프로젝트로 배우는

자바 웹 프로그래밍

Servlet, JSP, JDBC

황희정 지음

Chapter 05. JSP 기본 문법

한빛아카데미 Hanbit Academy, Inc.

목차

03. 액션

1. JSP 액션의 종류

- JSP 액션은 JSP 고유 기능으로 빈즈 클래스 연동 및 동적 페이지 관리를 위한 기능을 제공함.
- **<jsp:**action_name attribute="value" **/>** 형태를 가짐.
- 주로 사용하는 액션은 useBean, get/setProperty 이며 자바 클래스와의 연동을 위해 사용함.

  → 7장. JSP와 자바빈즈

- 액션(Action)은 JSP 주요 구성요소 중 하나로 다음과 같은 기능을 지원한다.
  - JSP 페이지간 흐름 제어
  - 자바 애플릿 지원
  - 자바 빈즈 컴포넌트와 JSP 상호작용 지원
- 특히 useBean 액션은 JSP에서 자바 빈즈 클래스와의 연동을 지원해주는 액션으로 잘 알아둘 필요가 있다.

03. 액션

- 대표적인 액션은 표5-3과 같음.

[표 5-3] 대표적인 액션의 종류

액션	사용 예	기능
include	<jsp:include page="xx.jsp" />	다른 페이지를 현재 페이지에 포함시킨다.
forward	<jsp:forward page="xx.jsp" />	현재 페이지의 제어를 다른 페이지로 전달한다.
useBean	<jsp:useBean scope="page" id="cls" class="xx.MyBean" />	xx패키지의 MyBean 클래스를 cls라는 이름으로 page 범위에서 사용할 것을 선언한다.
setProperty	<jsp:setProperty name="cls" property="xxx" />	useBean으로 선언된 빈즈 클래스의 setxxx( ) 메서드를 호출한다.
액션	**사용 예**	**기능**
getProperty	<jsp:getProperty name="cls" property="xxx" />	useBean으로 선언된 빈즈 클래스의 getxxx( ) 메서드를 호출한다.
plugin	<jsp:plugin type="applet/bean" code="class"> </jsp:plugin>	애플릿이나 빈즈 클래스를 플러그인 형태로 로딩한다.
param	<jsp:param name="user" value="홍길동" />	include, forward 액션에서 사용할 수 있는 파라미터를 설정한다.

03. 액션

2. include 액션

- include 액션은 다른 파일을 불러온다는 측면에서 include 지시어와 개념이 유사
- include 액션은 단순히 페이지를 포함하는 것 뿐만 아니라 포함될 페이지로 **파라미터를 전달하는 것이 가능**함
- include 지시어는 해당 파일을 포함시킨 후 컴파일하는 것에 비해, include 액션은 **실행 시점에서 해당 파일을 호출/실행하여 그 결과를 포함한다**는 점에서 차이가 있음.
- 동적으로 파일들을 핸들링 하기 때문에 과도한 사용은 성능상에 문제를 줄 수 있음
- include 액션은 동적인 페이지 를 포함시킬 경우에 사용하는 것이 좋고, include 지시어는 잘 바뀌지 않는 정적인 페이지를 포함 할 때 사용하는 것이 좋다

```
<jsp:include page="포함할 파일_이름" />
```

```
<jsp:include page="footer.jsp">
    <jsp:param name="email" value="test@test.net" />
    <jsp:param name="tel" value="000-000-0000" />
</jsp:include>
```

03. 액션

- **[실습] include 액션 사용하기(include_action.jsp)**

  - 교재 p.180 ~ 181 참고

- **[실습] include 액션에서 footer.jsp 호출하기(footer.jsp)**

  - 교재 p.181 참고

[그림 5-15] 실행 결과

[예제 5-10] include_action.jsp

include_action.jsp | footer.jsp | include_action.jsp

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

<html>
<head><title>include_action.jsp</title></head>
<body>
<h2>include_action.jsp 에서footer.jsp 호출</h2>
<hr>
include_action.jsp 에서 출력한 메시지 입니다.<BR>
<jsp:include page="footer.jsp">
    <jsp:param name="email" value="test@test.net" />
    <jsp:param name="tel" value="000-000-0000" />
</jsp:include>
</body>
</html>

```

[예제 5-11] footer.jsp

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

<HTML>
<HEAD><TITLE>footer.jsp</TITLE></HEAD>
<BODY>
footer.jsp 에서 출력한 메시지 입니다.
<HR>
<%= request.getParameter("email") %>,
<%= request.getParameter("tel") %>
</BODY>
</HTML>

```

- **전달된 파라미터는 request.getParameter(...) 메소드를 통해서 가져올 수 있다.**
  - request는 JSP 내장 객체로 6장에서 자세하게 다룸

03. 액션

include_action.jsp footer.jsp include_action.jsp

http://localhost:8080/jspbook_lab/ch05/include_action.jsp

include_action.jsp 에서footer.jsp 호출

include_action.jsp 에서 출력한 메시지 입니다.
footer.jsp 에서 출력한 메시지 입니다.

test@test.net, 000-000-0000

include_action[1] - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
<html>
<head><title>include_action.jsp</title></head>
<body>
<h2>include_action.jsp 에서footer.jsp 호출</h2>
<hr>
include_action.jsp 에서 출력한 메시지 입니다.<BR>


<HTML>
<HEAD><TITLE>footer.jsp</TITLE></HEAD>
<BODY>
footer.jsp 에서 출력한  메시지 입니다.
<HR>
test@test.net,
000-000-0000
</BODY>
</HTML>
</body>
</html>
```

03. 액션

3. forward 액션

- forward 액션은 include 액션과 사용법은 유사하지만 **요청 페이지를 다른 페이지로 전환할 때** 사용
- response 내장객체의 sendRedirect()와 유사하지만 포워드된 페이지에 **파라미터를 전달할 수 있다**는 점에서 차이가 있음
- 브라우저 URL 창에는 최초 요청 페이지가 표시 되기 때문에 처리 페이지 정보를 숨기거나 MVC 패턴의 컨트롤러와 같이 특정 기능 수행 후 다른 페이지로 이동해야 하는 경우 유용

```
<jsp:forward page="포워딩할 파일_이름" />
```

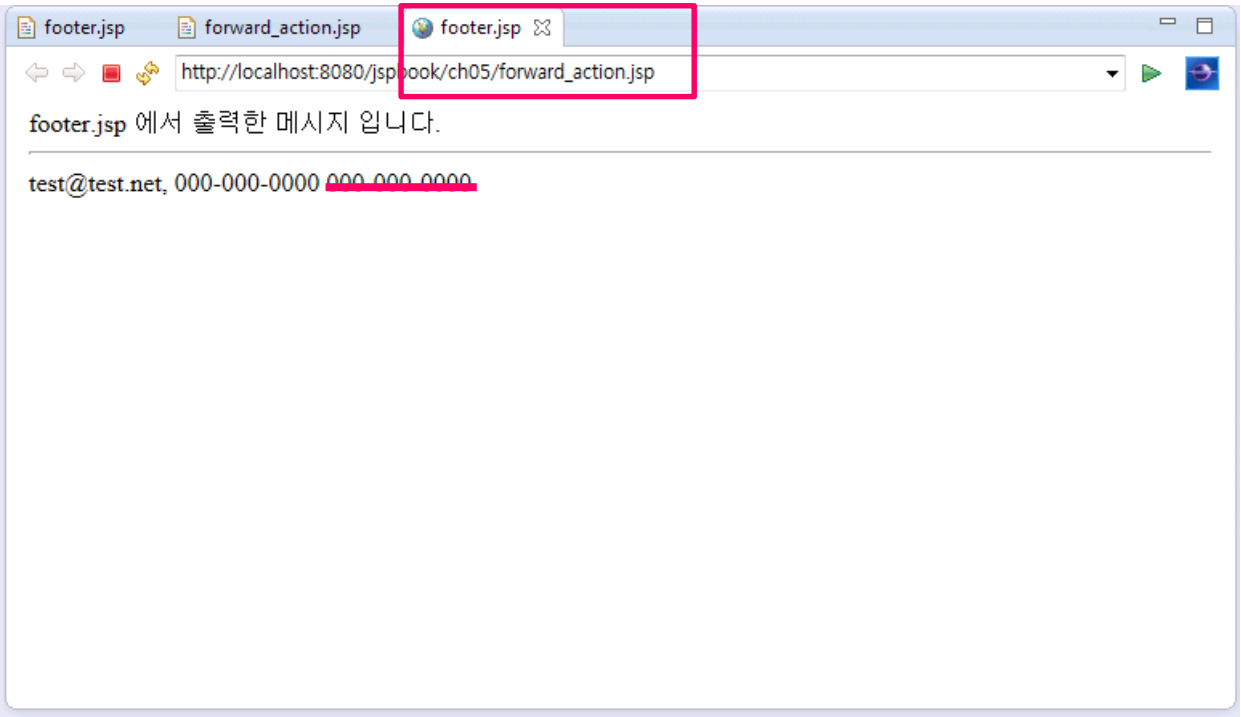
- 사용 예)

```
<jsp:forward page="footer.jsp" />
<jsp:param name="email" value="test@test.net" />
<jsp:param name="tel" value="000-000-0000" />
</jsp:forward>
```

03. 액션

[실습] forward 액션 사용하기(forward_action.jsp)

- 교재 p.182 ~ 183 참고



[그림 5-16] forward_action.jsp 실행 결과

[예제 5-12] forward_action.jsp

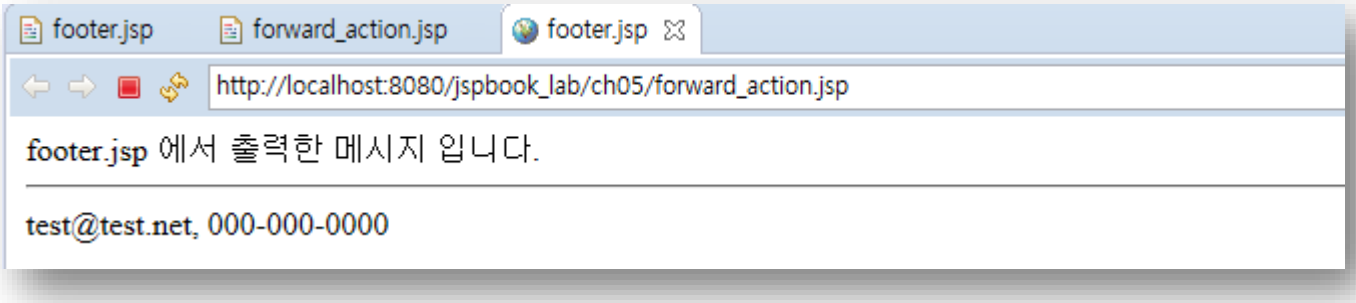
footer.jsp | forward_action.jsp | footer.jsp

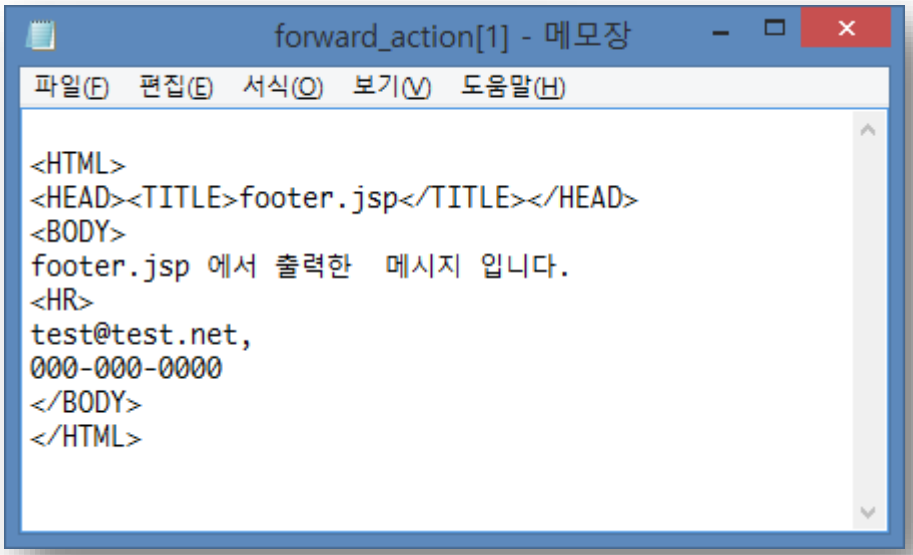
```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8" %>

<html>
<head><title>forwared_action.jsp</title></head>
<body>
<h2>forward_action.jsp 에서 footer.jsp 호출</h2>
<hr>
forward_action.jsp에서 호출한 메시지입니다. <br>
<jsp:forward page="footer.jsp">
    <jsp:param name="email" value="test@test.net" />
    <jsp:param name="tel" value="000-000-0000" />
</jsp:forward>
</body>
</html>

```

Forward Action (실행 결과)





include directive vs. include action

scriptlet1.jsp | include_test.jsp | include_directive.jsp | include directive | include_action.jsp

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

<html>
<head><title>scriptlet1</title></head>
<body>
<div align="center">
<H2>스크립트릿 테스트1 : 1-10까지 출력</H2>
<HR>
<%

    for(int i=1;i<=10;i++) {
        out.println(i+"<BR>");
    }

%>
</div>
</body>
</html>
```

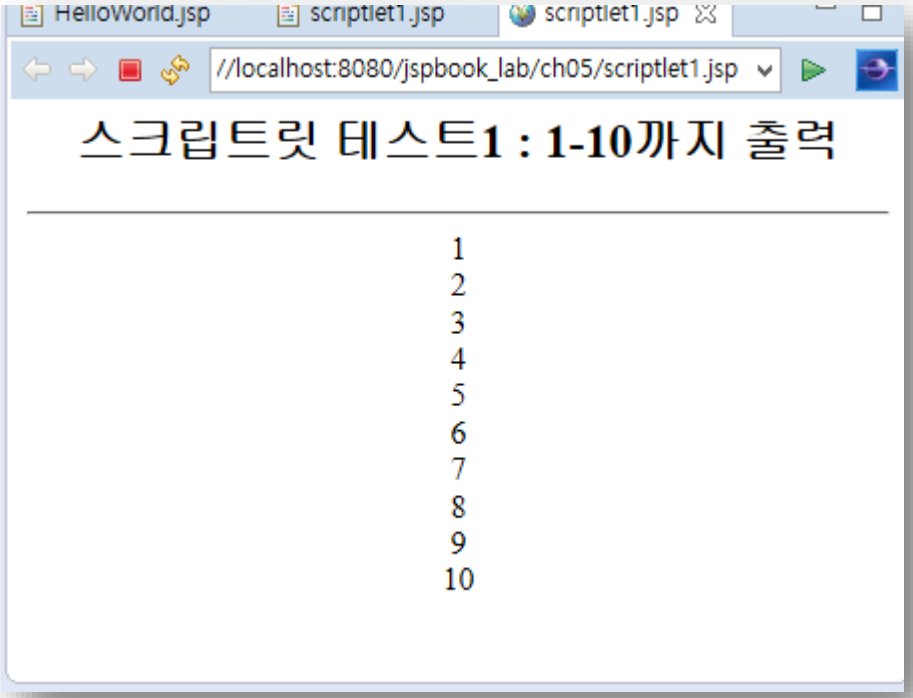
HelloWorld.jsp | scriptlet1.jsp | scriptlet1.jsp

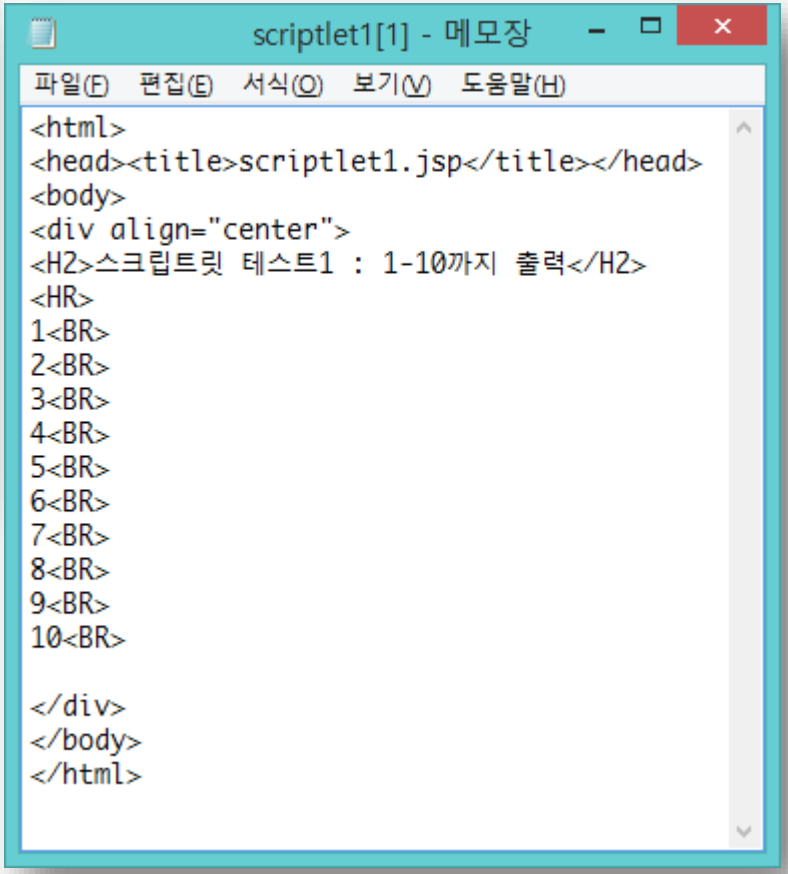
//localhost:8080/jspbook_lab/ch05/scriptlet1.jsp

스크립트릿 테스트1 : 1-10까지 출력

1
2
3
4
5
6
7
8
9
10

scriptlet1.jsp 실행 결과





include directive
vs.
include action

scriptlet1_jsp.java - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
  }

  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("\r\n");
      out.write("<html>\r\n");
      out.write("<head><title>scriptlet1.jsp</title></head>\r\n");
      out.write("<body>\r\n");
      out.write("<div align=\"center\">\r\n");
      out.write("<H2>스크립트릿 테스트1 : 1-10까지 출력</H2>\r\n");
      out.write("<HR>\r\n");


        for(int i=1;i<=10;i++) {
                out.println(i+"<BR>");
        }


      out.write("\r\n");
      out.write("</div>\r\n");
      out.write("</body>\r\n");
      out.write("</html>\r\n");
    } catch (java.lang.Throwable t) {
      if (!(t instanceof javax.servlet.jsp.SkipPageException)){
```

include directive vs. include action

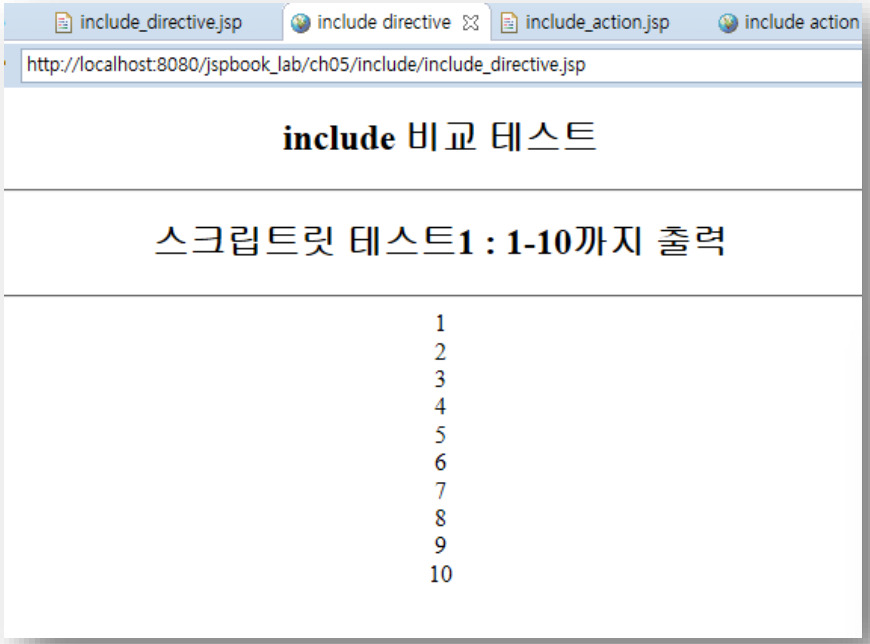
scriptlet1.jsp | include_directive.jsp | include_action.jsp

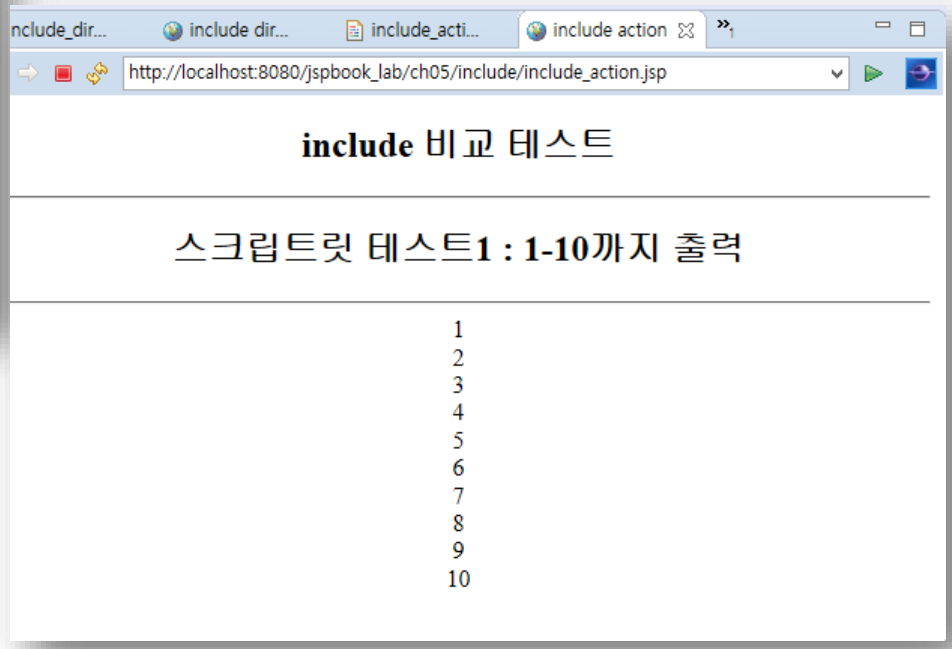
```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<html>
<head>
<title>include directive</title>
</head>
<body>
<div align="CENTER">
<H2> include 비교 테스트</H2>
<HR>
<%@ include file="scriptlet1.jsp" %>
</div>
</body>
</html>
```

scriptlet1.jsp | include_directive.jsp | include_action.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<html>
<head>
<title>include action</title>
</head>
<body>
<div align="CENTER">
<H2> include 비교 테스트</H2>
<HR>
<jsp:include page="scriptlet1.jsp" />
</div>
</body>
</html>
```

include directive vs. include action





include 지시어를 사용한 경우의 서블릿 클래스

include_005fdirective_jsp.java - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
      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=U
      out.write("<title>include directive</title>\r\n");
      out.write("</head>\r\n");
      out.write("<body>\r\n");
      out.write("<div align=\"CENTER\">\r\n");
      out.write("<H2> include 지시어 테스트</H2>\r\n");
      out.write("<HR>\r\n");
      out.write("\r\n");
      out.write("\r\n");
      out.write("<html>\r\n");
      out.write("<head><title>scriptlet1</title></head>\r\n");
      out.write("<body>\r\n");
      out.write("<div align=\"center\">\r\n");
      out.write("<H2>스크립트릿 테스트1 : 1-10까지 출력</H2>\r\n");
      out.write("<HR>\r\n");


        for(int i=1;i<=10;i++) {
                out.println(i+"<BR>");
        }


      out.write("\r\n");
      out.write("</div>\r\n");
      out.write("</body>\r\n");
      out.write("</html>\r\n");
      out.write("\r\n");
      out.write("</div>\r\n");
      out.write("</body>\r\n");
      out.write("</html>\r\n");
    } catch (java.lang.Throwable t) {
```

include_directive[1] - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
<html>
<head>
<title>include directive</title>
</head>
<body>
<div align="CENTER">
<H2> include 비교 테스트</H2>
<HR>


<html>
<head><title>scriptlet1</title></head>
<body>
<div align="center">
<H2>스크립트릿 테스트1 : 1-10까지 출력</H2>
<HR>
1<BR>
2<BR>
3<BR>
4<BR>
5<BR>
6<BR>
7<BR>
8<BR>
9<BR>
10<BR>

</div>
</body>
</html>

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

include 액션을 이용한 경우의 서블릿 클래스

include_005faction_jsp.java - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
      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>include action</title>\r\n");
      out.write("</head>\r\n");
      out.write("<body>\r\n");
      out.write("<div align=\"CENTER\">\r\n");
      out.write("<H2> include 지시어 테스트</H2>\r\n");
      out.write("<HR>\r\n");
      org.apache.jasper.runtime.JspRuntimeLibrary.include(request, response, "scriptlet1.jsp", out, false);
      out.write("\r\n");
      out.write("</div>\r\n");
      out.write("</body>\r\n");
      out.write("</html>\r\n");
    } catch (java.lang.Throwable t) {
      if (!(t instanceof javax.servlet.jsp.SkipPageException)){
        out = _jspx_out;
        if (out != null && out.getBufferSize() != 0)
          try {
            if (response.isCommitted()) {
              out.flush();
            } else {
              out.clearBuffer();
            }
          } catch (java.io.IOException e) {}
        if (_jspx_page_context != null) _jspx_page_context.handlePageException(t);
        else throw new ServletException(t);
      }
```

include_action[1] - 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

```
<html>
<head>
<title>include action</title>
</head>
<body>
<div align="CENTER">
<H2> include 비교 테스트</H2>
<HR>

<html>
<head><title>scriptlet1</title></head>
<body>
<div align="center">
<H2>스크립트릿 테스트1 : 1-10까지 출력</H2>
<HR>
1<BR>
2<BR>
3<BR>
4<BR>
5<BR>
6<BR>
7<BR>
8<BR>
9<BR>
10<BR>

</div>
</body>
</html>

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

03. 액션

4. plugin 액션

- plugin 액션은 웹 브라우저에서 자바 플러그인을 사용하여 자바 애플릿이나 자바 빈즈 컴포넌트를 실행 할 수 있게 한다.
- plugin 액션을 사용하면 자동으로 <OBJECT> 혹은 <EMBED>와 같은 태그를 통해 애플릿 등을 실행하도록 한다. 일반적으로 애플릿을 사용하는 경우가 드물기 때문에 참고만 하기 바란다.
- 사용법

```
<jsp:plugin type="bean|applet" code="objectCode" codebase="objectCodeBase"
{ align = alignment }
{ archive = archiveList }
{ height = height }
{ hspace = hspace }
{ name = name }
{ vspace = vspace }
{ width = width }
{ nspluginurl = url }
{ iepluginurl = url }
{ <jsp:params>
{<jsp:params name="paramName" value="paramValue" />}
</jsp:params> }
{<jsp:fallback> Plugin S/W를 지원하지 못하는 경우의 설명</jsp:fallback> }
</jsp:plugin>
```

03. 액션

5. useBean 액션

- 액션에서 가장 중요한 부분으로, JSP 빈즈를 다루는 7장에서 자세히 살펴볼 것이다.
- 여기서는 기본 표기 방법만을 살펴보기로 한다.
- 사용법

```
<jsp:useBean id="변수_이름" class="빈즈 클래스_이름"/>
<jsp:setProperty name="변수_이름" property="속성_이름"/>
<jsp:getProperty name="변수_이름" property="속성_이름"/>
```

  - useBean 액션은 빈즈 클래스를 사용하기 위한 구문이며 class 에 지정된 자바 빈즈 클래스를 id 라는 이름으로 사용할 수 있도록 해준다.
  - get/setProperty 액션은 브라우저에서 빈즈 클래스의 멤버 변수로 값을 저장하거나 가져오기 위한 구문이다.
  - get/setProperty는 빈즈 클래스의 getter/setter 메소드와 연동된다.

04. 선언과 표현식

1. 선언

- JSP 페이지에서 메서드나 멤버변수를 선언하기 위한 구문
- JSP 가 서블릿으로 변환된 자바 코드에서는 모든 내용이 _jspService() 메서드에 들어가기 때문에
- JSP 에서 선언한 변수는 로컬변수가 된다.
- 자바에서는 메서드 안에서 다른 메서드를 선언할 수 없기 때문에, 스트립트릿 안에서 메소드를 정의하면 컴파일 에러가 발생하게 된다.
- **<%! %>**는 JSP 페이지에서 이러한 제약 사항 없이 **소속변수와 소속메서드 선언 가능**
- 구조적으로 JSP 에서 자바 코드를 사용하는 것은 권장되지 않기 때문에 선언문의 사용 역시 권장되지 않음
- 사용 예)

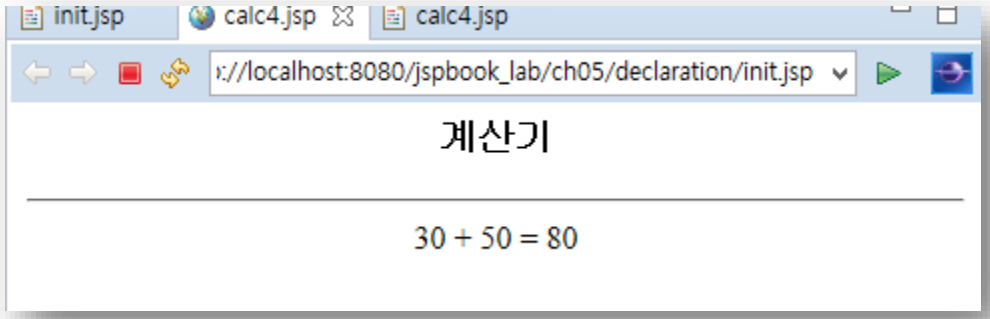
```
<%!
    // 여기에 선언하면 서블릿 클래스의 소속변수/소속메소드로 정의됨
    String str = "test";
    public boolean check() {
        return false;
    }
%>
```

init.jsp

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

<html>
<head><title>init.jsp</title></head>

<body>
<jsp:forward page="calc4.jsp">
    <jsp:param name="num1" value="30" />
    <jsp:param name="num2" value="50" />
    <jsp:param name="operator" value="+" />
</jsp:forward>
</body>
</html>
```



calc4.jsp

init.jsp | calc4.jsp | calc4.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%!
    // 멤버변수 선언
    int num1, num2 = 0;
    String op = "";

    // 연산자별 처리를 위한 멤버 메소드 선언
    public int calculator()  {
        int result = 0;

        if (op.equals("+"))  {
            result = num1 + num2;
        } else if (op.equals("-"))  {
            result = num1 - num2;
        } else if (op.equals("*"))  {
            result = num1 * num2;
        } else if (op.equals("/"))  {
            result = num1 / num2;
        }
        return result;
    }
%>
```

```jsp
<html>
<head><title>calc4.jsp</title></head>
<body>
<div align="CENTER">
<h3>계산기</h3>
<hr>
<%
    // 문자열 형태로 전달된 인자들을 int로 변환함
    String sNum1 = request.getParameter("num1");
    num1 = Integer.parseInt(sNum1);

    String sNum2 = request.getParameter("num2");
    num2 = Integer.parseInt(sNum2);

    op = request.getParameter("operator");

    out.println(num1 +" " + op + " " + num2 + " = " + calculator());

%>
</div>
</body>
</html>
```

calc4_jsp 서블릿 클래스

```
package org.apache.jsp.ch05.declaration;

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

public final class calc4_jsp extends
org.apache.jasper.runtime.HttpJspBase
    implements org.apache.jasper.runtime.JspSourceDependent
{

// 멤버변수 선언
 int num1, num2 = 0;
 String op = "";

 // 연산자별 처리를 위한 멤버 메소드 선언
public int calculator()  {
  int result = 0;

 if (op.equals("+"))  {
  result = num1 + num2;
 } else if (op.equals("-"))  {
  result = num1 - num2;
 } else if (op.equals("*"))  {
  result = num1 * num2;
 } else if (op.equals("/"))  {
  result = num1 / num2;
 }
 return result;
}
```

```
 public void _jspInit() {
 }

 public void _jspDestroy() {
 }

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

   final PageContext pageContext;
   HttpSession session = null;
   final ServletContext application;
   final ServletConfig config;
   JspWriter out = null;
   final Object page = this;
   JspWriter _jspx_out = null;
   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;
```

calc4_jsp 서블릿 클래스

```
out.write('\r');
out.write('\n');
out.write("\r\n");
out.write("\r\n");
out.write("<html>\r\n");
out.write("<head><title>calc4.jsp</title></head>\r\n");
out.write("<body>\r\n");
out.write("<div align=\"CENTER\">\r\n");
out.write("<h3>계산기</h3>\r\n");
out.write("<hr>\r\n");

// 문자열 형태로 전달된 인자들을 int로 변환함
String sNum1 = request.getParameter("num1");
num1 = Integer.parseInt(sNum1);

String sNum2 = request.getParameter("num2");
num2 = Integer.parseInt(sNum2);

op = request.getParameter("operator");

out.println(num1 +" " + op + " " + num2 + " = " + calculator());
```

```
      out.write("\r\n");
      out.write("</div>\r\n");
      out.write("</body>\r\n");
      out.write("</html>");
    } catch (java.lang.Throwable t) {
      if (!(t instanceof SkipPageException)){
        out = _jspx_out;
        if (out != null && out.getBufferSize() != 0)
          try {
            if (response.isCommitted()) {
              out.flush();
            } else {
              out.clearBuffer();
            }
          } catch (java.io.IOException e) {}
        if (_jspx_page_context != null)
            _jspx_page_context.handlePageException(t);
        else throw new ServletException(t);
      }
    } finally {

_jspxFactory.releasePageContext(_jspx_page_context);
    }
  } // end of _jspService
}
```

프로젝트로 배우는

자바 웹 프로그래밍

Servlet, JSP, JDBC

황희정 지음