

12장 JSP 스크립트 요소 기능

12.1 JSP 스크립트 요소

12.2 선언문 사용하기

12.3 스크립트릿 사용하기

12.4 표현식 사용하기

12.5 JSP 주석문 사용하기

12.6 스크립트 요소 이용해 실습하기

12.7 내장 객체(내장 변수) 사용하기

12.8 JSP 예외 처리하기

12.9 JSP welcome 파일 지정하기

12.10 스크립트 요소 이용해 회원 정보
조회하기

12.1 JSP 스크립트 요소

JSP 스크립트 요소(Scripting Element)

- JSP 페이지에서 여러 가지 동적인 처리를 제공하는 기능
- `<% %>` 기호 안에 자바 코드로 구현함
- `<% %>` 기호를 스크립트릿(scriptlet)이라고 부름

스크립트릿 종류

- 선언문(declaration tag): JSP에서 변수나 메서드를 선언할 때 사용
- 스크립트릿(scriptlet): JSP에서 자바 코드를 작성할 때 사용
- 표현식(expression tag): JSP에서 변수의 값을 출력할 때 사용

12.2 선언문 사용하기

선언문(Declaration Tag)

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

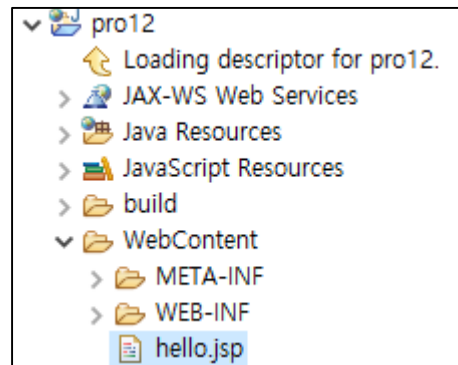
선언문 형식

<%! 멤버 변수 or 멤버 메서드 %>

12.2 선언문 사용하기

- 12.2.1 JSP에서 선언문 실습

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



12.2 선언문 사용하기

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

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;}
%>
```

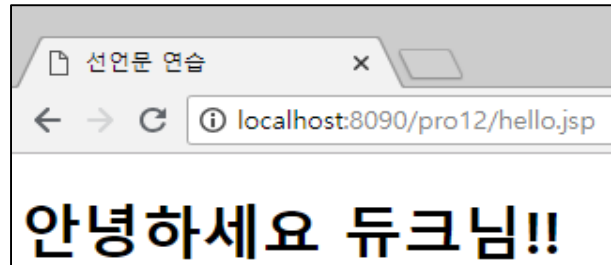
선언문을 이용해 멤버 변수 `name`과 멤버 메서드 `getName()`을 선언합니다.

```
<!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.

12.2 선언문 사용하기

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     private static final javax.servlet.jsp.JspFactory _jspxFactory =
23         javax.servlet.jsp.JspFactory.getDefaultFactory();
24
25     private static java.util.Map<java.lang.String,java.lang.Long> _jspx_dependants;
26
27     private static final java.util.Set<java.lang.String> _jspx_imports_packages;
28
29     private static final java.util.Set<java.lang.String> _jspx_imports_classes;
```

서블릿 클래스의 멤버 변수
와 멤버 메서드로 변환됩니다

12.3 스크립트릿 사용하기

스크립트릿(Scriptlet)

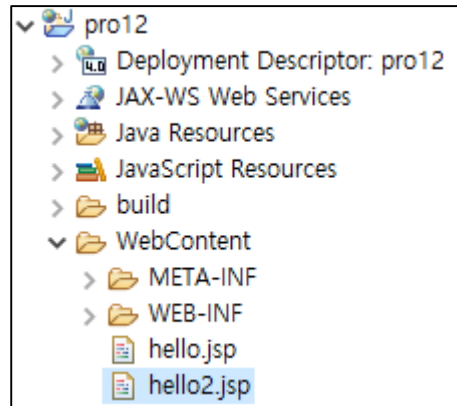
주로 초기 웹페이지에서 동적인 기능을 구현하기 위해서 사용됨

스크립트릿 형식

<% 자바 코드 %>

- 12.3.1 JSP에서 스크립트릿 실습

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



12.3 스크립트릿 사용하기

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"); %>

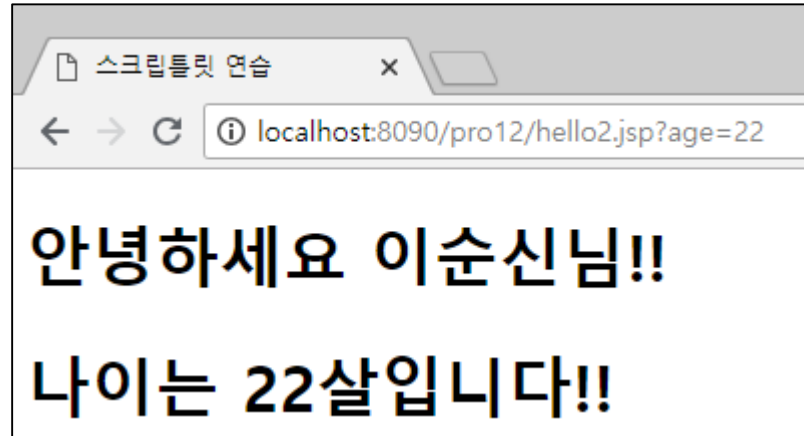
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
    <title>스크립트릿 연습</title>
</head>
<body>
    <h1>안녕하세요 <%=name %>님!!</h1>
    <h1>나이는 <%=age %>살입니다!!</h1>'
</body>
</html>
```

스크립트릿을 이용해 자바 코드를 작성합니다.

표현식을 이용해 전송된 나이를 출력합니다.

12.3 스크립트릿 사용하기

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



12.3 스크립트릿 사용하기

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("님!!</h1>\r\n");
136     out.write("    <h1>나이는 ");
137     out.print(age );
138     out.write("살입니다!!</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

12.4 표현식

표현식(Expression Tag)

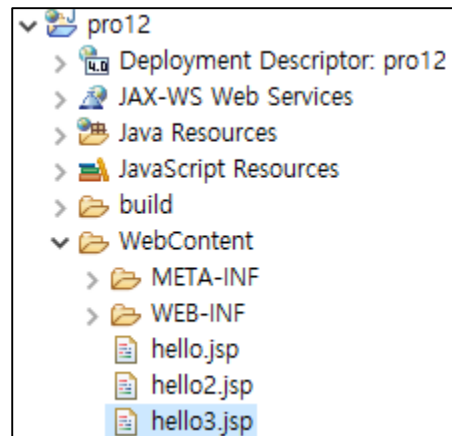
- JSP 페이지에서 원하는 위치에 값을 출력하는 기능

표현식 형식

<%=값 or 자바 변수 or 자바 식 %>

- 12.4.1 JSP에서 표현식 실습

1. 다음과 같이 hello3.jsp 파일을 준비합니다.



12.4 표현식

2. 다음과 같이 hello3.jsp를 작성합니다.

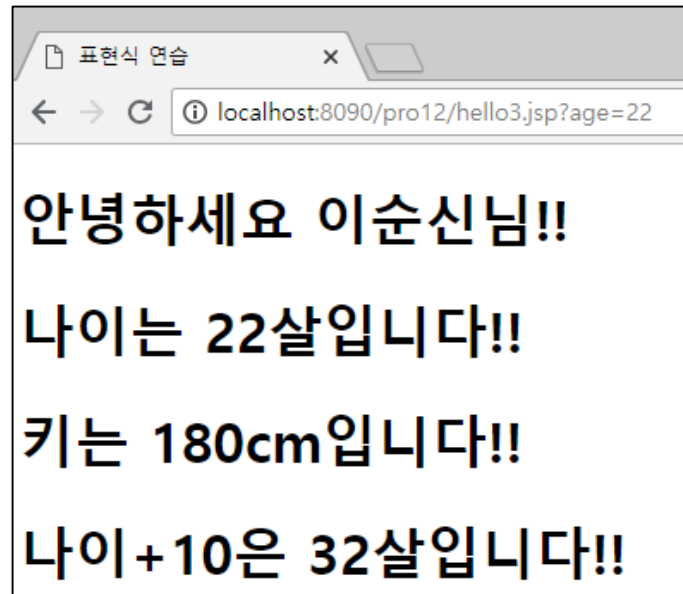
코드 12-3 pro12/WebContent/hello3.jsp

```
<%@ 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>
```

12.4 표현식

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



```

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.

12.4 표현식

선언문에 세미콜론(;) 추가

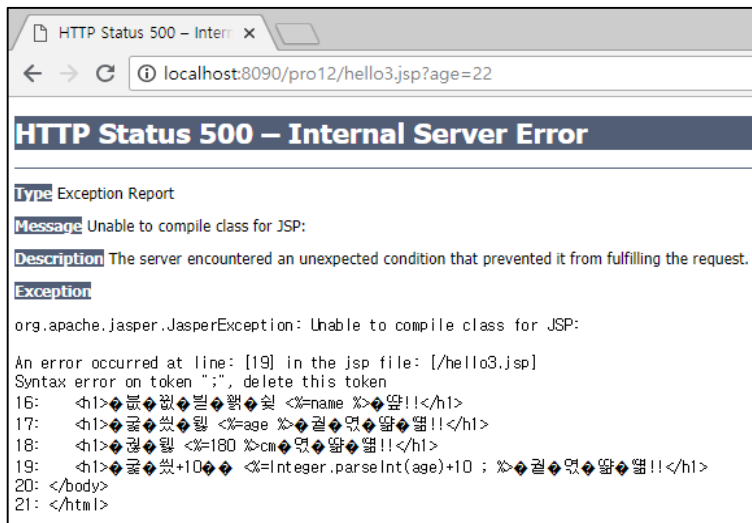
```

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!



12.5 JSP 주석문 사용하기

JSP에 사용되는 주석문

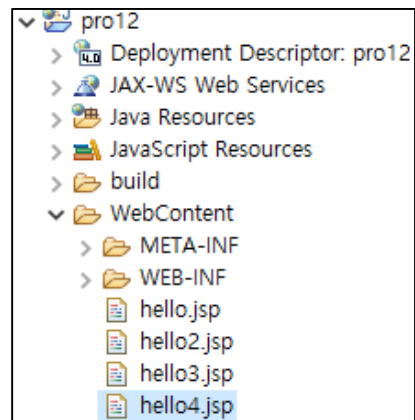
- HTML 주석
- 자바 주석
- JSP 주석

JSP 주석문 형식

`<%-- 내용 --%>`

• 12.5.1 JSP 페이지에서 주석문 사용하기

1. 다음과 같이 hello4.jsp 파일을 준비합니다.



12.5 JSP 주석문 사용하기

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>
```

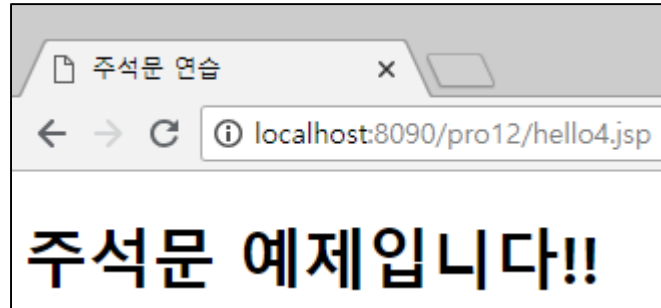
〈% %〉 안의 자바 코드에 대한 주석문

HTML 태그에 대한 주석문

JSP 페이지에 대한 주석문

12.5 JSP 주석문 사용하기

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.

12.5 JSP 주석문 사용하기

자바 주석문은 서블릿으로 변환 시 자바 주석문으로 표시됩니다.

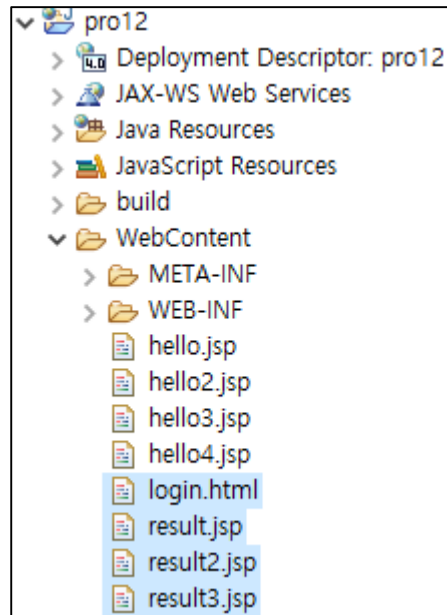
```
113 out = pageContext.getOut();
114 _jspx_out = out;
115
116 out.write("\r\n");
117 out.write("\r\n");
118
119 /*
120 String age=request.getParameter("age");
121 */
122
123 out.write(" \r\n");
124 out.write("<!DOCTYPE html>\r\n");
125 out.write("<!-- HTML 주석문입니다. -->\r\n");
126 out.write("<html>\r\n");
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.

12.6 스크립트 요소 이용해 실습하기

- 12.6.1 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.



12.6 스크립트 요소 이용해 실습하기

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.

12.6 스크립트 요소 이용해 실습하기

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>
```

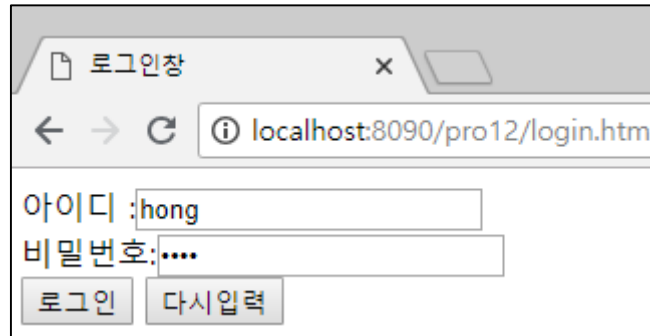
getParameter() 메서드를 이용해
입력 정보를 가져옵니다.

ID를 표현식으로 출력합니다.

비밀번호를 표현식으로 출력합니다.

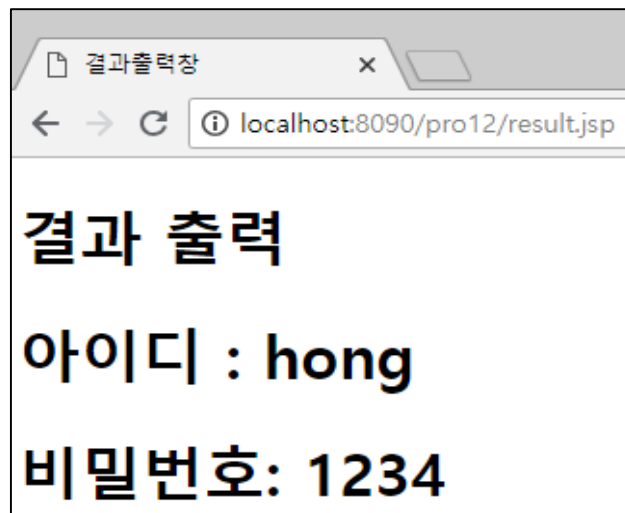
12.6 스크립트 요소 이용해 실습하기

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).

12.6 스크립트 요소 이용해 실습하기

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 pro12/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>
```

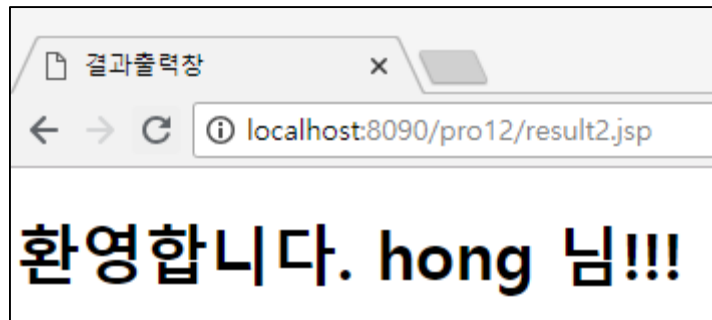
ID가 정상적으로 입력되었는지 체크합니다.

ID를 입력하지 않았을 경우 다시 로그인창으로 이동합니다.

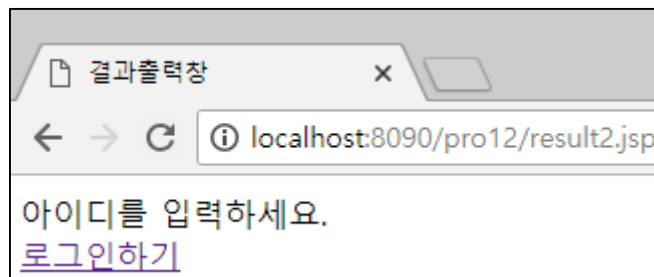
ID를 정상적으로 입력했을 경우 메시지를 표시합니다.

12.6 스크립트 요소 이용해 실습하기

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.



12.6 스크립트 요소 이용해 실습하기

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.

12.6 스크립트 요소 이용해 실습하기

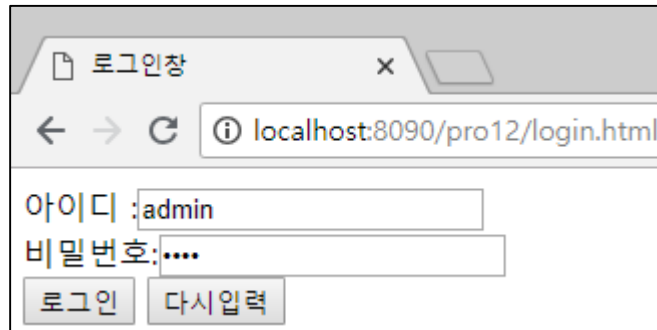
```
}else{
    if(user_id.equals("admin")){
    %>
    <h1>관리자로 로그인 했습니다.</h1>
    <form>
        <input type=button value="회원정보 삭제하기" />
        <input type=button value="회원정보 수정하기" />
    </form>
    <%
    }else{
    %>
        <h1> 환영합니다. <%=user_id %> 님!!!</h1>
    <%
    }
    }
    %>
</body>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

ID를 입력한 경우 ID가 admin인지 다시 체크합니다.

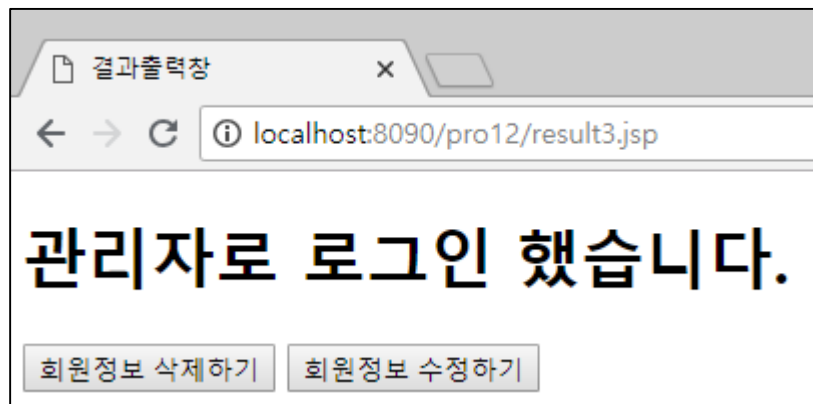
ID가 admin이면 관리자창을 나타냅니다.

12.6 스크립트 요소 이용해 실습하기

10. 다음은 admin으로 로그인했을 때의 실행 결과입니다.

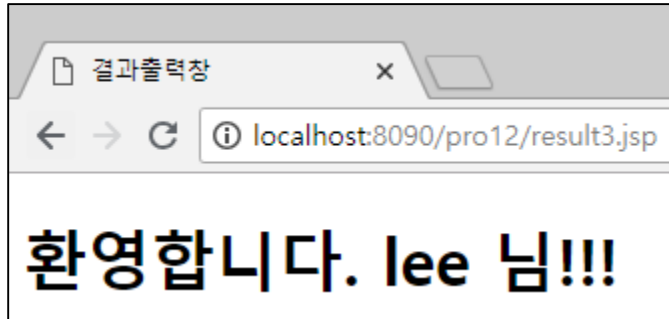


11. 관리자창이 나타납니다.



12.6 스크립트 요소 이용해 실습하기

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



12.6 스크립트 요소 이용해 실습하기

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
24 <h1>관리자로 로그인 했습니다.</h1>
25=<form>
26 <input type=button value="회원정보 삭제하기" />
27 <input type=button value="회원정보 수정하기" />
28 </form>
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

```

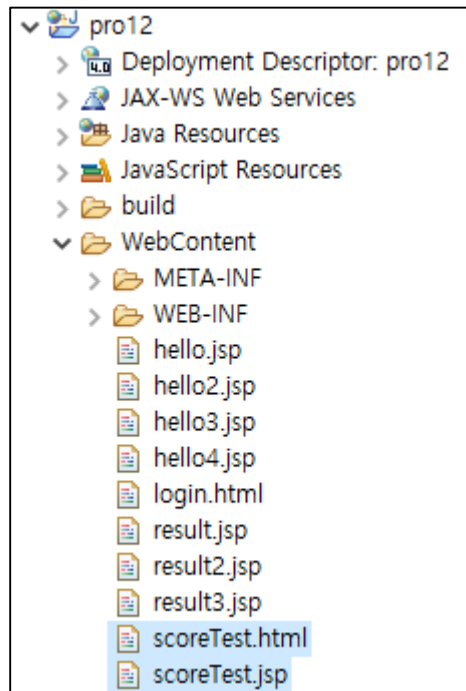
21:     }else{
22:         if(user_id.equals("admin")){

```

12.6 스크립트 요소 이용해 실습하기

- 12.6.1 학점 변환 예제

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



12.6 스크립트 요소 이용해 실습하기

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.

12.6 스크립트 요소 이용해 실습하기

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.

12.6 스크립트 요소 이용해 실습하기

```
<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.

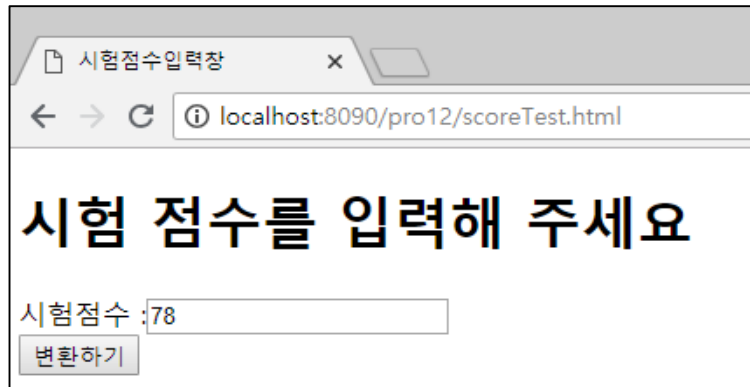
70~80점 사이면 C를 출력합니다.

60~70점 사이면 D를 출력합니다.

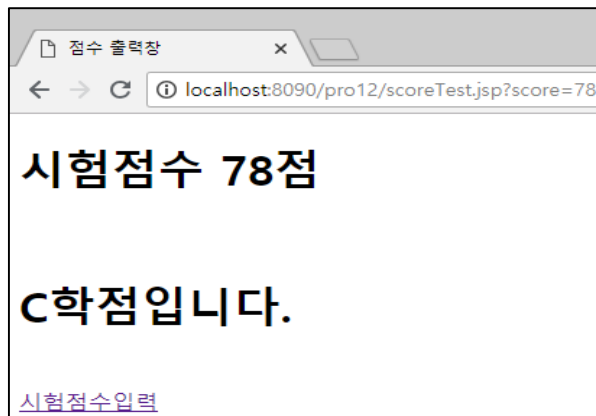
그 외 점수는 F를 출력합니다.

12.6 스크립트 요소 이용해 실습하기

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



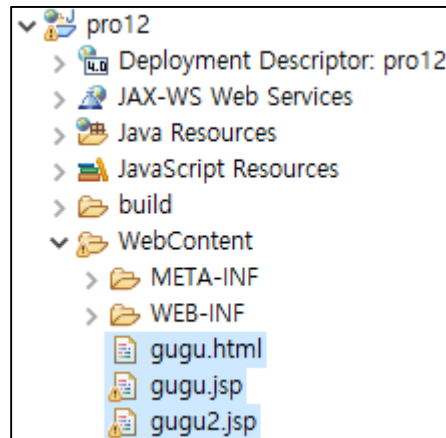
5. 시험 점수를 학점으로 변환하여 출력합니다.



12.6 스크립트 요소 이용해 실습하기

- 12.6.3 구구단 출력 예제

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



12.6 스크립트 요소 이용해 실습하기

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.

12.6 스크립트 요소 이용해 실습하기

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.

12.6 스크립트 요소 이용해 실습하기

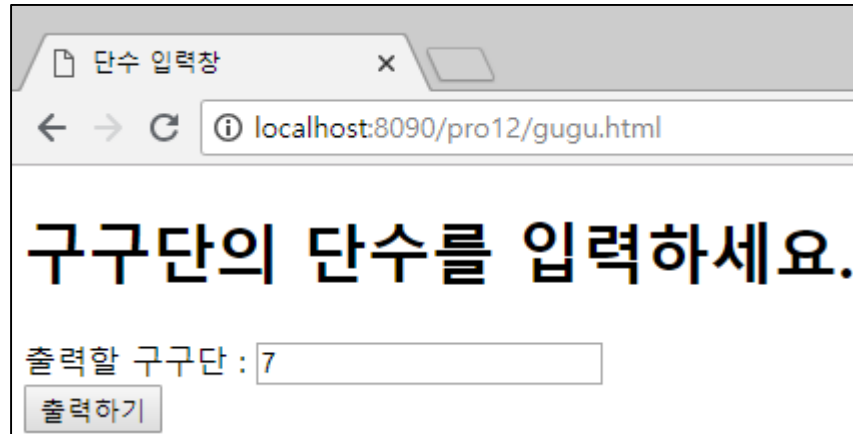
```
<td colspan='2'><%= dan %>단 출력 </td>
</tr>
<%
  for(int i=1; i<10;i++){
%>
    <tr align='center'>
      <td width='400'>
        <%=dan %> * <%=i %>
      </td>
      <td width='400'>
        <%=i*dan %>
      </td>
    </tr>
  <%
  }
%>
</table>
</body>
</html>
```

전송된 단수를 출력합니다.

for 반복문을 이용해 테이블의 각 행에 연속해서 구구단을 출력합니다.

12.6 스크립트 요소 이용해 실습하기

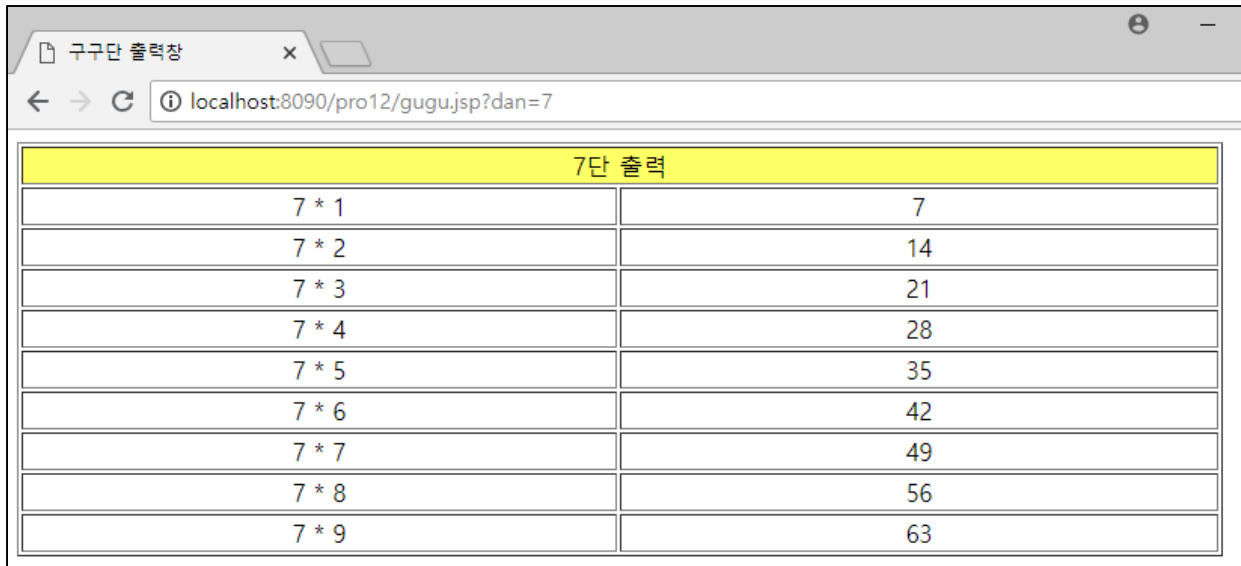
4. <http://localhost:8090/pro12/gugu.html>로 요청하여 입력창에서 단수를 입력한 후 전송합니다.



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

12.6 스크립트 요소 이용해 실습하기

5. for문을 이용해 구구단을 리스트로 출력합니다.



The screenshot shows a web browser window with the title '구구단 출력창' and the address bar displaying 'localhost:8090/pro12/gugu.jsp?dan=7'. The main content area contains a table with a yellow header row labeled '7단 출력'. Below the header, there are 9 rows of multiplication results for the number 7, ranging 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

12.6 스크립트 요소 이용해 실습하기

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

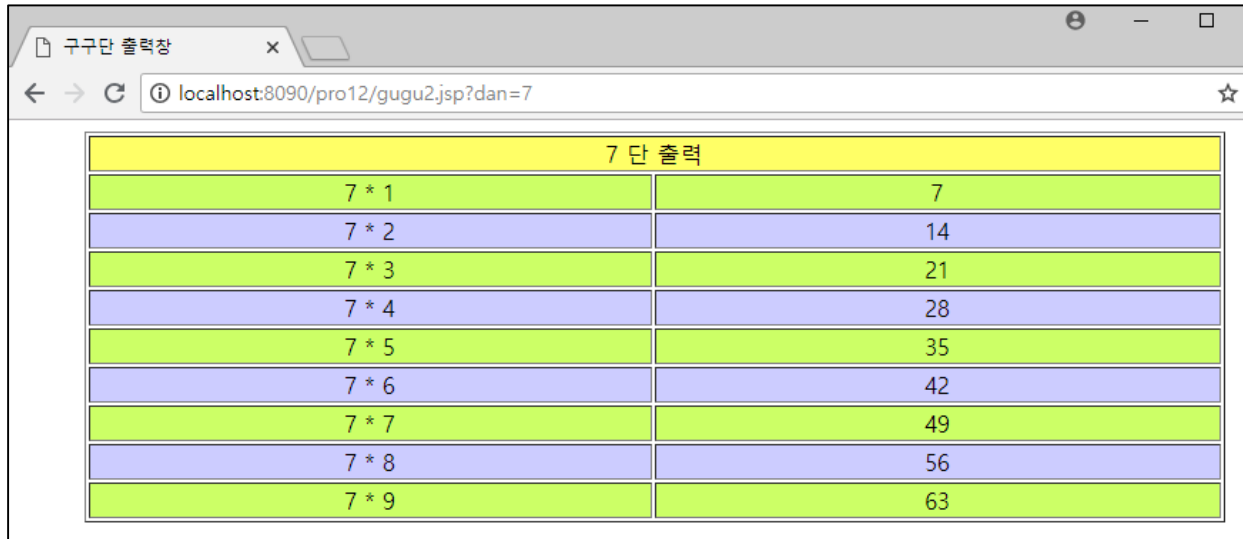
```
...
<%
    for(int i=1; i<10;i++){
%>
<%
    if(i%2==1){
%>
        <tr align=center bgcolor="#CCFF66">
<%
    }else{
%>
        <tr align=center bgcolor="#CCCCFF">
<%
    }
%>
...

```

테이블의 홀수 행과 짝수 행의 색깔을
다르게 표시합니다.

12.6 스크립트 요소 이용해 실습하기

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

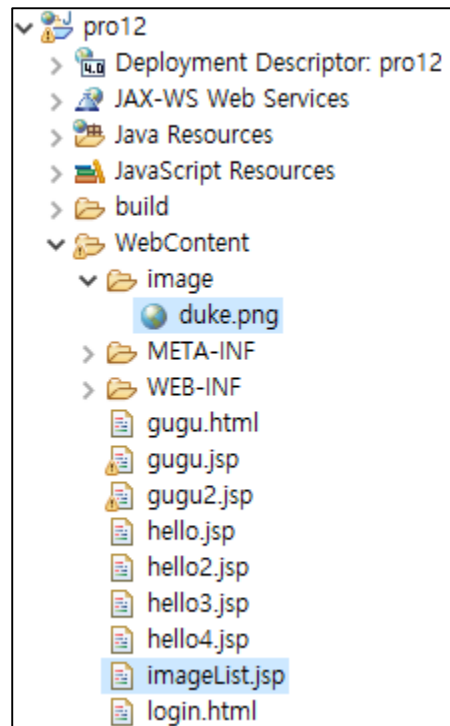


7 단 출력	
7 * 1	7
7 * 2	14
7 * 3	21
7 * 4	28
7 * 5	35
7 * 6	42
7 * 7	49
7 * 8	56
7 * 9	63

12.6 스크립트 요소 이용해 실습하기

- 12.6.4 이미지 리스트 출력 예제

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



12.6 스크립트 요소 이용해 실습하기

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.

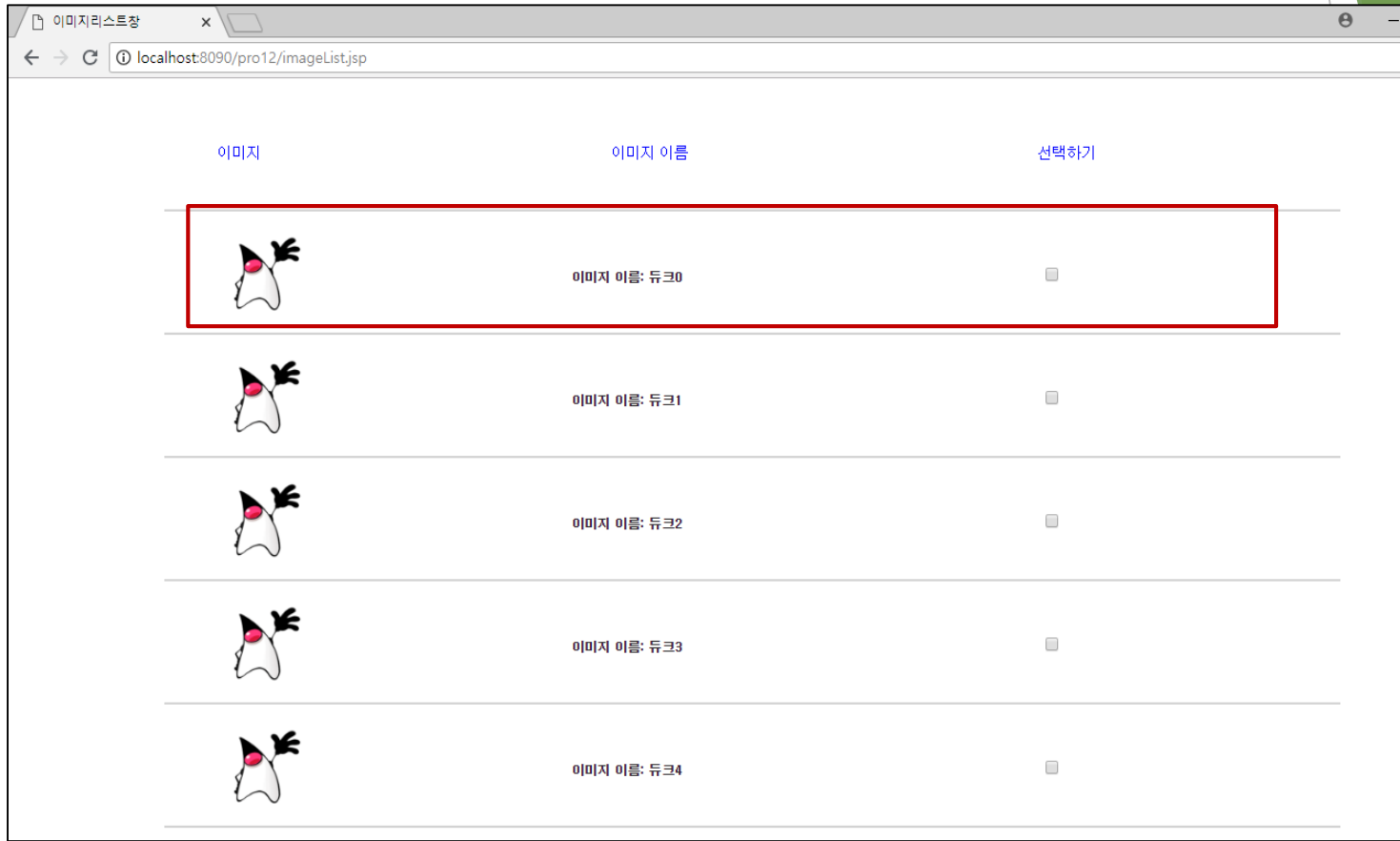
for 반복문을 이용해 태그를 연속해서 출력합니다.

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

image 폴더의 이미지를 나타냅니다.








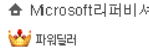
12.6 스크립트 요소 이용해 실습하기

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



12.6 스크립트 요소 이용해 실습하기

검색 상품 리스트 출력

플러스 상품 광고 ①			
	[삼성전자] 삼성컴퓨터 SSD장착 최대80% 할인	199,000원 400,000원 50% ↓ 무료배송	
삼성 컴퓨터 모음전		상품평 ★★★★	
	[삼성전자] 삼성 사무용 컴퓨터 i3 i5 듀얼 쿼드코어 PC 모음전	155,000원 무료배송	
삼성 사무용 컴퓨터		상품평 ★★★★	
	[LG전자] LG데스크탑 Z71EV-A X7P26 G4560 4G SSD 128 원도우선택	419,000원 445,740원 5% ↓ 무료배송	
원자(G4560 / 4G / SSD 128G / 유선키보드 / 윈도우선택) LG 데스크탑 PC Z71EV-A X7P26		상품평 ★★★★	
	중고 빠른부팅신속SSD넉넉한저장공간 DB-Z600 슬림PC	299,000원 무료배송	

12.6 스크립트 요소 이용해 실습하기

❖ JSP 프리컴파일(Precompile) 기능

브라우저에서 서블릿으로 최초 요청을 보내면 먼저 톰캣이 컴파일을 한 후 실행을 합니다. 따라서 톰캣은 시작 시 미리 서블릿을 메모리에 로드해서 사용하는 방법을 제공합니다(8.6절 참고). JSP도 최초 요청 시 변환 과정을 거치기 때문에 실행이 늦어지게 됩니다. 따라서 톰캣 컨테이너에서는 JSP Precompile 기능을 제공해 미리 JSP를 컴파일함으로써 요청 시 바로 처리할 수 있도록 하고 있습니다. 웹 애플리케이션 개발 시에는 JSP에 변경 사항이 자주 발생하므로 그다지 필요할 것 같지 않지만 실제 서비스를 제공할 때는 사용하면 좋은 기능입니다. 자세한 내용은 톰캣 홈페이지에서 확인하세요.

12.7내장 객체(내장 변수) 기능

내장 객체(내장 변수)

➤ JSP가 서블릿으로 변환 시 컨테이너가 자동으로 생성 시키는 서블릿 멤버 변수

```
79 public void _jspService(final javax.servlet.http.HttpServletRequest request, final javax.se
80     throws java.io.IOException, javax.servlet.ServletException {
81
82     if (!javax.servlet.DispatcherType.ERROR.equals(request.getDispatcherType())) {
83         final java.lang.String _jspx_method = request.getMethod();
84         if ("OPTIONS".equals(_jspx_method)) {
85             response.setHeader("Allow","GET, HEAD, POST, OPTIONS");
86             return;
87         }
88         if (!"GET".equals(_jspx_method) && !"POST".equals(_jspx_method) && !"HEAD".equals(_jspx
89             response.setHeader("Allow","GET, HEAD, POST, OPTIONS");
90             response.sendError(HttpServletResponse.SC_METHOD_NOT_ALLOWED, "JSPs only permit GET,
91             return;
92         }
93     }
94
95     final javax.servlet.jsp.PageContext pageContext;
96     javax.servlet.http.HttpSession session = null;
97     final javax.servlet.ServletContext application;
98     final javax.servlet.ServletConfig config;
99     javax.servlet.jsp.JspWriter out = null;
100     final java.lang.Object page = this;
101     javax.servlet.jsp.JspWriter _jspx_out = null;
102     javax.servlet.jsp.PageContext _jspx_page_context = null;
```

12.7 내장 객체(내장 변수) 기능

JSP에서 제공하는 내장 객체들

내장객체	서블릿	설명
request	javax.servlet.http.HttpServletRequest	클라이언트의 요청 정보를 저장합니다.
response	javax.servlet.http.HttpServletResponse	응답 정보를 저장합니다.
out	javax.servlet.jsp.JspWriter	JSP 페이지에서 결과를 출력합니다.
session	javax.servlet.http.HttpSession	세션 정보를 저장합니다.
application	javax.servlet.ServletContext	컨텍스트 정보를 저장합니다.
pageContext	javax.servlet.jsp.PageContext	JSP 페이지에 대한 정보를 저장합니다.
page	java.lang.Object	JSP 페이지의 서블릿 인스턴스를 저장합니다.
config	javax.servlet.ServletConfig	JSP 페이지에 대한 설정 정보를 저장합니다.
exception	java.lang.Exception	예외 발생 시 예외를 처리합니다.

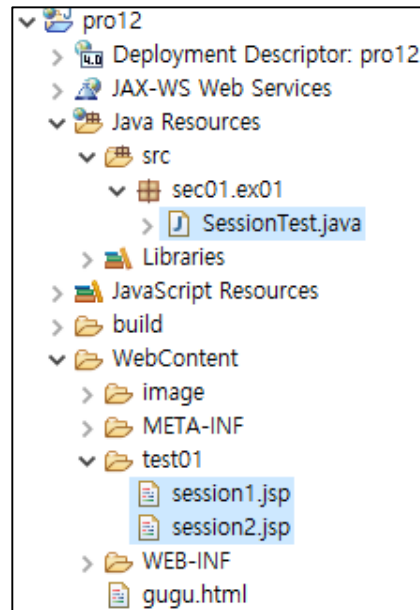
내장 객체들의 스코프

내장객체	서블릿	스코프
page	this	한 번의 요청에 대해 하나의 JSP 페이지를 공유합니다.
request	HttpServletRequest	한 번의 요청에 대해 같은 요청을 공유하는 JSP 페이지를 공유합니다.
session	HttpSession	같은 브라우저에서 공유합니다.
application	ServletContext	같은 애플리케이션에서 공유합니다.

12.7내장 객체(내장 변수) 기능

- 12.7.1 session 내장 객체에 데이터 바인딩 실습

1. JSP 파일이 많아지므로 test01 폴더를 만든 후 session1.jsp, session2.jsp 등 실습 파일들을 생성합니다.



12.7내장 객체(내장 변수) 기능

2. SessionTest 클래스를 다음과 같이 작성합니다.

코드 12-15 pro12/src/sec01/ex01/SessionTest.java

```
package sec01.ex01;

...

@WebServlet("/sess")

public class SessionTest extends HttpServlet {
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        response.setContentType("text/html;charset=utf-8");
        PrintWriter pw = response.getWriter();
        HttpSession session = request.getSession();
        session.setAttribute("name", "이순신");
        pw.println("<html><body>");
        pw.println("<h1>세션에 이름을 바인딩합니다.</h1>");
        pw.println("<a href='/pro12/test01/session1.jsp'>첫 번째 페이지로 이동하기 </a>");
        pw.println("</body></html>");
    }
}
```

session 객체를 가져옵니다.

session 객체에 name을 바인딩합니다.

12.7내장 객체(내장 변수) 기능

3. session1.jsp 파일을 다음과 같이 작성합니다.

코드 12-16 pro12/WebContent/test01/session1.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    String name=(String)session.getAttribute("name");
    session.setAttribute("address","서울시 강남구");
%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>session 내장 객체 테스트2</title>
</head>
<body>
    이름은 <%=name %>입니다. <br>
    <a href=session3.jsp>세 번째 페이지로 이동</a>
</body>
</html>
```

session 객체에 바인딩된 name 값을 가져옵니다.

session 객체에 address를 바인딩합니다.

12.7내장 객체(내장 변수) 기능

4. session2.jsp에서는 `getAttribute()`를 이용해 서블릿과 JSP에서 session에 바인딩된 name과 address 값을 가져옵니다.

코드 12-17 pro12/WebContent/test01/session2.jsp

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

<%
    String name=(String)session.getAttribute("name");
    String address = (String)session.getAttribute("address");
%>

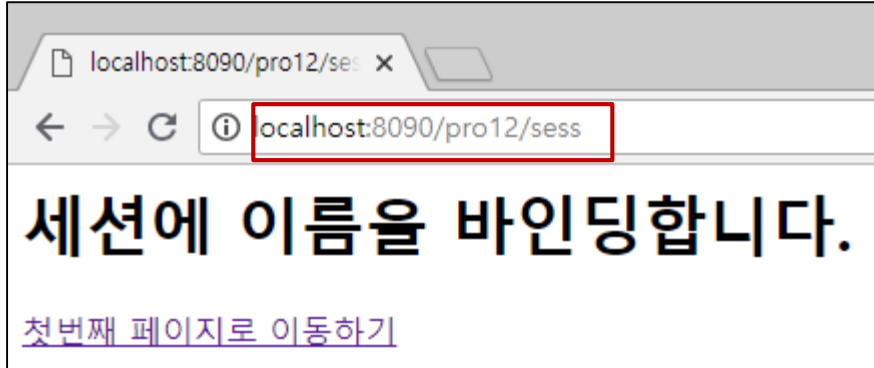
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>session 내장 객체 테스트3</title>
</head>

<body>
    이름은 <%=name %>입니다.<br>
    주소는 <%=address %>입니다. <br>
</body>
</html>
```

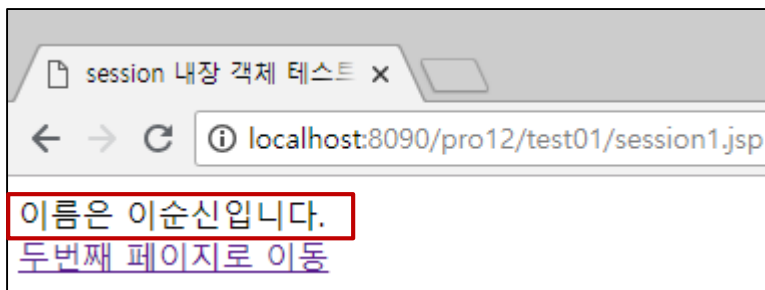
session 객체에 바인딩된
name 값과 address 값을
가져옵니다.

12.7내장 객체(내장 변수) 기능

5. 다음은 최초 서블릿에 요청한 결과입니다. 서블릿 요청 시 session 객체에 name을 바인딩합니다.

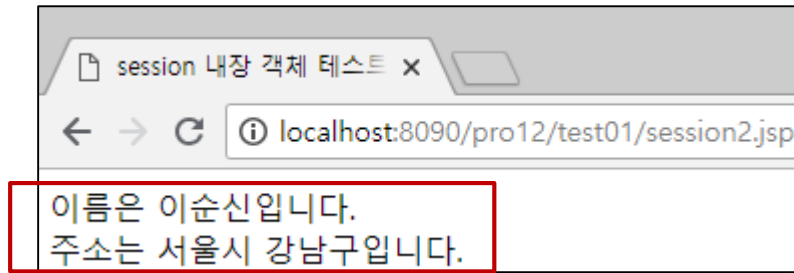


6. 첫번째 페이지로 이동하기 클릭 시 서블릿에서 바인딩한 name을 출력합니다.
두번째 페이지로 이동을 클릭합니다.



12.7내장 객체(내장 변수) 기능

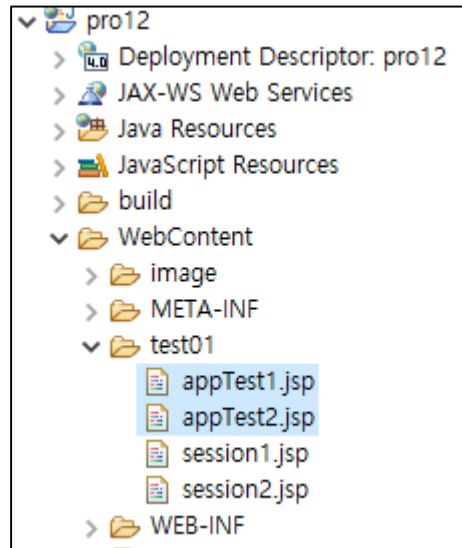
6. 서블릿과 첫 번째 JSP에서 바인딩한 이름(name)과 주소(address)를 출력합니다.



12.7내장 객체(내장 변수) 기능

- 12.7.2 application 내장 객체에 데이터 바인딩 실습

1. 다음과 같이 appTest1.jsp, appTest2.jsp 실습 파일을 준비합니다.



12.7내장 객체(내장 변수) 기능

2. appTest1.jsp를 다음과 같이 작성합니다. session과 application 내장 객체에 name과 address 값을 바인딩합니다.

코드 12-18 pro12/WebContent/test01/appTest1.jsp

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

```
<%
    session.setAttribute("name","이순신");
    application.setAttribute("address","서울시 성동구");
%>
```

이름과 주소를 session과 application 내장 객체에 바인딩합니다.

```
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>내장 객체 스코프 테스트1</title>
</head>
<body>
    <h1>이름과 주소를 저장합니다.</h1>
    <a href=appTest2.jsp>두 번째 웹 페이지로 이동</a>
</body>
</html>
```

12.7내장 객체(내장 변수) 기능

3. appTest2.jsp를 다음과 같이 작성합니다. 첫 번째 JSP에서 session과application 내장 객체에 바인딩한 값을 가져옵니다

코드 12-19 pro12/WebContent/test01/appTest2.jsp

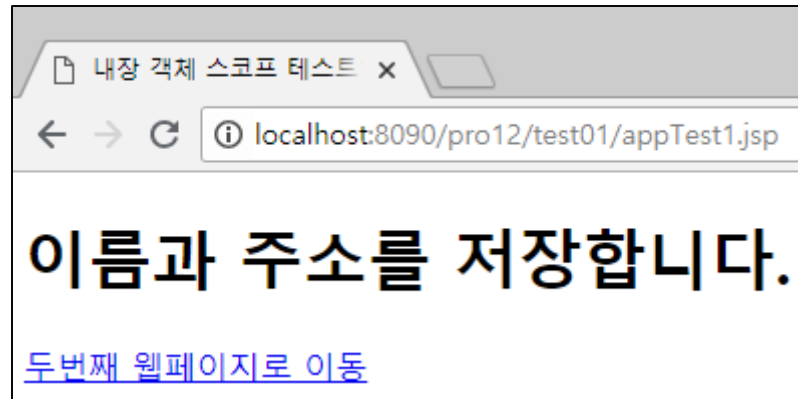
```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    String name=(String)session.getAttribute("name");
    String address=(String )application.getAttribute("address");
%>

<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>내장 객체 스코프 테스트2</title>
</head>
<body>
    <h1>이름은 <%=name %>입니다.</h1>
    <h1>주소는 <%=address %>입니다.</h1>
</body>
</html>
```

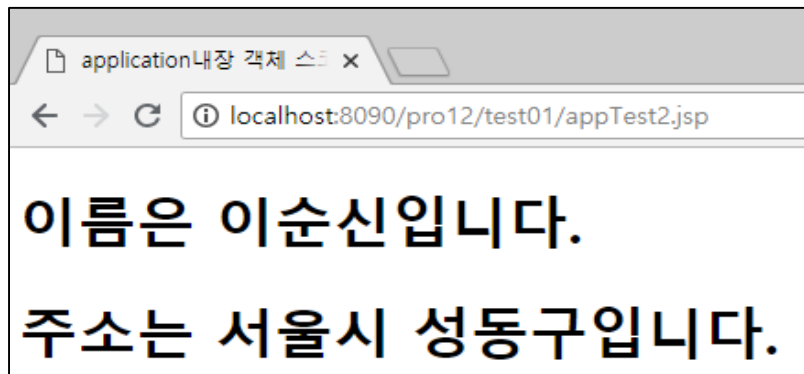
첫 번째 웹 페이지에서 저장한 데이터를 session과 application 내장 객체에서 가져옵니다.

12.7내장 객체(내장 변수) 기능

4. `http://localhost:8090/pro12/appTest1.jsp`로 요청합니다. 첫 번째 JSP에서 `name`과 `address`를 `session`과 `application`에 바인딩합니다.

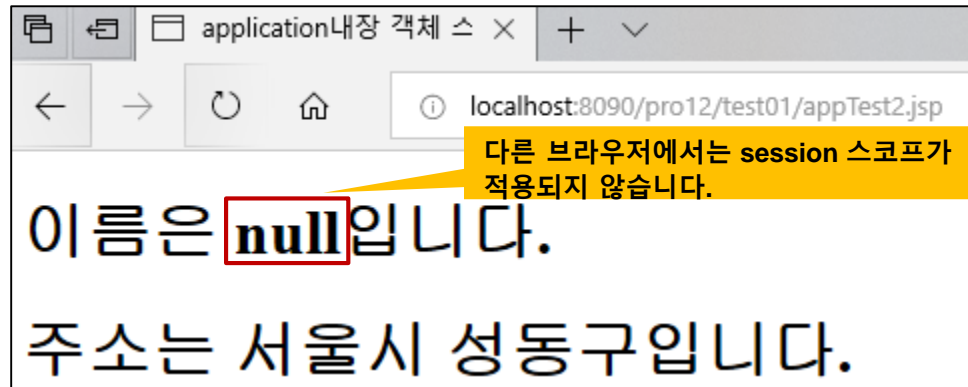


5. 같은 브라우저에서 요청할 경우 두 번째 JSP에서 `session`과 `application`에 접근할 수 있습니다.



12.7내장 객체(내장 변수) 기능

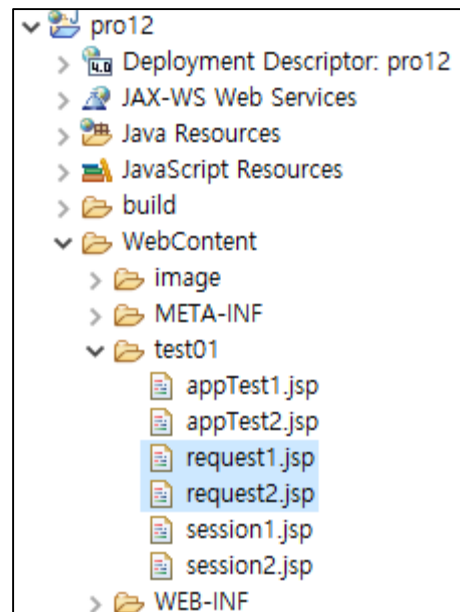
하지만 익스플로러에서는 application의 값에만 접근할 수 있습니다.



12.7내장 객체(내장 변수) 기능

- 12.7.3 request 내장 객체에 데이터 바인딩 실습

1. request 내장 객체 실습 파일인 request1.jsp, request2.jsp를 준비합니다



12.7내장 객체(내장 변수) 기능

2. 첫 번째 JSP인 request1.jsp를 다음과 같이 작성합니다.

코드 12-20 pro12/WebContent/request1.jsp

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

```
<%
    request.setAttribute("name", "이순신");
    request.setAttribute("address", "서울시 강남구");
    %>
```

request 객체에 setAttribute()를 이용해
name과 address를 바인딩합니다.

```
<!DOCTYPE html>
<html>
<head>...</head>
<body>
```

```
<%
    RequestDispatcher dispatch = request.getRequestDispatcher("request2.jsp");
    dispatch.forward(request, response);
    %>
```

request 객체를 다른 JSP로 포워드합니다.

```
</body>
</html>
```

12.7내장 객체(내장 변수) 기능

3. 두 번째 JSP인 request2.jsp를 다음과 같이 작성합니다.

코드 12-21 pro12/WebContent/request2.jsp

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

```
<%  
    String name=(String)request.getAttribute("name");  
    String address=(String )request.getAttribute("address");  
%>
```

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>...</head>
```

```
<body>
```

```
<h1>이름은 <%=name %>입니다.</h1>
```

```
<h1>주소는 <%=address %>입니다.</h1>
```

```
</body>
```

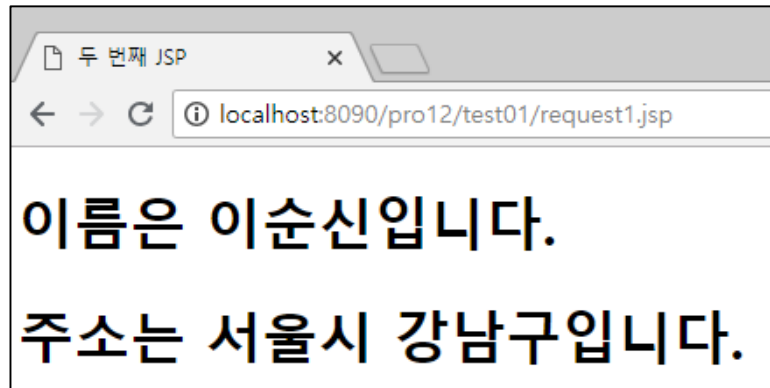
```
</html>
```

첫 번째 JSP 페이지에서 포워딩된 request 객체에서 getAttribute())를 이용해 정보를 가져옵니다.

이전 JSP에서 전송된 정보를 출력합니다.

12.7내장 객체(내장 변수) 기능

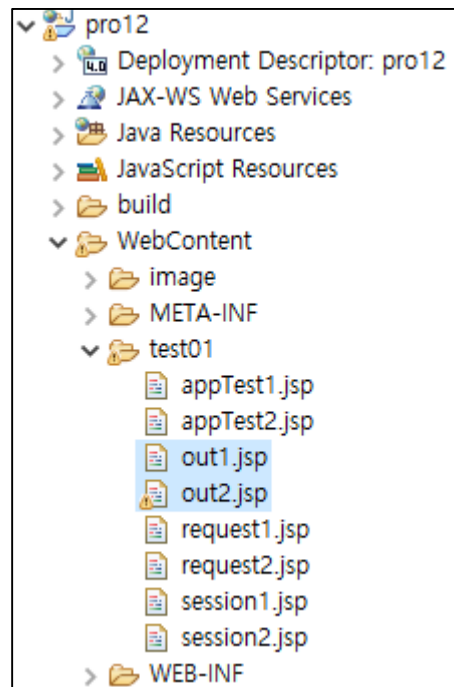
4. 브라우저에서 request1.jsp로 요청하면 request 객체에 바인딩한 후 request2.jsp로 포워딩하여 이름과 주소를 출력합니다.



12.7내장 객체(내장 변수) 기능

- 12.7.4 out 내장 객체 이용해 데이터 출력하기

1. 다음과 같이 실습 파일 out1.jsp, out2.jsp를 준비합니다.



12.7내장 객체(내장 변수) 기능

2. 첫 번째 JSP 페이지인 out1.jsp를 작성합니다. 이름과 나이를 두 번째 JSP로 전송합니다.

코드 12-22 pro12/WebContent/test01/out1.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>...</head>
<body>
    <form method="post" action="out2.jsp">
        이름:<input type="text" name="name"><br>
        나이: <input type="text" name="age"><br>
        <input type="submit" value="전송">
    </form>
</body>
</html>
```

12.7 내장 객체(내장 변수) 기능

3. 두 번째 JSP 페이지인 out2.jsp를 작성합니다.

코드 12-23 pro12/WebContent/test01/out2.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    request.setCharacterEncoding( "utf-8" );
    String name=request.getParameter("name");
    String age=request.getParameter("age");
%>

<!DOCTYPE html>
<html>
<head>...</head>
<body>
<%
    if(name!=null ||name.length()!=0){
%>
    <h1><%=name %> ,<%=age %> </h1>
<%
    }else{
%>
    <h1>이름을 입력하세요</h1>
```

name과 age의 값을 표현식으로 출력합니다.

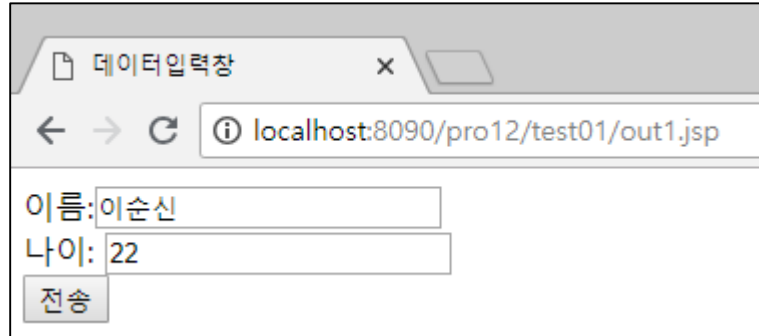
12.7 내장 객체(내장 변수) 기능

```
<%  
    }  
%>  
  
<%  
    if(name!=null ||name.length()!=0){  
%>  
        <h1><% out.println(name+" , "+age); %>.</h1>  
%>  
    }else{  
%>  
        <h1>이름을 입력하세요</h1>  
%>  
    }  
%>  
</body>  
</html>
```

name과 age의 값을 out 내장 객체로 출력합니다.

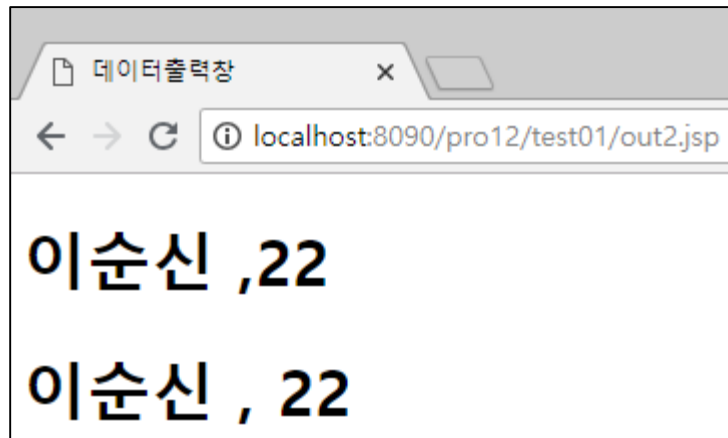
12.7 내장 객체(내장 변수) 기능

4. 브라우저에서 요청하여 이름과 나이를 입력한 후 전송합니다.



A screenshot of a web browser window titled "데이터입력창". The address bar shows "localhost:8090/pro12/test01/out1.jsp". The form contains two input fields: "이름:" with the value "이순신" and "나이:" with the value "22". Below the inputs is a button labeled "전송".

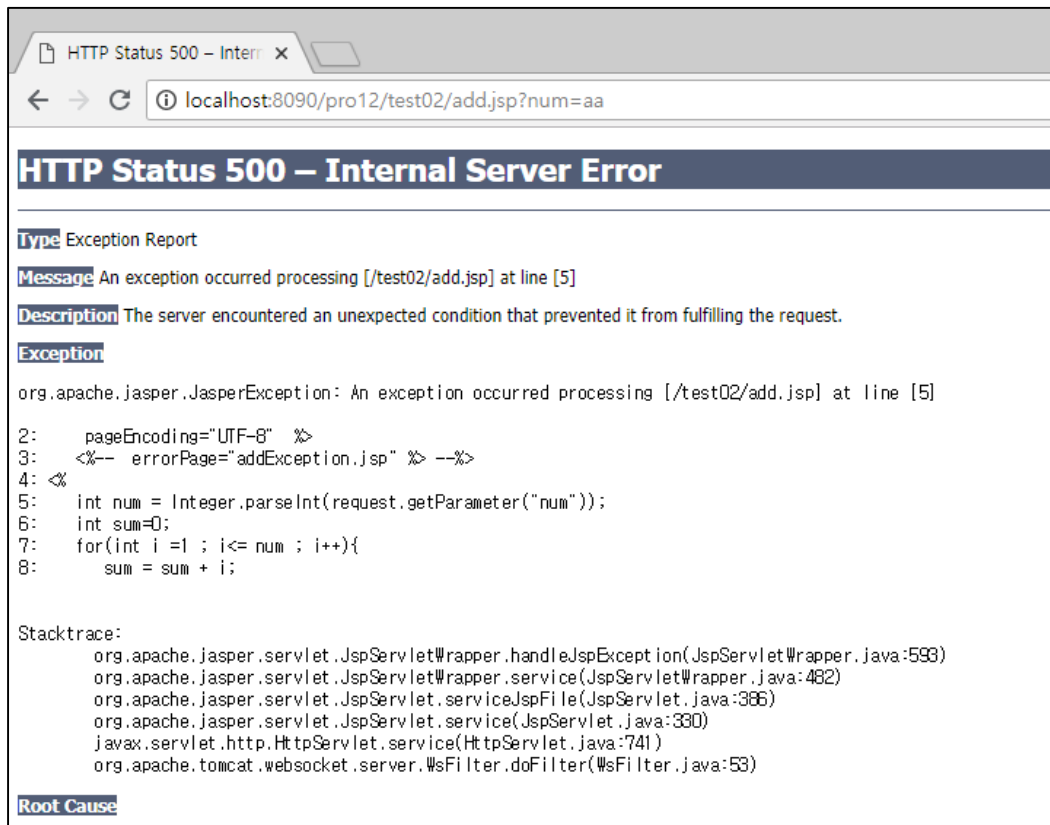
5. 전달받은 정보를 표현식과 out 내장 객체로 출력합니다.



A screenshot of a web browser window titled "데이터출력창". The address bar shows "localhost:8090/pro12/test01/out2.jsp". The page displays two lines of text: "이순신 ,22" and "이순신 , 22".

12.8 JSP 페이지 예외 처리하기

- 사용자 입장에선 예외 발생 시 웹 페이지에 코드 출력 시 사이트에 큰 문제가 발생한 것으로 인식함
- 예외 처리 전용 페이지로 예외 처리 시 신뢰 있고 친화적인 웹 페이지가 가능



```
HTTP Status 500 – Internal Server Error

Type Exception Report
Message An exception occurred processing [/test02/add.jsp] at line [5]
Description The server encountered an unexpected condition that prevented it from fulfilling the request.
Exception
org.apache.jasper.JasperException: An exception occurred processing [/test02/add.jsp] at line [5]
2:   pageEncoding="UTF-8" %>
3:   <%-- errorPage="addException.jsp" %> --%>
4: <%
5:   int num = Integer.parseInt(request.getParameter("num"));
6:   int sum=0;
7:   for(int i=1 ; i<= num ; i++){
8:     sum = sum + i;

Stacktrace:
org.apache.jasper.servlet.JspServletWrapper.handleJspException(JspServletWrapper.java:593)
org.apache.jasper.servlet.JspServletWrapper.service(JspServletWrapper.java:482)
org.apache.jasper.servlet.JspServlet.serviceJspFile(JspServlet.java:386)
org.apache.jasper.servlet.JspServlet.service(JspServlet.java:330)
javax.servlet.http.HttpServlet.service(HttpServlet.java:741)
org.apache.tomcat.websocket.server.WsFilter.doFilter(WsFilter.java:53)

Root Cause
```

12.8 JSP 페이지 예외 처리하기

- 12.8.1 JSP 페이지 예외 처리 과정

- JSP 예외 처리 페이지 만드는 과정

① 예외 처리 담당 JSP를 만듭니다.

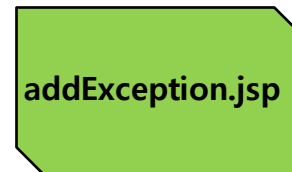
```
<%@ page isErrorPage='true' %>
```



② 예외 발생 시 예외처리 담당 JSP 파일을 지정합니다.

```
<%@ page errorPage='addException.jsp' %>
```

- 실습예제 예외 처리 과정

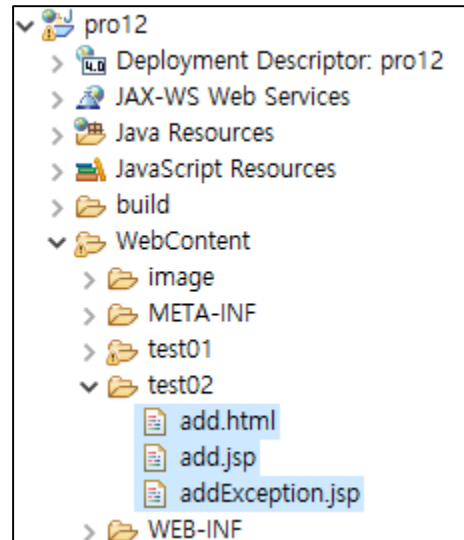


exception 내장 객체를 사용해서
예외 처리를 합니다.

12.8 JSP 페이지 예외 처리하기

- 12.8.2 JSP 페이지 예외 처리 실습

1. 실습을 위해 WebContent 아래 test02 폴더를 만들고 다음과 같이 add.html, add.jsp, addException.jsp 파일들을 준비합니다.



12.8 JSP 페이지 예외 처리하기

2. add.html을 다음과 같이 작성합니다. 입력창에서 숫자를 입력 받아 action에 지정한 add.jsp로 전송합니다.

코드 12-24 pro12/WebContent/test02/add.html

```
<!DOCTYPE html>
<html>
<head>
  <title>합계</title>
</head>
<body>
  자연수를 입력하세요.
  <form action='add.jsp' >————— 입력한 값을 add.jsp로 전송합니다.
    1부터 <input type='text' name='num'>
    <input type='submit' value='계산하기'>
  </form>
</body>
</html>
```

12.8 JSP 페이지 예외 처리하기

3. add.jsp를 다음과 같이 작성합니다.

코드 12-25 pro12/WebContent/test02/add.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"
    errorPage="addException.jsp" %>
<%
    int num = Integer.parseInt(request.getParameter("num"));
    int sum=0;
    for(int i =1 ; i<= num ; i++){
        sum = sum + i;
    }
%>

<!DOCTYPE html>
<html>
<head>
    <title>합계 구하기</title>
</head>
<body>
    <h2>합계 구하기</h2>
    <h1>1부터 <%=num %>까지의 합은 <%=sum %>입니다</h1>
</body>
</html>
```

예외 발생 시 예외를 처리할 JSP 페이지를 지정합니다.

12.8 JSP 페이지 예외 처리하기

4. 또 다른 JSP 페이지인 addException.jsp를 다음과 같이 작성합니다.

코드 12-26 pro12/WebContent/test02/addException.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"
    isErrorPage="true"%>
<!DOCTYPE html>
<html>
<head>
    <title>에러 페이지</title>
</head>
<body>
    ===== toString() 내용 ===== <br>
    <h1><%= exception.toString() %></h1>
    ===== getMessage()내용 =====<br>
    <h1><%=exception.getMessage()%></h1>
    ===== printStackTrace() 내용 =====<br>
    <h1><%= exception.printStackTrace(); %></h1>
    <h3>
        숫자만 입력 가능합니다. 다시 시도하세요.
        <a href='add.html'>다시 하기</a>
    </h3>
</body>
```

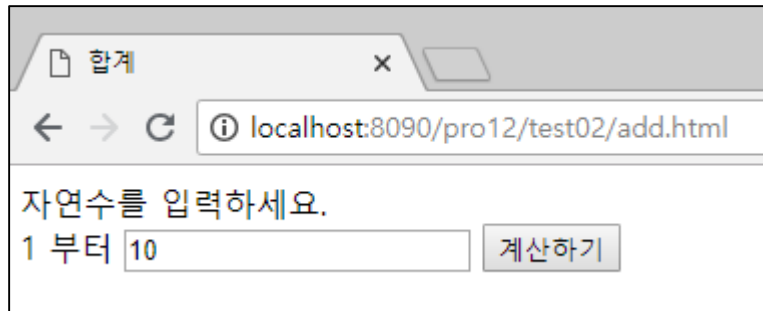
다른 JSP 페이지에서 예외 발생 시 예외를 처리하는 예외 페이지로 지정합니다.

exception 내장 객체를 사용해 예외 처리를 합니다.

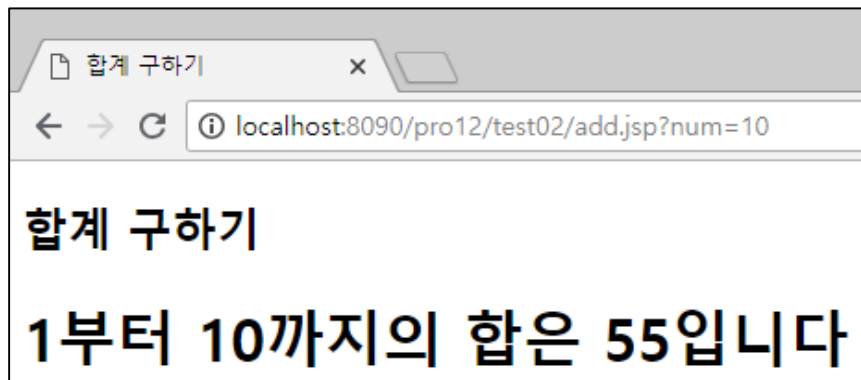
이클립스 콘솔로 예외 메시지를 출력합니다.

12.8 JSP 페이지 예외 처리하기

5. `http://localhost:8090/pro12/test02/add.html`로 요청하여 입력창에 정상적인 숫자를 입력한 후 계산하기를 클릭합니다.

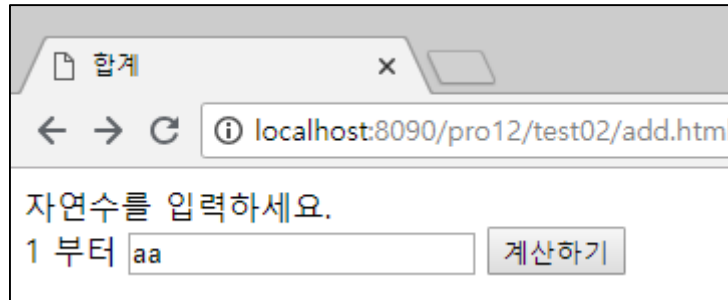


6. 정상적인 결과가 출력됩니다.



12.8 JSP 페이지 예외 처리하기

7. 이번에는 문자를 입력해 볼까요?

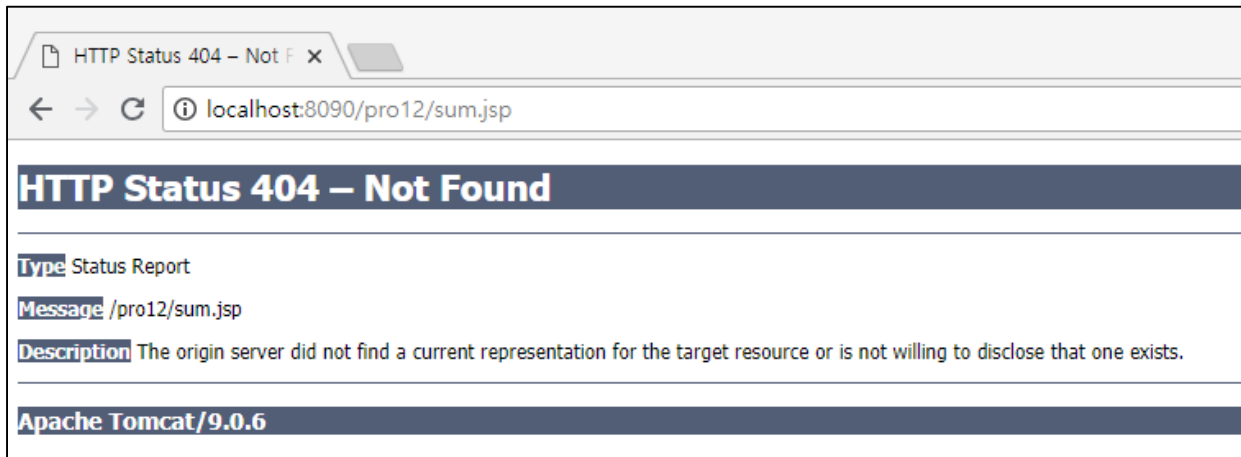


8. 문자는 처리 시 예외가 발생합니다. 다음과 같이 예외 처리 페이지에서 예외를 처리합니다.

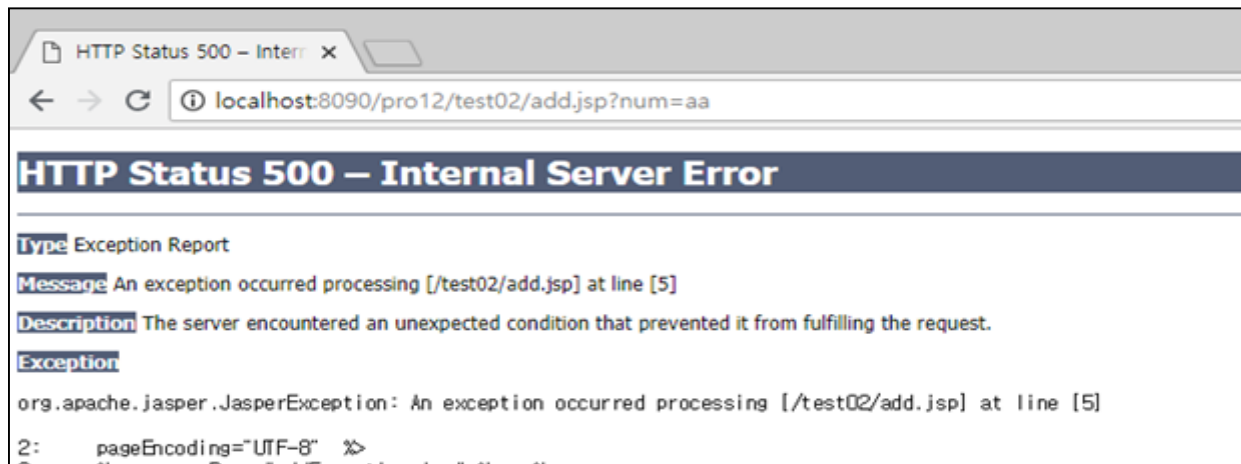


12.8 JSP 페이지 예외 처리하기

- 12.8.3 JSP 페이지의 오류 페이지 종류
 - 존재하지 않는 sum.jsp 요청 시 표시되는 404 에러



- 컨테이너 처리 중 JSP에서 에러 발생 시 표시하는 500 에러



12.8 JSP 페이지 예외 처리하기

- 12.8.4 에러 코드에 따른 예외 페이지 지정

- 전체 JSP 페이지에 대해 발생하는 오류에 따라서 화면에 표시되는 각각의 예외 처리 JSP 페이지를 지정함

web.xml에 지정 방법

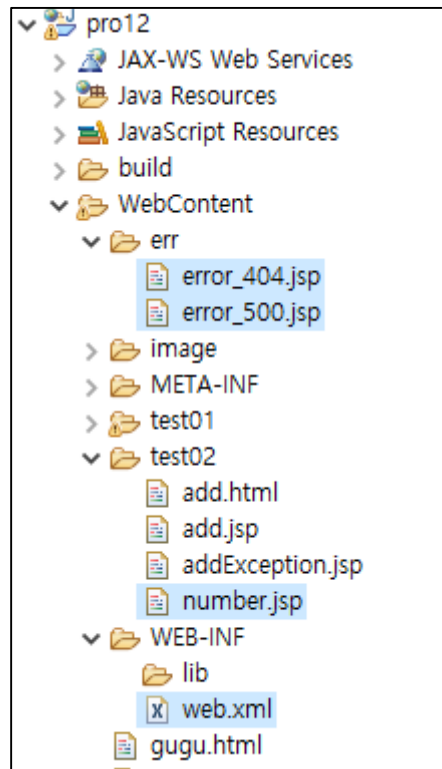
코드 web.xml

```
<error-page>  
  <error-code>에러코드</error-code>  
  <location>오류 페이지 위치</location>  
</error-page>
```


12.8 JSP 페이지 예외 처리하기

web.xml에 예외 페이지 지정하기 실습

1. WebContent 하위에 오류 페이지들이 위치할 err 폴더를 만들고 error_404.jsp, error_500.jsp 파일을 준비합니다.



12.8 JSP 페이지 예외 처리하기

2. web.xml에 <error-page> 태그를 이용해 각각의 에러 코드에 대해 처리할 오류 페이지가 있는 경로를 지정합니다.

코드 12-27 pro12/WebContent/WEB-INF/web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.
jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
```

```
    <error-page>
        <error-code>404</error-code>
        <location>/err/error_404.jsp</location>
    </error-page>

    <error-page>
        <error-code>500</error-code>
        <location>/err/error_500.jsp</location>
    </error-page>
</web-app>
```

404와 500 오류 발생 시 예외 처리를 할 페이지를 지정합니다.

12.8 JSP 페이지 예외 처리하기

3. 404 오류를 처리하는 JSP 페이지인 error_404.jsp를 다음과 같이 작성합니다.

코드 12-28 pro12/WebContent/err/error_404.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>404 예외 처리 페이지</title>
</head>
<body>
    <h1>요청한 페이지는 존재하지 않습니다.</h1>
</body>
</html>
```

12.8 JSP 페이지 예외 처리하기

4. 500 오류를 처리하는 JSP 페이지인 error_500.jsp를 다음과 같이 작성합니다.

코드 12-29 pro12/WebContent/err/error_500.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>500 예외 처리 페이지</title>
</head>
<body>
    <br>
    <h1>죄송합니다. 서비스 실행 중 오류가 발생했습니다.</h1>
    <h1>잠시 후 다시 시도해 보세요.</h1>
</body>
</html>
```

한 단계 위에 있는 image 폴더의 이미지를 표시합니다.

12.8 JSP 페이지 예외 처리하기

5. 브라우저 요청 시 예외를 발생시키는 number.jsp를 다음과 같이 작성합니다.

코드 12-30 pro12/WebContent/test02/number.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%
    int num = Integer.parseInt(request.getParameter("num"));
%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>테스트 페이지</title>
</head>

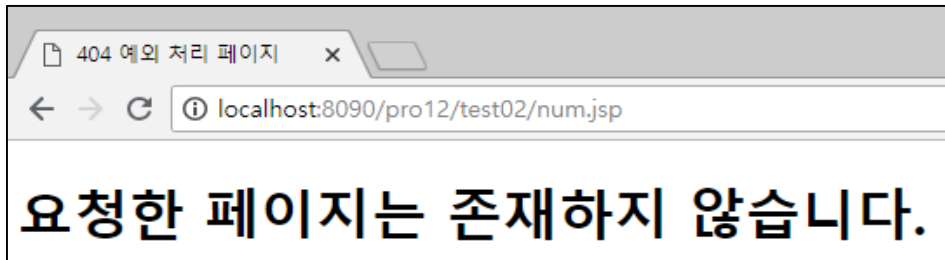
<body>
    <h1>쇼핑몰 중심 JSP 입니다!!!!</h1>
</body>
</html>
```

예외를 강제로 발생시킵니다.

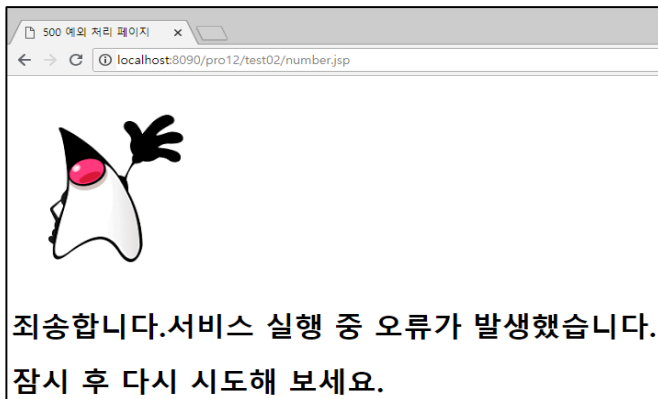
12.8 JSP 페이지 예외 처리하기

6. 이제 각각의 예외를 고의로 발생시켜 볼까요?

먼저 존재하지 않는 `http://localhost:8090/pro12/test02/num.jsp`를 요청한 결과를 확인해 봅시다.



7. 실행 중 예외를 발생시키는 `http://localhost:8090/pro12/test02/number.jsp`를 요청합니다.



❖ 만약 한 개의 JSP 페이지에 페이지 디렉티브의 `errorPage` 속성과 `web.xml`이 같이 지정되어 있으면 페이지 디렉티브의 `errorPage`가 우선적으로 나타납니다.

12.9 JSP welcome 파일 지정하기

welcome 파일 리스트

- web.xml에 웹 애플리케이션의 홈페이지 설정 기능

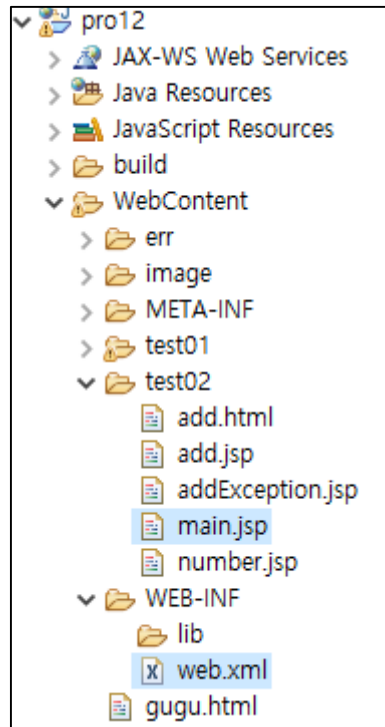
코드 web.xml

```
<welcome-file-list>
  <welcome-file>jsp 또는 html 파일 이름1</welcome-file>
  <welcome-file>jsp 또는 html 파일 이름2</welcome-file>
  ...
</welcome-file-list>
```

12.9 JSP welcome 파일 지정하기

welcome 파일 실습

1. 다음과 같이 test02 폴더 하위에 main.jsp 파일과 web.xml 파일을 준비합니다.



12.9 JSP welcome 파일 지정하기

2. web.xml에 <welcome-file-list> 태그 경로를 포함하여 홈페이지에 해당하는 파일들을 나열합니다.

코드 12-31 pro12/WebContent/WEB-INF/web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.
jcp.org/xml/ns/javaee" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd" id="WebApp_ID" version="3.1">
  <welcome-file-list>
    <welcome-file>/test02/main.jsp</welcome-file>
    <welcome-file>/test02/add.jsp</welcome-file>
    <welcome-file>/test02/add.html</welcome-file>
  </welcome-file-list>
</web-app>
```

여러 개의 welcome 파일을 지정합니다.

12.9 JSP welcome 파일 지정하기

3. 첫 번째 홈페이지인 main.jsp 페이지를 다음과 같이 작성합니다.

코드 12-32 pro12/WebContent/test02/main.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>홈페이지</title>
</head>
<body>
    <br>
    <h1>안녕하세요</h1>
    <h1>쇼핑몰 중심 JSP 홈페이지 입니다!!!</h1>
</body>
</html>
```

12.9 JSP welcome 파일 지정하기

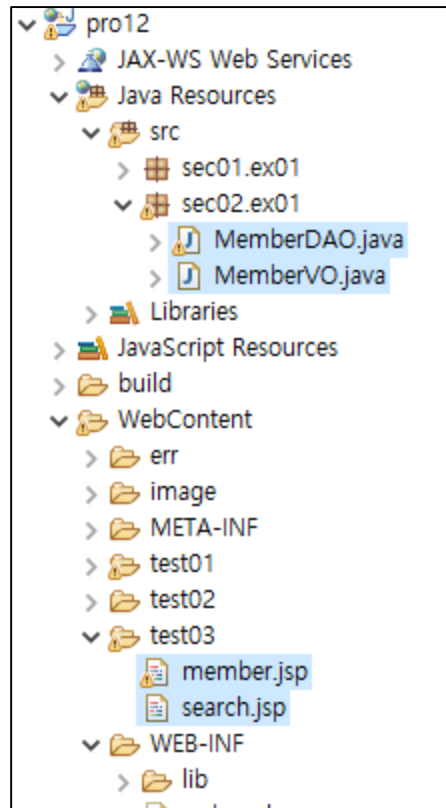
4. 톰캣을 다시 실행한 후 브라우저에서 컨텍스트 이름(/pro12)으로 요청합니다.



- ❖ 개발을 모두 마치고 실제 서비스를 제공할 때는 웹 사이트에 대한 도메인 이름을 구한 후 웹 호스팅 업체에서 제공하는 방법으로 브라우저에서 도메인 이름으로 요청해야 합니다. 그리고 다시 컨텍스트 이름으로 재요청하도록 설정하면 됩니다.

12.10 스크립트 요소 이용해 회원 정보 조회

1. sec02.ex01 패키지를 생성한 후 MemberVO, MemberDAO 클래스를 복사해 붙여 넣습니다.
그리고 test03 폴더에 member.jsp, search.jsp 파일을 추가합니다.



12.10 스크립트 요소 이용해 회원 정보 조회

2. 데이터베이스의 회원을 조회하는 JSP 페이지인 search.jsp를 다음과 같이 작성합니다.

코드 12-33 pro12/WebContent/test03/search.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="UTF-8">
    <title>회원 검색창</title>
</head>
<body>
    <form method="post" action="member.jsp">
        이름:<input type="text" name="name"><br>
        <input type="submit" value="조회하기">
    </form>
</body>
</html>
```

이름을 member.jsp로 전송합니다.

12.10 스크립트 요소 이용해 회원 정보 조회

3. member.jsp를 다음과 같이 작성합니다.

코드 12-34 pro12/WebContent/test03/member.jsp

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    import="java.util.*"
    import="sec02.ex01.*"
    pageEncoding="UTF-8"
%>
<!DOCTYPE html>
<html>
<head>
<style>
    h1 {
        text-align: center;
    }
</style>
<meta charset="UTF-8">
<title>회원 정보 출력창</title>
</head>

<body>
    <h1>회원 정보 출력</h1>
```

〈h1〉 태그의 텍스트를 중앙에 정렬합니다.

12.10 스크립트 요소 이용해 회원 정보 조회

<%

request.setCharacterEncoding("utf-8");

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

전송된 이름을 가져옵니다.

MemberVO memberVO = new MemberVO();

memberVO.setName(_name);

MemberDAO dao=new MemberDAO();

List membersList=dao.listMembers(memberVO);

memberVO를 listMembers() 메서드로 전달하여 조회 조건에 해당하는 회원 정보를

%>

<table border=1 width=800 align=center>

<tr align=center bgcolor="#FFFF66">

<td>아이디</td>

<td>비밀번호</td>

<td>이름</td>

<td>이메일</td>

<td>가입일자</td>

</tr>

<%

for (int i=0; i < membersList.size(); i++){

MemberVO vo=(MemberVO) membersList.get(i);

String id=vo.getId();

String pwd=vo.getPwd();

String name=vo.getName();

String email=vo.getEmail();

Date joinDate=vo.getJoinDate();

MemberDAO에서 조회한 회원 정보를 for 반복문을 이용해 테이블의 행으로 출력합니다.

%>

12.10 스크립트 요소 이용해 회원 정보 조회

```
<tr align=center>
  <td><%= id %></td>
  <td><%= pwd %></td>
  <td><%= name %></td>
  <td><%= email %></td>
  <td><%= joinDate %></td>
</tr>

<%
}
%>
</table>
</body>
</html>
```

12.10 스크립트 요소 이용해 회원 정보 조회

4. MemberDAO 클래스를 다음과 같이 작성합니다.

코드 12-35 pro12/src/sec02.ex01/MemberDAO.java

```
package sec02.ex01;

...

public class MemberDAO {
    private Connection con;
    private PreparedStatement pstmt;
    private DataSource dataFactory;

    ...

    public List listMembers(MemberVO memberVO) {
        List membersList = new ArrayList();
        String _name=memberVO.getName();
        try {
            con = dataFactory.getConnection();
            String query = "select * from t_member ";
            if((_name!=null && _name.length()!=0)){
                query+=" where name=?";
                pstmt = con.prepareStatement(query);
                pstmt.setString(1, _name);
            }else {
                pstmt = con.prepareStatement(query);
            }
            System.out.println("prepareStatement: " + query);
            ResultSet rs = pstmt.executeQuery();
            while (rs.next()) {
                String id = rs.getString("id");
                String pwd = rs.getString("pwd");
                String name = rs.getString("name");
                String email = rs.getString("email");
                Date joinDate = rs.getDate("joinDate");
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
        return membersList;
    }
}
```

조회할 이름을 가져옵니다.

_name 값이 존재하면 SQL문에 where절을 추가하여 해당 이름으로 조회합니다.

첫 번째 '?'에 전달된 이름을 지정합니다.

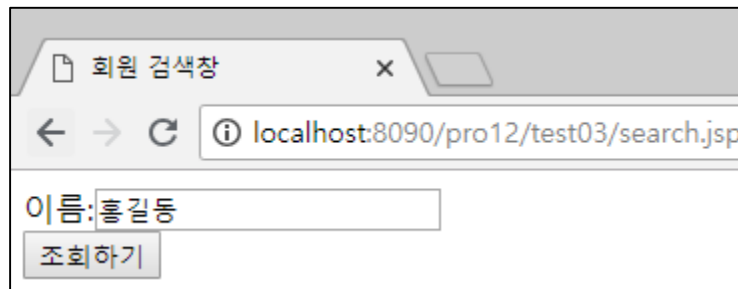
_name 값이 없으면 모든 회원 정보를 조회합니다.

12.10 스크립트 요소 이용해 회원 정보 조회

```
        MemberVO vo = new MemberVO();
        vo.setId(id);
        vo.setPwd(pwd);
        vo.setName(name);
        vo.setEmail(email);
        vo.setJoinDate(joinDate);
        membersList.add(vo);
    }
    rs.close();
    pstmt.close();
    con.close();
} catch (Exception e) {
    e.printStackTrace();
}
return membersList;
}
}
```

12.10 스크립트 요소 이용해 회원 정보 조회

5. `http://localhost:8090/pro12/test03/search.jsp`로 요청한 다음 조회할 이름을 입력하고 `member.jsp`로 전송합니다.



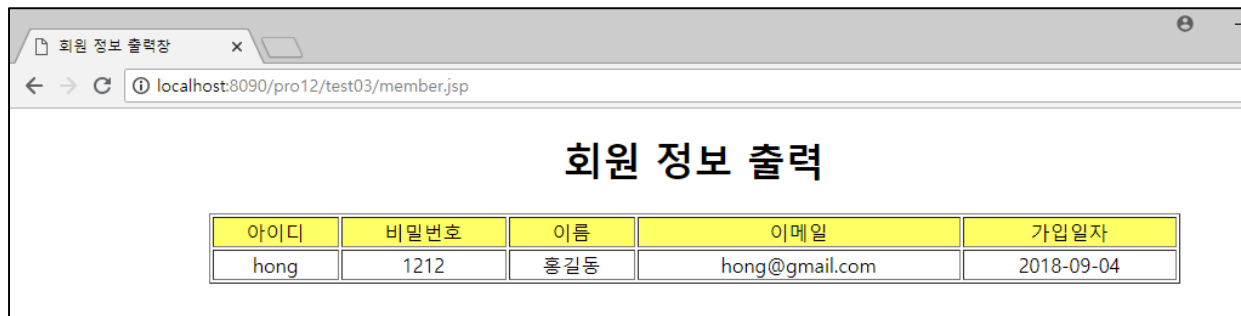
회원 검색창

← → ↻ ⓘ localhost:8090/pro12/test03/search.jsp

이름:홍길동

조회하기

6. 그러면 조회한 회원 정보가 출력됩니다.



회원 정보 출력

아이디	비밀번호	이름	이메일	가입일자
hong	1212	홍길동	hong@gmail.com	2018-09-04

12.10 스크립트 요소 이용해 회원 정보 조회

만약 이름을 입력하지 않고 조회할 경우에는 모든 회원 정보가 출력됩니다.



The screenshot shows a web browser window with the title '회원 정보 출력창' and the address bar displaying 'localhost:8090/pro12/test03/member.jsp'. The main content area is titled '회원 정보 출력' and contains a table with 5 columns: 아이디, 비밀번호, 이름, 이메일, and 가입일자. The table lists four members: hong, lee, kim, and park.

아이디	비밀번호	이름	이메일	가입일자
hong	1212	홍길동	hong@gmail.com	2018-09-04
lee	1212	이순신	lee@test.com	2018-09-04
kim	1212	김유신	kim@jweb.com	2018-09-04
park	1234	박찬호	park@test.com	2018-09-04