



JSTL 사용하기

뇌를 자극하는 JSP & Servlet

Contents

❖ 학습목표

- JSTL이란 JSP 페이지를 작성할 때 유용하게 사용할 수 있는 여러 가지 커스텀 액션과 함수가 포함되어 있는 라이브러리이다. 이 라이브러리는 인터넷에서 무상으로 다운로드 받을 수 있는데 이 장에서는 라이브러리를 다운로드 받아서 설치하고 사용하는 방법을 알아보자.

❖ 내용

- JSTL이란?
- JSTL 설치하기
- 코어 라이브러리 사용하기
- 포매팅 라이브러리 사용하기
- 함수 라이브러리 사용하기



1. JSTL이란?

- JSTL은 JSP 표준 태그 라이브러리(JSP Standard Tag Library)의 약어이다.
- 라이브러리란 여러 프로그램이 공통으로 사용하는 코드를 모아놓은 코드의 집합이다.
- JSTL을 가지고 할 수 있는 일
 - 간단한 프로그램 로직의 구사(자바의 변수 선언, if 문, for 문 등에 해당하는 로직)
 - 다른 JSP 페이지 호출(<c:redirect>, <c:import>)
 - 날짜, 시간, 숫자의 포맷
 - JSP 페이지 하나를 가지고 여러 가지 언어의 웹 페이지 생성
 - 데이터베이스로의 입력, 수정, 삭제, 조회
 - XML 문서의 처리
 - 문자열을 처리하는 함수 호출
- 문자열을 처리하는 함수 호출을 제외한 나머지 기능들은 모두 커스텀 액션 형태로 제공된다.



1. JSTL이란?

- JSTL에 있는 <c:forEach> 커스텀 액션은 자바의 for 문과 비슷한 기능을 한다.

```
<c:forEach begin= "1" end= "10" >  
  <H5>안녕하세요, 여러분!</H5>  
</c:forEach>
```

} 시작 태그와 끝 태그 사이에 있는 코드를
10번 반복해서 출력합니다.

- JSTL에 있는 <fmt:formatNumber> 커스텀 액션은 수치 값을 포맷하는 기능을 한다.

```
<fmt:formatNumber value= "3.14159" pattern= "#.00" />
```

주어진 수치를 소수점 이하 2자리까지 끊어서 출력합니다

- JSTL에는 커스텀 액션만 있는 게 아니라 익스프레션 언어에서 사용할 수 있는 EL 함수도 있다.

```
${fn:toUpperCase( "Hello ")}
```

이 함수는 'HELLO' 라는 문자열을 리턴합니다



1. JSTL이란?

[표 9-1] JSTL을 구성하는 작은 라이브러리들

라이브러리	기능	URI 식별자	접두어
코어	일반 프로그래밍 언어에서 제공하는 것과 유사한 변수 선언, 실행 흐름의 제어 기능을 제공하고, 다른 JSP 페이지로 제어를 이동하는 기능도 제공합니다.	http://java.sun.com/jsp/jstl/core	c
포매팅	숫자, 날짜, 시간을 포매팅하는 기능과 국제화, 다국어 지원 기능을 제공합니다.	http://java.sun.com/jsp/jstl/fmt	fmt
데이터베이스	데이터베이스의 데이터를 입력/수정/삭제/조회하는 기능을 제공합니다.	http://java.sun.com/jsp/jstl/sql	sql
XML 처리	XML 문서를 처리할 때 필요한 기능을 제공합니다.	http://java.sun.com/jsp/jstl/xml	x
함수	문자열을 처리하는 함수를 제공합니다.	http://java.sun.com/jsp/jstl/functions	fn



1. JSTL이란?

- JSP 페이지에서 앞 페이지의 접두어를 사용하기 위해서는 taglib 지시자를 이용해서 라이브러리의 URI 식별자와 접두어를 연결해야 한다.
- taglib 지시자는 다른 지시자와 마찬가지로 <%@으로 시작해서 %>로 끝난다.
- taglib 지시자에는 uri와 prefix라는 두 개의 애트리뷰트를 써야 하고, 이 두 애트리뷰트에 각각 URI 식별자와 접두어를 값으로 주어야 한다.

```
<%@taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core " %>
```

↑
접두어

↑
라이브러리를 식별하는 URI



2. JSTL 설치하기

❖ JSTL 1.2 다운로드 받기

- 웹 브라우저를 열고 주소 창에 <https://jstl.java.net/>라는 URL을 입력한다.

The screenshot shows the JSTL 1.2 website. A black box highlights the 'Download JSTL' section, which states: 'Get the latest version of JSTL'. An arrow points from this section to a callout box on the right. The callout box contains the text: 'The current JSTL can be downloaded from the maven repositories:' followed by a bulleted list: '• JSTL API' and '• JSTL Implementation'. The website also features a 'Get Started' section with a magnifying glass icon, a 'Get Involved' section with a wrench icon, and a 'GlassFish Community' sidebar with links to Downloads, Mailing lists, Membership, Developing JSTL, Subversion repository, and IssueTracker. The footer includes 'Terms of Use', 'Privacy Policy', and 'Copyright ©2013-2013 (revision)'.

2. JSTL 설치하기

❖ JSTL 다운로드 받기

- Downloads를 선택한다.

JSTL API

[javax.servlet.jsp.jstl-api-1.2.1.jar](#)

JSTL Implementation

[javax.servlet.jsp.jstl-1.2.1.jar](#)

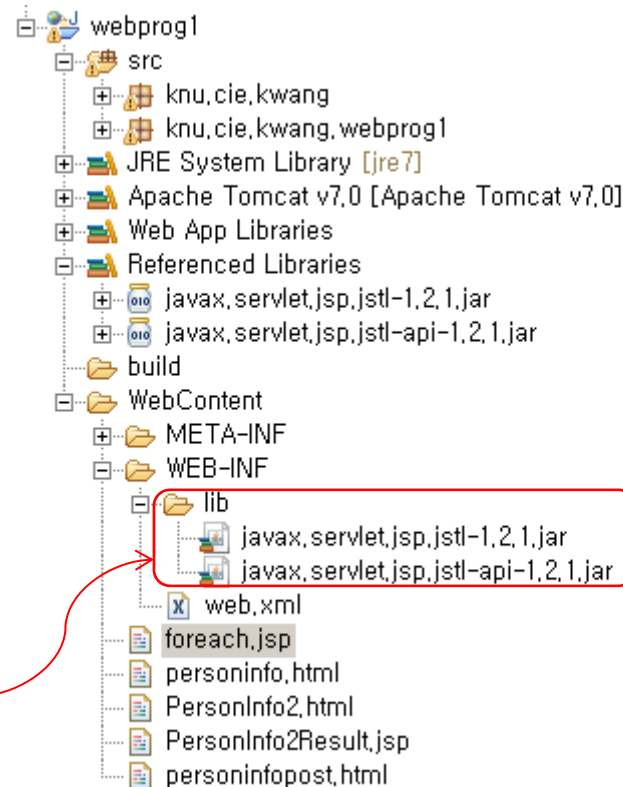


2. JSTL 설치하기

❖ 톰캣에 JSTL 설치하기

- webprog1의 WEB-INF/lib 서브디렉토리를 만들고, JSTL을 설치한다.

① WEB-INF/lib 디렉토리에
JSTL 파일을 복사



② build path에 jar 파일들을 추가



3. 코어 라이브러리 사용하기

❖ <c:set> 커스텀 액션의 사용 방법

- <c:set>은 변수를 선언하고 초기값을 대입하는 커스텀 액션이다.
- 자바 프로그램에서 변수를 선언할 때는 기본적으로 변수의 타입과 이름을 기술하고, 선택적으로 초기값을 기술한다.

```
int num=100;
```

변수의 타입 변수의 이름 초기값

- <c:set> 커스텀 액션을 이용해서 변수를 선언할 때는 변수의 타입을 쓰지 않는다.

```
<c:set var= "num" value= "100" />
```

변수의 이름 초기값

- value 애트리뷰트 값 위치에 EL 식을 쓸 수도 있다.

```
<c:set var= "sum" value= "${num1+num2}" />
```

value 애트리뷰트 값으로 EL 식을 쓸 수도 있습니다

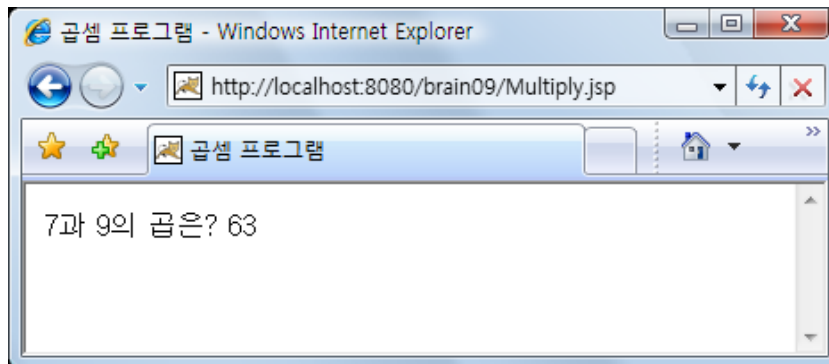


3. 코어 라이브러리 사용하기

❖ <c:set> 커스텀 액션의 사용 방법

[예제 9-1] <c:set> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<c:set var= "num1" value= "7" />
<c:set var= "num2" value= "9" />
<c:set var= "result" value= "${num1*num2}" />
<HTML>
  <HEAD><TITLE>곱셈 프로그램</TITLE></HEAD>
  <BODY>
    ${num1}과 ${num2}의 곱은? ${result}
  </BODY>
</HTML>
```



[그림 9-9] 예제 9-1의 실행 결과



3. 코어 라이브러리 사용하기

❖ <c:set> 커스텀 액션의 사용 방법

- <c:set> 액션을 이용해서 선언한 변수는 page 데이터 영역의 애트리뷰트가 된다.
- <c:set> 태그에 scope 애트리뷰트를 추가하고 page, request, session, application 중 한 값을 지정하면 선언된 변수가 page, request, session, application 데이터 영역의 애트리뷰트가 되도록 지정하는 것도 가능하다.

```
<c:set var= "PRICE " value= "15000 " scope= "request " />
```

변수가 저장될 데이터 영역

- scope 애트리뷰트에 request라는 값을 지정하고 나서 forward 메서드를 통해 다른 JSP 페이지를 호출하면 그 JSP 페이지 안에서도 선언된 변수를 사용할 수 있다.



3. 코어 라이브러리 사용하기

❖ <c:set> 커스텀 액션의 사용 방법

[예제 9-2] <c:set> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<c:set var= "CODE" value= "80012" scope= "request" />
<c:set var= "NAME" value= "온습도계" scope= "request" />
<c:set var= "PRICE" value= "15000" scope= "request" />
<jsp:forward page= "ProductInfoView.jsp" />
```

호출

request 데이터 영역에
데이터를 저장합니다

```
<% @page contentType= "text/html; charset=euc-kr" %>
<HTML>
  <HEAD><TITLE>상품 정보</TITLE></HEAD>
  <BODY>
    <H3>상품 정보</H3>
    상품코드: ${CODE} <BR>
    상품명: ${NAME} <BR>
    단가: ${PRICE}원 <BR>
  </BODY>
</HTML>
```

request 데이터 영역에 있는
데이터 값을 가져다가 출력합니다



3. 코어 라이브러리 사용하기

❖ <c:remove> 커스텀 액션의 사용 방법

- <c:set> 액션을 이용해서 선언한 변수는 page, request, session, application 데이터 영역의 애트리뷰트로 저장되므로, 자바 변수와 달리 인위적으로 삭제해야 할 필요가 있다.
- <c:remove> 커스텀 액션은 이런 애트리뷰트를 삭제하는 기능을 한다.

```
<c:remove var= "num" />
```

↑
변수의 이름

- 위 코드는 page, request, session, application 데이터 영역에 저장되어 있는 num이라는 이름의 애트리뷰트를 모두 찾아서 제거한다. 특정 영역의 애트리뷰트만 제거하려면 scope 애트리뷰트를 사용하면 된다.

```
<c:remove var= "code" scope= "request" />
```

↑
request 데이터 영역에 있는 변수를 제거합니다



3. 코어 라이브러리 사용하기

❖ <c:if> 커스텀 액션의 사용 방법

- <c:if> 커스텀 액션은 자바 프로그램의 if 문과 비슷한 역할을 한다.
- 자바 프로그램에서 if 문을 작성하는 방법은 다음과 같다.

조건식

```
if (num1 > num2) {  
    System.out.println( "num1이 더 큼니다. " );  
}
```

조건식의 결과가 true일 때만 실행되는 명령문

- <c:if> 커스텀 액션에서는 조건식을 괄호 안에 쓰는 것은 아니라, test라는 이름의 애트리뷰트 값으로 지정해야 한다.

조건식

```
<c:if test= "${num1 > num2}">  
    num1이 더 큼니다.  
</c:if>
```

조건식의 결과가 true일 때만 출력되는 코드

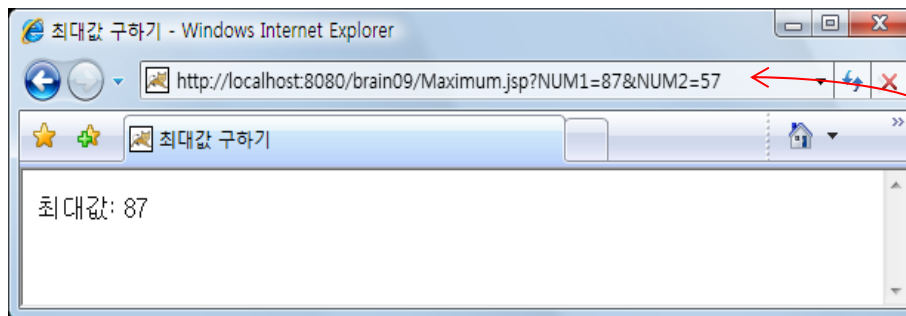


3. 코어 라이브러리 사용하기

❖ <c:if> 커스텀 액션의 사용 방법

[예제 9-3] <c:if> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr"%>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<HTML>
  <HEAD><TITLE>최대값 구하기</TITLE></HEAD>
  <BODY>
    최대값:
    <c:if test= "${param.NUM1 - param.NUM2 >= 0}">
      ${param.NUM1}
    </c:if>
    <c:if test= "${param.NUM1 - param.NUM2 < 0}">
      ${param.NUM2}
    </c:if>
  </BODY>
</HTML>
```



URL 뒤에 이런 식으로 입력 데이터 값을 직접
쓰세요

[그림 9-11] 예제 9-3의 실행 결과



3. 코어 라이브러리 사용하기

❖ <c:choose> 커스텀 액션의 사용 방법

- <c:choose> 커스텀 액션은 자바 프로그램의 switch 문과 비슷한 역할을 한다.
- <c:when>, <c:otherwise>라는 커스텀 액션과 함께 사용되며, 두 커스텀 액션은 각각 switch 문의 case, default 절과 비슷한 역할을 한다.
- 자바 프로그램의 switch 문의 문법은 다음과 같다.

비교의 기준이 되는 변수

```
switch (num) {  
  case 0 :  
    System.out.println( "처음 뵙겠습니다. " );  
    break;  
  case 1 :  
    System.out.println( "반갑습니다. " );  
    break;  
  default :  
    System.out.println( "안녕하세요. " );  
    break;  
}
```

} 첫 번째 조건을 만족할 때
실행되는 명령문

} 두 번째 조건을 만족할 때
실행되는 명령문

} 아무 조건도 만족하지
않을 때 실행되는 명령문



3. 코어 라이브러리 사용하기

❖ <c:choose> 커스텀 액션의 사용 방법

- <c:choose> 커스텀 액션의 전체적인 구조는 switch 문과 비슷하지만, 변수의 이름이 아니라 조건식을 <c:when> 커스텀 액션을 test 애트리뷰트에 EL 식 형태로 지정해야 한다.

조건식을 직접
기술했습니다.

```
<c:choose>
  <c:when test= "${num == 0}">
    처음 뵙겠습니다. <BR>
  </c:when>
  <c:when test= "${num == 1}">
    반갑습니다. <BR>
  </c:when>
  <c:otherwise>
    안녕하세요. <BR>
  </c:otherwise>
</c:choose>
```

첫 번째 조건을 만족할 때
출력할 코드

두 번째 조건을 만족할 때
출력할 코드

아무 조건도 만족하지 않을 때
출력할 코드

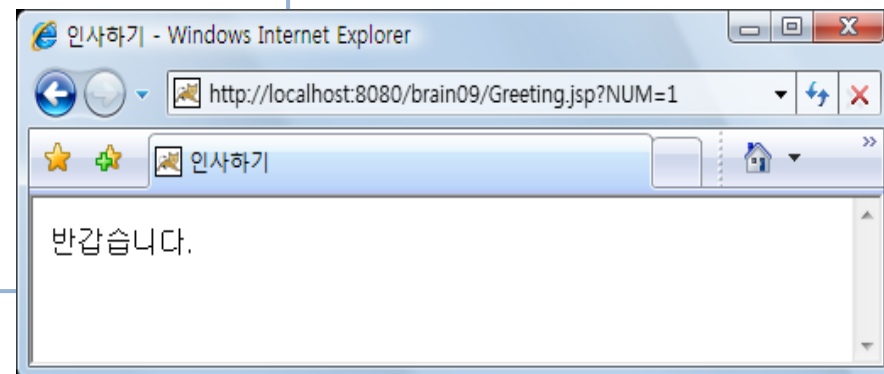


3. 코어 라이브러리 사용하기

❖ <c:choose> 커스텀 액션의 사용 방법

[예제 9-4] <c:choose> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<HTML>
  <HEAD><TITLE>인사하기</TITLE></HEAD>
  <BODY>
    <c:choose>
      <c:when test= "${param.NUM == 0}">
        처음 뵙겠습니다. <BR>
      </c:when>
      <c:when test= "${param.NUM == 1}">
        반갑습니다. <BR>
      </c:when>
      <c:otherwise>
        안녕하세요. <BR>
      </c:otherwise>
    </c:choose>
  </BODY>
</HTML>
```



[그림 9-11] 예제 9-3의 실행 결과



3. 코어 라이브러리 사용하기

❖ <c:forEach> 커스텀 액션의 사용 방법

- <c:forEach> 커스텀 액션은 자바 프로그램의 for 문에 해당하는 기능을 제공한다. 이것을 이용하면 특정 HTML 코드를 지정된 횟수만큼 반복해서 출력할 수 있다.

카운터의 초기값 반복 종료의 기준값 카운터를 증가시키는 식

```
for (int cnt = 0; cnt < 10; cnt++) {  
    System.out.println( "야호" );  
}
```

반복 실행할 명령문

- <c:forEach> 액션을 사용할 때는 begin과 end라는 이름의 애트리뷰트를 쓰고, 거기에 각각 카운터 변수의 시작 값과 끝 값을 지정하면 된다.

시작 값 끝 값

```
<c:forEach begin= "1 " end= "10 ">  
    야호<BR>  
</c:forEach>
```

반복 출력할 명령문



3. 코어 라이브러리 사용하기

❖ <c:forEach> 커스텀 액션의 사용 방법

- 반복 출력할 코드 안에서 카운터 변수의 값을 사용해야 할 경우에는 <c:forEach> 태그 안에 var라는 애트리뷰트를 쓰고, 그 값으로 카운터 변수의 이름을 지정하면 된다.

카운터 변수

```
<c:forEach var= "cnt" begin= "1" end= "10">  
    ${cnt} <BR>  
</c:forEach>
```

- 카운터 변수의 값은 기본적으로 1씩 증가하지만, 그 값을 바꾸려면 <c:forEach> 태그에 step이라는 애트리뷰트를 추가하고 증가치를 지정하면 된다.

증가치

```
<c:forEach var= "cnt" begin= "1" end= "10" step= "2">  
    ${cnt} <BR>  
</c:forEach>
```

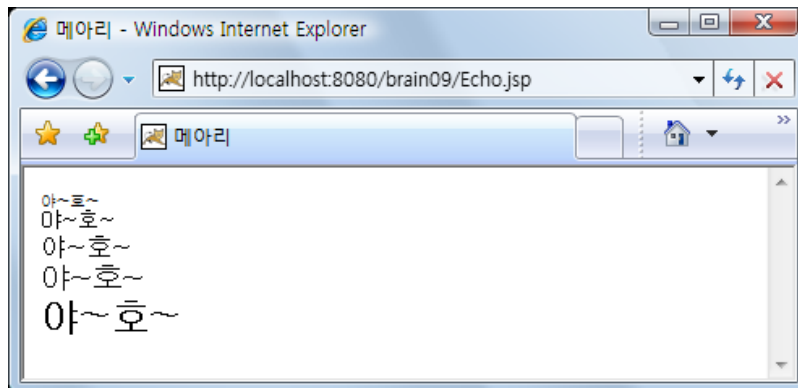


3. 코어 라이브러리 사용하기

❖ <c:forEach> 커스텀 액션의 사용 방법

[예제 9-5] <c:forEach> 커스텀 액션의 사용 예 (1)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<HTML>
  <HEAD><TITLE>메아리</TITLE></HEAD>
  <BODY>
    <c:forEach var= "cnt" begin= "1" end= "5">
      <FONT size=${cnt}> <b>아~호~</b> <BR>
    </c:forEach>
  </BODY>
</HTML>
```



[그림 9-13] 예제 9-5의 실행 결과



3. 코어 라이브러리 사용하기

❖ <c:forEach> 커스텀 액션의 사용 방법

- <c:forEach> 커스텀 액션의 items 애트리뷰트를 이용하면 여러 개의 항목으로 구성된 데이터를 순서대로 출력하는 일도 할 수 있다.

배열의 각 항목을 저장할 변수 배열의 이름

```
<c:forEach var= "str" items= "${arr}">  
    ${str} <BR>  
</c:forEach>
```

- <c:forEach> 액션의 items 애트리뷰트를 이용해서 처리할 수 있는 데이터
 - 배열
 - java.util.Collection 객체
 - java.util.Iterator 객체
 - java.util Enumeration 객체
 - java.util.Map 객체
 - 콤마(,)로 구분된 항목들을 포함한 문자열



3. 코어 라이브러리 사용하기

❖ <C:forEach> 커스텀 액션의 사용 방법

[예제 9-6] <c:forEach> 커스텀 액션의 사용 예 (2)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<%
    String arr[] = { "불고기 백반", "오므라이스", "콩국수" };
    request.setAttribute( "MENU", arr);
%>
<jsp:forward page= "LunchMenuView.jsp" />
```

}

request 데이터 영역에
배열을 저장합니다.

호출

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<HTML>
  <HEAD><TITLE>구내 식당</TITLE></HEAD>
  <BODY>
    <H3>오늘의 점심 메뉴입니다.</H3>
    <UL>
      <c:forEach var= "dish" items= "${MENU}">
        <LI>${dish}</LI>
      </c:forEach>
    </UL>
  </BODY>
</HTML>
```

}

배열 항목을 순서대로 가
져다가 출력합니다.



3. 코어 라이브러리 사용하기

❖ <c:forTokens> 커스텀 액션의 사용 방법

- <c:forTokens> 커스텀 액션은 자바의 for 문과 java.util.StringTokenizer 클래스의 기능을 합친 것 같은 기능을 제공한다.
- 이 액션에는 items, delims, var라는 3개의 애트리뷰트를 써야 한다. items 애트리뷰트에는 토큰을 포함하는 문자열을, delims 애트리뷰트에는 토큰 분리에 사용할 구획 문자를, var 애트리뷰트에는 분리된 토큰을 대입할 변수의 이름을 써야 한다.

토큰을 대입할 변수 토큰을 포함한 문자열 구획 문자

```
<c:forTokens var= "pet" items= "햄스터 이구나나 소라게" delims= " " >  
    ${pet} <BR>  
</c:forTokens>
```

- 토큰의 구획 문자로 한 종류 이상의 문자를 지정할 수도 있다.

토큰을 대입할 변수 토큰을 포함한 문자열 구획 문자

```
<c:forTokens var= "fruit" items= "딸기*키위/체리-참외" delims= "*/-" >  
    ${fruit} <BR>  
</c:forTokens>
```

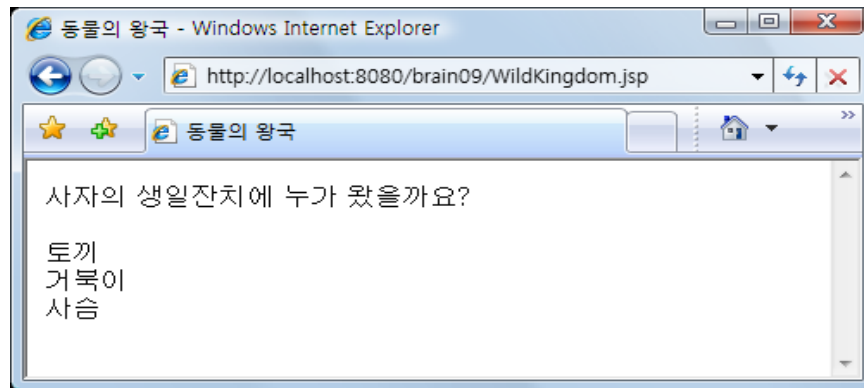


3. 코어 라이브러리 사용하기

❖ <c:forTokens> 커스텀 액션의 사용 방법

[예제 9-7] <c:forTokens> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<HTML>
  <HEAD><TITLE>동물의 왕국</TITLE></HEAD>
  <BODY>
    사자의 생일잔치에 누가 왔을까요? <BR><BR>
    <c:set var= "guests" value= "토끼^^거북이~사슴" />
    <c:forTokens var= "animal" items= "${guests}" delims= "^~" >
      ${animal} <BR>
    </c:forTokens>
  </BODY>
</HTML>
```



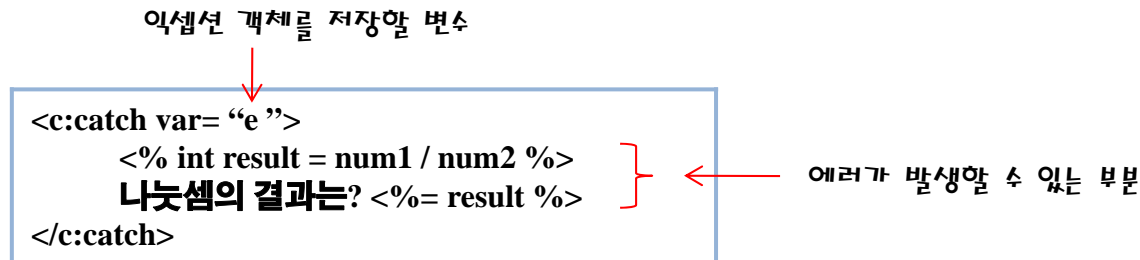
[그림 9-15] 예제 9-7의 실행 결과



3. 코어 라이브러리 사용하기

❖ <c:catch> 커스텀 액션의 사용 방법

- <c:catch> 커스텀 액션은 자바 프로그래밍 언어의 try문과 비슷한 기능을 한다.
- <c:catch> 커스텀 액션의 시작 태그와 끝 태그 사이에서 에러가 발생하면 실행의 흐름이 곧바로 <c:catch> 액션 다음에 있는 코드로 넘어간다.



- <c:catch> 커스텀 액션은 자바의 try 블록에 해당하는 일만 하기 때문에 catch 블록에 해당하는 일은 별도로 코딩해야 한다.



3. 코어 라이브러리 사용하기

❖ <c:catch> 커스텀 액션의 사용 방법

- var 애트리뷰트에 지정된 변수(익셉션 객체가 저장되는 변수)는 <c:catch> 액션의 범위 밖에서도 EL 식을 통해 사용할 수 있으므로, 이를 이용해서 에러 처리를 하면 된다.

익셉션이 발생했는지 체크하는 조건식

```
<c:if test= "${e != null}" >
```

```
    에러 메시지: ${e.message}
```

```
</c:if>
```

← 에러 메시지를 출력하는 코드

- `${e.message}`라는 EL 식은 익셉션 객체 `e`에 대해 `getMessage` 메서드를 호출하는 일을 한다.

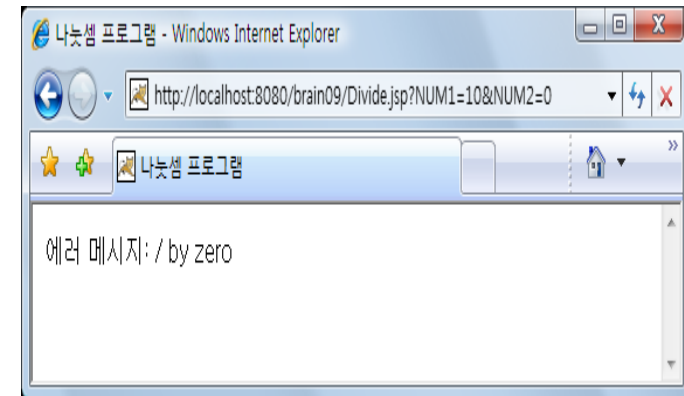


3. 코어 라이브러리 사용하기

❖ <c:catch> 커스텀 액션의 사용 방법

[예제 9-8] <c:catch> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<%
    String str1 = request.getParameter( "NUM1 " );
    String str2 = request.getParameter( "NUM2 " );
    int num1 = Integer.parseInt(str1);
    int num2 = Integer.parseInt(str2);
%>
<HTML>
    <HEAD><TITLE>나눗셈 프로그램</TITLE></HEAD>
    <BODY>
        <c:catch var= "e">
            <% int result = num1 / num2; %>
            나눗셈의 결과는? <%= result %>
        </c:catch>
        <c:if test= "${e != null}" >
            에러 메시지: ${e.message}
        </c:if>
    </BODY>
</HTML>
```



[그림 9-16] 예제 9-8의 실행 결과



3. 코어 라이브러리 사용하기

❖ <c:redirect> 커스텀 액션의 사용 방법

- <c:redirect> 커스텀 액션은 sendRedirect 메서드를 통해 다른 웹 자원을 호출하는 일을 한다.
- 호출할 웹 자원의 URL은 url 애트리뷰트를 이용해서 지정하면 된다.

```
<c:redirect url= "http://www.hanb.co.kr" />
```

↑
호출할 웹 자원의 URL

[예제 9-9] <c:redirect> 커스텀 액션의 사용 예

```
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>  
<c:redirect url= "Multiply.jsp" >  
    <c:param name= "NUM1" value= "5" />  
    <c:param name= "NUM2" value= "25" />  
</c:redirect>
```



3. 코어 라이브러리 사용하기

❖ <c:import> 커스텀 액션의 사용 방법

- <c:import> 커스텀 액션은 <jsp:include> 표준 액션과 비슷하지만 다른 JSP 페이지 뿐만 아니라 다른 종류의 웹 자원도 호출할 수 있다는 점이 다르다.
- 호출할 웹 자원의 URL은 url 애트리뷰트를 이용해서 지정하면 된다.

```
<c:import url= "http://www.hanb.co.kr/binfo/BrainSeries.jsp" />
```

↑
호출할 웹 자원의 URL

- 호출할 웹 자원에 데이터를 넘겨주려면 <c:import> 커스텀 액션의 시작 태그와 끝 태그 사이에 <c:param> 커스텀 액션을 쓰면 된다.

```
<c:import url= "http://www.hanb.co.kr/AdScrap.jsp" >  
  <c:param name= "product" value= "TV" />  
  <c:param name= "ad_index" value= "007" />  
</c:import>
```

↑
데이터 이름

↑
데이터 값



3. 코어 라이브러리 사용하기

❖ <c:url> 커스텀 액션의 사용 방법

- <c:url> 커스텀 액션은 <c:set> 커스텀 액션과 마찬가지로 변수의 선언에 사용되지만, URL을 쉽게 다룰 수 있는 방법을 제공한다는 점이 다르다.
- <c:url>의 기본적인 사용방법은 <c:set>과 동일하다. var 애트리뷰트에 변수 이름을 지정하고, value 애트리뷰트에 변수의 초기값을 지정하면 된다.

```
<c:url var= "myUrl " value= "http://localhost:8080/brain09/Add.jsp " >
```

↑
변수 이름

↑
변수 값

- <c:url>의 시작 태그와 끝 태그 사이에 <c:param> 커스텀 액션을 쓰면, URL 뒤에 쿼리 스트링 형태로 덧붙는 데이터를 지정할 수 있다.

```
<c:url var= "myUrl " value= "http://localhost:8080/brain09/Add.jsp " >  
  <c:param name= "NUM1 " value= "999 " />  
  <c:param name= "NUM2 " value= "1 " />  
</c:url>
```

↑
데이터 이름

↑
데이터 값

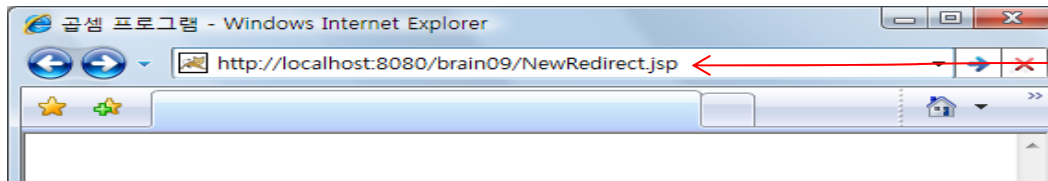


3. 코어 라이브러리 사용하기

❖ <c:url> 커스텀 액션의 사용 방법

[예제 9-10] <c:url> 커스텀 액션의 사용 예

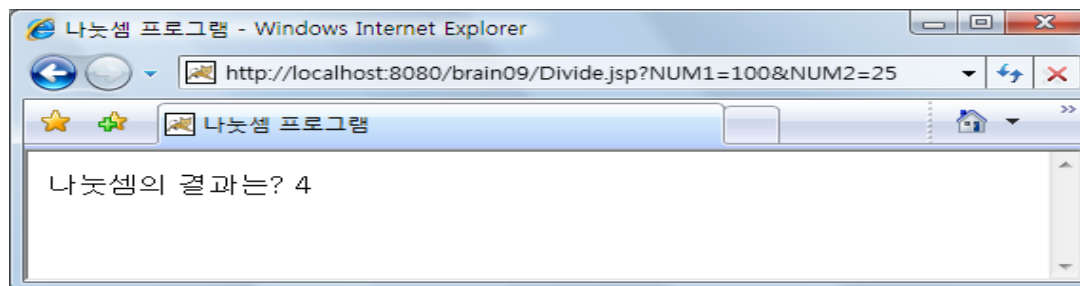
```
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core " %>
<c:url var= "next" value= "Divide.jsp" >
    <c:param name= "NUM1 " value= "100 " />
    <c:param name= "NUM2 " value= "25 " />
</c:url>
<c:redirect url= "${next} " />
```



① (예제 9-10)의 URL을
입력하고 Enter 키를
누르면



② <c:redirect> 태그의 url 애트리뷰트
값에 해당하는
웹 페이지가 나타납니다.



3. 코어 라이브러리 사용하기

❖ <c:out> 커스텀 액션의 사용 방법

- <c:out> 커스텀 액션은 데이터를 출력할 때 사용하는 커스텀 액션인데, 웹 브라우저에 의해 특수한 문자로 해석되는 <, >, &, ', “를 포함하는 데이터를 출력할 때 편리하다.
- 출력할 데이터는 value 애트리뷰트에 지정하면 된다.

```
<c:out value= “<INPUT>은 <FORM>의 서브엘리먼트입니다.” />
```

이 두 태그는 HTML 태그로 해석되지 않고,
웹 브라우저 상에 그대로 나타납니다

[예제 9-11] <c:out> 커스텀 액션의 사용 예 (1)

```
<% @page contentType= “text/html; charset=euc-kr ” %>
<% @taglib prefix= “c” uri= “http://java.sun.com/jsp/jstl/core ” %>
<HTML>
  <HEAD><TITLE>HTML 문법 설명</TITLE></HEAD>
  <BODY>
    <H3>FONT 태그에 대하여</H3>
    <c:out value= “<FONT size=7>커다란 글씨</FONT>는 다음과 같은 출력을 합니다.” /> <BR><BR>
    <c:out value= “<FONT size=7>커다란 글씨</FONT> ” escapeXml= “false ” />
  </BODY>
</HTML>
```



3. 코어 라이브러리 사용하기

❖ <c:out> 커스텀 액션의 사용 방법

- default 애트리뷰트를 이용하면 출력할 데이터의 디폴트 값을 지정할 수 있다. 이 방법은 EL 식의 결과를 출력할 때 유용하다.

```
<c:out value= "${str}" default= "No Data" />
```

이 값이 없으면 이 값을 대신 출력합니다

[예제 9-12] <c:out> 커스텀 액션의 사용 예 (2)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<HTML>
  <HEAD><TITLE>간단한 인사</TITLE></HEAD>
  <BODY>
    안녕하세요, <c:out value= "${param.ID}" default= "guest" />님
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 날짜와 시각을 포맷하는 <fmt:formatDate> 커스텀 액션

- <fmt:formatDate>는 날짜와 시각을 포맷하는 커스텀 액션이다.
- 이 액션에는 출력할 날짜와 시각을 java.util.Date 클래스 타입의 객체로 넘겨줘야 하므로 먼저 이 클래스의 객체를 만들어야 한다.

```
Date date = new Date();
```

현재의 날짜와 시각을 포함한
Date 객체를 생성합니다.

- <fmt:formatDate> 커스텀 액션의 value 애트리뷰트에 Date 객체를 지정하면 그 객체가 포함하고 있는 날짜가 YYYY.MM.DD 포맷으로 출력된다.

```
<fmt:formatDate value= "${date}" />
```

Date 객체



4. 포매팅 라이브러리 사용하기

❖ 날짜와 시각을 포맷하는 <fmt:formatDate> 커스텀 액션

- <fmt:formatDate> 커스텀 액션은 시각을 출력하기 위한 용도로도 사용 될 수 있다.

```
<fmt:formatDate value= "${date} " type= "time " />
```

시각을 출력하라고 지시하는 애트리뷰트 값

- type 애트리뷰트에 date라는 값을 지정하면 날짜가 출력되고, both라는 값을 넘겨주면 날짜와 시각이 모두 출력된다. 이 중 date는 디폴트 값이다.

```
<fmt:formatDate value= "${date} " type= "both " />
```

날짜와 시각을 모두 출력하라고 지시하는
애트리뷰트 값



4. 포매팅 라이브러리 사용하기

❖ 날짜와 시각을 포맷하는 <fmt:formatDate> 커스텀 액션

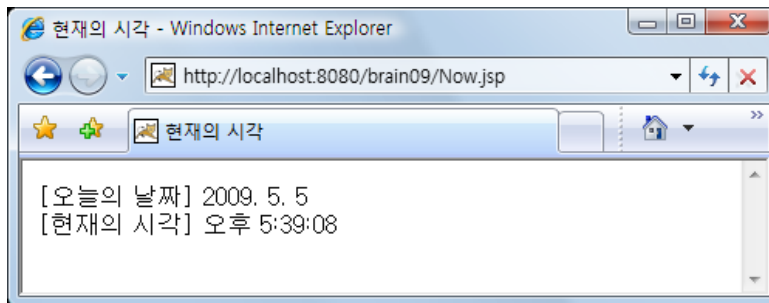
java.util.Date 클래스를 사용하기 위해 필요합니다

[예제 9-13] <fmt:formatDate> 커스텀 액션의 사용 예 (1)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @page import= "java.util.*" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<c:set var= "date" value= "<%= new Date() %>" />
<HTML>
  <HEAD><TITLE>현재의 시각</TITLE></HEAD>
  <BODY>
    [오늘의 날짜] <fmt:formatDate value= "${date}" /> <BR>
    [현재의 시각] <fmt:formatDate value= "${date}" type= "time" />
  </BODY>
</HTML>
```

<fmt:formatDate> 액션에서 Date 객체를 사용하기 위해서는 <c:set> 액션으로 선언한 변수에 저장해야 합니다.

날짜와 시각을 출력합니다.



[그림 9-21] 예제 9-13의 실행 결과



4. 포매팅 라이브러리 사용하기

❖ 날짜와 시각을 포맷하는 <fmt:formatDate> 커스텀 액션

- `dateStyle` 애트리뷰트 `full`, `long`, `medium`, `short` 중 한 값을 넘겨주면 날짜를 다른 포맷으로 출력할 수 있다.

```
<fmt:formatDate type="date" value="${date}" dateStyle="long" />
```

↑
날짜를 '2009년 5월 5일 (화)' 포맷으로
출력하도록 지시합니다

- `timeStyle` 애트리뷰트에 `full`, `long`, `medium`, `short` 중 한 값을 넘겨주면 시각도 다른 포맷으로 출력할 수 있다.

```
<fmt:formatDate type="time" value="${date}" timeStyle="full" />
```

↑
시각을 '오후 1시 31분 42초 KST'
포맷으로 출력하도록 지시합니다

- `type` 애트리뷰트에 `both` 값을 지정해서 날짜와 시각을 한꺼번에 출력할 때는 `dateStyle`과 `timeStyle` 애트리뷰트를 함께 쓸 수 있다.

```
<fmt:formatDate type="both" value="${date}" dateStyle="long" timeStyle="short" />
```

↑ ↑
날짜를 '2009년 5월 5일 (화)' 포맷으로, 시각을 '오후 2:50'
포맷으로 출력하도록 지시합니다.



4. 포매팅 라이브러리 사용하기

❖ 날짜와 시각을 포맷하는 <fmt:formatDate> 커스텀 액션

[예제 9-14] <fmt:formatDate> 커스텀 액션의 사용 예 (2)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @page import= "java.util.*" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<c:set var= "date" value= "<%= new Date() %>" />
<HTML>
  <HEAD><TITLE>현재의 시각</TITLE></HEAD>
  <BODY>
    [S] <fmt:formatDate value= "${date}" type= "both" dateStyle= "short" timeStyle= "short" /> <BR>
    [M] <fmt:formatDate value= "${date}" type= "both" dateStyle= "medium" timeStyle= "medium" /> <BR>
    [L] <fmt:formatDate value= "${date}" type= "both" dateStyle= "long" timeStyle= "long" /> <BR>
    [F] <fmt:formatDate value= "${date}" type= "both" dateStyle= "full" timeStyle= "full" />
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 날짜와 시각을 포맷하는 <fmt:formatDate> 커스텀 액션

- 시각의 포맷도 pattern 애트리뷰트를 이용해서 지정할 수 있다.

```
<fmt:formatDate value= "${date}" type= "time" pattern= "(a) hh:mm:ss" />
```

시각을 '(오후) 5:52:03' 포맷으로 출력하도록 지시합니다

[예제 9-15] <fmt:formatDate> 커스텀 액션의 사용 예 (3)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @page import= "java.util.*" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<c:set var= "date" value= "<%= new Date() %>" />
<HTML>
  <HEAD><TITLE>현재의 시각</TITLE></HEAD>
  <BODY>
    [오늘의 날짜] <fmt:formatDate value= "${date}" type= "date" pattern= "yyyy/MM/dd (E)" /> <BR>
    [현재의 시각] <fmt:formatDate value= "${date}" type= "time" pattern= "(a) hh:mm:ss" />
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 수치를 포맷하는 <fmt:formatNumber> 커스텀 액션

- 출력할 수치 값은 <fmt:formatNumber>의 value 애트리뷰트에 지정하면 된다.

```
<fmt:formatNumber value= "10000" />
```

출력할 수치 데이터

- 세 자리마다 쉼표를 찍은 포맷으로 출력하려면 groupingUsed라는 애트리뷰트를 추가하고, 그 값으로 true를 지정하면 된다.

```
<fmt:formatNumber value= "1234500" groupingUsed= "true" />
```

주어진 값을 '1,234,500' 포맷으로 출력하도록 지시합니다

- pattern 애트리뷰트를 사용하면 소수점 아래의 숫자를 원하는 만큼 끊거나 늘려서 표시할 수 있다.

```
<fmt:formatNumber value= "3.14158" pattern= "#.##" />
```

주어진 값을 소수점 아래 2자리까지 끊어서 출력하도록 지시합니다.



4. 포매팅 라이브러리 사용하기

❖ 수치를 포맷하는 <fmt:formatNumber> 커스텀 액션

- pattern 애트리뷰트의 값에서 0이라고 쓴 위치는 표시할 유효숫자가 없으면 0으로 채워진다.

```
<fmt:formatNumber value= "10.5 " pattern= "#.00 " />
```

주어진 값을 소수점 아래 2자리까지 끊어서 출력하도록 지시합니다

[예제 9-16] <fmt:formatNumber> 커스텀 액션의 사용 예 (1)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<HTML>
  <HEAD><TITLE>숫자 포맷</TITLE></HEAD>
  <BODY>
    첫번째 수: <fmt:formatNumber value= "1234500" groupingUsed= "true" /> <BR>
    두번째 수: <fmt:formatNumber value= "3.14158" pattern= "#.###" /> <BR>
    세번째 수: <fmt:formatNumber value= "10.5 " pattern= "#.00" />
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 수치를 포맷하는 <fmt:formatNumber> 커스텀 액션

- type 애트리뷰트에 percent라는 값을 지정하면 주어진 수치를 퍼센트 단위로 표시할 수 있다.

```
<fmt:formatNumber value= "0.5" type= "percent" />
```

주어진 수치를 퍼센트 단위로 포맷하여
출력하도록 지시합니다

- type 애트리뷰트에 currency라는 값을 지정하면 주어진 수치가 금액에 적합한 포맷으로 만들어져서 출력된다.

```
<fmt:formatNumber value= "2500000" type= "currency" />
```

주어진 수치를 금액으로 표시하여
출력하도록 지시합니다

- 화폐 단위를 표시하기 위해서는 currencySymbol이라는 애트리뷰트를 이용하면 된다.

```
<fmt:formatNumber value= "2500000 " type= "currency " currencySymbol= "₩" />
```

금액 앞에 붙는 화폐 단위 표시

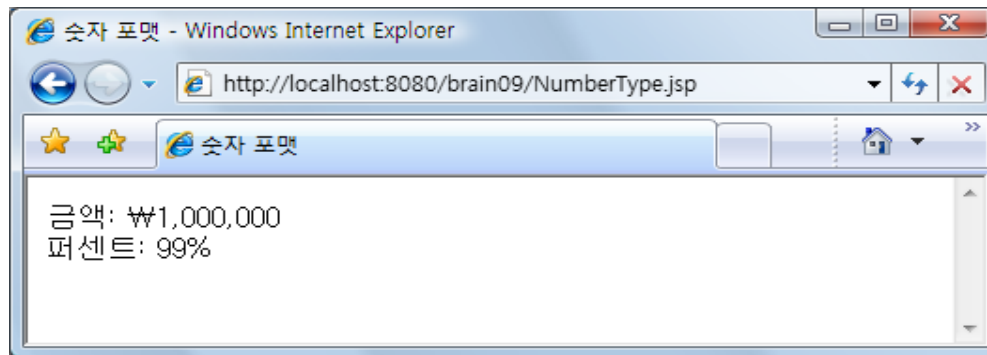


4. 포매팅 라이브러리 사용하기

❖ 수치를 포맷하는 <fmt:formatNumber> 커스텀 액션

[예제 9-17] <fmt:formatNumber> 커스텀 액션의 사용 예 (2)

```
<% @page contentType= "text/html; charset=euc-kr"%>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<HTML>
  <HEAD><TITLE>숫자 포맷</TITLE></HEAD>
  <BODY>
    금액: <fmt:formatNumber value= "1000000 " type= "currency" currencySymbol= "₩" /> <BR>
    퍼센트: <fmt:formatNumber value= "0.99" type= "percent" />
  </BODY>
</HTML>
```



[그림 9-25] 예제 9-17의 실행 결과



4. 포매팅 라이브러리 사용하기

❖ 지역을 설정하는 <fmt:setLocale> 커스텀 액션

- <fmt:setLocale> 커스텀 액션은 출력할 데이터의 포맷을 특정 지역에 맞게 설정하고자 할 때 사용하는 액션이다.
- <fmt:setLocale> 커스텀 액션을 이용해서 특정 지역을 설정하기 위해서는 value 애트리뷰트에 언어 코드 또는 국가코드_언어코드를 지정하면 된다.

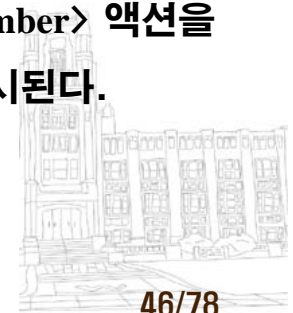
```
<fmt:setLocale value= "en" />
```

언어 코드

```
<fmt:setLocale value= "us_en" />
```

국가 코드 및 언어 코드

- 위의 액션이 실행되고 나면 날짜와 시각이 영어권에 맞게 포맷되고, <fmt:formatNumber> 액션을 이용해서 출력되는 모든 금액 앞에는 달러를 의미하는 \$기호가 자동으로 붙어서 표시된다.



4. 포매팅 라이브러리 사용하기

❖ 지역을 설정하는 <fmt:setLocale> 커스텀 액션

[예제 9-18] <fmt:setLocale> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @page import= "java.util.*" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<c:set var= "date" value= "<%= new Date() %>" />
<HTML>
  <HEAD><TITLE>나라마다 다른 포맷</TITLE></HEAD>
  <BODY>
    <H3>우리나라의 포맷</H3>
    <fmt:setLocale value= "ko_kr" />
    금액: <fmt:formatNumber value= "1000000" type= "currency" /> <BR>
    일시: <fmt:formatDate value= "${date}" type= "both" dateStyle= "full" timeStyle= "full" /> <BR>
    <H3>미국의 포맷</H3>
    <fmt:setLocale value= "en_us" />
    금액: <fmt:formatNumber value= "1000000" type= "currency" /> <BR>
    일시: <fmt:formatDate value= "${date}" type= "both" dateStyle= "full" timeStyle= "full" /> <BR>
    <H3>일본의 포맷</H3>
    <fmt:setLocale value= "ja_jp" />
    금액: <fmt:formatNumber value= "1000000" type= "currency" /> <BR>
    일시: <fmt:formatDate value= "${date}" type= "both" dateStyle= "full" timeStyle= "full" /> <BR>
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 시간대를 설정하는 <fmt:timeZone>과 <fmt:setTimeZone> 커스텀 액션

- <fmt:timeZone>과 <fmt:setTimeZone>은 시간대마다 다른 날짜와 시각을 자동으로 계산해서 표시하기 위해 필요한 커스텀 액션이다.
- <fmt:timeZone> 커스텀 액션의 시작 태그에 value 애트리뷰트를 쓰고 거기에 특정 시간대에 해당하는 지역 이름을 지정하면, 이 액션의 시작 태그와 끝 태그 사이에서 출력되는 날짜와 시각은 그 시간대에 맞게 표시된다.

```
<fmt:timeZone value="America/New_York" >  
    날짜: <fmt:formatDate value= "${date}" type="date" />  
    시각: <fmt:formatDate value= "${date}" type="time" />  
</fmt:timeZone>
```



4. 포매팅 라이브러리 사용하기

❖ 시간대를 설정하는 <fmt:timeZone>과 <fmt:setTimeZone> 커스텀 액션

[예제 9-19] <fmt:timeZone>의 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @page import= "java.util.*" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<c:set var= "date" value= "<%= new Date() %>" />
<HTML>
  <HEAD><TITLE>세계시 프로그램</TITLE></HEAD>
  <BODY>
    서울: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    <fmt:timeZone value= "Asia/Hong_Kong" >
      홍콩: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    </fmt:timeZone>
    <fmt:timeZone value= "Europe/London" >
      런던: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    </fmt:timeZone>
    <fmt:timeZone value= "America/New_York" >
      뉴욕: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    </fmt:timeZone>
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 시간대를 설정하는 <fmt:timeZone>과 <fmt:setTimeZone> 커스텀 액션

- <fmt:setTimeZone> 커스텀 액션은 <fmt:timeZone>처럼 시간대를 설정하는 기능을 하지만, 시작 태그와 끝 태그 사이에만 영향을 미치는 것이 아니라, 이 액션 다음의 모든 코드에 영향을 미친다.

```
<fmt:setTimeZone value= "Europe/London " />
```

시간대의 기준이 되는 지역 이름

[예제 9-20] <fmt:setTimeZone> 커스텀 액션의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr " %>
<% @page import= "java.util.* " %>
<% @taglib prefix= "c " uri= "http://java.sun.com/jsp/jstl/core " %>
<% @taglib prefix= "fmt " uri= "http://java.sun.com/jsp/jstl/fmt " %>
<c:set var= "date " value= "<%= new Date() %> " />
<HTML>
  <HEAD><TITLE>세계시 프로그램</TITLE></HEAD>
  <BODY>
    서울: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    <fmt:setTimeZone value= "Asia/Hong_Kong" />
    홍콩: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    <fmt:setTimeZone value= "Europe/London" />
    런던: <fmt:formatDate value= "${date}" type= "both" /> <BR>
    <fmt:setTimeZone value= "America/New_York" />
    뉴욕: <fmt:formatDate value= "${date}" type= "both" /> <BR>
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- <fmt:setBundle>, <fmt:bundle>, <fmt:message> 커스텀 액션을 사용하면 하나의 JSP 페이지지만 가지고 서로 다른 언어로 기술된 둘 이상의 웹 페이지를 생성할 수 있다.
- 우선 웹 페이지마다 서로 다른 언어로 기술되어야 할 부분을 추출해서 프로퍼티 파일(property file)로 만들어 놓아야 한다.

< 한글 데이터가 저장된 프로퍼티 파일 >

```
TITLE=회사 소개  
GREETING=이 사이트를 방문해주셔서 감사합니다.  
BODY=당사는 소프트웨어 개발을 주업무로 하는 회사입니다.  
COMPANY_NAME=(주) 듀크 소프트웨어
```

< 영문 데이터가 저장된 프로퍼티 파일 >

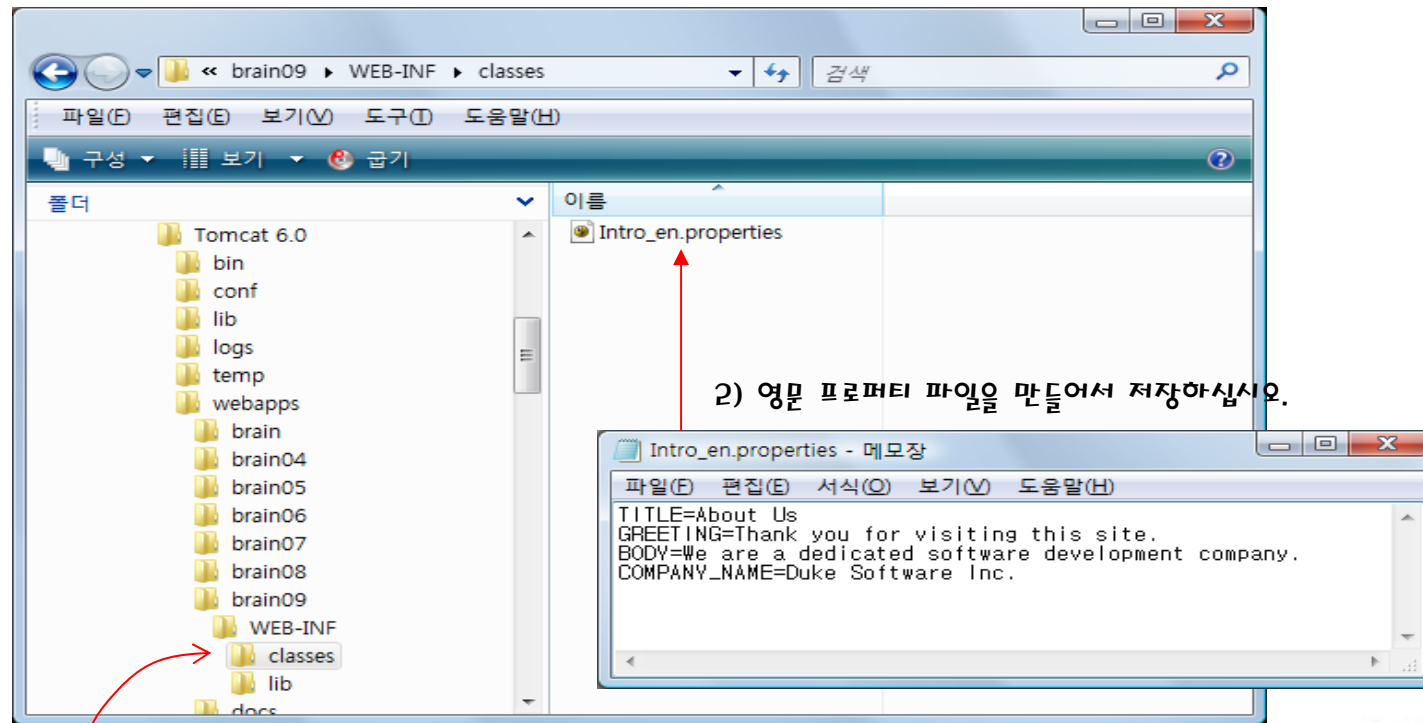
```
TITLE=About Us  
GREETING=Thank you for visiting this site.  
BODY=We are a dedicated software development company.  
COMPANY_NAME=Duke Software Inc.
```

- 프로퍼티 파일은 대표명_ISO언어코드.properties라는 이름으로 만들어서 WB-INF/classes 디렉터리에 저장해야 한다.



4. 포매팅 라이브러리 사용하기

- ❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션
 - 영문 프로퍼티 파일은 일반 텍스트 에디터를 이용해서 만들어 저장하면 된다.



1) brain09 웹 애플리케이션의 WEB-INF/
classes 서브디렉터리로 가십시오.

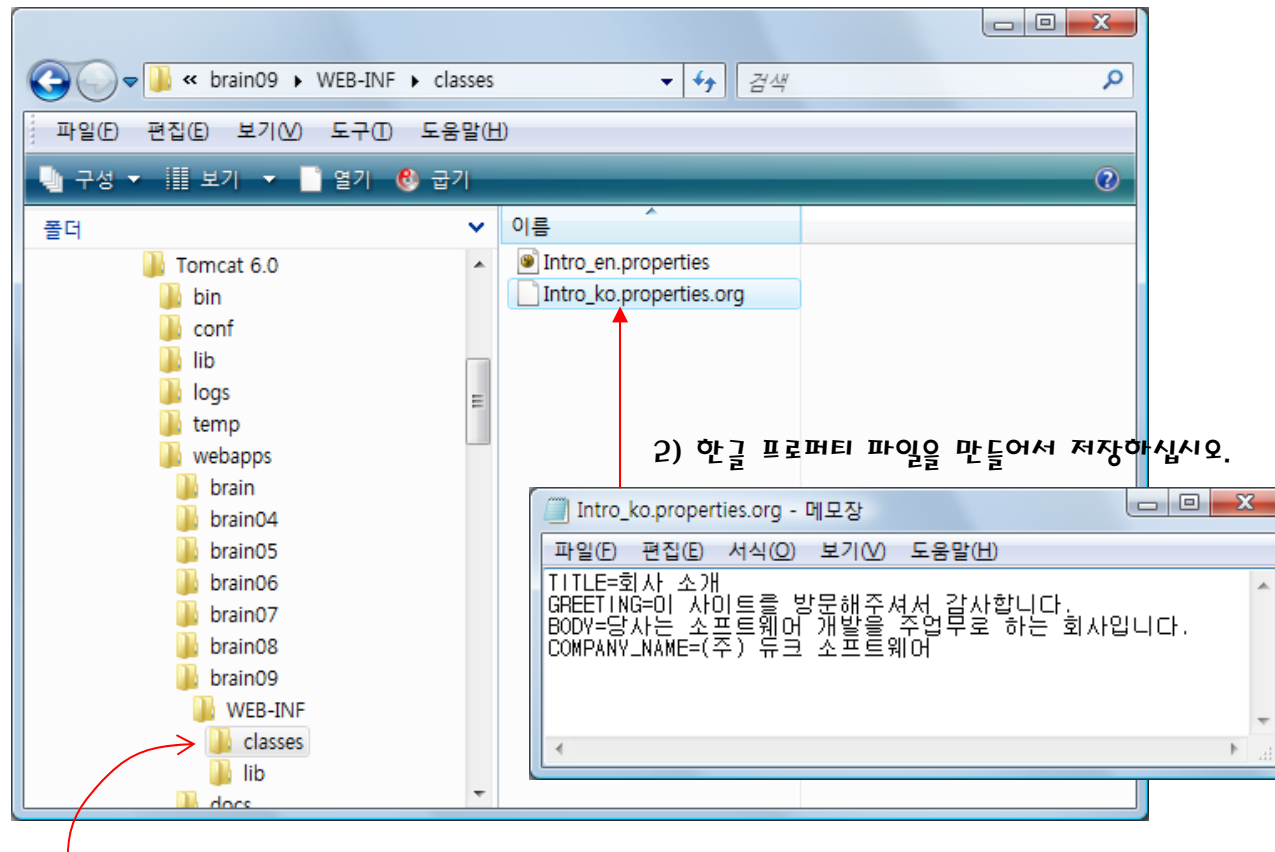
[그림 9-30] 영문 프로퍼티 파일을 만드는 방법



4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- 한글 프로퍼티 파일은 일반 텍스트 에디터를 이용해서 만들어 저장한 다음에 `ascii2native.exe` 프로그램을 이용하여 변환해야 한다.



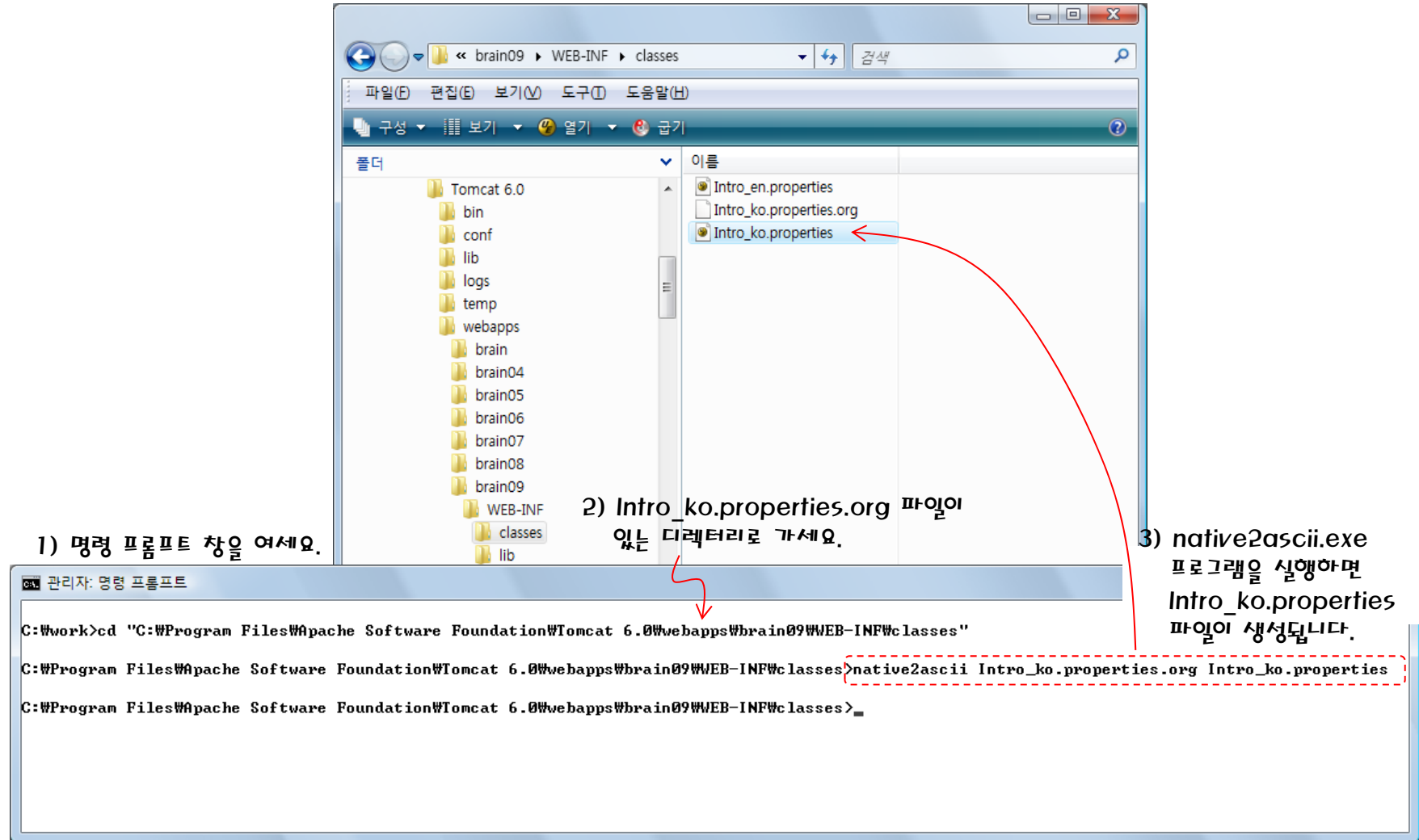
1) brain09 웹 애플리케이션의 WEB-INF/classes 서브디렉터리로 가십시오.

[그림 9-31] 한글 프로퍼티 파일을 만드는 방법 (1)



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션



[그림 9-32] 한글 프로퍼티 파일을 만드는 방법 (2)

4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- <fmt:setBundle> 액션은 프로퍼티 파일의 대표명을 지정하는 역할을 한다.

```
<fmt:setBundle basename= "Intro " />
```

↑
프로퍼티 파일의 대표명

- <fmt:message> 커스텀 액션은 프로퍼티 파일에 있는 데이터를 가져다가 출력하는 역할을 한다.

```
<fmt:message key= "TITLE " />
```

↑
프로퍼티 파일에 있는 데이터의 키

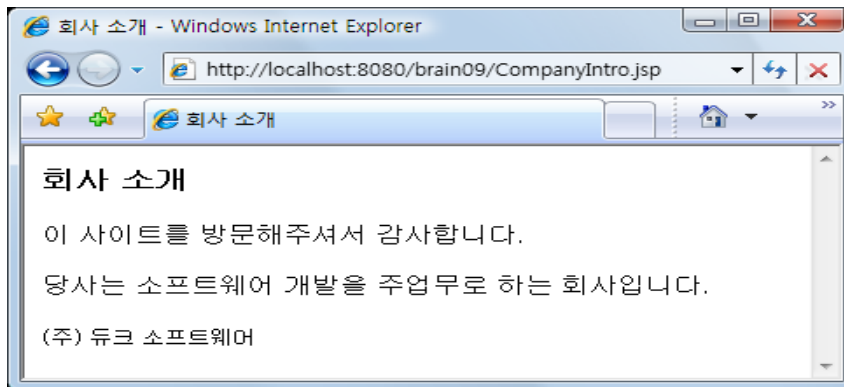


4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

[예제 9-21] <fmt:setBundle> 커스텀 액션의 사용 예 (1)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<fmt:setBundle basename= "Intro" />
<HTML>
  <HEAD><TITLE><fmt:message key= "TITLE" /></TITLE></HEAD>
  <BODY>
    <H3><fmt:message key= "TITLE" /></H3>
    <fmt:message key= "GREETING" /> <BR><BR>
    <fmt:message key= "BODY" /> <BR><BR>
    <FONT size=2><fmt:message key= "COMPANY_NAME" /></FONT>
  </BODY>
</HTML>
```

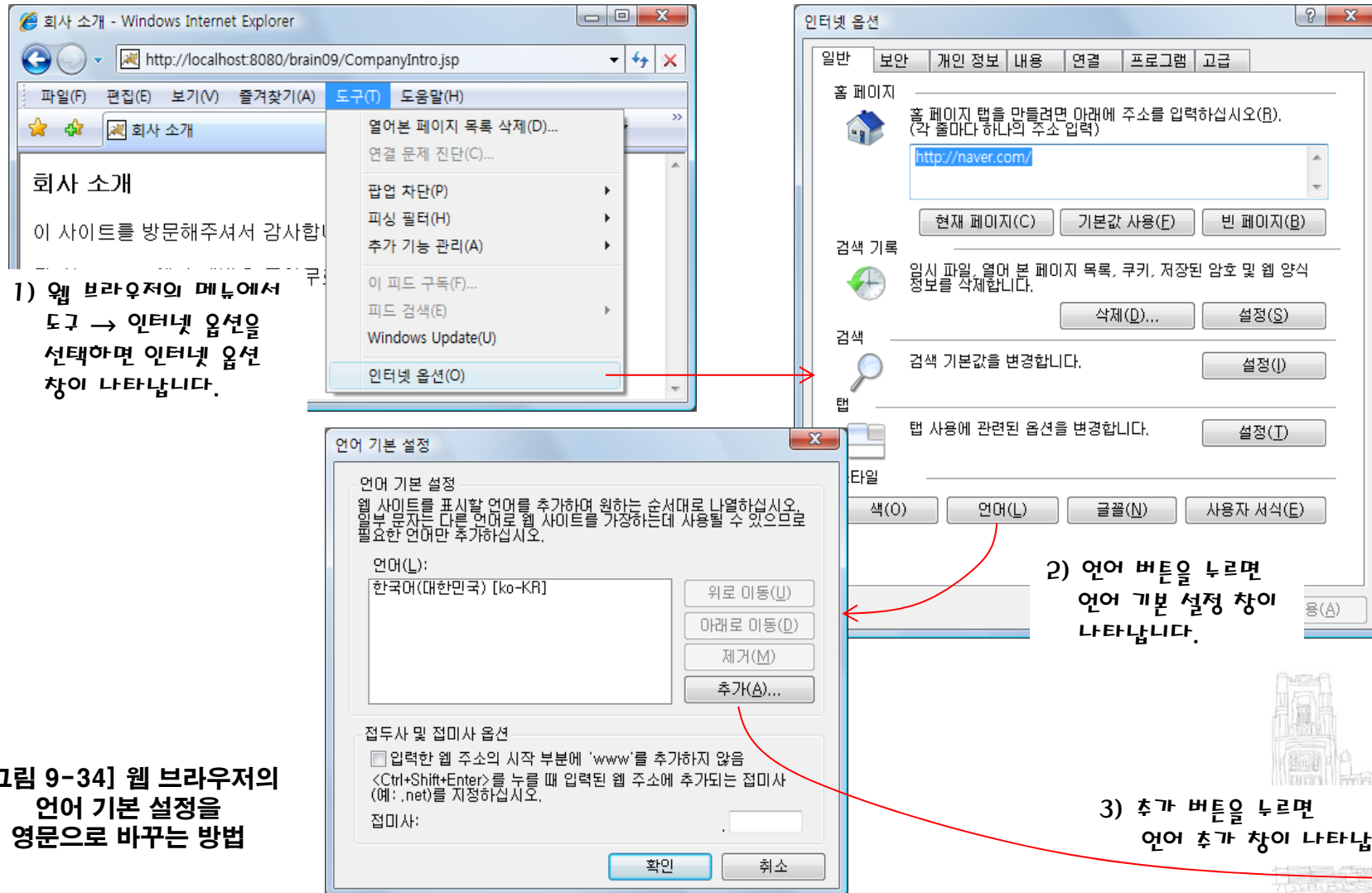


[그림 9-33] 예제 9-21의 실행 결과 (1)



4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션



[그림 9-34] 웹 브라우저의 언어 기본 설정을 영문으로 바꾸는 방법

4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

언어 기본 설정

언어 기본 설정
웹 사이트를 표시할 언어를 추가하여 원하는 순서대로 나열하십시오.
일부 문자는 다른 언어로 웹 사이트를 가장하는데 사용될 수 있으므로
필요한 언어만 추가하십시오.

언어(L):

- 한국어(대한민국) [ko-KR]
- 영어(미국) [en-US]

위로 이동(U) | 아래로 이동(D) | 제거(R) | 추가(A)...

언어 추가

언어(L):

- 아루트어(러시아) [sah-RU]
- 에스토니아어(에스토니아) [et-EE]
- 영어(남아프리카 공화국) [en-ZA]
- 영어(뉴질랜드) [en-NZ]
- 영어(말레이시아) [en-MY]
- 영어(미국) [en-US]
- 영어(벨리즈) [en-BZ]
- 영어(싱가포르) [en-SG]
- 영어(마셜랜드) [en-IE]
- 영어(영국) [en-GB]
- 영어(오스트레일리아) [en-AU]
- 영어(인도) [en-IN]

사용자 지정(B):

확인 | 취소

4) 영어(미국)을 찾아서 선택한 다음에
확인 버튼을 누르면 이 창은 사라집니다.

5) 영어(미국)을 선택하고
위로 이동 버튼을 누르십시오.

언어 기본 설정

언어 기본 설정
웹 사이트를 표시할 언어를 추가하여 원하는 순서대로 나열하십시오.
일부 문자는 다른 언어로 웹 사이트를 가장하는데 사용될 수 있으므로
필요한 언어만 추가하십시오.

언어(L):

- 영어(미국) [en-US]
- 한국어(대한민국) [ko-KR]

위로 이동(U) | 아래로 이동(D) | 제거(R) | 추가(A)...

접두사 및 접미사 옵션

☐ 입력한 웹 주소의 시작 부분에 'www'를 추가하지 않음
<Ctrl+Shift+Enter>를 누를 때 입력된 웹 주소에 추가되는 접미사
(예: .net)을 지정하십시오.

접미사: _____

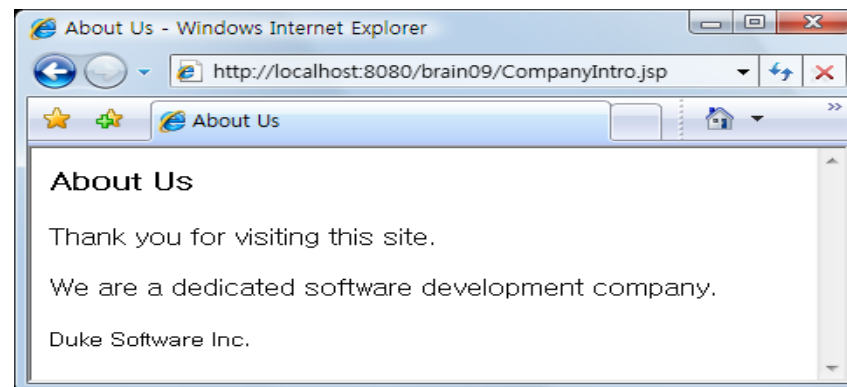
확인 | 취소

b) 영어(미국) 항목이 제일 위에 위치하면
모든 설정이 끝난 것입니다.



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션



[그림 9-35] 예제 9-21의 실행 결과 (2)



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- <fmt:message> 커스텀 액션에 var 애트리뷰트를 사용하면 프로퍼티 파일의 데이터가 출력되는 것이 아니라 변수에 저장된다.

```
<fmt:message var= "title" key= "TITLE" />
```

↑
변수 이름

[예제 9-22] <fmt:setBundle> 커스텀 액션의 사용 예 (2)

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<fmt:setBundle basename= "Intro" />
<fmt:message var= "title" key= "TITLE" />
<fmt:message var= "greeting" key= "GREETING" />
<fmt:message var= "body" key= "BODY" />
<fmt:message var= "companyName" key= "COMPANY_NAME" />
<HTML>
  <HEAD><TITLE>${title}</TITLE></HEAD>
  <BODY>
    <H3>${title}</H3>
    ${greeting} <BR><BR>
    ${body} <BR><BR>
    <FONT size=2>${companyName}</FONT>
  </BODY>
</HTML>
```



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- <fmt:bundle> 커스텀 액션도 <fmt:setBundle>처럼 프로퍼티 파일의 대표명 지정에 사용되지만, 시작 태그와 끝 태그 사이에 있는 코드만 영향을 미친다는 점이 다르다.

```
<fmt:bundle basename= "Intro" >
  <fmt:message var= "title" key= "TITLE" />
  <fmt:message var= "greeting" key= "GREETING" />
  <fmt:message var= "body" key= "BODY" />
  <fmt:message var= "companyName" key= "COMPANY_NAME" />
</fmt:bundle>
```

- <fmt:bundle> 커스텀 액션은 프로퍼티 파일이 적용되는 코드의 범위를 한 눈에 알아볼 수 있게 만들기 때문에 코드의 가독성을 높여주는 효과가 있다.



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- 프로퍼티 파일에도 변수를 사용할 수 있다. 단, 이 때는 영문으로 된 변수 이름이 아니라 숫자로 된 인덱스 값으로 변수를 표시해야 한다.

GREETING=안녕하세요, {0}님 {1}번째 방문하셨습니다.

아이디가 들어갈 부분

방문 횟수가 들어갈 부분

- 변수에 값을 대입하는 일은 JSP 페이지 안에서 해야 한다. 그런 일은 <fmt:message> 액션의 시작 태그와 끝 태그 사이에 <fmt:param> 액션을 써서 할 수 있다.

```
<fmt:message var="greeting" key="GREETING" >
```

```
  <fmt:param>Spiderman</fmt:param> ← {0} 위치에 들어갈 변수 값을 지정합니다.
```

```
  <fmt:param>3</fmt:param> ← {1} 위치에 들어갈 변수 값을 지정합니다.
```

```
</fmt:message>
```



4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

- 변수를 포함하는 프로퍼티 파일을 사용하는 예제를 작성하기 위해 다음과 같은 두 개의 프로퍼티 파일을 만들자

< 한글 데이터가 저장된 프로퍼티 파일 >

```
TITLE=환영 인사
GREETING=안녕하세요, {0}님. {1}번째 방문이시군요.
BODY=새로운 게임이 추가되었습니다. \
      즐거운 시간 보내시기 바랍니다.
COMPANY_NAME=(주) 듀크게임
```

< 영문 데이터가 저장된 프로퍼티 파일 >

```
TITLE=WELCOME
GREETING=Hi, {0}. You have visited this site {1} times.
BODY=New games are added. Have a good time.
COMPANY_NAME=Duke Games Inc.
```



4. 포매팅 라이브러리 사용하기

❖ 다국어를 지원하는 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

[예제 9-23] 프로퍼티 데이터에 변경 가능한 값을 대입해서 출력하는 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<%
    request.setAttribute( "ID ", "Spiderman ");
    request.setAttribute( "VNUM ", new Integer(3));
%>
<jsp:forward page= "WelcomeView.jsp" />
```

호출

request 데이터 영역에 아이디와 방문 횟수를 저장합니다

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<fmt:bundle basename= "Welcome" >
    <fmt:message var= "title" key= "TITLE" />
    <fmt:message var= "greeting" key= "GREETING" >
        <fmt:param>${ID}</fmt:param>
        <fmt:param>${VNUM}</fmt:param>
    </fmt:message>
    <fmt:message var= "body" key= "BODY" />
    <fmt:message var= "companyName" key= "COMPANY_NAME" />
</fmt:bundle>
<HTML>
    <HEAD><TITLE>${title}</TITLE></HEAD>
    <BODY>
        {greeting} <BR><BR>
        ${body} <BR><BR>
        <FONT size=2>${companyName}</FONT>
    </BODY>
</HTML>
```

아이디와 방문 횟수를 포함한 인사 말을 만듭니다

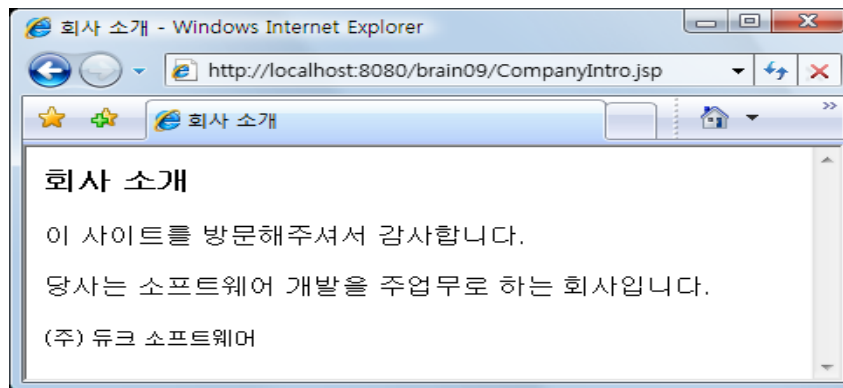
인사말을 출력합니다.



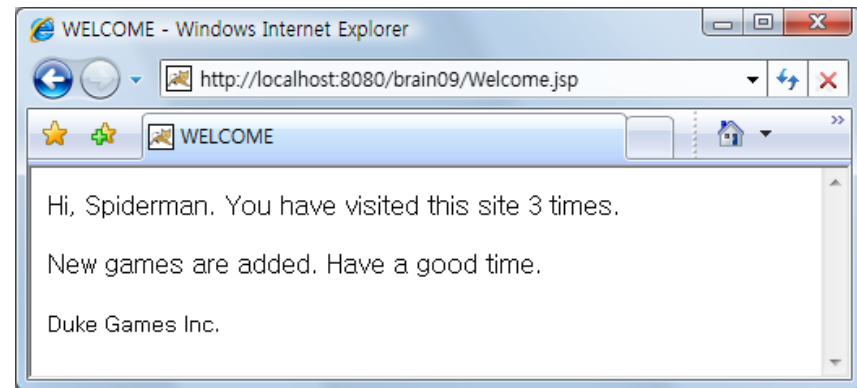
4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션

웹 브라우저가 기본적으로 한글을 사용하도록 설정되어 있을 때



웹 브라우저가 기본적으로 영어를 사용하도록 설정되어 있을 때



[그림 9-38] 예제 9-23의 실행 결과



4. 포매팅 라이브러리 사용하기

❖ POST 메서드로 전송된 한글 입력 데이터를 받기 위해 필요한 커스텀 액션

[예제 9-24] 프로퍼티 데이터에 변수 값을 첨가해서 편집하는 예

아이디를 입력받는 HTML 문서

```
<HTML>
  <HEAD>
    <META http-equiv= "Content-Type " content= "text/html; charset=euc-kr ">
    <TITLE>한글로 인사하기</TITLE>
  </HEAD>
  <BODY>
    <FORM ACTION=HelloResult.jsp METHOD=POST>
      한글 아이디를 입력하세요. <BR>
      <INPUT TYPE=TEXT NAME=ID> <BR><BR>
      <INPUT TYPE=SUBMIT VALUE= '확인' >
    </FORM>
  </BODY>
</HTML>
```

POST 메서드를 이용해서 아이디를 입력받습니다

입력받은 아이디를 가지고 인사말을 출력하는 JSP 페이지

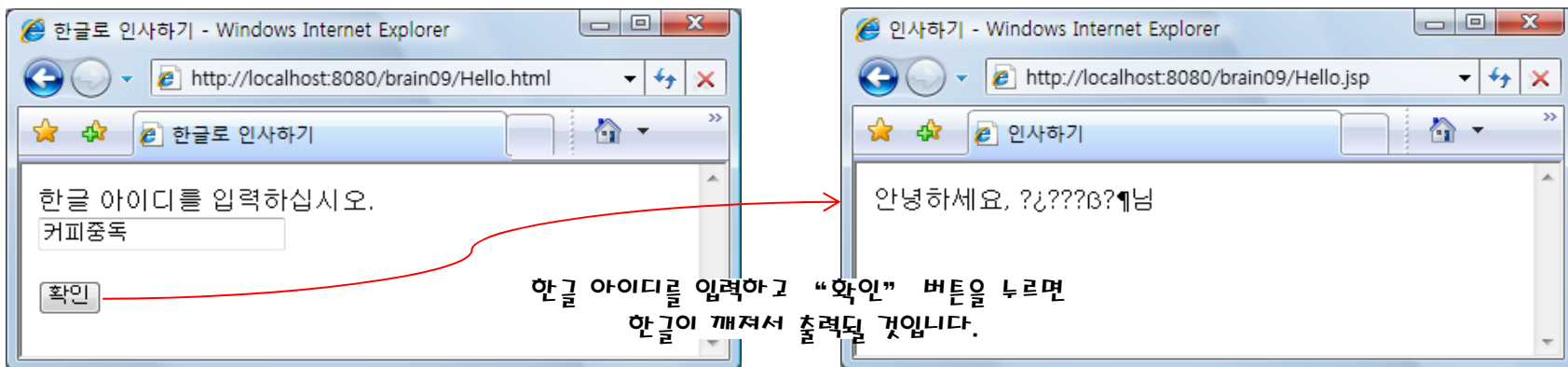
```
<% @page contentType= "text/html; charset=euc-kr "%>
<HTML>
  <HEAD><TITLE>인사하기</TITLE></HEAD>
  <BODY>
    안녕하세요, ${param.ID}님
  </BODY>
</HTML>
```

익스프레션 언어를 이용해서 아이디를 포함한 인사말을 출력합니다.



4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션



[그림 9-39] 예제 9-24의 실행 결과

4. 포매팅 라이브러리 사용하기

❖ POST 메서드로 전송된 한글 입력 데이터를 받기 위해 필요한 커스텀 액션

- POST 메서드로 전송된 한글 입력 데이터를 올바르게 가져오기 위해서는 `setCharacterEncoding` 메서드를 호출해야 한다.
- `<fmt:requestEncoding>` 커스텀 액션은 내부적으로 `setCharacterEncoding` 메서드를 호출한다.

```
<fmt:requestEncoding value= "UTF-8" />
```

↑
한글 코드 이름

[예제 9-25] <fmt:requestEncoding> 커스텀 액션의 사용 예

입력받은 아이디를 가지고 인사말을 출력하는 JSP 페이지

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @taglib prefix= "fmt" uri= "http://java.sun.com/jsp/jstl/fmt" %>
<fmt:requestEncoding value= "UTF-8" />
<HTML>
  <HEAD><TITLE>인사하기</TITLE></HEAD>
  <BODY>
    안녕하세요, ${param.ID}님
  </BODY>
</HTML>
```

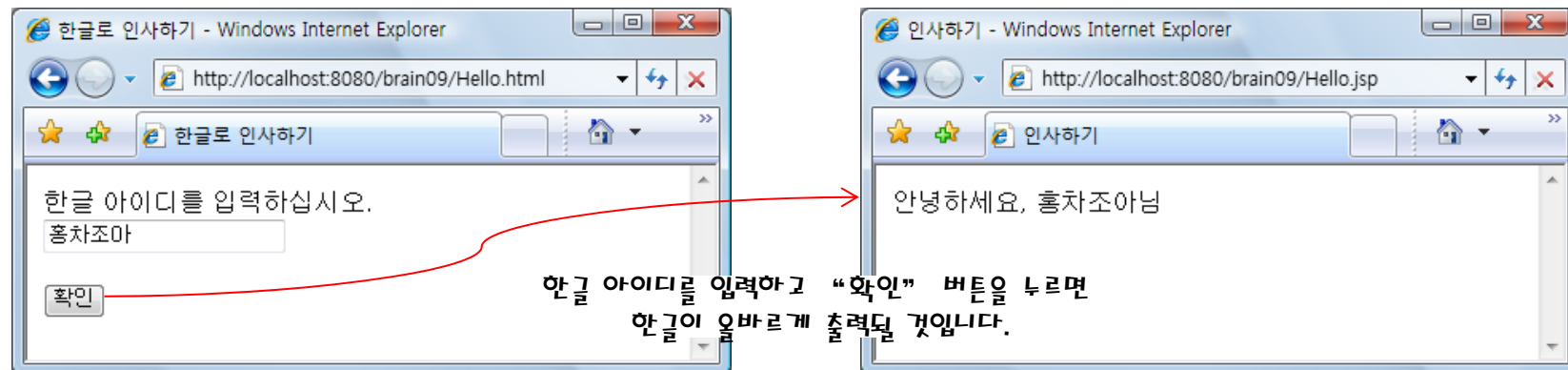
`<fmt:requestEncoding>` 액션을 사용하기 위해 필요합니다.

한글 데이터를 입력받을 수 있도록 만듭니다



4. 포매팅 라이브러리 사용하기

❖ 다국어 지원을 위한 <fmt:setBundle>과 <fmt:bundle> 커스텀 액션



[그림 9-39] 예제 9-24의 실행 결과

5. 함수 라이브러리 사용하기

- JSTL의 함수 라이브러리는 익스프레션 언어의 EL 식 안에서 사용할 수 있는 EL 함수들의 라이브러리이다.
- toUpperCase라는 함수는 문자열에 포함된 모든 영문 소문자를 영문 대문자로 바꾸는 함수이다.

```
${fn:toUpperCase( "Hello. " )}
```

이 함수는 'HELLO.' 를 리턴합니다.

- substring이라는 함수는 문자열의 부분자열을 가져오는 함수이다.

```
${fn:substring( "도레미파솔라시도" , 3, 6 )}
```

이 함수는 '파솔라' 를 리턴합니다



5. 함수 라이브러리 사용하기

■ JSTL의 함수 라이브러리에 있는 함수들

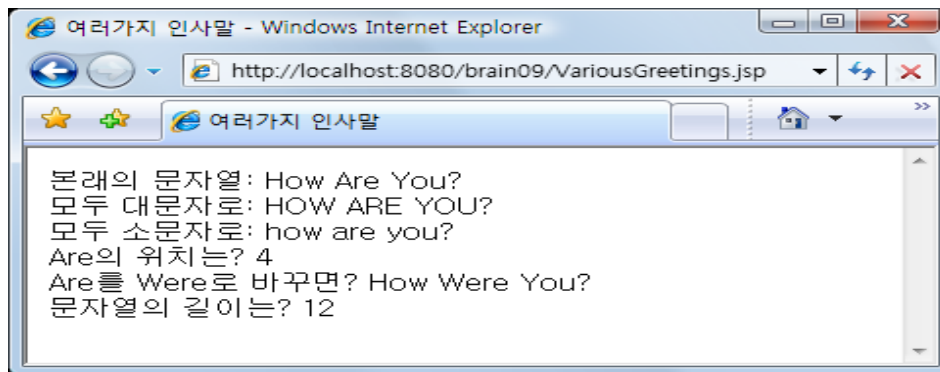
함수	제공하는 기능
substring(str, index1, index2)	str의 index1부터 index2 - 1까지의 부분자열을 리턴
substringAfter(str1, str2)	str1에서 str2를 찾아서 그 후의 부분자열을 리턴
substringBefore(str1, str2)	str1에서 str2를 찾아서 그 전의 부분자열을 리턴
toUpperCase(str)	모든 소문자를 대문자로 치환한 값을 리턴
toLowerCase(str)	모든 대문자를 소문자로 치환한 값을 리턴
trim(str)	문자열에서 앞뒤 공백 문자를 제거한 결과를 리턴
replace(str, src, dest)	str 문자열에 포함된 src를 모두 dest로 치환한 결과를 리턴
indexOf(str1, str2)	str1에 포함된 str2의 시작 인덱스를 리턴
startsWith(str1, str2)	str1이 str2로 시작하면 true, 그렇지 않으면 false를 리턴
endsWith(str1, str2)	str1이 str2로 끝나면 true, 그렇지 않으면 false를 리턴
contains(str1, str2)	str1이 str2를 포함하면 true, 그렇지 않으면 false를 리턴
containsIgnoreCase(str1, str2)	str1이 str2를 포함하면 true, 그렇지 않으면 false를 리턴. contains 함수와는 달리 대소문자를 구별하지 않고 비교함
split(str1, str2)	str1을 str2를 기준으로 분리해서 만든 부분자열들의 배열을 리턴
join(arr, str2)	arr 배열의 모든 항목을 합쳐서 리턴. 항목 사이에는 str2가 들어감
escapeXml(str)	HTML 문법에 의해 특수 문자로 취급되는 모든 문자(표 9-2 참조)를 이스케이프 시퀀스로 치환한 결과를 리턴
length(obj)	obj가 문자열이면 문자열의 길이, List나 Collection이면 항목의 수를 리턴



5. 함수 라이브러리 사용하기

[예제 9-26] 함수 라이브러리의 사용 예

```
<% @page contentType= "text/html; charset=euc-kr" %>
<% @page import= "java.util.*" %>
<% @taglib prefix= "c" uri= "http://java.sun.com/jsp/jstl/core" %>
<% @taglib prefix= "fn" uri= "http://java.sun.com/jsp/jstl/functions" %>
<c:set var= "greeting" value= "How Are You?" />
<HTML>
  <HEAD><TITLE>여러가지 인사말</TITLE></HEAD>
  <BODY>
    본래의 문자열: ${greeting} <BR>
    모두 대문자로: ${fn:toUpperCase(greeting)} <BR>
    모두 소문자로: ${fn:toLowerCase(greeting)} <BR>
    Are의 위치는? ${fn:indexOf(greeting, "Are")} <BR>
    Are를 Were로 바꾸면? ${fn:replace(greeting, "Are", "Were")} <BR>
    문자열의 길이는? ${fn:length(greeting)} <BR>
  </BODY>
</HTML>
```



[그림 9-41] 예제 9-26의 실행 결과





Thank You !

뇌를 자극하는 JSP & Servlet