p>HTML 코드에 Java 코드가 삽입</p> 
예시	<code>out.println("<HTML>");</code>	<code><% for (int i=0; i<10; i++) { %></code>
특징	Business 로직 처리에 적합	Presentation 로직 처리에 적합

※ JSP 기술의 목표는 Servlet의 Business 로직으로부터 Presentation 로직을 분리하는데 있다

❖ Template Page

- html 내에 코드를 포함하는 방법

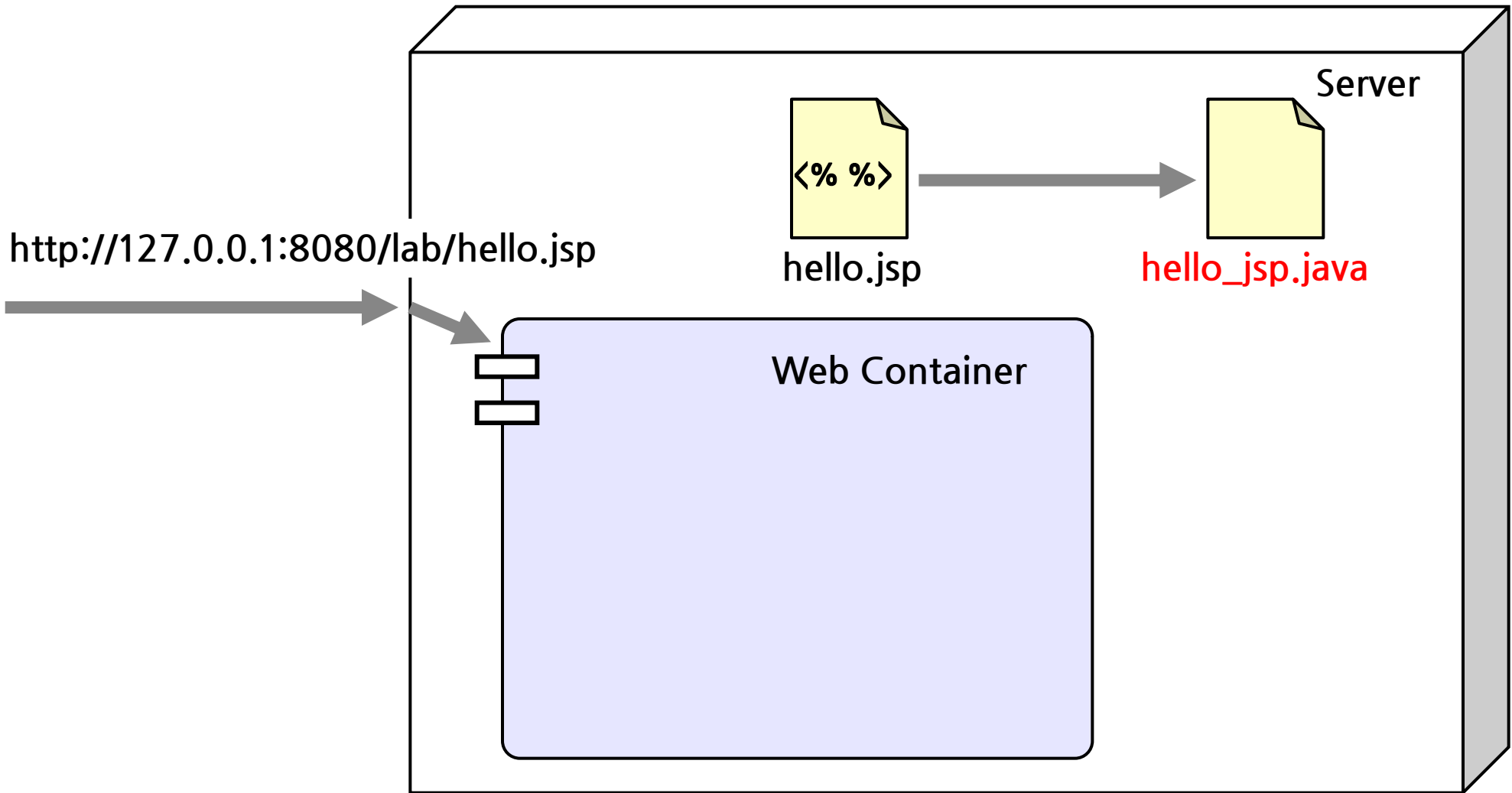
❖ html 내에 코드를 포함하는 기술

- PHP : Apache
- ASP (Active Server Page) : Microsoft
- JSP (Java Server Page) : Sun Microsystems

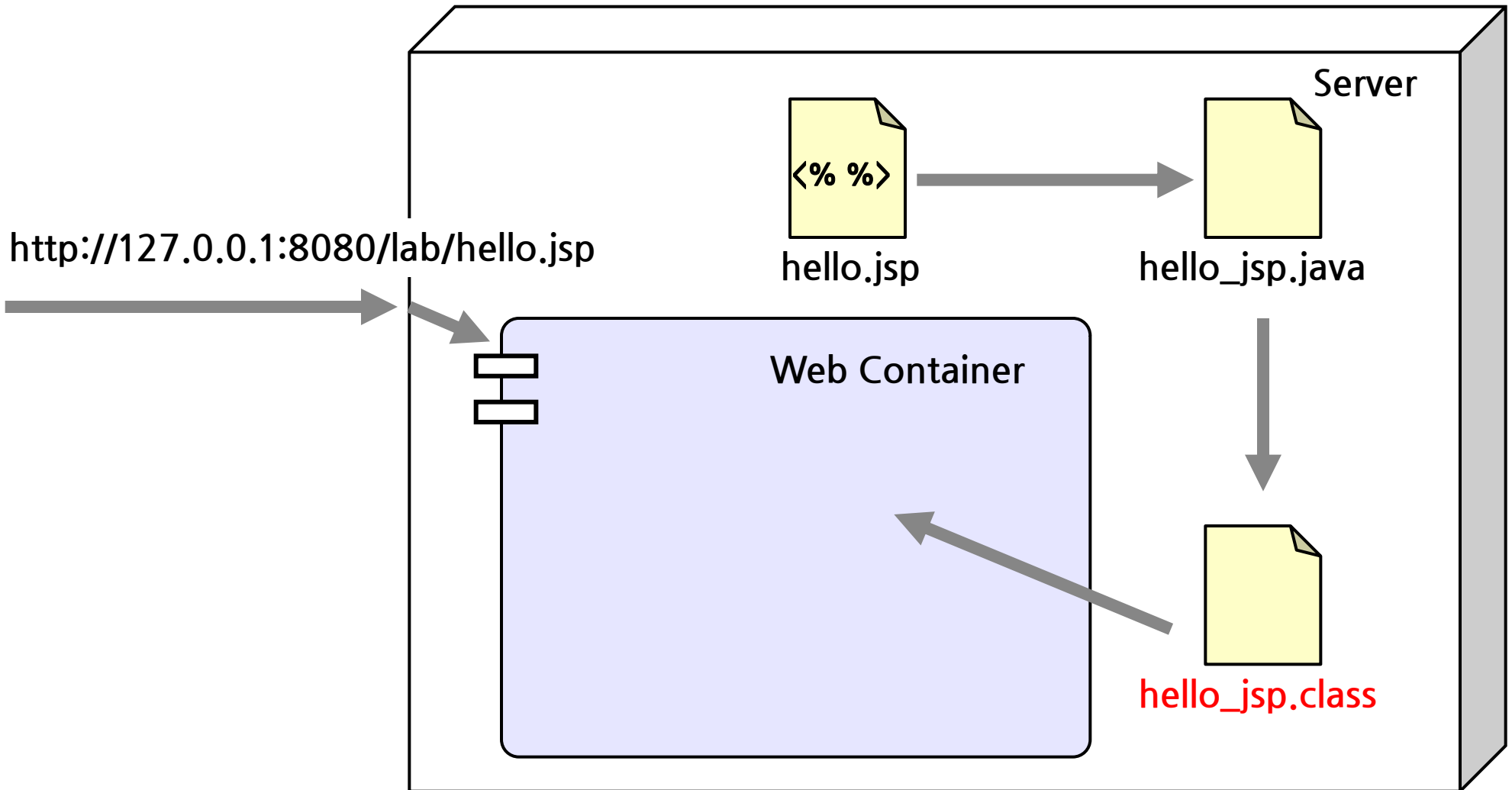
❖ JSP

- Java 코드를 포함하는 html
- JSP page는 Web Container에 의해 Servlet으로 변환 되며, 변환된 Servlet은 해당 JSP page에 대한 요청을 처리한다. → 결국 Servlet

1. JSP에 대한 첫 요청 시, Web Container에 의해 JSP(.jsp)는 Servlet(.java)으로 변환된다.



2. Web Container에 의해 Servlet(.java)은 Class 파일(.class)로 컴파일되고,
Web Container의 JVM에 의해 컴파일 된 클래스 파일이 load된다.



- ❑ JSP 파일이 변경되지 않는다면, .jsp 파일에 대한 컴파일은 다시 일어나지 않는다.
- ❑ JSP 파일이 변경될 때 마다, Web Container는 **translation, compile, load, initialization** 과정을 수행한다.

※ 주의 : 구 버전의 JSP 파일을 overwrite 할 경우 제대로 반영이 되지 않는 경우가 발생할 수 있다.

- ❑ JSP의 배포 환경은 HTML과 동일 → WEB_ROOT 폴더 하단

6.1 JSP 란?

6.2 JSP Elements

6.3 내장 객체

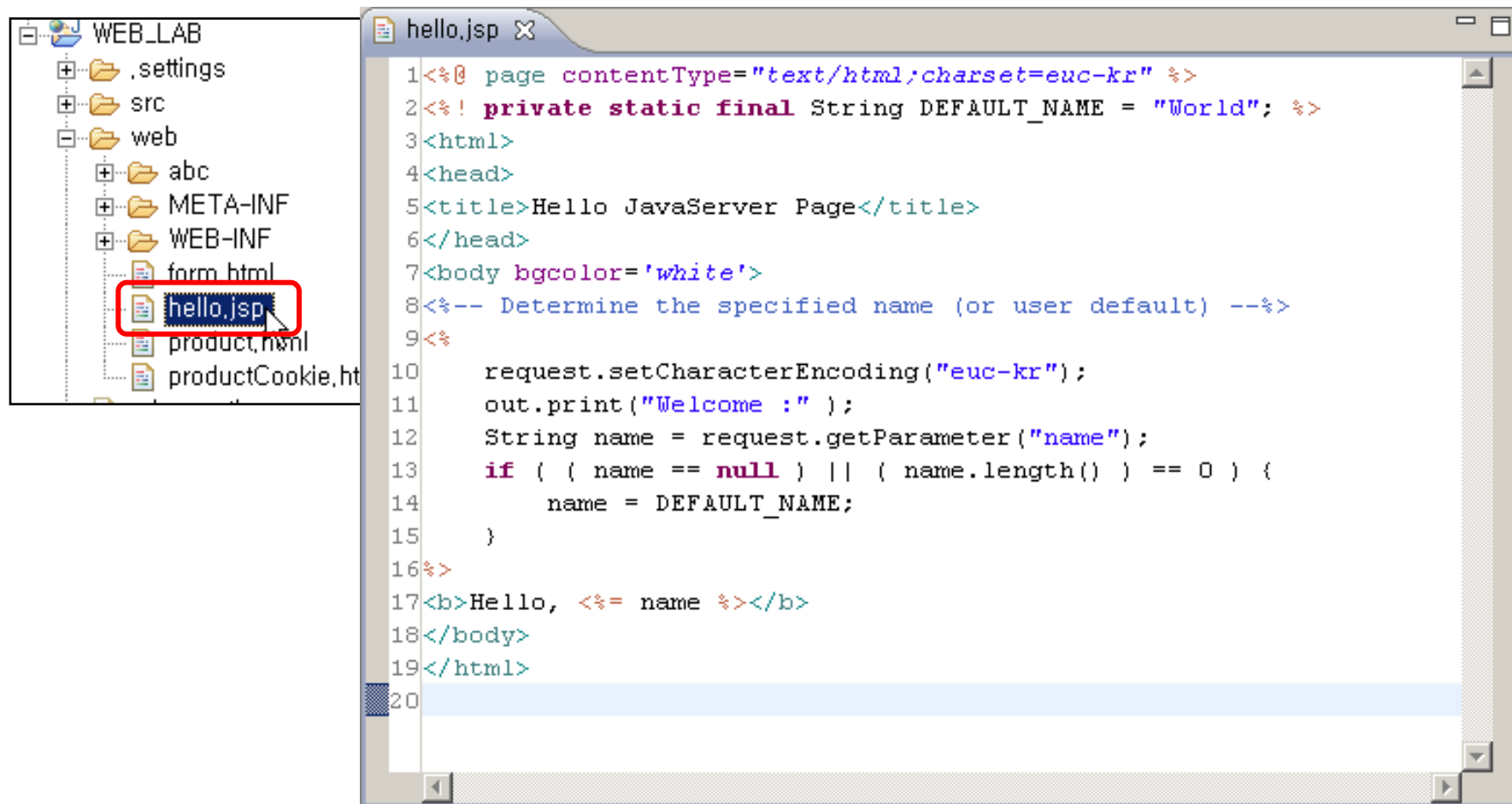
6.4 JSP 지시자 태그

6.5 JSP Exception 처리

6.6 JSP 디버깅

Comments tag	<code><%-- 주식 --%></code>
Directive tag	<code><%@ 지시자 %></code>
Declaration tag	<code><%! 선언문 %></code>
Scriptlet tag	<code><% 코드 %></code>
Expression tag	<code><%= 표현식 %></code>

- 강사제공폴더로부터 hello.jsp 파일을 복사하여 WEB_ROOT 인 WEB_LAB/web 폴더에 복사한다.



hello.jsp

```
<%@ page contentType="text/html;charset=euc-kr" %>
```

Directive tag

```
<%! private static final String DEFAULT_NAME = "World"; %>
```

Declaration tag

```
<html>
```

```
<head>
```

```
<title>Hello JavaServer Page</title>
```

```
</head>
```

```
<body>
```

Comments tag

```
<%-- Determine the specified name (or user default) --%>
```

```
<%
```

```
    request.setCharacterEncoding("euc-kr");
```

```
    out.print("Welcome :" );
```

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

```
    if ( ( name == null ) || ( name.length() ) == 0 ) {
```

```
        name = DEFAULT_NAME;
```

```
    }
```

Scriptlet tag

```
%>
```

```
<b>Hello, <%= name %></b>
```

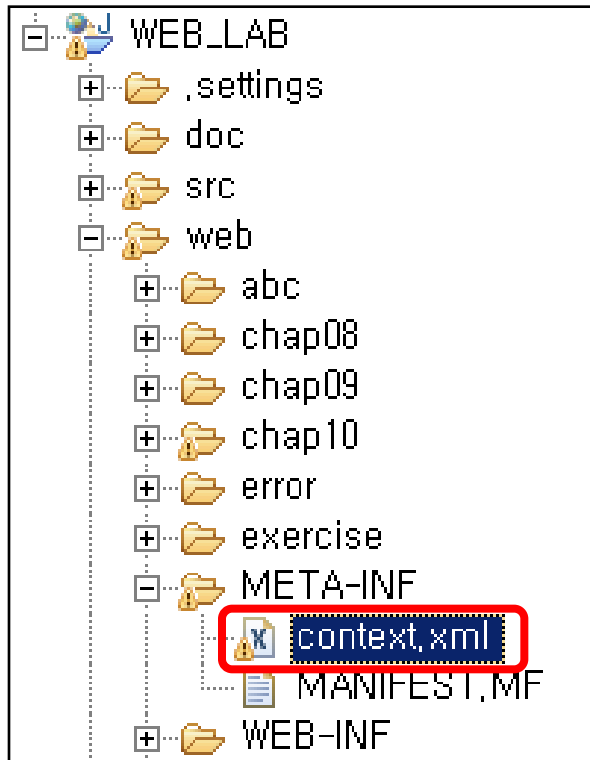
Expression tag

```
</body>
```

```
</html>
```

□ JSP 파일이 서블릿으로 translation 및 compile 될 위치 지정

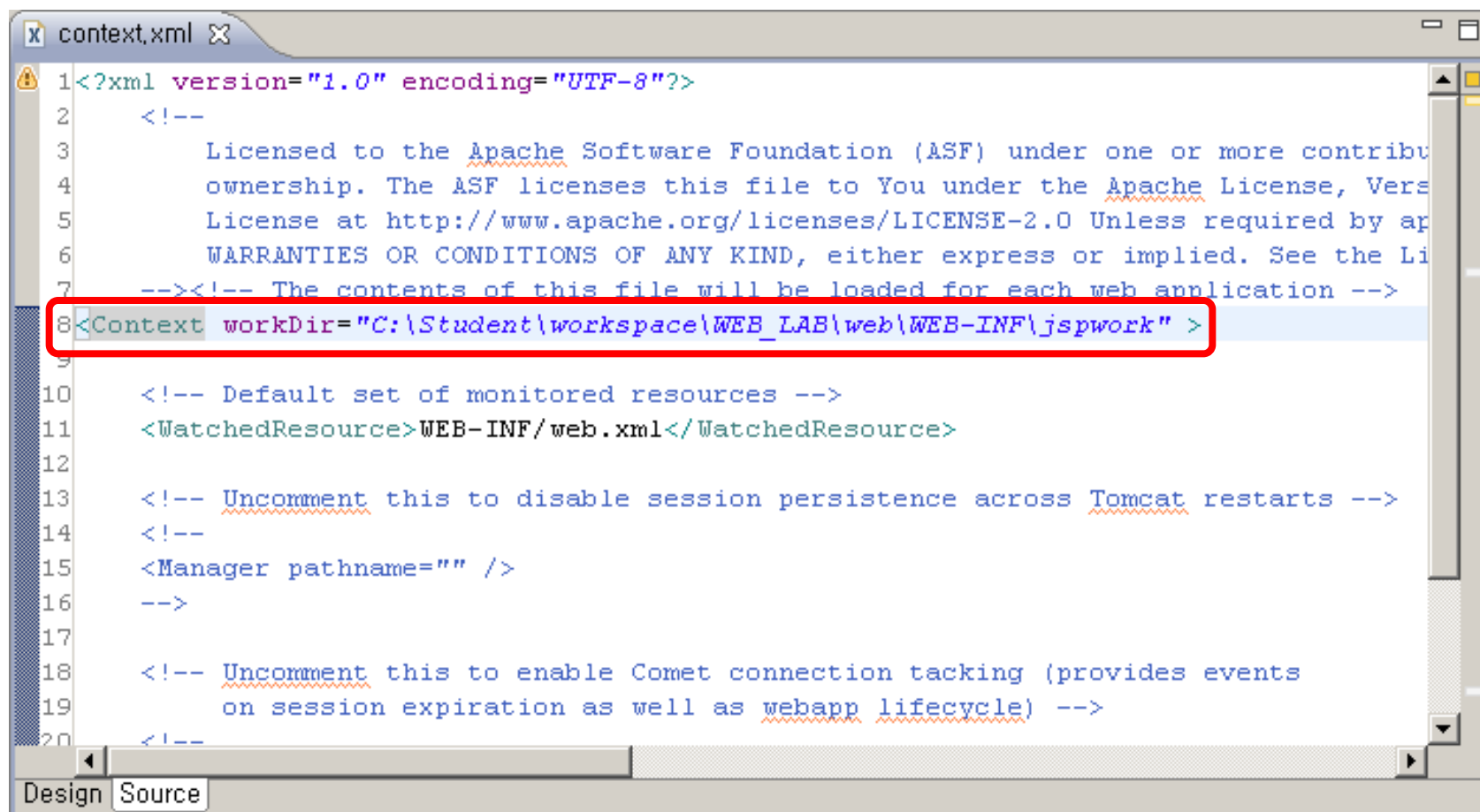
1. WEB-LAB 프로젝트 > web > META-INF > context.xml 추가



- JSP 파일이 서블릿으로 translation 및 compile 될 위치 지정

2. Context 태그에

`workDir="C:\Student\workspace\WEB_LAB\web\WEB-INF\jspwork"` 추가



```
1<?xml version="1.0" encoding="UTF-8"?>
2  <!--
3      Licensed to the Apache Software Foundation (ASF) under one or more contrib
4      ownership. The ASF licenses this file to You under the Apache License, Vers
5      License at http://www.apache.org/licenses/LICENSE-2.0 Unless required by ap
6      WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied. See the Li
7  --><!-- The contents of this file will be loaded for each web application -->
8  <Context workDir="C:\Student\workspace\WEB_LAB\web\WEB-INF\jspwork" >
9
10     <!-- Default set of monitored resources -->
11     <WatchedResource>WEB-INF/web.xml</WatchedResource>
12
13     <!-- Uncomment this to disable session persistence across Tomcat restarts -->
14     <!--
15     <Manager pathname="" />
16     -->
17
18     <!-- Uncomment this to enable Comet connection tacking (provides events
19          on session expiration as well as webapp lifecycle) -->
20     <!--
```

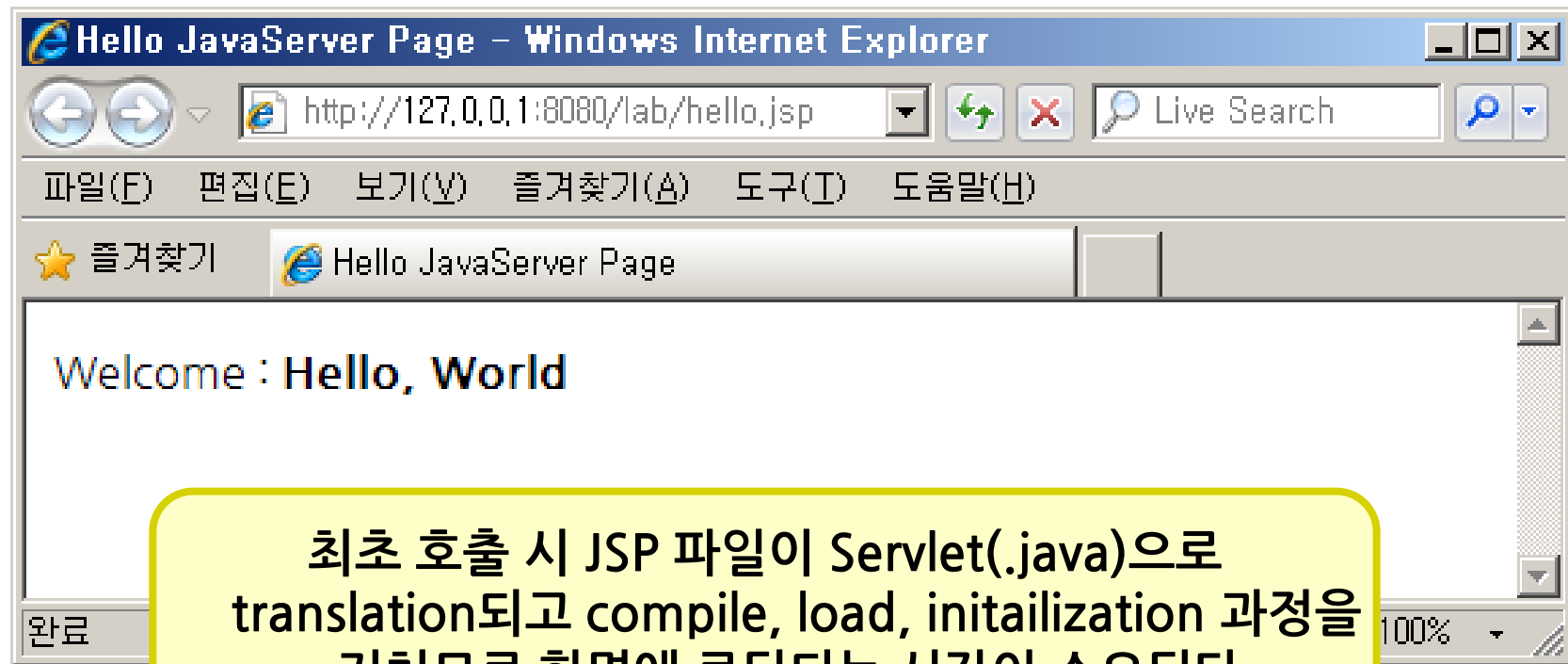
□ context.xml (context 설정 파일)

```
<Context workDir="C:\Student\workspace\WEB_LAB\web\WEB-INF\jspwork" >
```

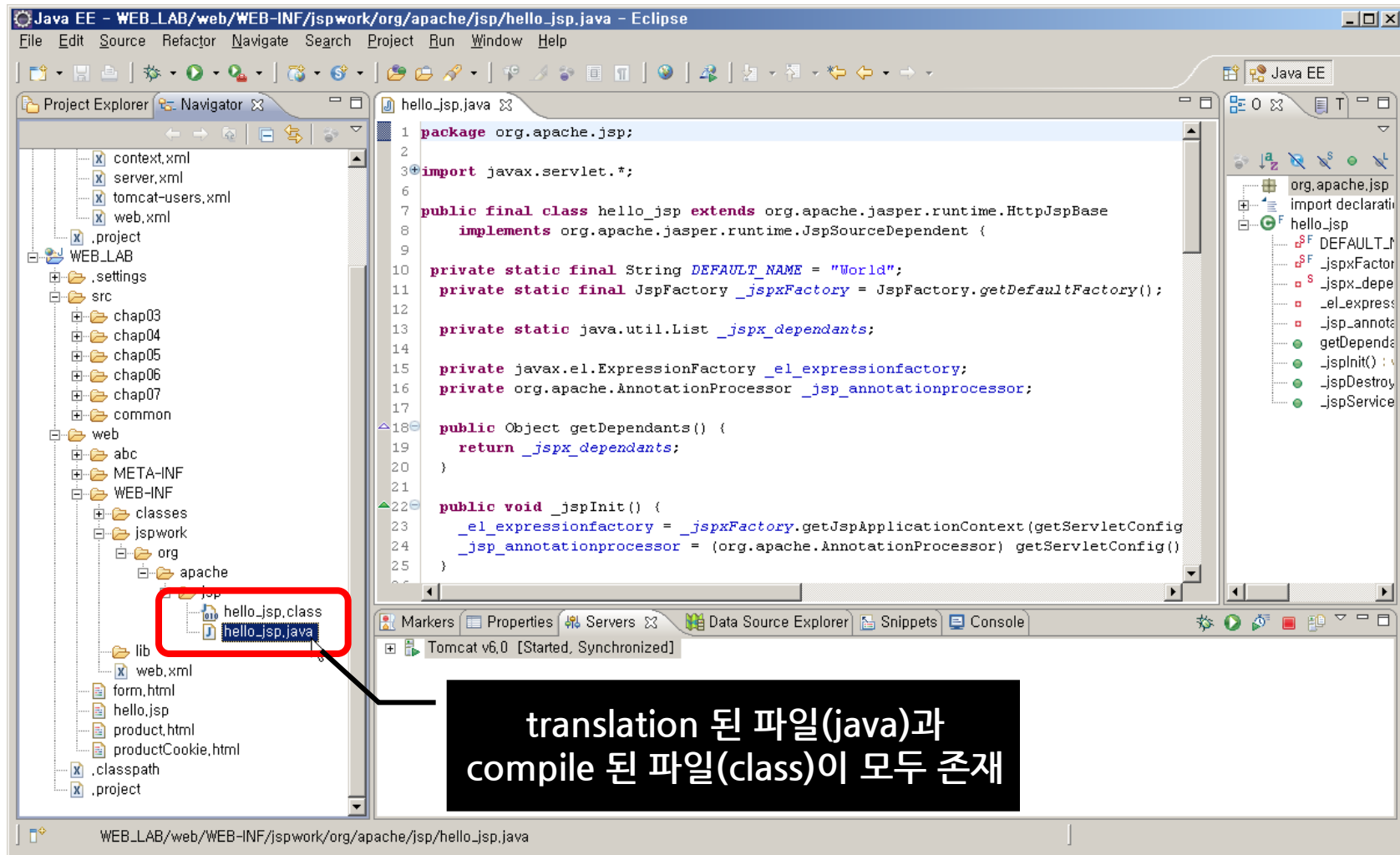
JSP 파일이 서블릿으로 translation 및
compile 될 위치 지정

- Context 태그의 workDir 속성으로 JSP 파일이 서블릿으로 변환되어 위치할 디렉토리를 지정할 수 있다.
- 또한 서블릿이 컴파일되어 생성된 class 파일도 해당 디렉토리에 위치하게 된다.

- 설정 완료 후 Tomcat 과 이클립스를 재가동하고 웹브라우저에서 <http://127.0.0.1:8080/lab/hello.jsp> 를 실행하여 결과를 확인한다.



- ❑ WEB_LAB\web\WEB-INF\jspwork 폴더로 이동하여 translation 된 java 파일을 확인해보자.



- hello_jsp.java 파일과 hello.jsp 를 비교해보자.

```
private static final String DEFAULT_NAME = "World";
```

Declaration tag

```
response.setContentType("text/html; charset=euc-kr");
```

Directive tag

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

```
    request.setCharacterEncoding("euc-kr");  
    out.print("Welcome :");  
    String name = request.getParameter("name");  
    if ( ( name == null ) || ( name.length() ) == 0 ) {  
        name = DEFAULT_NAME;  
    }
```

Scriptlet tag

```
    out.write("\r\n");  
    out.write("<b>Hello ");  
    out.print( name );  
    out.write("</b>\r\n");  
    out.write("</body>\r\n");  
    out.write("</html>\r\n");
```

Expression tag

❖ HTML comments

- HTTP 응답으로 전송되지만, 화면에는 보이지 않는다.

```
<!-- This is HTML Comments -->
```

❖ JSP comments

- JSP 파일 내에서만 존재하고, 변환된 Servlet 코드에는 포함되지 않는다.

```
<%-- This is JSP Comments --%>
```

❖ Java comments

- 변환된 Servlet 코드에는 포함되지만, HTTP 응답으로 전송되지 않는다.

```
<% /* This is Java Comments */  
%>
```

- 전체 JSP page에 영향을 미치는 정보를 기술할 때 쓰인다.

```
<%@ 지시자 [attr="value"] * %>
```

- 지시자

- page, include, taglib 등 3종류가 있다.

```
<%@ page import="java.io.*" %>  
<%@ include file="abc.html" %>  
<%@ taglib tagdir="/WEB-INF/tags/abc" prefix="abc" %>
```

□ Servlet 클래스의 멤버 변수/메소드에 해당하는 코드를 작성할 때 사용된다.

- 멤버 변수 선언

```
<%! public static final String DEFAULT_NAME="World"; %>  
<%! int counter = 0; %>
```

- 멤버 메소드 선언

```
<%! public String getName(HttpServletRequest request) {  
    return request.getParameter("name");  
}  
%>
```

□ `_jspService` 메소드의 로컬 변수와 코드를 작성할 때 사용된다.

- 로컬변수 선언

```
<% int i = 0; %>
```

- 메소드 내용 코드

```
<%  
    if ( i > 10 ) {  
    %>  
    I is a big number.  
    <%  
        } else {  
    %>  
    I is a small number.  
    <%  
        }  
    %>
```

□ out.print() 의 역할을 한다.

▪ 예제

```
<b>Ten is <%= ( 2 * 5 ) %></b>
```

```
The current day and time is: <%= new java.util.Date() %>
```

▪ 주의 : ‘;’ 을 붙이지 않는다.

```
<%= new java.util.Date(); %>
```

Translation



```
out.print( new java.util.Date(); ); → error !!
```

6.1 JSP 란?

6.2 JSP Elements

6.3 내장 객체

6.4 JSP 지시자 태그

6.5 JSP Exception 처리

6.6 JSP 디버깅

hello.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<%! private static final String DEFAULT_NAME = "World"; %>
<html>
<head>
<title>Hello JavaServer Page</title>
</head>
<body>
<%-- Determine the specified name (or user default) --%>
<%
    request.setCharacterEncoding("euc-kr");
    out.print("Welcome :" );
    String name = request.getParameter("name");
    if ( ( name == null ) || ( name.length() ) == 0 ) {
        name = DEFAULT_NAME;
    }
%>
<b>Hello, <%= name %></b>
</body>
</html>
```

JSP에서 기본적으로 제공하는 내장 객체

❖ Scriptlet tag와 Expression tag에서 사용할 수 있도록 암시적으로 선언된 변수

Variable Name	Description
request	HttpServletRequest 객체 참조 변수
response	HttpServletResponse 객체 참조 변수
out	JspWriter 객체 참조 변수
session	HttpSession 객체 참조 변수
application	ServletContext 객체 참조 변수
page	자바 클래스의 this와 동일
exception	발생 하는 Throwable 객체에 대한 참조 변수

6.1 JSP 란?

6.2 JSP Elements

6.3 내장 객체

6.4 JSP 지시자 태그

6.5 JSP Exception 처리

6.6 JSP 디버깅

□ 하나의 JSP 전체에 대한 지시구문을 선언할 때 사용

- 여러 개의 page 구문을 사용할 수 있지만, import 속성을 제외하고는 한 페이지에 한 번씩만 선언할 수 있다.
- page 지시어는 JSP 파일의 어느 위치에 와도 상관 없으나, 가장 첫 부분에 사용하는 것이 좋다.

```
<%@ page import="java.util.Date" %>  
<%@ page contentType="text/html" %>
```

❖ import

- 변환될 서블릿 클래스에 필요한 자바 클래스의 import 문을 정의한다.
- java.lang, javax.servlet, javax.servlet.http, javax.servlet.jsp 는 기본적으로 import 되어 있다.
- 여러 package import시 ‘,’기호를 이용하여 구분한다.

```
<%@ page import="java.io.*, java.util.Date" %>
```

❖ contentType

- MIME 타입과 문자 인코딩을 설정한다.

```
<%@ page contentType="text/html; charset=euc-kr" %>
```

❖ isErrorPage

- 현재 페이지가 JSP 오류 처리용 페이지인지를 정의한다.
- 값은 true 또는 false(default)
- true인 경우, exception 내장 객체를 사용할 수 있다.

```
<%@ page isErrorPage="true" %>
```

❖ errorPage

- 해당 JSP 페이지가 발생시키는 모든 runtime exception을 처리할 다른 JSP페이지를 지정한다.
- 값은 상대적인 URL이다.

```
<%@ page errorPage="/error/errorForm.jsp" %>
```

❖ include 지시자를 사용하면 다른 페이지(JSP, HTML)를 포함할 수 있다.

□ 문법

```
<%@ include file="페이지 경로" %>
```

□ Example

```
<%@ include file="footer.html" %>
```

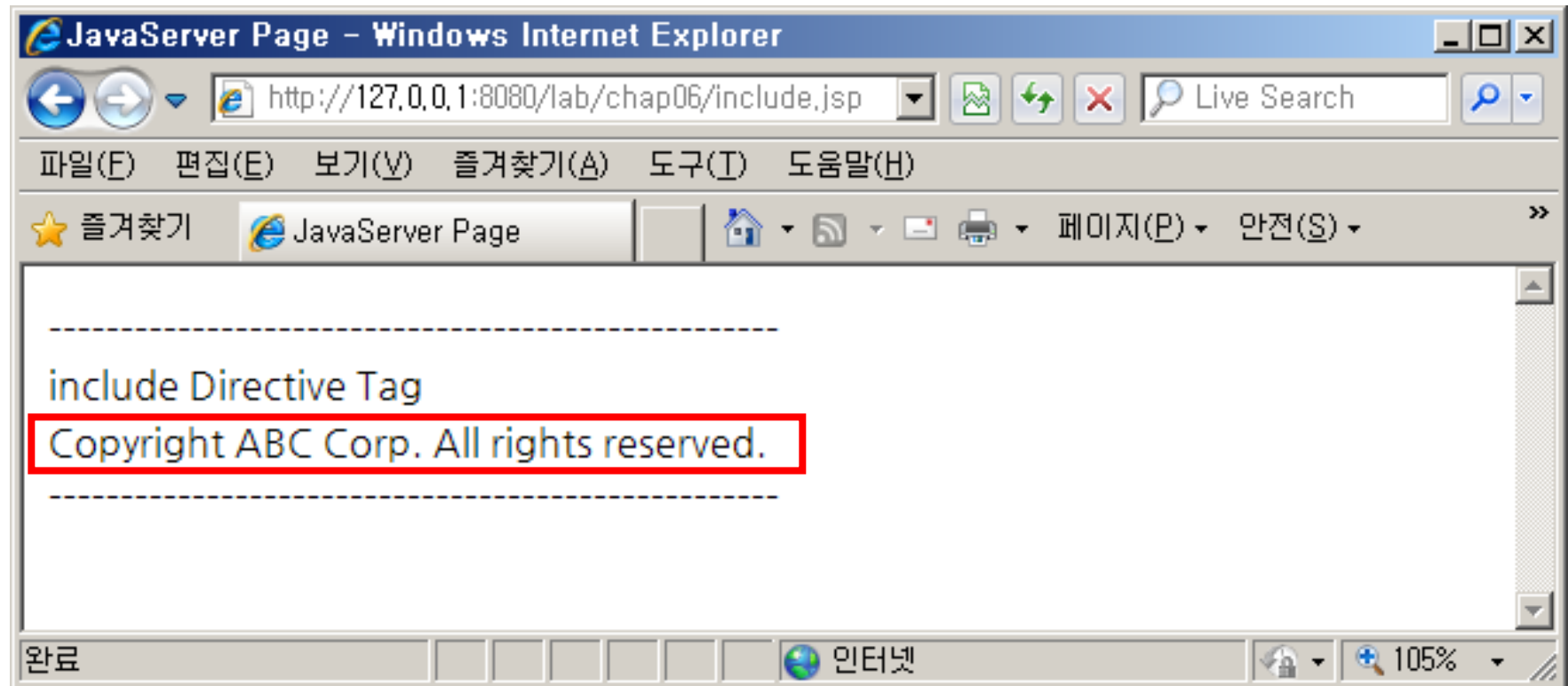
- 포함될 page - /chap06/footer.html

Copyright ABC Corp. All rights reserved.

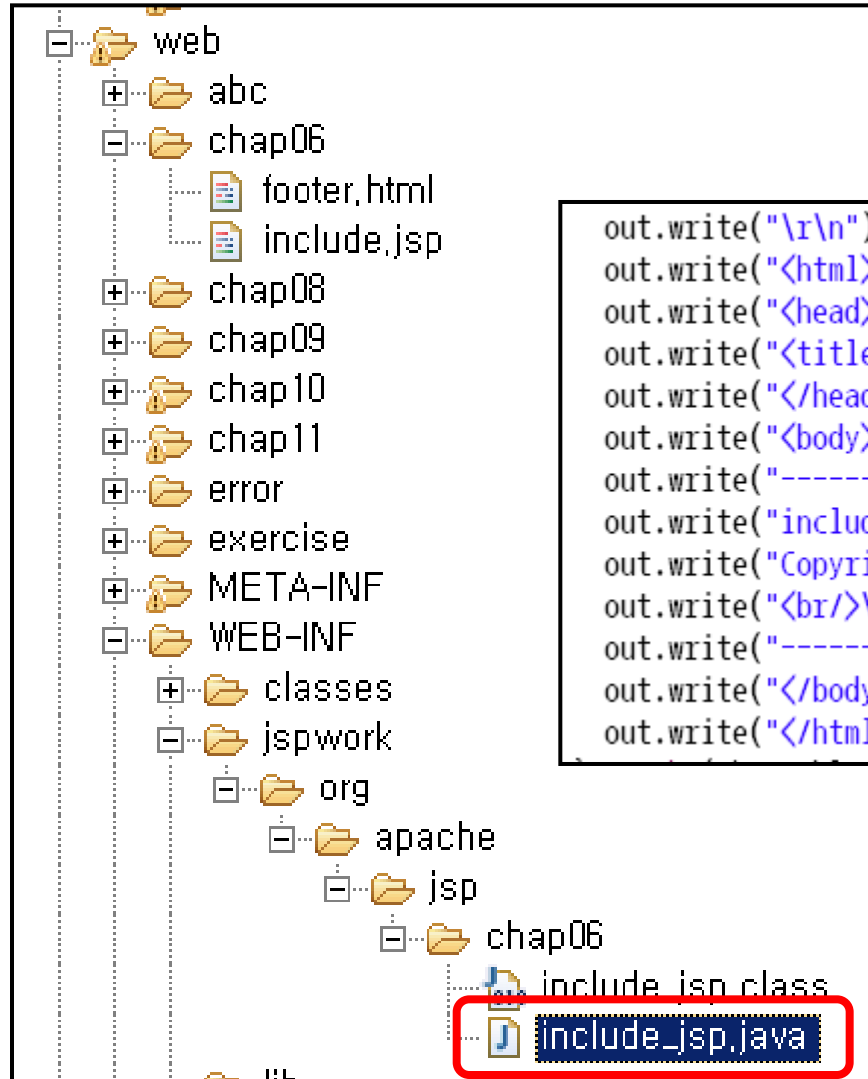
- 포함하는 page - /chap06/include.jsp

```
<%@ page contentType="text/html; charset=euc-kr" %>
<html>
<head>
<title>JavaServer Page</title>
</head>
<body>
-----<br/>
include Directive Tag<br/>
<%@ include file="/chap06/footer.html" %><br/>
-----
</body>
</html>
```

□ <http://127.0.0.1:8080/lab/chap06/include.jsp>



□ WEB_LAB\web\WEB-INF\jspwork\org\apache\jsp\chap06



include_jsp.java

```
out.write("\r\n");
out.write("<html>\r\n");
out.write("<head>\r\n");
out.write("<title>JavaServer Page</title>\r\n");
out.write("</head>\r\n");
out.write("<body>\r\n");
out.write("-----<br/>\r\n");
out.write("include Directive Tag<br/>\r\n");
out.write("Copyright ABC Corp. All rights reserved.");
out.write("<br/>\r\n");
out.write("-----\r\n");
out.write("</body>\r\n");
out.write("</html>\r\n");
```

include_jsp.java

```
out.write("<html>WrWn");
out.write("<head>WrWn");
out.write("<title>JavaServer Page</title>WrWn");
out.write("</head>WrWn");
out.write("<body>WrWn");
out.write("-----<%@ include file="/chap06/footer.html" %>
out.write("include Directive Tag<br/>WrWn");
out.write("Copyright ABC Corp. All rights reserved.");
out.write("<br/>WrWn");
out.write("-----WrWn");
out.write("</body>WrWn");
out.write("</html>WrWn");
```

6.1 JSP 란?

6.2 JSP Elements

6.3 내장 객체

6.4 JSP 지시자 태그

 **6.5 JSP Exception 처리**

6.6 JSP 디버깅

- ❑ JSP 페이지에서 발생하는 Exception을 처리하기 위해서는 별도의 예외 처리 페이지를 지정한다.
- ❑ 하나의 JSP 페이지에 대한 예외 처리 페이지는 하나만 지정할 수 있기 때문에 예외마다 다른 예외 처리는 불가능하다.

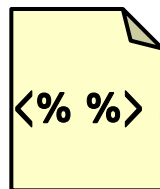
- throw_error.jsp

```
<%@page errorPage="/error/exceptionPage.jsp"%>
```

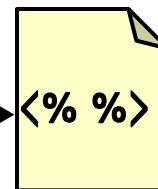
- exceptionPage.jsp

```
<%@page isErrorPage="true"%>
```

/throws_error.jsp



/error/exceptionPage.jsp



Web Container는 오류를 catch하여
에러 처리 페이지로 전달한다.

□ JSP 페이지 작성

throws_error.jsp

```
<%@ page contentType="text/html;charset=euc-kr"  
    errorPage="error/exceptionPage.jsp" %>
```

```
<html>  
<head>  
    <title>Demonstrate Error Page</title>  
</head>
```

```
<body>  
<% String str = ' null; %>  
<%= str.length() %>  
</body>  
</html>
```

□ 예외 처리 페이지 작성

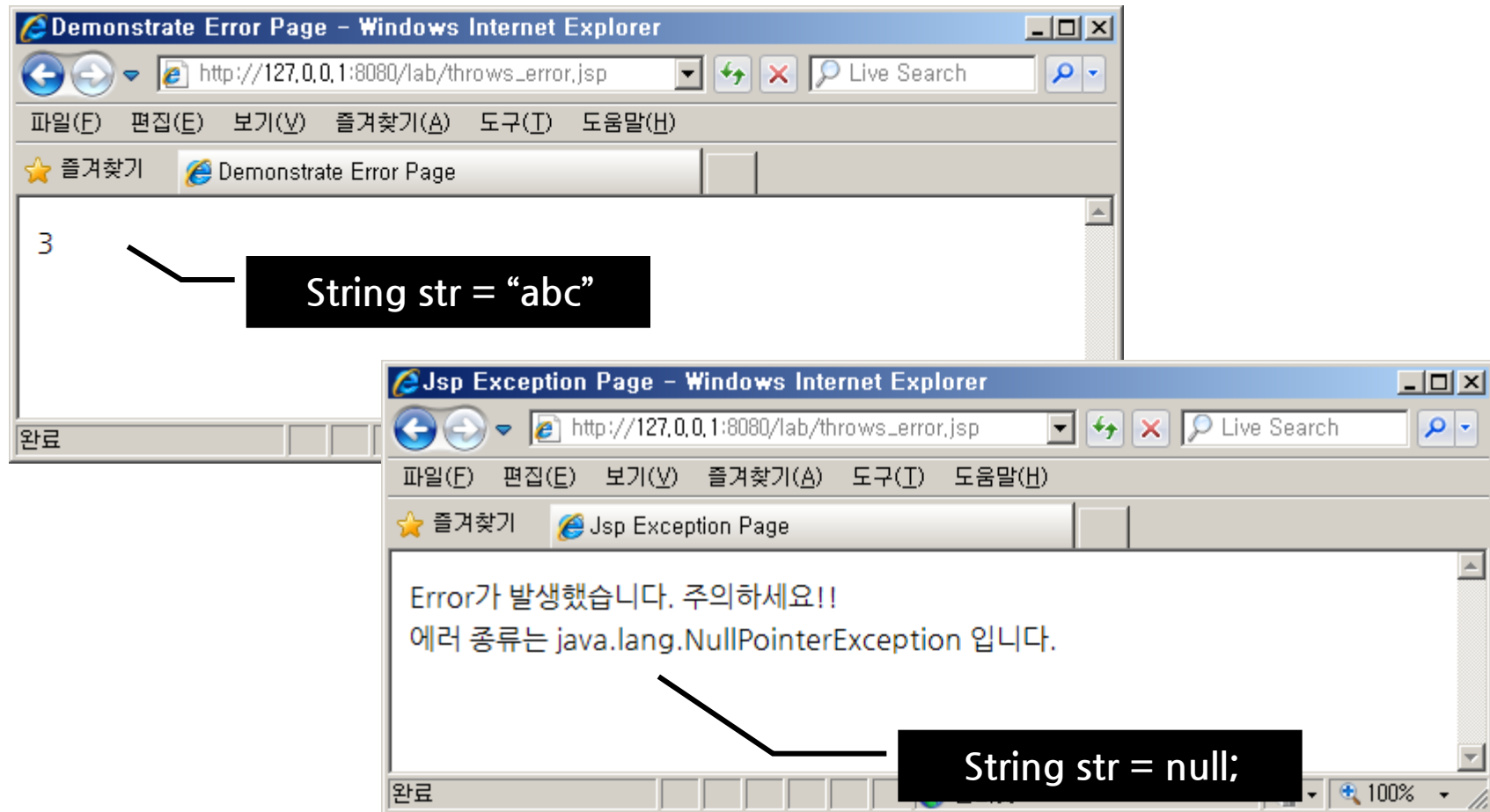
/error/exceptionPage.jsp

```
<%@ page contentType="text/html;charset=euc-kr"
    isErrorPage="true" %>

<html>
<head>
    <title>Jsp Exception Page</title>
</head>

<body>
    Error가 발생했습니다. 주의하세요!!<br/>
    에러 종류는 <%=exception.getClass().getName()%> 입니다.
</body>
</html>
```

❑ http://127.0.0.1:8080/lab/throws_error.jsp 실행



6.1 JSP 란?

6.2 JSP Elements

6.3 내장 객체

6.4 JSP 지시자 태그

6.5 JSP Exception 처리

 **6.6 JSP 디버깅**

Translation Time Error

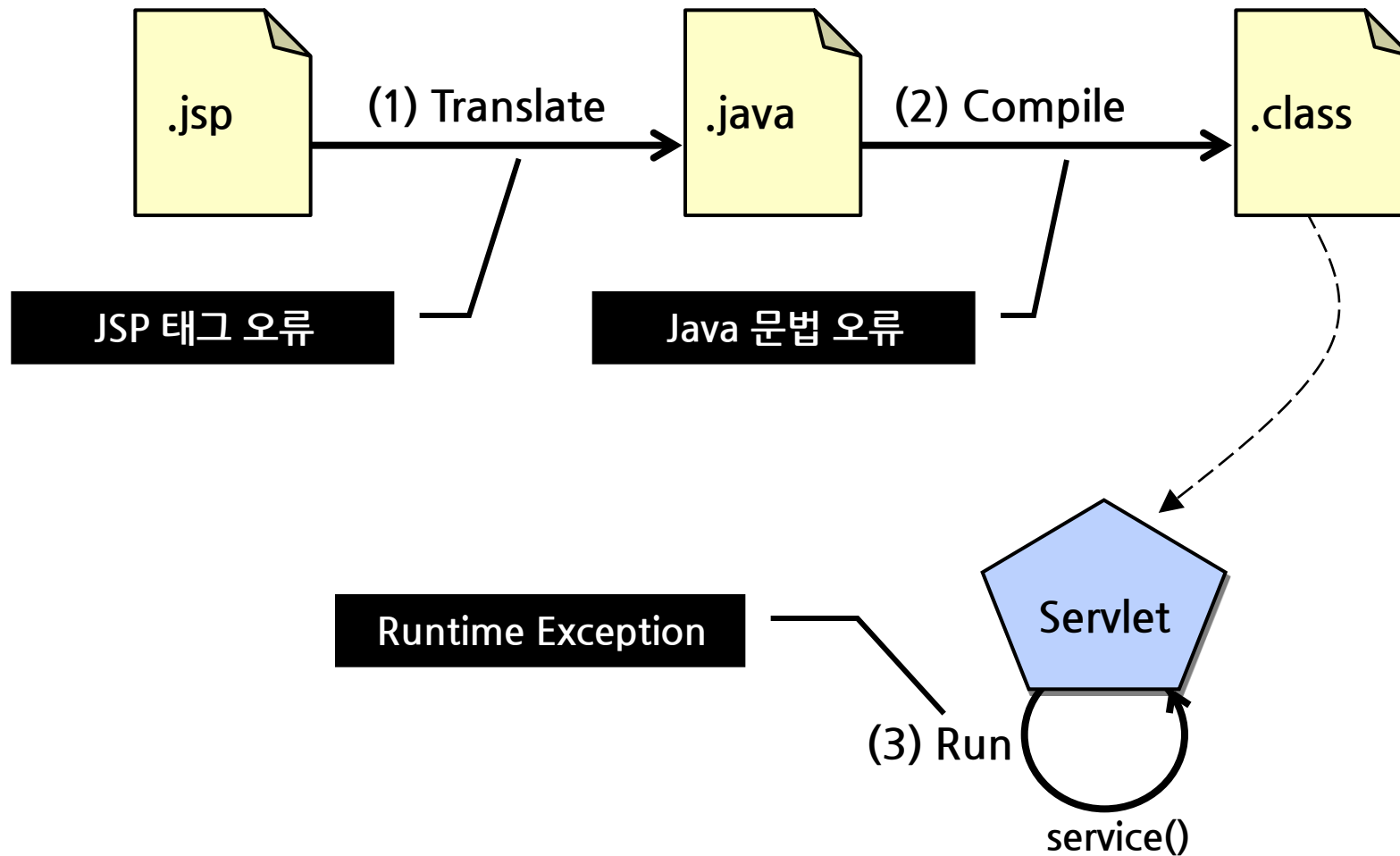
- JSP 파일을 Java 파일로 변환할 때 발생하는 오류
- Web Container가 JSP 페이지의 Scripting elements의 구문을 분석할 수 없을 때 발생
예) “%>” 없이 “<%!” 만 사용한 경우

Compile Time Error

- Java 파일을 Class 파일로 컴파일 할 때 발생하는 오류
- 자바 문법 오류
예) 자바 문장을 끝낼 때, “;” 사용하지 않음

Runtime error

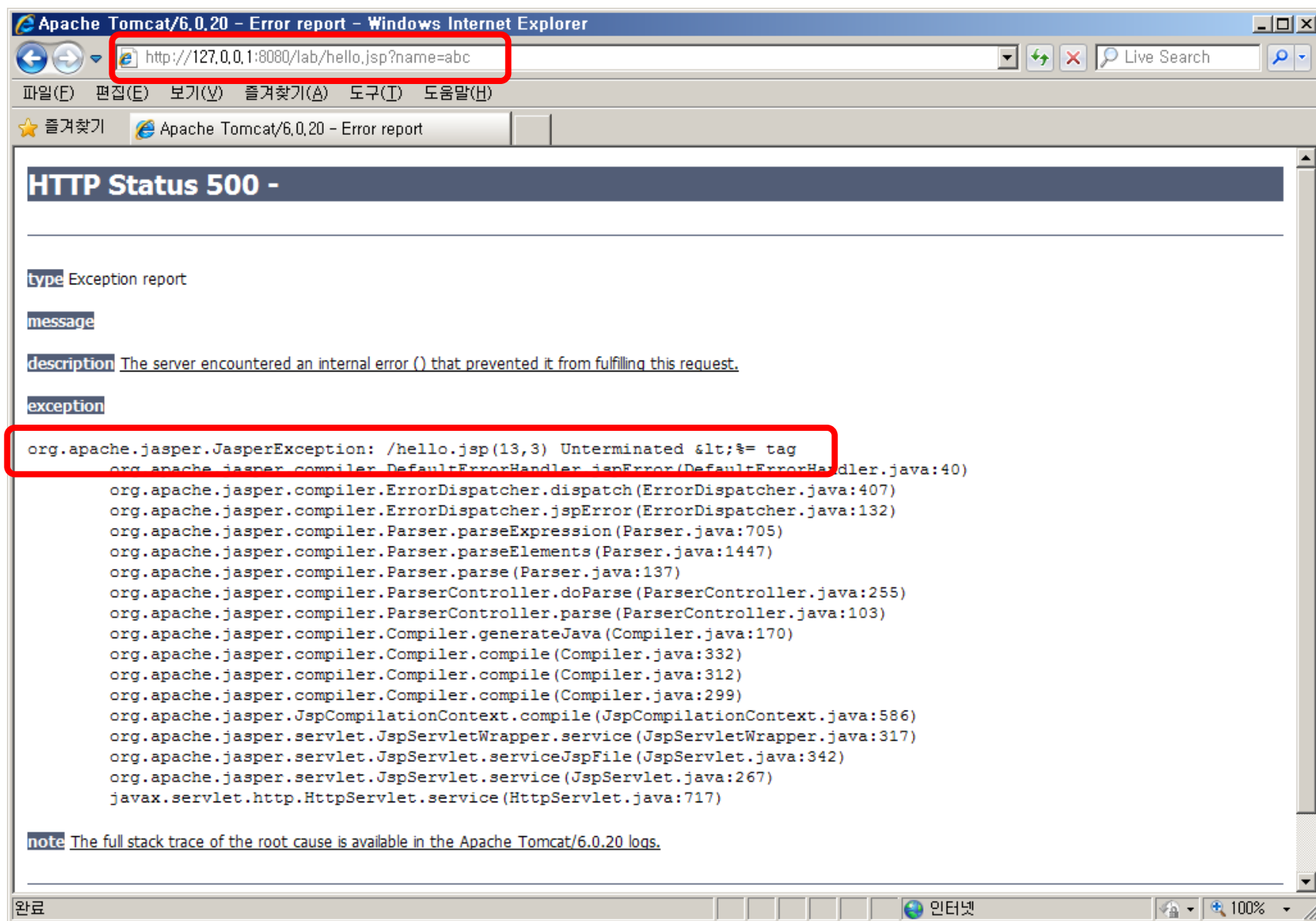
- 실행 도중, Exception이 발생하는 경우
예) getParameter() 인자에 HTML form 태그에 없는 이름을 사용하면, getParameter() 는 null을 return하는데 이 값을 가지고 조작하는 경우 NullPointerException 발생



- hello.jsp 내용을 수정한 후 웹 브라우저에서 `http://127.0.0.1:8080/lab/hello.jsp?name=abc` 를 호출한다.

hello.jsp

```
<html>
<head>
<title>Hello JavaServer Page</title>
</head>
<body>
<%-- Determine the specified name (or user default) --%>
<%
    request.setCharacterEncoding("euc-kr");
    out.print("Welcome :" );
%>
<%= request.getParameter("name")
</body>
</html>
```

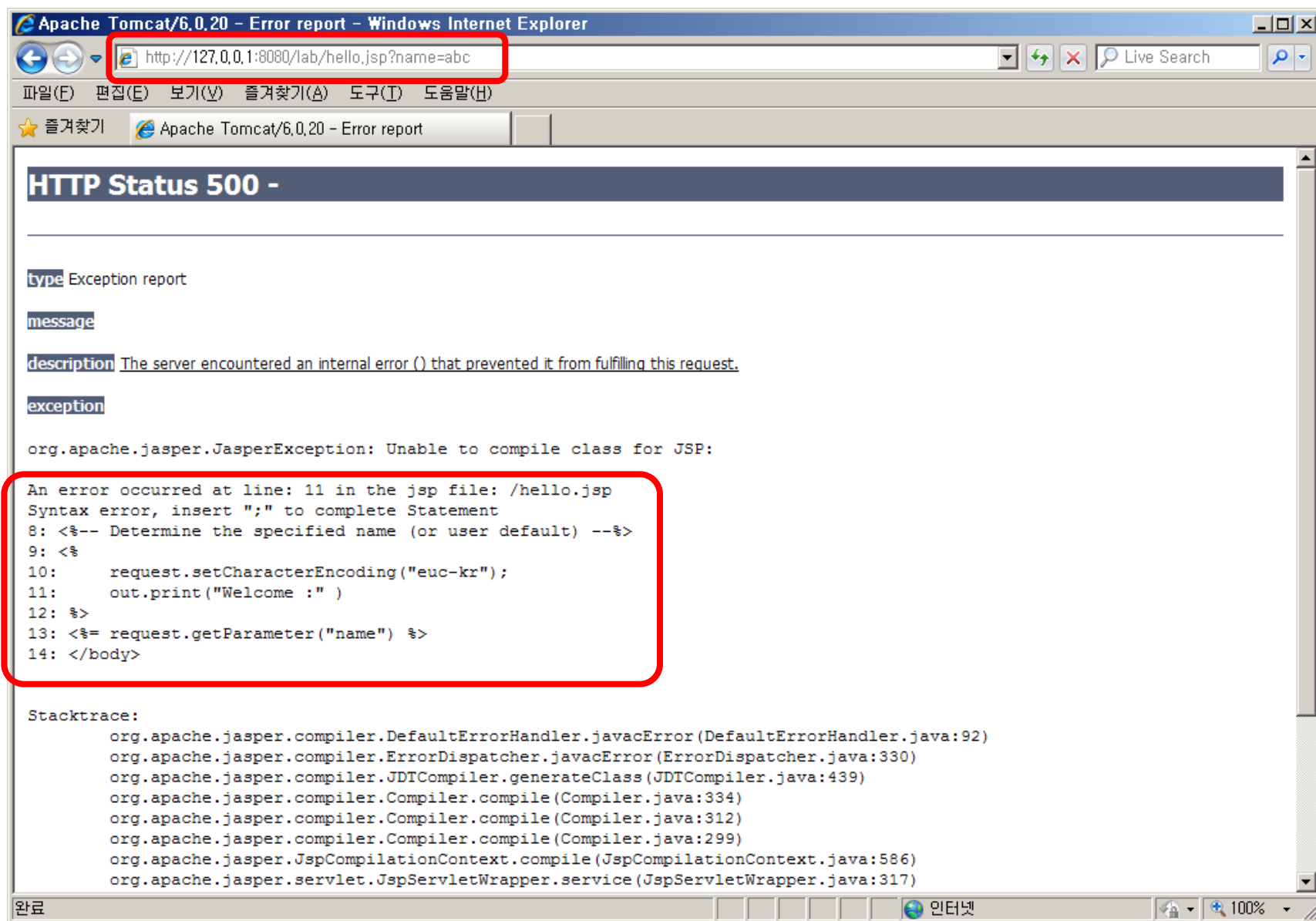


- hello.jsp 내용을 수정한 후 웹 브라우저에서 `http://127.0.0.1:8080/lab/hello.jsp?name=abc` 를 호출한다.

hello.jsp

```
<html>
<head>
<title>Hello JavaServer Page</title>
</head>
<body>
<%-- Determine the specified name (or user default) --%>
<%
    request.setCharacterEncoding("euc-kr");
    out.print("Welcome :" )
%>
<%= request.getParameter("name") %>
</body>
</html>
```

세미콜론(:) 누락



- hello.jsp 내용을 수정한 후 웹 브라우저에서 `http://127.0.0.1:8080/lab/hello.jsp?name2=abc` 를 호출한다.

hello.jsp

```
<html>
<head>
<title>Hello JavaServer Page</title>
</head>
<body>
<%-- Determine the specified name (or user default) --%>
<%
    request.setCharacterEncoding("euc-kr");
    out.print("Welcome :" );
    String name = request.getParameter("name");
    if ( name.equals("abc") ){
        out.println( name );
    }
%>
</body>
</html>
```

name의 null 여부를
검사하지 않고 사용

Apache Tomcat/6.0.20 - Error report - Windows Internet Explorer

http://127.0.0.1:8080/lab/hello.jsp?name2=abc

파일(F) 편집(E) 보기(V) 즐겨찾기(A) 도구(T) 도움말(H)

☆ 즐겨찾기 Apache Tomcat/6.0.20 - Error report

HTTP Status 500 -

type Exception report

message

description The server encountered an internal error () that prevented it from fulfilling this request.

exception

org.apache.jasper.JasperException: An exception occurred processing JSP page /hello.jsp at line 13

```
10:     request.setCharacterEncoding("euc-kr");
11:     out.print("Welcome :" );
12:     String name = request.getParameter("name");
13:     if ( name.equals("abc") ) {
14:         out.println( name );
15:     }
16: %>
```

Stacktrace:

```
org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:505)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:416)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:342)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:267)
javax.servlet.http.HttpServlet.service(HttpServlet.java:717)
```

root cause

```
java.lang.NullPointerException
    org.apache.jsp.hello_jsp._jspService(hello_jsp.java:68)
    org.apache.jasper.runtime.HttpJspBase.service(HttpJspBase.java:70)
    javax.servlet.http.HttpServlet.service(HttpServlet.java:717)
    org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:374)
    org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:342)
    org.apache.jasper.servlet.JspServlet.service(JspServlet.java:267)
    javax.servlet.http.HttpServlet.service(HttpServlet.java:717)
```

완료 인터넷 100%