

5

내장 객체와 액션 태그



이 장에서 다룰 내용

1

내장 객체

2

영역 객체와 속성

3

액션 태그



내장 객체

□ 내장 객체의 개요

- 웹 컨테이너가 제공하는 고정된 이름의 객체
- 공통적으로 요구되는 javax.servlet 패키지 아래 8개의 객체와 예외 처리를 위한 java.lang 패키지 아래 1개의 객체를 각각 JSP 스펙에서 정해진 이름의 객체로 제공

내장 객체 변수명	클래스/인터페이스 타입	설명
request	javax.servlet.ServletException (javax.servlet.http.HttpServletRequest)	클라이언트의 HTTP 요청 정보를 저장한 객체 (HTTP 헤더 정보, 파라미터 등)
response	javax.servlet.ServletResponse (javax.servlet.http.HttpServletResponse)	HTTP 요청에 대한 응답 정보를 저장한 객체
session	javax.servlet.http.HttpSession	클라이언트의 세션 정보를 저장한 객체
pageContext	javax.servlet.jsp.PageContext	페이지 실행에 필요한 컨텍스트 정보를 저장한 객체
out	javax.servlet.jsp.JspWriter	응답 페이지 전송을 위한 출력 스트림 객체
application	javax.servlet.ServletContext	동일한 어플리케이션의 컨텍스트 정보를 저장한 객체
config	javax.servlet.ServletConfig	해당 페이지의 서블릿 설정 정보(초기화 정보)를 저장한 객체
page	java.lang.Object (javax.servlet.jsp.HttpJspPage)	해당 페이지 서블릿 객체(인스턴스)
exception	java.lang.Throwable	예외 처리를 위한 객체



내장 객체

□ request 객체

- 사용자의 요청에 관련된 정보를 얻기 위해 사용하는 객체
- 요청 파라미터와 관련된 메소드들

리턴 타입	메소드명	설명
String	getParameter(String name)	name이란 이름으로 지정된 파라미터에 할당된 값을 리턴한다. 지정된 이름의 파라미터가 없으면 null을 리턴한다.
String[]	getParameterValues(String name)	name이란 이름으로 지정된 파라미터의 모든 값을 String 배열로 리턴한다. 하나의 이름으로 여러 개의 값을 가질 수 있는 checkbox와 같은 태그를 사용했을 때에 주로 사용되며 하나의 이름에 하나의 값만 가지는 파라미터는 getParameter(String name) 메소드를 사용하는 것이 좋다.
Enumeration	getParameterNames()	요청에 포함된 모든 파라미터 이름을 java.util.Enumeration 객체로 리턴한다.



내장 객체

■ 예제

• requestTest1_Form.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <html>
4  <head>
5  <title>Request Test</title>
6  </head>
7  <body>
8  <h1>Request 예제입니다.</h1>
9  <form action="requestTest1.jsp" method="post">
10 <table border="1" width="400">
11     <tr>
12         <td>이름</td>
13         <td><input type="text" name="name"></td>
14     </tr>
15     <tr>
16         <td>성별</td>
17         <td>
18             남<input type="radio" name="gender" value="male">
19             여<input type="radio" name="gender" value="female">
20         </td>
21     </tr>
22     <tr>
```



내장 객체

```
23         <td>취미</td>
24         <td>
25             독서<input type="checkbox" name="hobby" value="독서">
26             게임<input type="checkbox" name="hobby" value="게임">
27             TV시청<input type="checkbox" name="hobby" value="TV시청">
28             축구<input type="checkbox" name="hobby" value="축구">
29             기타<input type="checkbox" name="hobby" value="기타">
30         </td>
31     </tr>
32     <tr>
33         <td colspan="2"><input type="submit" value="전송"></td>
34     </tr>
35 </table>
36 </form>
37 </body>
38 </html>
```



내장 객체

- requestTest1.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <%request.setCharacterEncoding("euc-kr");%>
4  <html>
5  <head>
6  <title>Request Test</title>
7  </head>
8  <body>
9  <h1>Request 예제입니다.</h1>
10 <table border="1" width="400">
11     <tr>
12         <td>이름</td>
13         <td><%=request.getParameter("name") %></td>
14     </tr>
15     <tr>
16         <td>성별</td>
17         <td>
18             <%if(request.getParameter("gender").equals("male")) {%>   남자
19             <%} else {%>여자<%} %>
20         </td>
21     </tr>
22     <tr>
```



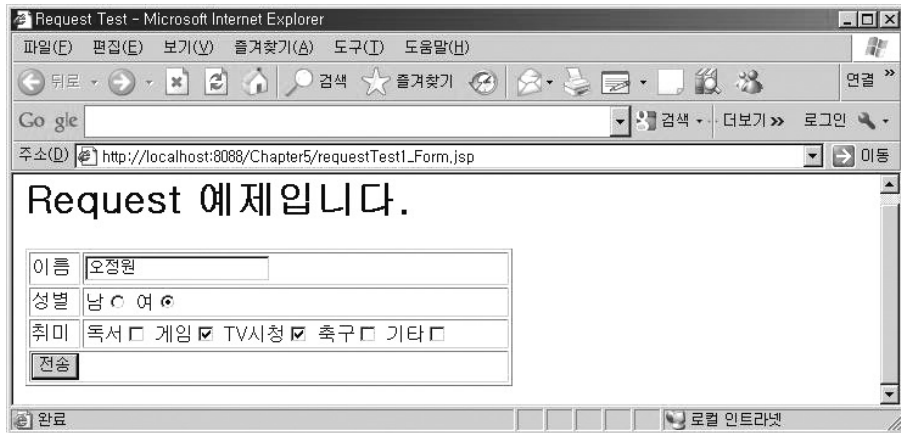
내장 객체

[illegible]

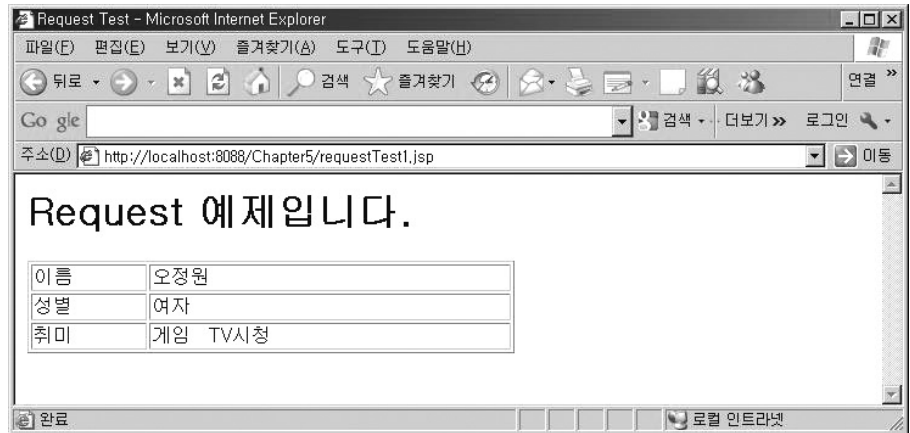
내장 객체

```
25 | </table>
26 | </body>
27 | </html>
```

■ 실행 결과



- requestTest1_Form.jsp -



- requestTest1.jsp -



내장 객체

■ HTTP 헤더 정보와 관련된 메소드들

리턴 타입	메소드명	설명
String	getHeader(String headerName)	HTTP 요청 헤더에 headerName으로 지정된 이름으로 할당된 값을 리턴한다. headerName으로 지정된 이름이 없을 경우 null을 리턴한다.
Enumeration	getHeaders(String headerName)	headerName으로 지정된 이름으로 할당된 모든 값을 java.util.Enumeration 객체로 리턴한다.
Enumeration	getHeaderNames()	HTTP 요청 헤더에 포함된 모든 헤더 이름을 java.util.Enumeration 객체로 리턴한다.
int	getIntHeader(String headerName)	headerName 헤더의 값을 int 타입으로 리턴한다. 지정된 헤더값을 int로 변환할 수 없을 경우에는 NumberFormatException이 발생하고 headerName 헤더가 없을 경우에는 -1을 리턴한다.



내장 객체

■ 예제

• requestTest2.jsp

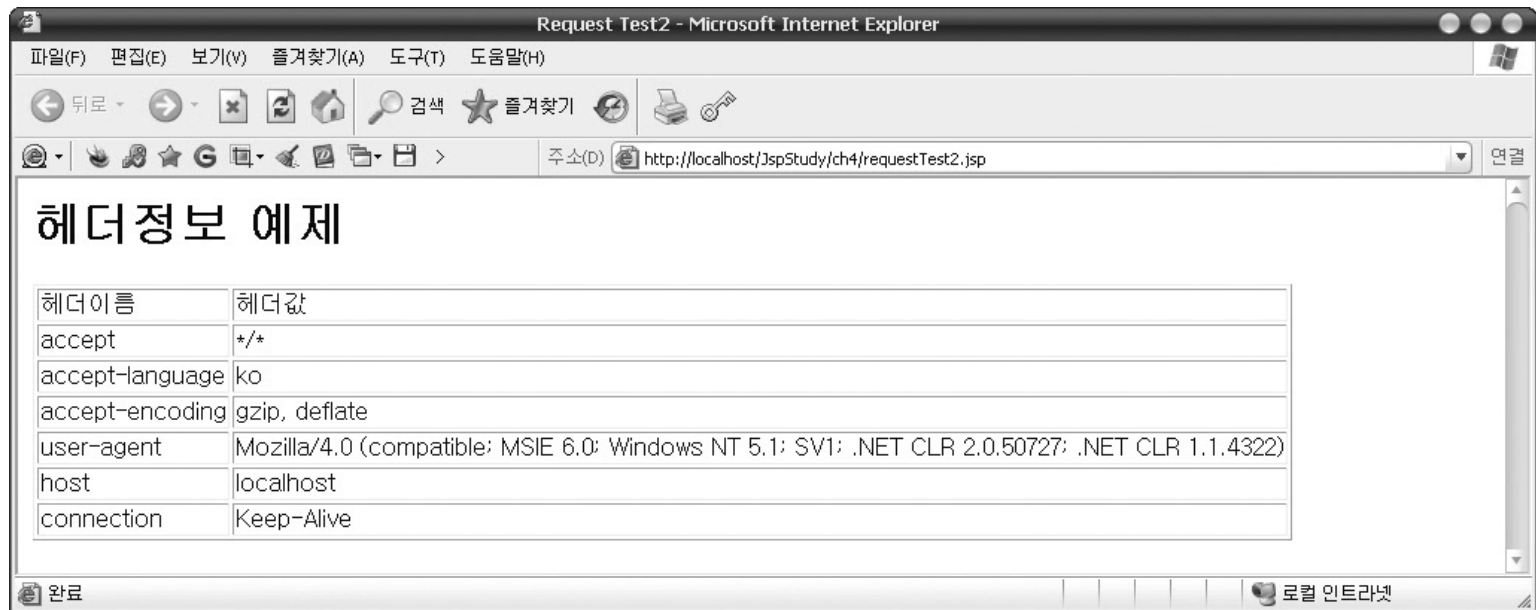
```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <%@page import="java.util.Enumeration"%>  
4  <html>  
5  <head>  
6  <title>Request Test2</title>  
7  </head>  
8  <body>  
9  <h1>헤더정보 예제</h1>  
10 <table border="1">  
11     <tr>  
12         <td>헤더이름</td>  
13         <td>헤더값</td>  
14     </tr>  
15 <%  
16 Enumeration e=request.getHeaderNames();  
17 while(e.hasMoreElements()){  
18     String headerName=(String)e.nextElement();  
19 %>
```



내장 객체

```
20         <tr>
21             <td><%=headerName %></td>
22             <td><%=request.getHeader(headerName) %>
23         </td>
24     <%=}%>
25 </table>
26 </body>
27 </html>
```

■ 실행 결과



내장 객체

■ HTTP 헤더 정보와 관련된 메소드들

리턴 타입	메소드명	설명
HttpSession	getSession()	요청한 클라이언트에 지정된 HttpSession 객체를 반환한다. 이전에 생성된 HttpSession 객체가 없으면 새로운 객체를 생성해 할당한다.
HttpSession	getSession(Boolean create)	create가 true일 경우 getSession() 메소드와 동일한 결과를 리턴하지만 create를 false로 지정하면 이전에 생성된 HttpSession 객체가 없을 경우 null을 리턴한다.
String	getRequestedSessionId()	요청한 클라이언트에 지정된 세션의 ID를 문자열로 리턴한다.
boolean	isRequestedSessionIdValid()	요청에 포함된 클라이언트의 세션 ID가 유효하면 true를, 아니면 false를 리턴한다.



내장 객체

■ 쿠키, URL/URI, 요청 방식과 관련된 메소드들

리턴 타입	메소드명	설명
Cookie[]	getCookies()	HTTP 요청 메시지의 헤더에 포함된 쿠키를 javax.servlet.http.Cookie 배열로 리턴한다.
String	getServerName()	서버의 도메인명을 문자열로 리턴한다.
int	getServerPort()	서버의 포트 번호를 int형으로 리턴한다.
StringBuffer	getReqeustURL()	요청 URL을 StringBuffer로 리턴한다.
String	getRequestURI()	요청 URI를 문자열로 리턴한다.
String	getQueryString()	요청에 사용된 쿼리 문장을 문자열로 리턴한다.
String	getRemoteHost()	클라이언트의 호스트 이름을 문자열로 리턴한다.
String	getRemoteAddr()	클라이언트의 IP 주소를 문자열로 리턴한다.
String	getProtocol()	요청에 사용된 프로토콜 이름을 문자열로 리턴한다.
String	getMethod()	요청에 사용된 요청 방식(GET, POST 등)을 문자열로 리턴한다.
String	getContextPath()	해당 JSP 페이지의 컨텍스트 경로를 문자열로 리턴한다.



내장 객체

■ 예제

• requestTest3.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <html>
4  <head>
5  <title>Request Test3</title>
6  </head>
7  <body>
8  <h1>쿠키, URL/URI, 요청방식에 관련된 정보 예제</h1>
9  <table border="1">
10     <tr>
11         <td>쿠키정보</td>
12     <%
13     Cookie[] cookie=request.getCookies();
14     if(cookie==null){
15     %>
16         <td>쿠키가 존재하지 않습니다</td>
17     <%
18     } else {
19         for(int i=0; i<cookie.length;i++){
20             %>
21             <td><%=cookie[i].getName()%><%=cookie[i].getValue()%>&nbsp;&nbsp;&nbsp;</td>
22             <%
```



내장 객체

```
23     }
24 }
25 %>
26 </tr>
27 <tr>
28     <td>서버 도메인명</td>
29     <td><%=request.getServerName() %>
30 </td>
31 <tr>
32     <td>서버 포트번호</td>
33     <td><%=request.getServerPort() %>
34 </td>
35 <tr>
36     <td>요청 URL</td>
37     <td><%=request.getRequestURL() %>
38 </td>
39 <tr>
40     <td>요청 URI</td>
41     <td><%=request.getRequestURI() %>
42 </td>
43 <tr>
44     <td>요청 쿼리</td>
```



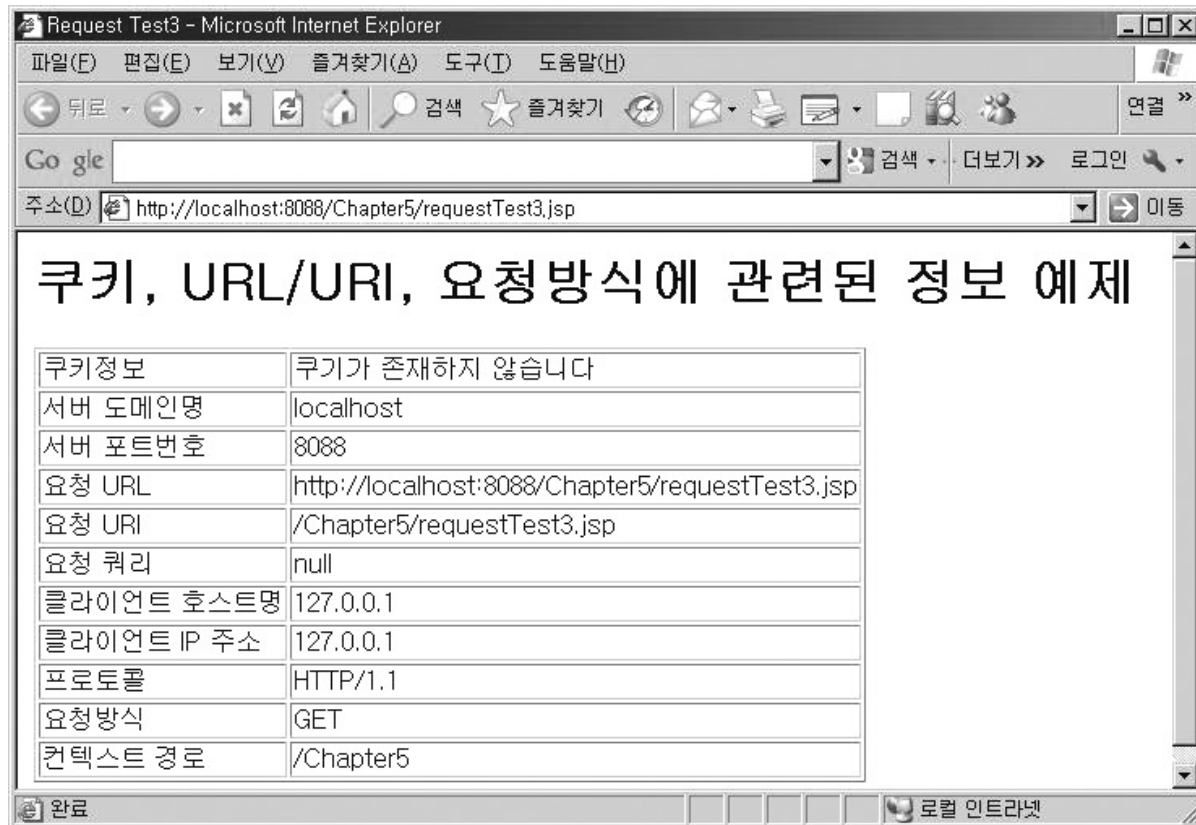
내장 객체

```
46         <td><%=request.getQueryString() %>
47     </td>
48 <tr>
49     <td>클라이언트 호스트명</td>
50     <td><%=request.getRemoteHost() %>
51 </td>
52 <tr>
53     <td>클라이언트 IP 주소</td>
54     <td><%=request.getRemoteAddr() %>
55 </td>
56 <tr>
57     <td>프로토콜</td>
58     <td><%=request.getProtocol() %>
59 </td>
60 <tr>
61     <td>요청방식</td>
62     <td><%=request.getMethod() %>
63 </td>
64 <tr>
65     <td>컨텍스트 경로</td>
66     <td><%=request.getContextPath() %>
67 </td>
68 </table>
69 </body>
70 </html>
```



내장 객체

■ 실행결과



Request Test3 - Microsoft Internet Explorer

주소(D) http://localhost:8088/Chapter5/requestTest3.jsp

쿠키, URL/URI, 요청방식에 관련된 정보 예제

쿠키정보	쿠키가 존재하지 않습니다
서버 도메인명	localhost
서버 포트번호	8088
요청 URL	http://localhost:8088/Chapter5/requestTest3.jsp
요청 URI	/Chapter5/requestTest3.jsp
요청 쿼리	null
클라이언트 호스트명	127.0.0.1
클라이언트 IP 주소	127.0.0.1
프로토콜	HTTP/1.1
요청방식	GET
컨텍스트 경로	/Chapter5

완료 로컬 인트라넷



내장 객체

□ response 객체

- 클라이언트의 요청에 대한 HTTP 응답(HTTP Response)을 나타내는 객체
- 관련 메서드

리턴 타입	메소드명	설명
없음	setHeader(String headerName, String value)	응답에 포함될 헤더 정보에 headerName의 이름으로 value 값을 설정해 추가한다.
없음	addCookie(Cookie cookie)	javax.servlet.http.Cookie 형식의 쿠키를 응답 헤더에 추가한다. 쿠키에 대해서는 Chapter 8에서 자세히 설명하도록 하겠다.
없음	sendRedirect(String url)	지정된 URL로 요청을 재전송한다.
없음	setContentType(String type)	응답 페이지의 contentType을 설정한다.



내장 객체

■ 예제

• responseTest1.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <%  
4      response.sendRedirect("responseTest2.jsp");  
5  %>
```



내장 객체

■ 예제

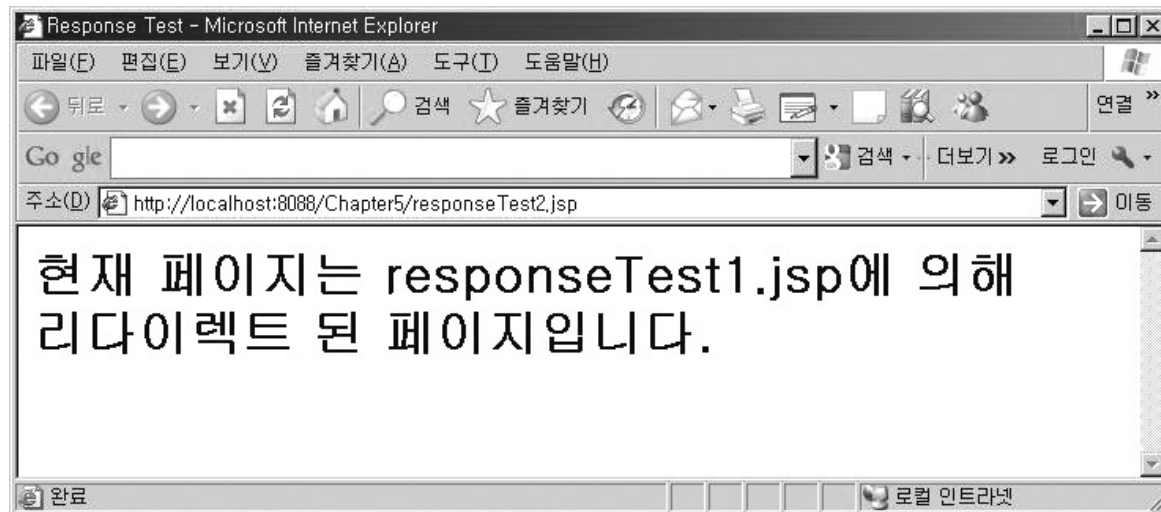
• responseTest2.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <html>
4  <head>
5  <title>Response Test</title>
6  </head>
7  <body>
8  <h1>
9  현재 페이지는 responseTest1.jsp에 의해<br>
10 리다이렉트 된 페이지입니다.
11 </h1>
12 </body>
13 </html>
```



내장 객체

■ 실행 결과



내장 객체

□ pageContext 객체

- JSP 페이지와 관련된 프로그램에서 다른 내장 객체를 얻어내거나 현재 페이지의 요청과 응답의 제어권을 다른 페이지로 넘겨주는 데 사용
- request, session, application과 같은 내장 객체의 속성을 제어
- 관련 메서드

리턴 타입	메소드명	설명
ServletRequest	getRequest()	클라이언트의 요청 정보를 담고 있는 객체를 리턴한다(request 내장 객체를 리턴한다).
ServletResponse	getResponse()	요청에 대한 응답 객체를 리턴한다(response 내장 객체를 리턴한다).
JspWriter	getOut()	응답 출력 스트림을 리턴한다(out 내장 객체를 리턴한다).
Object	getPage()	서블릿 인스턴스 객체를 리턴한다(page 내장 객체를 리턴한다).
ServletConfig	getServletConfig()	서블릿의 초기 설정 정보를 담고 있는 객체를 리턴한다(config 내장 객체를 리턴한다).
ServletContext	getServletContext()	서블릿의 실행 환경 정보를 담고 있는 객체를 리턴한다(application 내장 객체를 리턴한다).



내장 객체

HttpSession	getSession()	클라이언트의 세션 정보를 담고 있는 객체를 리턴한다(session 내장 객체를 리턴한다).
없음	forward(String url)	현재 페이지의 요청과 응답에 관한 제어권을 URL로 지정된 주소로 영구적으로 넘긴다. forward된 페이지의 요청 처리가 종료되면 응답도 종료된다.
없음	include(String url)	현재 페이지의 요청과 응답에 관한 제어권을 URL로 지정된 주소로 임시로 넘긴다. include된 페이지의 처리가 끝나면 제어권은 다시 원래의 페이지로 돌아온다. 따라서 include로 지정된 페이지의 내용을 원래 페이지에 삽입하는 효과를 가진다.

■ 예제

• pageContextTest1.jsp (교재 136p 참조)

```

1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <%
4      pageContext.forward("pageContextTest2.jsp");
5  %>

```



내장 객체

- **pageContextTest2.jsp**

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <html>  
4  <head>  
5  <title>pageContext Test</title>  
6  </head>  
7  <body>  
8  <%  
9  pageContext.include("pageContextTest3.jsp");  
10 %>  
11 <h2>pageContext의 forward 메소드로 포워딩된 페이지입니다.</h2>  
12 </body>  
13 </html>
```

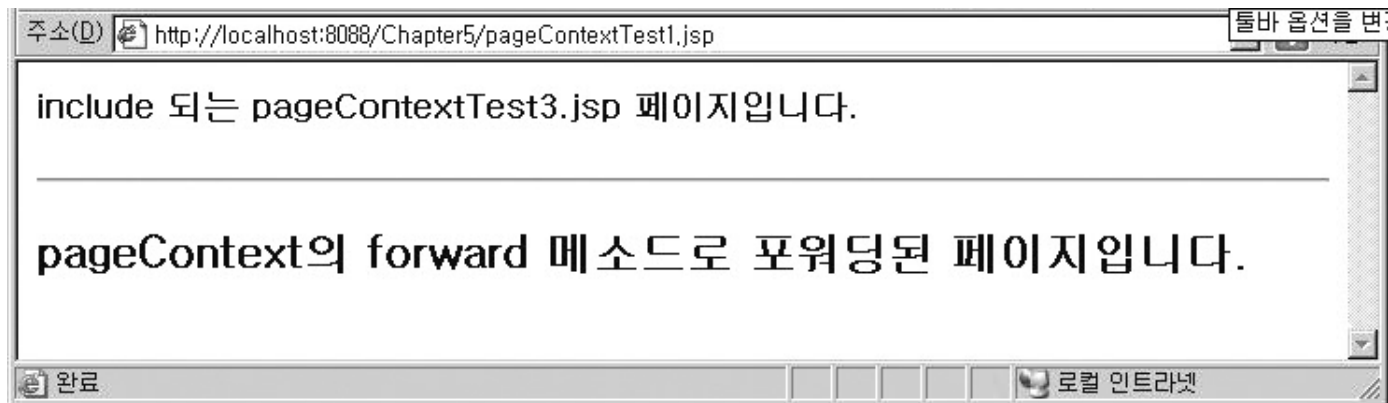


내장 객체

- **pageContextTest3.jsp**

```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2   pageEncoding="EUC-KR"%>  
3 <h3>include 되는 pageContextTest3.jsp 페이지입니다.</h3>  
4 <hr>
```

- **실행 결과**



내장 객체

□ session 객체

- 클라이언트와 서버와의 연결 유지에 사용
- 관련 메서드

리턴 타입	메소드명	설명
String	getId()	해당 세션의 세션 ID를 문자열로 리턴한다. 세션 ID는 session 객체 생성 시에 웹 컨테이너에 의해 자동으로 할당된다.
long	getCreationTime()	1970년 1월 1일 00시 00분 00초(epoch)부터 해당 세션이 생성된 순간까지의 경과 시간을 밀리 초로 계산하여 long형으로 리턴한다.
long	getLastAccessedTime()	epoch로부터 해당 세션에 마지막으로 접근된 시간까지의 경과 시간을 밀리 초로 계산하여 long형으로 리턴한다.
int	getMaxInactiveInterval()	클라이언트의 요청이 없을 시 서버가 해당 세션을 유지하도록 지정된 시간을 초 단위의 정수로 리턴한다.
없음	invalidate()	세션의 속성 값으로 저장된 모든 객체를 반납하여 해당 세션을 종료 시킨다.
boolean	isNew()	새로운 세션이면 true를 리턴하고 기존 세션이 유지되고 있으면 false를 리턴한다.
없음	setMaxInactiveInterval(int seconds)	클라이언트의 요청이 없더라도 세션을 유지할 시간을 초 단위의 정수값으로 설정한다. 음수로 설정할 경우 세션은 무효화(invalidate) 되지 않는다.



내장 객체

■ 예제

• sessionTest1.jsp

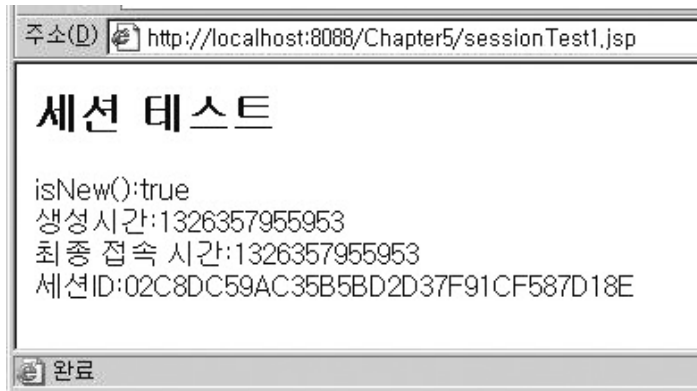
```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <%
4      session.setMaxInactiveInterval(10);
5      %>
6  <html>
7  <head>
8  <title>Session Test</title>
9  </head>
10 <body>
11 <h2>세션 테스트</h2>
12 isNew():<%=session.isNew()%><br>
13 생성시간:<%=session.getCreationTime()%><br>
14 최종 접속 시간:<%=session.getLastAccessedTime()%><br>
15 세션ID:<%=session.getId()%><br>
16 </body>
17 </html>
```



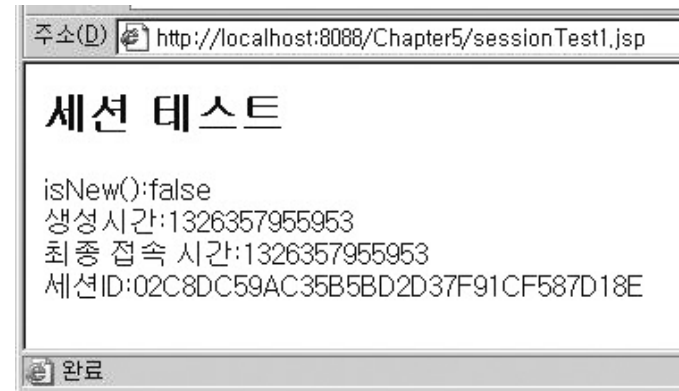
내장 객체

■ 실행 결과

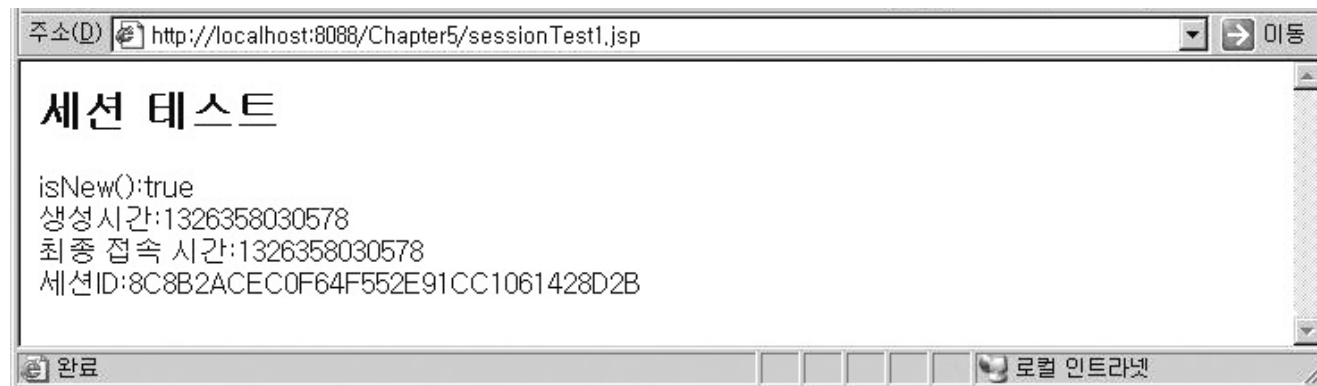
• sessionTest1.jsp



- 최초 실행시 -



- 10초 되기전 새로 고침 -



- 10초 지난후 새로 고침 -



내장 객체

□ application 객체

- 해당 웹 애플리케이션의 실행 환경을 제공하는 서버의 정보와 서버 측 자원에 대한 정보를 얻어내거나 해당 어플리케이션의 이벤트 로그를 다루는 메소드들을 제공
- 관련 메소드

리턴 타입	메소드명	설명
int	getMajorVersion()	Servlet API 스펙의 Major 버전을 int로 리턴한다.
int	getMinorVersion()	Servlet API 스펙의 Minor 버전을 int로 리턴한다.
String	getServerInfo()	서블릿/JSP 컨테이너의 이름과 버전을 문자열로 리턴한다.
String	getMimeType(String file)	서버에 존재하는 file이란 이름을 가진 파일의 MIME 타입을 문자열로 리턴한다.
java.net.URL	getResource(String path)	path로 지정된 경로의 자원을 URL 객체로 리턴한다. 자원이 존재하지 않으면 null을 리턴한다.
InputStream	getResourceAsStream(String path)	path로 지정된 경로의 자원을 InputStream 객체로 리턴한다. 자원이 존재하지 않으면 null을 리턴한다.
String	getRealPath(String path)	path로 지정된 경로의 자원을 서버의 실제 파일 시스템 상의 경로로 바꾸어 문자열로 리턴한다.



내장 객체

없음	<code>log(String msg)</code>	문자열 <code>msg</code> 를 서블릿 로그 파일에 기록한다.
없음	<code>log(String msg, java.lang.Throwable exception)</code>	문자열 <code>msg</code> 와 예외의 <code>StackTrace</code> 정보를 로그 파일에 기록한다.

■ 예제

- **applicationTest1.jsp**

```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"
2   pageEncoding="EUC-KR"%>
3 <html>
4 <head>
5 <title>Application Test</title>
6 </head>
7 <body>
8 <h2>application 테스트</h2>
9 <table border="1">
```



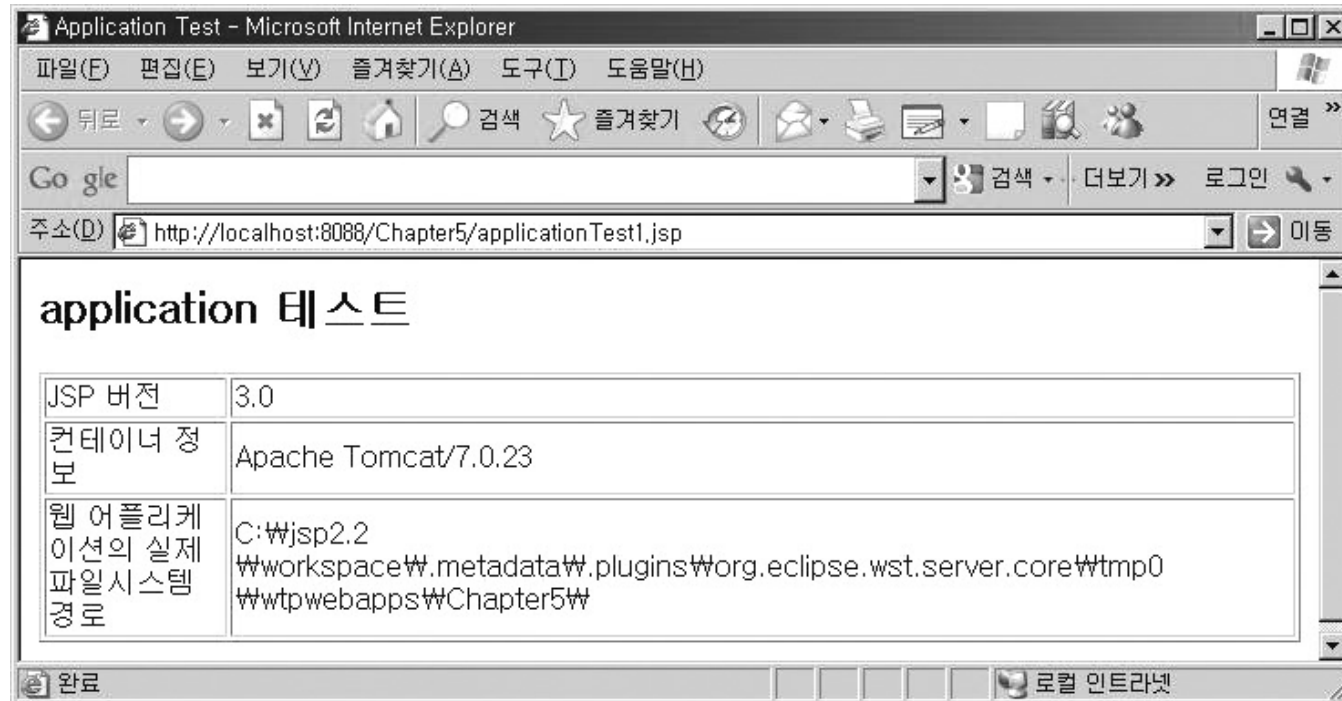
내장 객체

```
10      <tr>
11          <td>JSP 버전</td>
12          <td><%=application.getMajorVersion() %>.<%=application.getMinorVersion() %></
13      td>
14      </tr>
15      <tr>
16          <td>컨테이너 정보</td>
17          <td><%=application.getServerInfo() %></td>
18      </tr>
19      <tr>
20          <td>웹 애플리케이션의 실제 파일시스템 경로</td>
21          <td><%=application.getRealPath("/") %></td>
22      </tr>
23  </table>
24  </body>
25  </html>
```



내장 객체

■ 실행 결과



내장 객체

□ out 객체

- 서블릿/JSP 컨테이너가 응답 페이지를 만들기 위해 사용하는 출력 스트림 객체
- 관련 메서드

리턴 타입	메소드명	설명
없음	clear()	출력 버퍼에 저장된 내용을 버린다. 만일 이미 버퍼가 다 채워져서 클라이언트로 전송되었을 경우에는 예외를 발생시킨다.
없음	clearBuffer()	출력 버퍼에 저장된 내용을 버린다. clear() 메소드와는 다르게 버퍼에 담긴 내용이 이미 전송된 이후에도 예외를 발생시키지 않고 현재 저장되어 있는 버퍼만을 버린다.
없음	flush()	현재 버퍼에 저장되어 있는 내용을 클라이언트로 전송하고 버퍼를 비운다.
없음	close()	출력 버퍼를 클라이언트로 전송하고 출력 스트림을 종료한다.
boolean	isAutoFlush()	page 지시어의 autoFlush 속성으로 지정된 값을 리턴한다. 즉 출력 버퍼가 다 채워졌을 때 버퍼 내용을 클라이언트로 전송하도록 지정되어 있으면 true를 리턴하고, 출력 버퍼가 다 채워졌을 때 예외가 발생하도록 지정되어 있으면 false를 리턴한다.
int	getBufferSize()	출력 버퍼의 크기를 바이트 단위로 계산하여 정수 값으로 리턴한다.



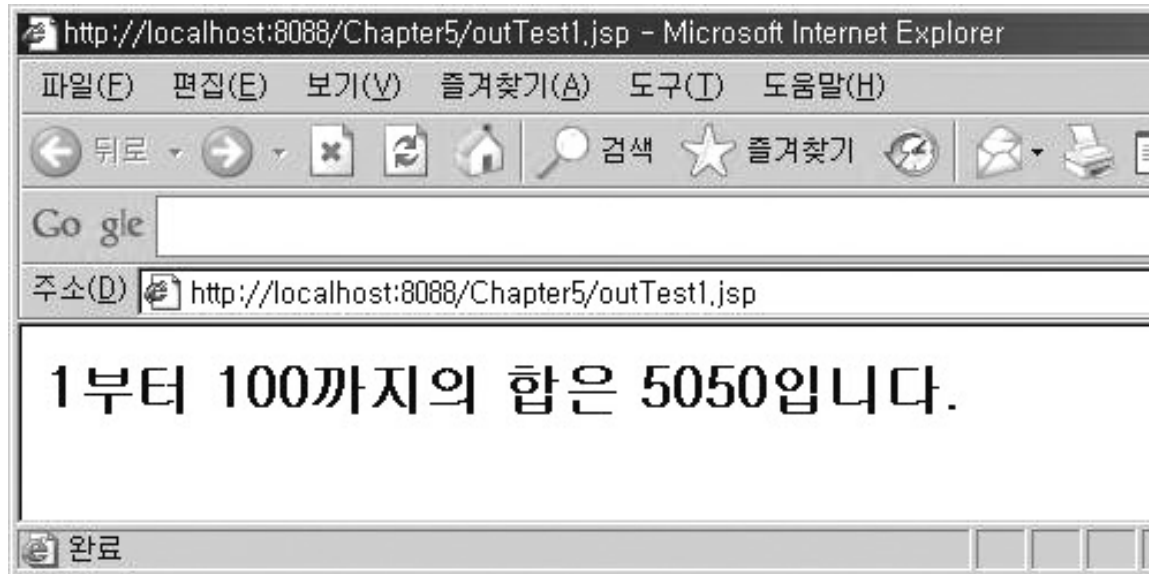
내장 객체

int	getRemaining()	출력 버퍼의 남은 양을 바이트 단위로 계산하여 정수 값으로 리턴한다.
없음	print(String str)	출력 스트림으로 str 문자열을 출력한다.

■ 예제

• outTest1.jsp

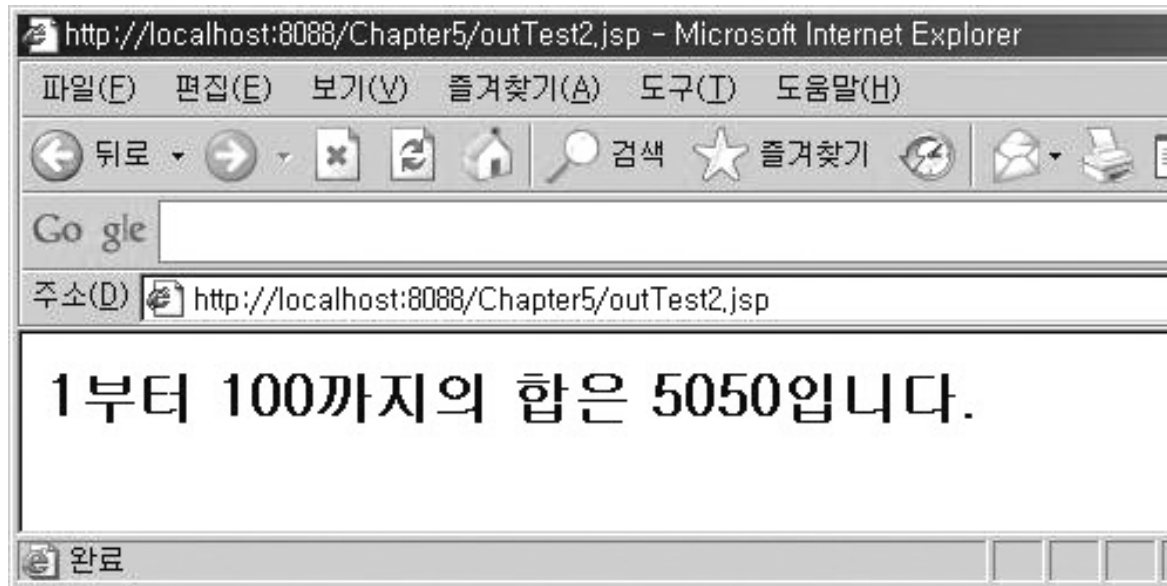
```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2   pageEncoding="EUC-KR"%>  
3 <h2>1부터 100까지의 합은  
4 <%  
5 int sum=0;  
6 for(int i=1;i<=100;i++){  
7     sum+=i;  
8 }  
9 out.print(sum+"입니다.</h2>");  
10 %>
```



내장 객체

- outTest1.jsp

```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2   pageEncoding="EUC-KR"%>  
3 <h2>1부터 100까지의 합은  
4 <%  
5 int sum=0;  
6 for(int i=1;i<=100;i++){  
7     sum+=i;  
8 }  
9 %>  
10 <%=sum %>입니다.</h2>
```



내장 객체

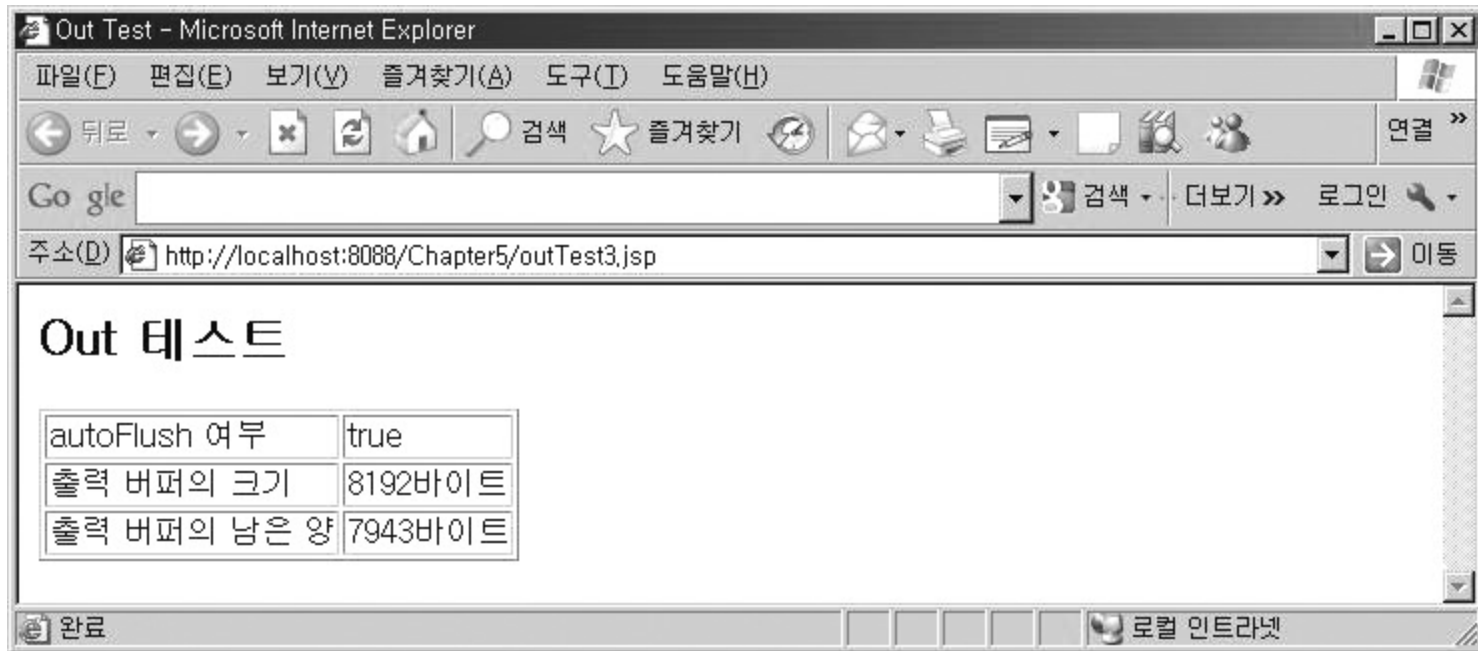
- outTest3.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <html>  
4  <head>  
5  <title>Out Test</title>  
6  </head>  
7  <body>  
8  <h2>Out 테스트</h2>  
9  <table border="1">  
10     <tr>  
11         <td>autoFlush 여부</td>  
12         <td><%=out.isAutoFlush() %></td>  
13     </tr>  
14     <tr>  
15         <td>출력 버퍼의 크기</td>  
16         <td><%=out.getBufferSize() %>바이트</td>  
17     </tr>
```



내장 객체

```
18         <tr>
19             <td>출력 버퍼의 남은 양</td>
20             <td><%=out.getRemaining() %>바이트</td>
21         </tr>
22     </table>
23 </body>
24 </html>
```



내장 객체

□ config 객체

- JSP 페이지가 서블릿 클래스로 변환되어 서블릿 인스턴스가 생성될 때 참조해야 할 초기 설정 정보들을 저장해 놓은 객체
- 관련 메서드

리턴 타입	메소드명	설명
String	getInitParameter(String init_paramName)	컨테이너의 설정 파일에 저장되어 있는 초기 파라미터 값 중 init_paramName의 이름을 가진 파라미터 값을 리턴한다. init_paramName의 이름을 가진 파라미터가 없을 경우 null을 리턴한다.
Enumeration	getInitParameterNames()	컨테이너의 설정 파일에 저장되어 있는 모든 초기 파라미터 이름을 Enumeration 타입으로 리턴한다.
String	getServletName()	해당 서블릿의 이름을 문자열로 리턴한다.



내장 객체

■ 예제

• configTest1.jsp

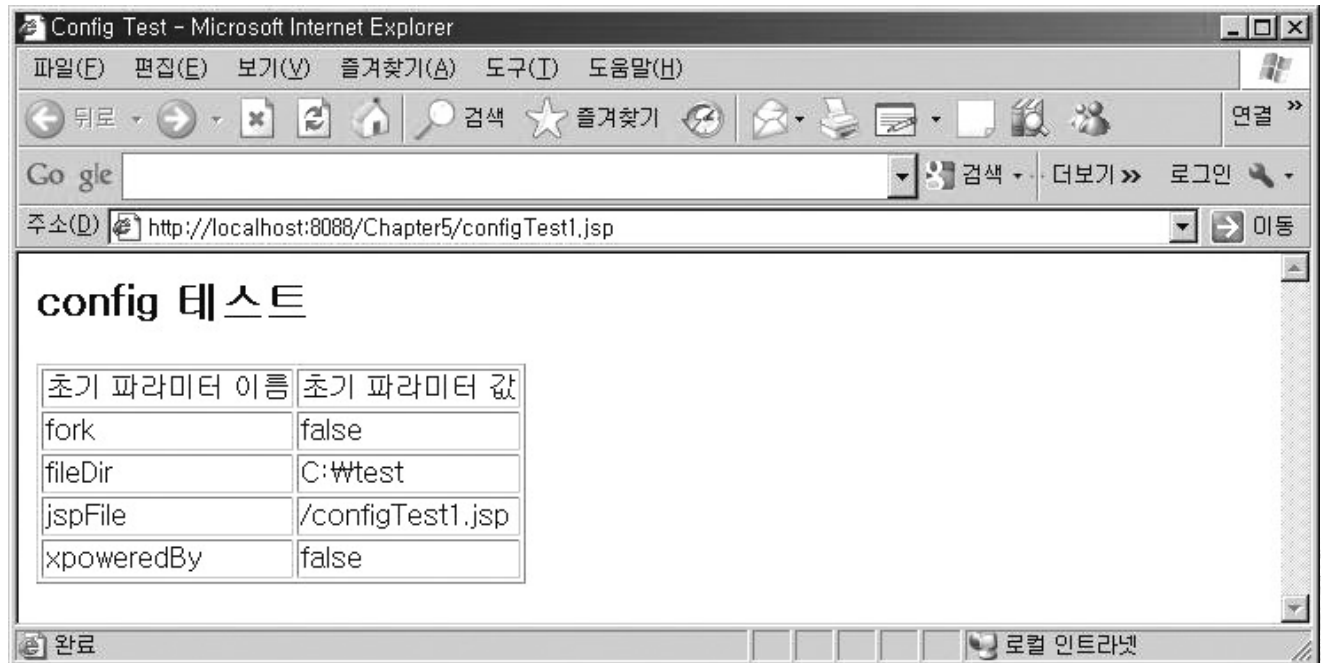
```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <%@page import="java.util.Enumeration"%>  
4  <html>  
5  <head>  
6  <title>Config Test</title>  
7  </head>  
8  <body>  
9  <h2>config 테스트</h2>  
10 <table border="1">  
11     <tr>  
12         <td>초기 파라미터 이름</td>  
13         <td>초기 파라미터 값</td>  
14     </tr>  
15 <%  
16 Enumeration e=config.getInitParameterNames();  
17 while(e.hasMoreElements()){  
18     String init_paramName=(String)e.nextElement();  
19     %>
```



내장 객체

```
20         <tr>
21             <td><%=init_paramName %></td>
22             <td><%=config.getInitParameter(init_paramName) %></td>
23         </tr>
24     <%
25     }
26     %>
27 </table>
28 </body>
29 </html>
```

■ 실행 결과



내장 객체

- **web.xml**

```
1 <display-name>Chapter5</display-name>
2   <servlet>
3
4       <servlet-name>configTest1</servlet-name>
5       <jsp-file>/configTest1.jsp</jsp-file>
6       <init-param>
7           <param-name>fileDir</param-name>
8           <param-value>C:\test</param-value>
9       </init-param>
10    </servlet>
11    <servlet-mapping>
12        <servlet-name>configTest1</servlet-name>
13        <url-pattern>/configTest1.jsp</url-pattern>
14    </servlet-mapping>
```



내장 객체

□ page 객체

- JSP 페이지에 의해 생성되는 서블릿 인스턴스 자체를 나타내는 객체

□ exception 객체

- JSP 페이지에서 예외가 발생하였을 경우 그 예외를 처리할 에러 페이지를 사용자가 지정한 경우에 해당 에러 페이지에 전달되는 예외 객체
- 관련 메서드

리턴 타입	메소드명	설명
String	getMessage()	에러 메시지를 문자열로 리턴한다.
없음	printStackTrace()	해당 에러의 StackTrace 정보를 출력한다.
없음	printStackTrace(PrintWriter out)	해당 에러의 StackTrace 정보를 PrintWriter 객체 out으로 출력한다.
없음	printStackTrace(PrintStream out)	해당 에러의 StackTrace 정보를 PrintStream 객체 out으로 출력한다.

영역 객체와 속성

□ 영역(Scope)과 속성(Attribute)

- 속성(Attribute)
 - 공유되는 데이터
- 영역(Scope)
 - 속성을 공유할 수 있는 유효 범위

영역	영역객체	속성의 유효 범위
page	pageContext	해당 페이지가 클라이언트에 서비스를 제공하는 동안에만 유효 (서블릿 인스턴스의 <code>_jspService()</code> 메소드가 실행되는 동안에만 유효)
request	request	클라이언트의 요청이 처리되는 동안 유효 (포워딩 또는 include를 이용하는 경우 여러 개의 페이지에서도 요청 정보가 계속 유지되므로 request 영역의 속성을 여러 페이지에서 공유 할 수 있다.)
session	session	세션이 유지되는 동안 유효 (하나의 브라우저에 1개의 세션이 생성되므로 같은 웹 브라우저 내에서 실행되는 페이지들이 속성을 공유할 수 있다.)
application	application	웹 애플리케이션이 실행되고 있는 동안 유효 (웹 컨테이너에서 해당 어플리케이션은 오직 하나만이 실행되므로 4가지 영역 중 가장 큰 영역에 해당한다.)



영역 객체와 속성

□ 속성과 관련된 메소드들

- **pageContext, request, session, application** 내장 객체들이 동일하게 정의하고 있는 메소드

리턴 타입	메소드명	해설
Object	getAttribute(String key)	key값으로 등록되어 있는 속성을 Object 타입으로 리턴(key값에 해당하는 속성이 없을 경우 null을 리턴)
Enumeration	getAttributeNames()	해당 영역에 등록되어 있는 모든 속성들의 이름을 Enumeration 타입으로 리턴
없음	setAttribute(String key, Object obj)	해당 영역에 key 값의 이름으로 obj 객체를 등록
없음	removeAttribute(String key)	key 값으로 등록되어 있는 속성을 제거



영역 객체와 속성

■ 예제

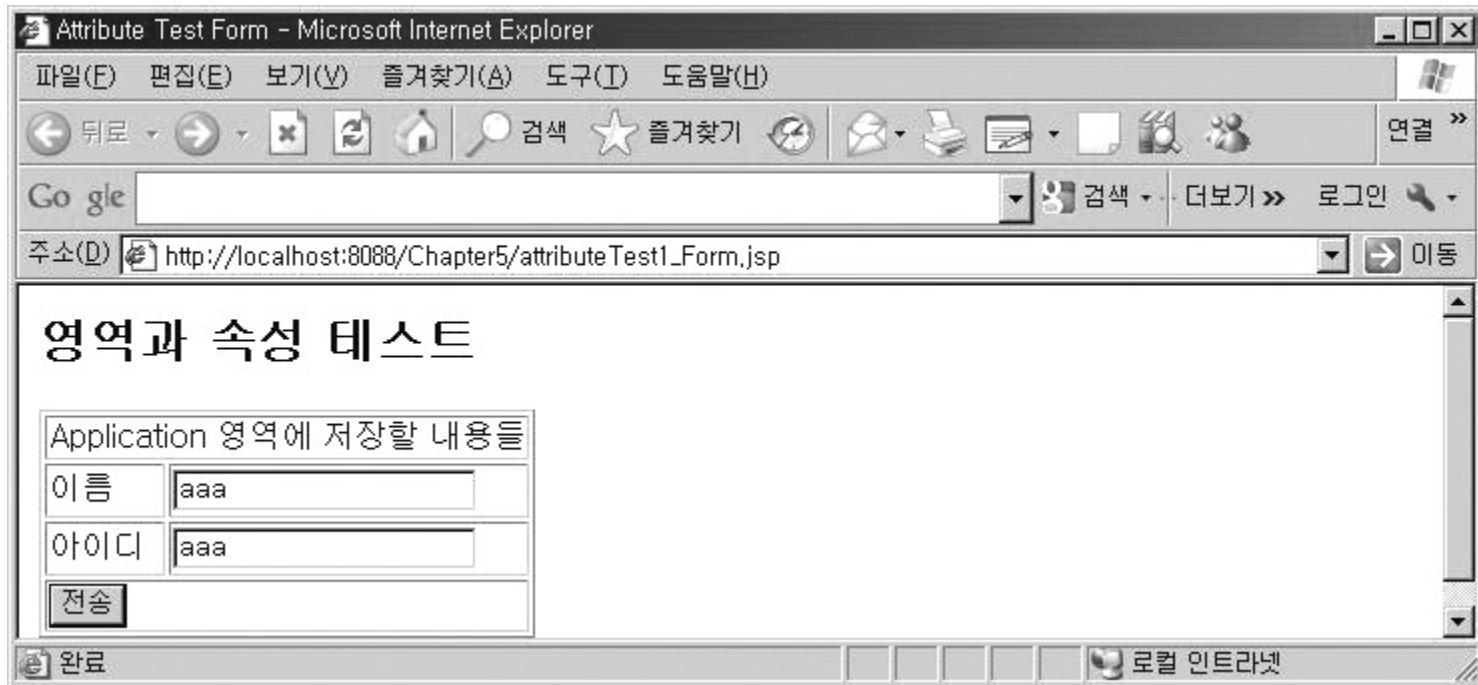
• attributeTest1_Form.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <html>  
4  <head>  
5  <title>Attribute Test Form</title>  
6  </head>  
7  <body>  
8  <h2>영역과 속성 테스트</h2>  
9  <form action="attributeTest1.jsp" method="post">  
10 <table border="1">  
11     <tr><td colspan="2">Application 영역에 저장할 내용들</td></tr>  
12     <tr>  
13         <td>이름</td>  
14         <td><input type="text" name="name"></td>  
15     </tr>  
16     <tr>  
17         <td>아이디</td>  
18         <td><input type="text" name="id"></td>  
19     </tr>
```



영역 객체와 속성

```
20         <tr>
21             <td colspan="2"><input type="submit" value="전송"></td>
22         </tr>
23     </table>
24 </form>
25 </body>
26 </html>
```



영역 객체와 속성

- **attributeTest1.jsp**

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <html>  
4  <head>  
5  <title>Attribute Test</title>  
6  </head>  
7  <body>  
8  <h2>영역과 속성 테스트</h2>  
9  <%  
10 request.setCharacterEncoding("euc-kr");  
11 String name=request.getParameter("name");  
12 String id=request.getParameter("id");  
13 if(name!=null&&id!=null){  
14     application.setAttribute("name",name);  
15     application.setAttribute("id",id);  
16 }  
17 %>
```



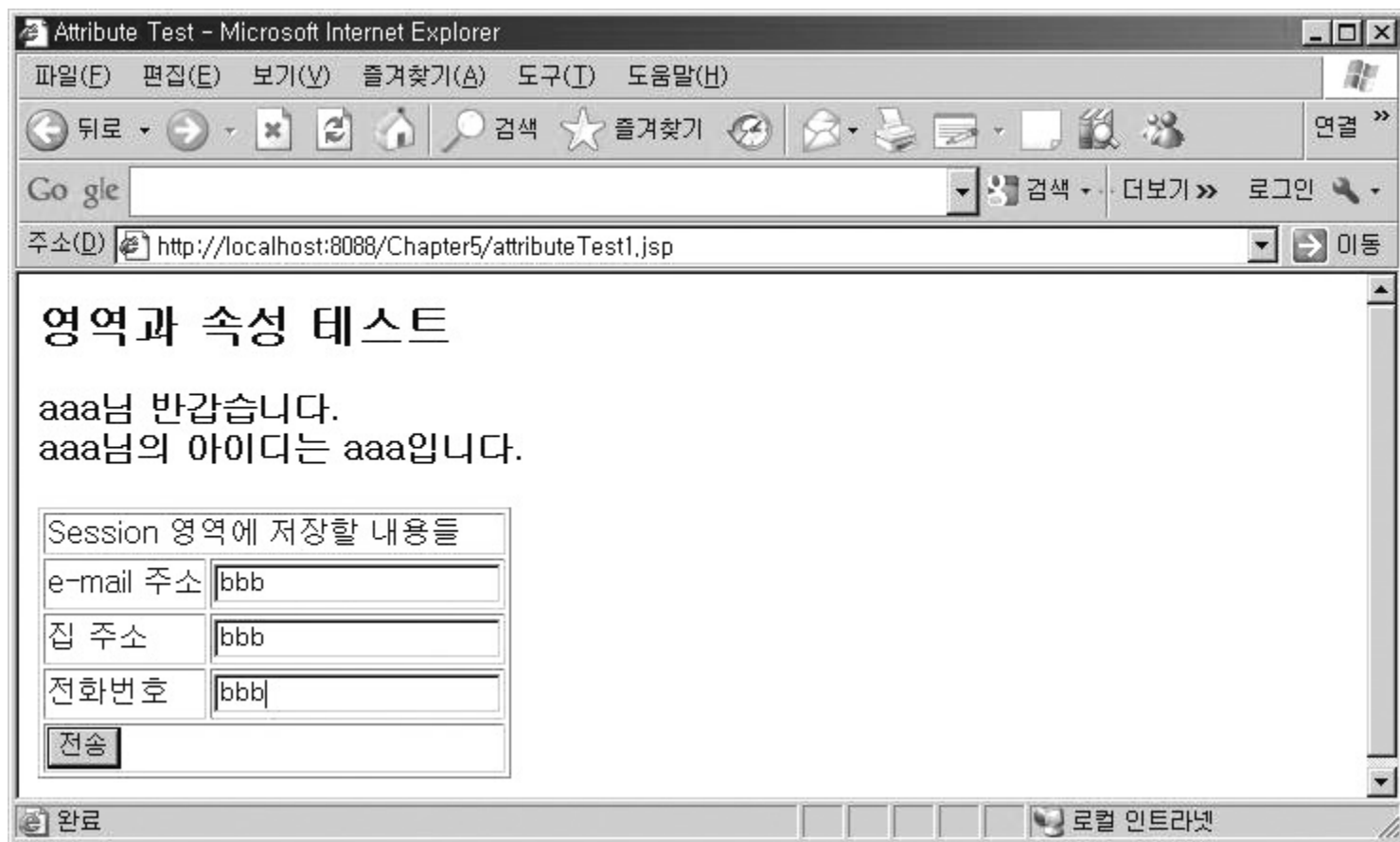
영역 객체와 속성

```
18 <h3><%=name %>님 반갑습니다.<br><%=name %>님의 아이디는 <%=id %>입니다.</h3>
19 <form action="attributeTest2.jsp" method="post">
20 <table border="1">
21     <tr><td colspan="2">Session 영역에 저장할 내용들</td></tr>
22     <tr>
23         <td>e-mail 주소</td>
24         <td><input type="text" name="email"></td>
25     </tr>
26     <tr>
27         <td>집 주소</td>
28         <td><input type="text" name="address"></td>
29     </tr>
30     <tr>
31         <td>전화번호</td>
32         <td><input type="text" name="tel"></td>
33     </tr>
34     <tr>
35         <td colspan="2"><input type="submit" value="전송"></td>
36     </tr>
37 </table>
```



영역 객체와 속성

```
38 </form>
39 </body>
40 </html>
```



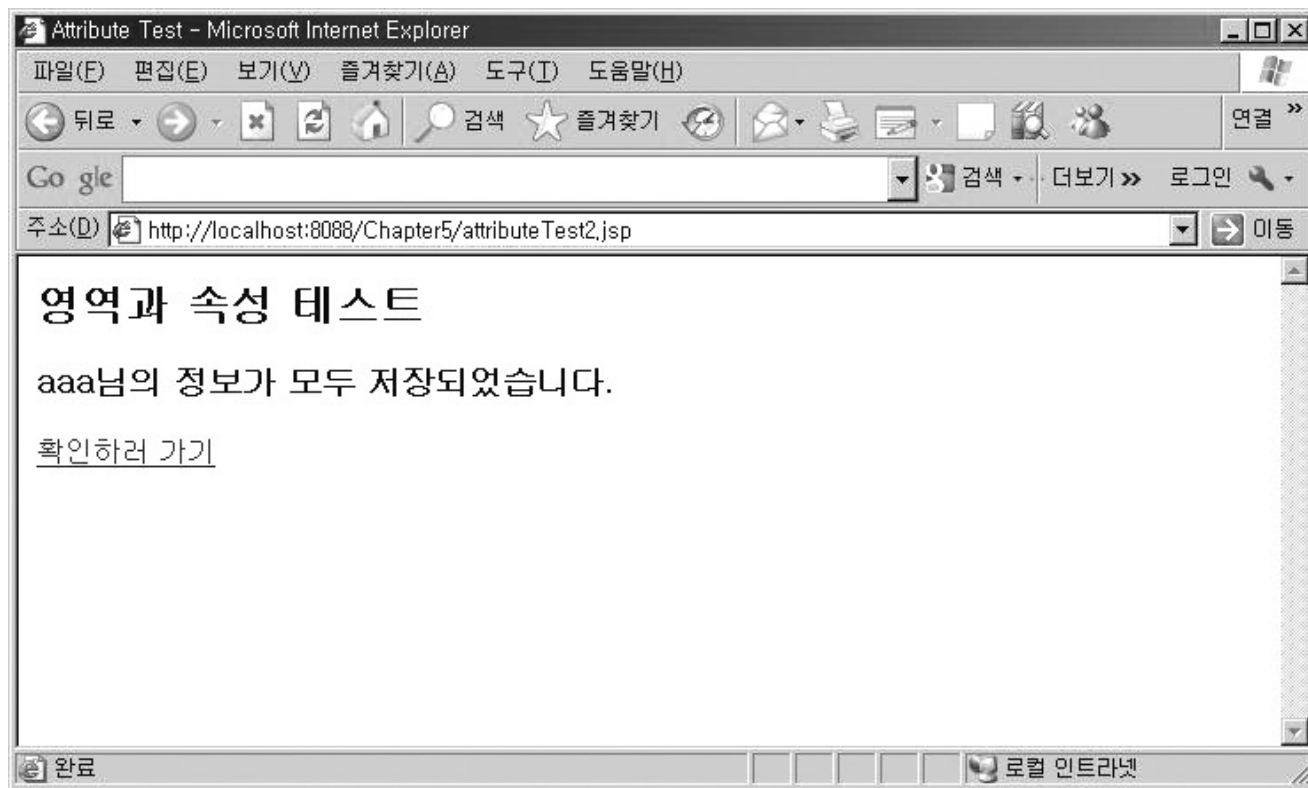
영역 객체와 속성

- **attributeTest2.jsp**

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <html>  
4  <head>  
5  <title>Attribute Test</title>  
6  </head>  
7  <body>  
8  <h2>영역과 속성 테스트</h2>  
9  <%  
10 request.setCharacterEncoding("euc-kr");  
11 String email=request.getParameter("email");  
12 String address=request.getParameter("address");  
13 String tel=request.getParameter("tel");  
14 session.setAttribute("email",email);  
15 session.setAttribute("address",address);  
16 session.setAttribute("tel",tel);  
17  
18 String name=(String)application.getAttribute("name");  
19 %>
```

영역 객체와 속성

```
20 <h3><%=name %>님의 정보가 모두 저장되었습니다.</h3>  
21 <a href="attributeTest3.jsp">확인하러 가기</a>  
22 </body>  
23 </html>
```



영역 객체와 속성

- **attributeTest3.jsp**

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2      pageEncoding="EUC-KR"%>  
3  <%@page import="java.util.Enumeration"%>  
4  <html>  
5  <head>  
6  <title>Attribute Test</title>  
7  </head>  
8  <body>  
9  <h2>영역과 속성 테스트</h2>  
10 <table border="1">  
11     <tr><td colspan="2">Application 영역에 저장된 내용들</td></tr>  
12     <tr>  
13         <td>이름</td>  
14         <td><%=application.getAttribute("name") %></td>  
15     </tr>  
16     <tr>  
17         <td>아이디</td>  
18         <td><%=application.getAttribute("id") %></td>  
19     </tr>  
20 </table>
```



영역 객체와 속성

```
21 <br>
22 <table border="1">
23     <tr><td colspan="2">Session 영역에 저장된 내용들</td></tr>
24     <%
25     Enumeration e=session.getAttributeNames();
26     while(e.hasMoreElements()){
27         String attributeName=(String)e.nextElement();
28         String attributeValue=(String)session.getAttribute(attributeName);
29         %>
30         <tr>
31             <td><%=attributeName %></td>
32             <td><%=attributeValue %></td>
33         </tr>
34         <%
35     }
36     %>
37 </table>
38 </body>
39 </html>
```



영역 객체와 속성

- attributeTest3.jsp

주소(D) http://localhost:8088/Chapter5/attributeTest3.jsp

영역과 속성 테스트

Application 영역에 저장된 내용들	
이름	aaa
아이디	aaa

Session 영역에 저장된 내용들	
address	bbb
email	bbb
tel	bbb

주소(D) http://localhost:8088/Chapter5/attributeTest3.jsp

영역과 속성 테스트

Application 영역에 저장된 내용들	
이름	aaa
아이디	aaa

Session 영역에 저장된 내용들	
---------------------	--

- 브라우저를 닫았다가 재실행 -

- 서버를 종료 후 다시 실행 -

주소(D) http://localhost:8088/Chapter5/attributeTest3.jsp

영역과 속성 테스트

Application 영역에 저장된 내용들	
이름	null
아이디	null

Session 영역에 저장된 내용들	
---------------------	--

영역 객체와 속성

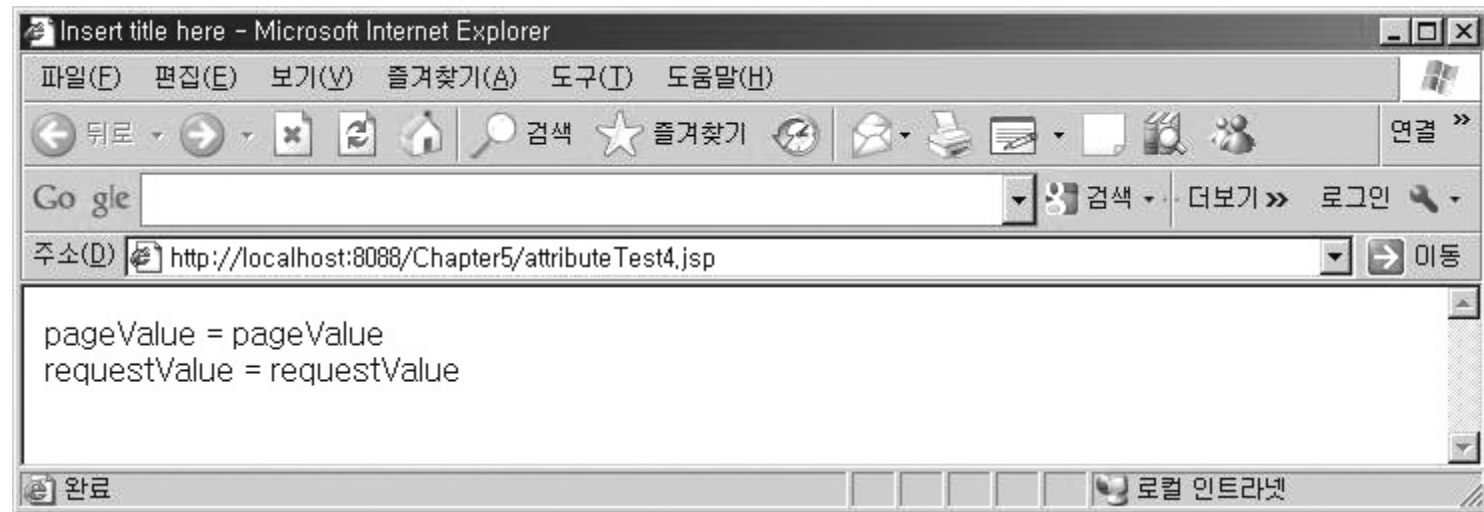
■ page 영역과 request 영역 비교

• attributeTest4.jsp

```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2      pageEncoding="EUC-KR"%>
3  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/
4      loose.dtd">
5  <html>
6  <head>
7      <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
8      <title>Insert title here</title>
9  </head>
10 <body>
11 <%
12     pageContext.setAttribute("pageScope", "pageValue");
13     request.setAttribute("requestScope", "requestValue");
14 %>
15
16     pageValue = <%=pageContext.getAttribute("pageScope") %><br>
17     requestValue = <%=request.getAttribute("requestScope") %>
18 </body>
19 </html>
```



영역 객체와 속성



영역 객체와 속성

- **attributeTest5.jsp**

- 포워드 기능을 이용해서 페이지를 다른 페이지로 변경해서 page 영역을 소멸시키고 request영역은 유지

```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"
2   pageEncoding="EUC-KR"%>
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/
4 loose.dtd">
5 <html>
6 <head>
7 <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
8 <title>Insert title here</title>
9 </head>
10 <body>
11 <%
12   pageContext.setAttribute("pageScope", "pageValue");
13   request.setAttribute("requestScope", "requestValue");
14   %>
15 <jsp:forward page="requestTest5Result.jsp"></jsp:forward>
16 </body>
17 </html>
```

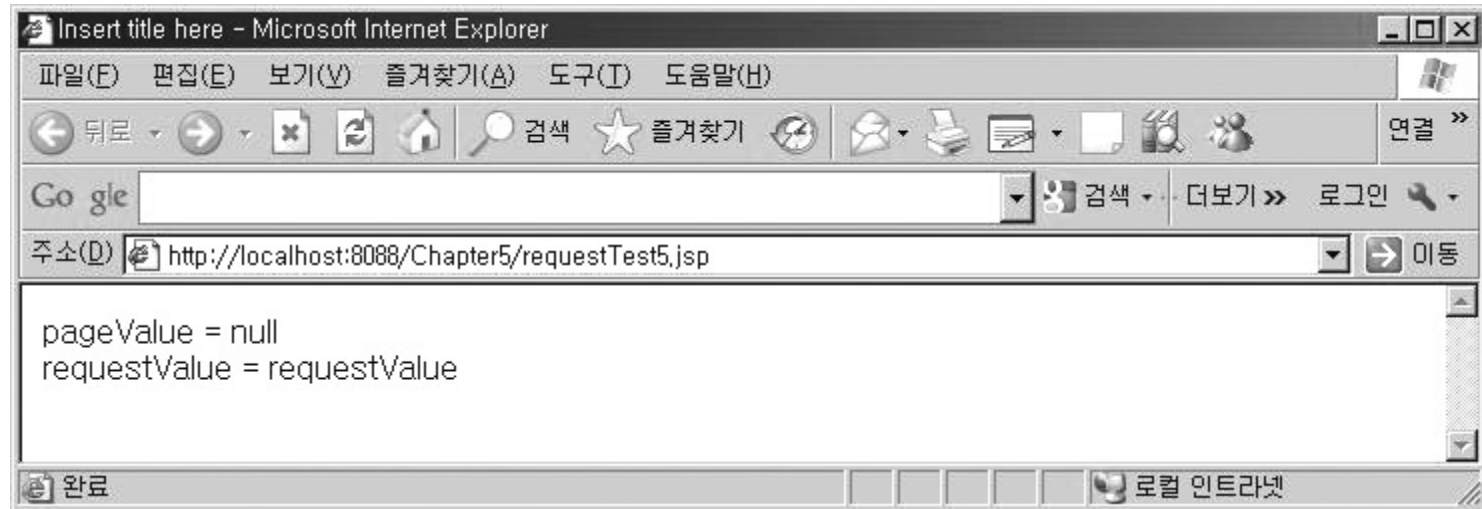
영역 객체와 속성

- **attributeTest5Result.jsp**

```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2   pageEncoding="EUC-KR"%>  
3 <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/  
4 loose.dtd">  
5 <html>  
6 <head>  
7 <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">  
8 <title>Insert title here</title>  
9 </head>  
10 <body>  
11 pageValue = <%=pageContext.getAttribute("pageScope") %><br>  
12 requestValue = <%=request.getAttribute("requestScope") %>  
13 </body>  
14 </html>
```



영역 객체와 속성



액션 태그

□ 액션 태그의 개요

- JSP 페이지에서 자바 코드 등의 스크립트 언어를 사용하지 않고도 (즉, HTML 태그 형태로) 다른 페이지의 서블릿이나 자바빈의 객체에 접근할 수 있도록 태그를 이용해 구현된 기능
- 액션 태그를 통해서 개발자는 페이지의 흐름을 제어하거나 자바빈의 속성을 읽고 쓰며 애플릿을 사용하는 등의 다양한 기능을 활용 가능
- 사용자에게 보여지는 프레젠테이션 부분과 사용자의 요청을 처리하는 비즈니스 로직 부분(프로그램 부분)을 분리하는 것이 가능
- JSP에서 제공하는 액션 태그
 - 페이지 흐름 제어 액션(forward/include 액션)
 - 자바빈 사용 액션(useBean 액션)
 - 애플릿 사용 액션(plugin 액션)



액션 태그

□ forward 액션

```
<jsp:forward page="이동할 페이지" />  
<jsp:forward page="이동할 페이지"></jsp:forward>
```

- **pageContext** 내장 객체의 **forward()** 메소드가 태그로 구현된 기능
- 현재 페이지의 요청과 응답에 관한 처리권을 **page** 속성에 지정된 이동할 페이지로 영구적으로 넘기는 기능 수행

```
<jsp:forward page="/test/forward/forward.jsp" />  
<jsp:forward page="forward/forward.jsp" />
```

- 표현식을 사용해 동적으로 지정하는 것도 가능

```
<jsp:forward page='<%=nextPage %>' />
```



액션 태그

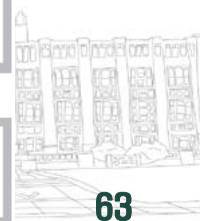
- **forward** 태그를 사용하여 이동할 페이지에 파라미터를 넘겨줄 필요가 있을 때에는 하위 태그인 `<jsp:param />` 태그 사용

```
<jsp:forward page="이동할 페이지">  
    <jsp:param name="파라미터 이름1" value="파라미터 값1" />  
    <jsp:param name="파라미터 이름2" value="파라미터 값2" />  
    ...  
</jsp:forward>
```

- `<jsp:param />` 태그를 사용하여 지정한 파라미터는 GET 방식으로 전송할 때처럼 이동할 페이지의 주소 뒤에 직접 붙여서 전송하는 것도 가능

```
<jsp:forward page="test/forward.jsp">  
    <jsp:param name="id" value="hongkildong" />  
    <jsp:param name="password" value="abc" />  
</jsp:forward>
```

```
<jsp:forward page="test/forward.jsp?id=hongkildong&password=abc" />
```



액션 태그

■ 예제

• forwardTest.jsp

```
1 <body>
2 <h2>포워드 액션 테스트</h2>
3 <form action="forwardTest1.jsp" method="POST">
4 <input type="hidden" name="forwardPage" value="forwardTest2.jsp">
5 <table>
6     <tr>
7         <td>이름    </td>
8         <td><input type="text" name="name"></td>
9     </tr>
10    <tr>
11        <td>나이    </td>
12        <td><input type="text" name="age"></td>
13    </tr>
14    <tr>
15        <td>주소    </td>
16        <td><input type="text" name="address"></td>
17    </tr>
18    <tr><td><input type="submit" value="전송"></td></tr>
```



액션 태그

```
19 | </table>
20 | </form>
21 | </body>
```

- **forwardTest1.jsp**

```
1 | <%@ page language="java" contentType="text/html; charset=EUC-KR"
2 |   pageEncoding="EUC-KR"%>
3 | <%request.setCharacterEncoding("euc-kr"); %>
4 | <html>
5 | <body>
6 | <jsp:forward page='<%=request.getParameter("forwardPage") %>' >
7 |   <jsp:param name="tel" value="034-1234-5678"/>
8 | </jsp:forward>
9 | </body>
10 | </html>
```



액션 태그

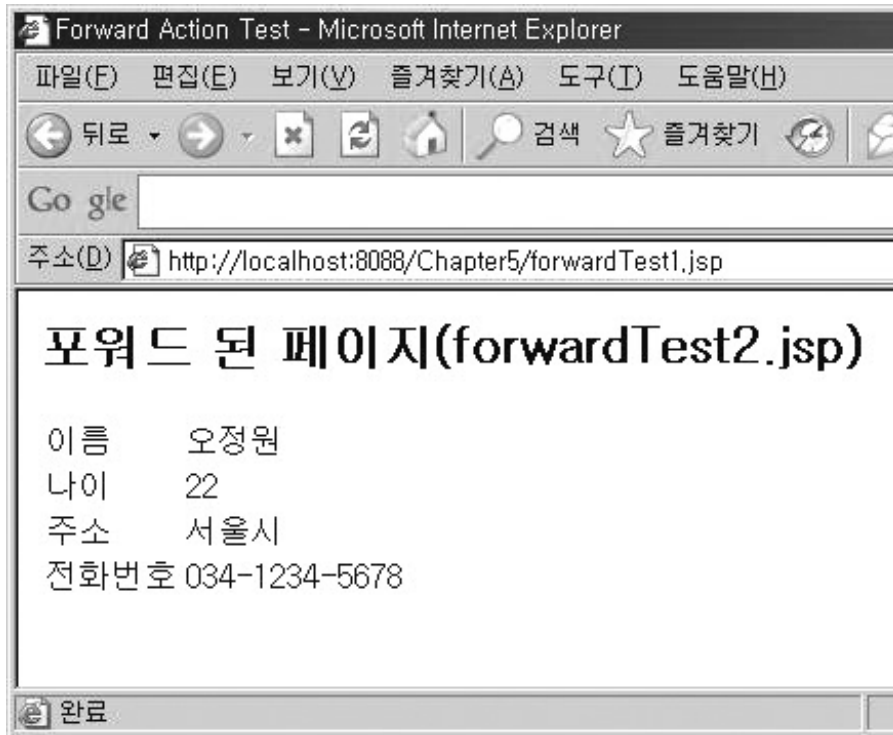
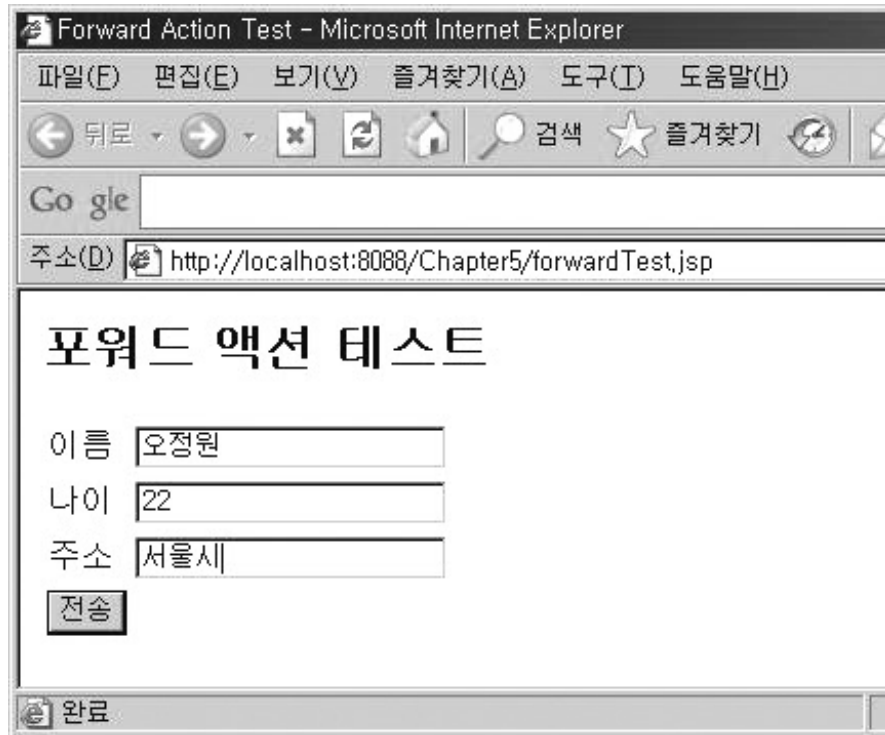
- forwardTest2.jsp

```
1 <body>
2 <h2>포워드 된 페이지(forwardTest2.jsp)</h2>
3 <table>
4     <tr>
5         <td>이름</td>
6         <td><%=request.getParameter("name") %></td>
7     </tr>
8     <tr>
9         <td>나이</td>
10        <td><%=request.getParameter("age") %></td>
11    </tr>
12    <tr>
13        <td>주소</td>
14        <td><%=request.getParameter("address") %></td>
15    </tr>
16    <tr>
17        <td>전화번호</td>
18        <td><%=request.getParameter("tel") %></td>
19    </tr>
20 </table>
21 </body>
```



액션 태그

■ 실행 결과



액션 태그

□ include 액션 태그

- 임시로 제어권을 include되는 페이지로 넘겼다가 그 페이지의 처리가 끝나면 처리 결과를 원래 페이지로 리턴하고 다시 원래의 페이지로 제어권을 반환하는 방식

```
<jsp:include page="포함될 페이지" flush="false"/>  
<jsp:include page="포함될 페이지" flush="false"></jsp:include>
```

- include 액션 태그에서도 파라미터 값 전달

```
<jsp:include page="이동할 페이지">  
    <jsp:param name="파라미터 이름1" value="파라미터 값1" />  
    <jsp:param name="파라미터 이름2" value="파라미터 값2" />  
    ...  
</jsp:include>
```



액션 태그

- **지정한 파라미터는 GET 방식으로 전송할 때처럼 이동할 페이지의 주소 뒤에 붙여서 전송하는 것도 가능**

```
<jsp:include page="test/include.jsp">  
    <jsp:param name="id" value="hongkildong" />  
    <jsp:param name="password" value="abc" />  
    ...  
</jsp:include>
```

```
<jsp:include page="test/include.jsp?id=hongkildong&password=abc" />
```



액션 태그

■ 예제

• includeTest.jsp

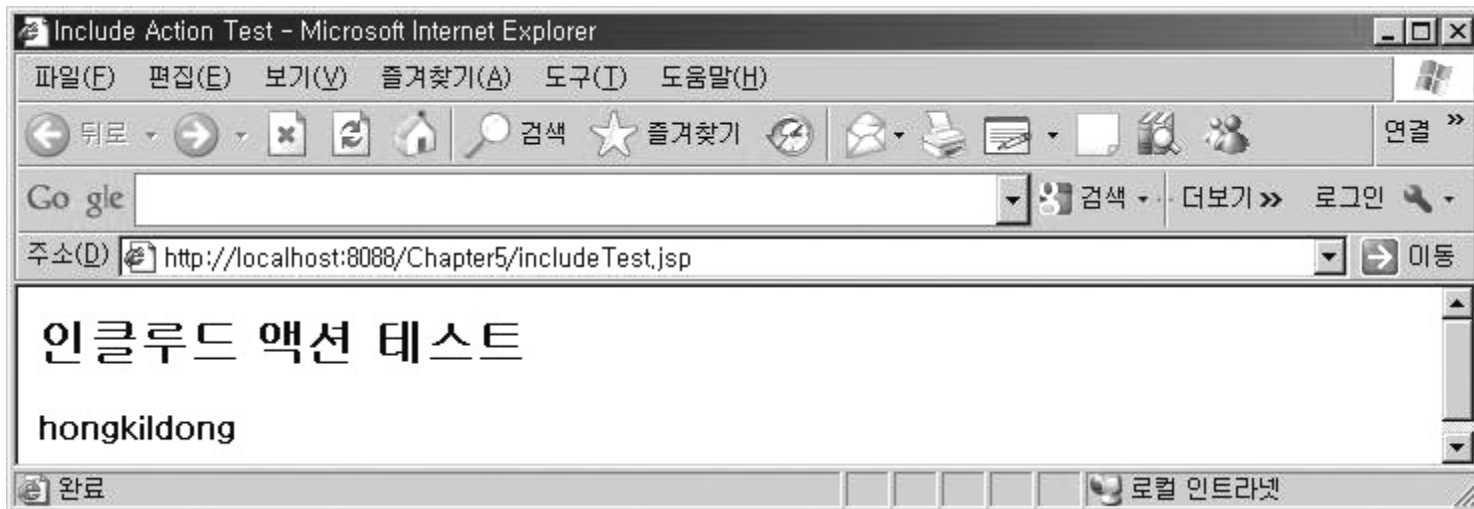
```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2     pageEncoding="EUC-KR"%>  
3  <html>  
4  <head>  
5  <title>Include Action Test</title>  
6  </head>  
7  <body>  
8  <h2>인클루드 액션 테스트</h2>  
9  <jsp:include page="includeTest2.jsp">  
10     <jsp:param name="name" value="hongkildong"/>  
11 </jsp:include>  
12 </body>  
13 </html>
```



액션 태그

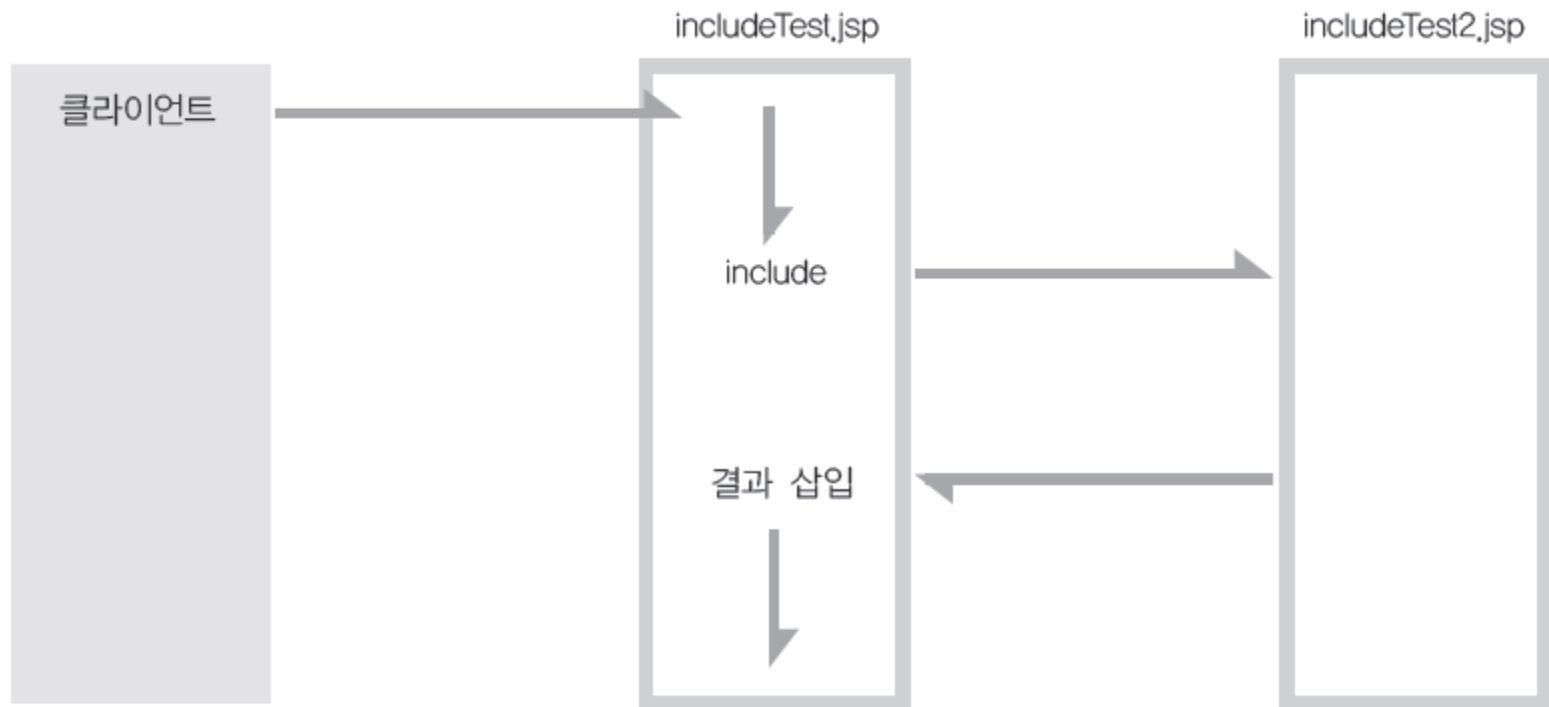
- includeTest2.jsp

```
1 <%@ page language="java" contentType="text/html; charset=EUC-KR"  
2   pageEncoding="EUC-KR"%>  
3 <%  
4     String name=request.getParameter("name");  
5   %>  
6 <html>  
7 <body>  
8 <b><%=name%></b>  
9 </body>  
10 </html>
```



액션 태그

- **include** 액션 태그 사용 시 실행 순서



[그림 5-1] include 액션 태그 사용 시 실행 순서



액션 태그

□ plugin 액션

- 태그는 자바 애플릿 또는 자바빈즈 컴포넌트를 클라이언트로 다운로드 받아서 사용할 수 있도록 각 브라우저에 맞는 HTML 코드를 생성해주는 역할 수행
- 서버의 부하를 줄여주는 장점이 있다.

```
<jsp:plugin type="플러그인 타입"
codebase="클래스 파일의 위치" code="불러올 클래스 파일"
width="가로" height="세로">
    <jsp:params>
        <jsp:param name="파라미터 이름" value="파라미터 값"/>
    </jsp:params>
</jsp:plugin>
```



액션 태그

■ plugin 태그 속성

속성	사용법	설명
type	type="applet bean"	사용할 플러그인 타입을 설정한다. (applet 또는 bean)
code	code="code.class"	플러그인으로 사용할 클래스 파일을 설정한다.
width	width="800"	플러그인이 표시될 영역의 넓이를 지정한다.
height	height="600"	플러그인이 표시될 영역의 높이를 지정한다.
codebase	codebase="/"	클래스 파일의 위치를 지정한다.
name	name="plugin1"	플러그인을 구별하기 위한 이름을 지정한다.
align	align="left"	플러그인의 정렬 방식을 설정한다.
archive	archive="plugin.jar"	Jar 파일로 묶어 놓은 클래스 파일들을 지정한다.
vspace	vspace="10"	플러그인의 수직 여백을 설정한다.
hspace	hspace="10"	플러그인의 수평 여백을 설정한다.
jreversion	jreversion="1.5"	플러그인을 실행할 JRE 버전을 입력한다.
iepluginurl	iepluginurl="url"	익스플로러 전용 플러그인의 다운로드 URL을 입력한다.
nspluginurl	nspluginurl="url"	넷스케이프 전용 플러그인의 다운로드 URL을 입력한다.



액션 태그

■ 예제

• pluginTest.jsp

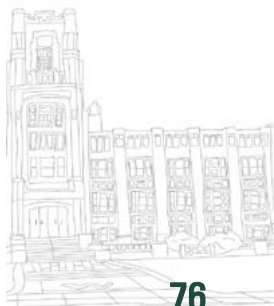
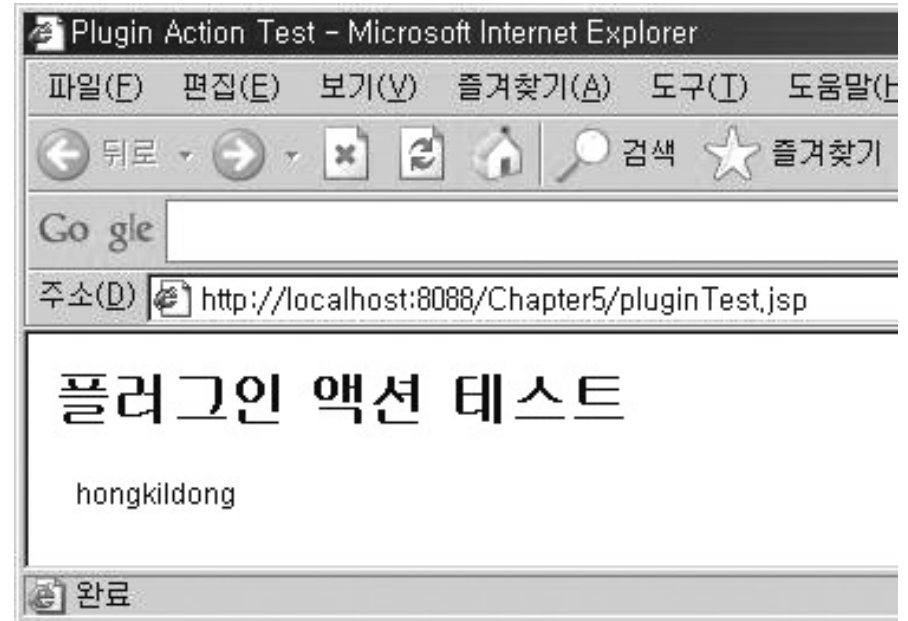
```
1  <%@ page language="java" contentType="text/html; charset=EUC-KR"
2     pageEncoding="EUC-KR"%>
3  <html>
4  <head>
5  <title>Plugin Action Test</title>
6  </head>
7  <body>
8  <h2>플러그인 액션 테스트</h2>
9  <jsp:plugin type="applet" codebase="." code="PluginTest" width="100" height="100">
10     <jsp:params>
11         <jsp:param name="name" value="hongkildong"/>
12     </jsp:params>
13     <jsp:fallback>플러그인 태그를 지원하지 않습니다.</jsp:fallback>
14 </jsp:plugin>
15 </body>
16 </html>
```

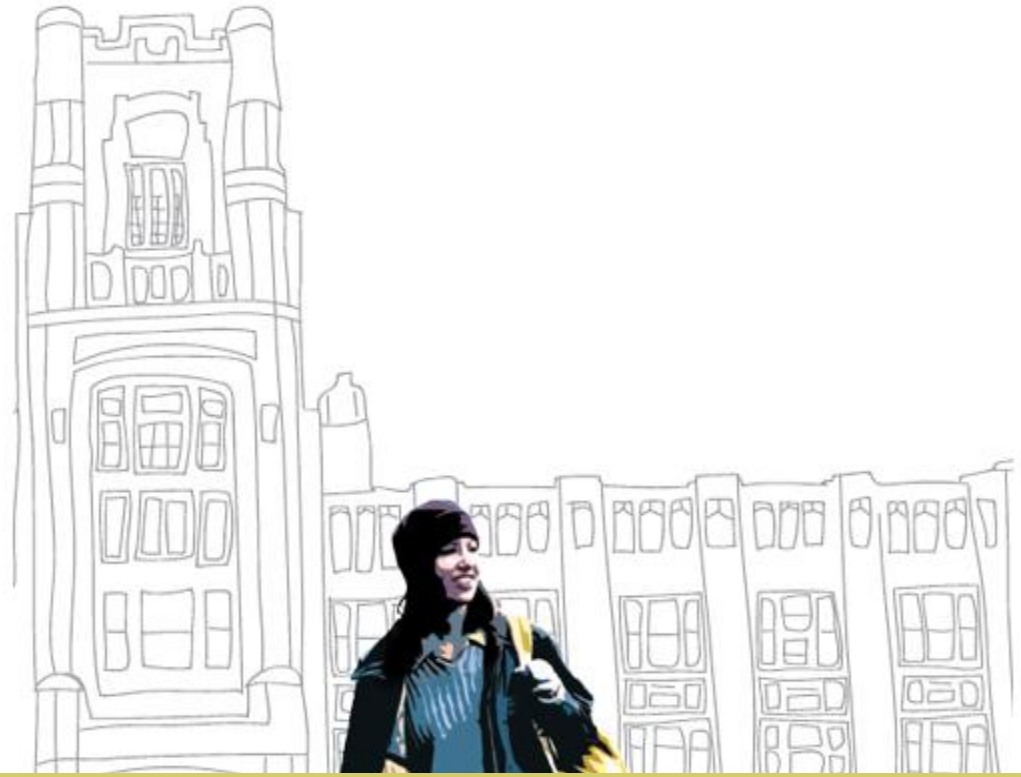


액션 태그

- pluginTest.java

```
1 import java.awt.*;
2 import java.applet.Applet;
3
4 public class PluginTest extends Applet {
5     String name;
6
7     public void init(){
8         name=getParameter("name");
9     }
10
11     public void paint(Graphics g){
12         g.drawString(name,10,10);
13     }
14 }
```





Thank You

