



JSP 14강 – JSTL

양 명 속

[now4ever7@gmail.com]



목차

- JSTL의 개요
- JSTL이 제공하는 태그의 종류
 - JSTL core
 - JSTL fmt
 - JSTL functions



JSTL의 개요

- JSP 에는 XML 처럼 사용자가 태그를 정의해서 사용하는 것이 가능함
- 사용자 정의 태그를 커스텀 태그라고 하는데 이들 중 자주 사용하는 것을 표준으로 만들어 놓은 것이 JSTL(JSP Standard Tag Library)
- JSTL(JSP Standard Tag Library)
 - 자주 사용하는 커스텀 태그(Custom Tag)의 표준
 - 많은 JSP 어플리케이션을 간단한 태그로 캡슐화함
 - 액션태그는 일종의 시스템이 제공하는 커스텀 태그
 - 예) 자바빈 객체를 생성하는 코드를 <jsp:useBean>액션 태그로 캡슐화

```
<jsp:useBean id="vo" class="study.BoardVO">
```



JSTL의 개요

- JSTL을 사용하면 작업을 수행하는 코드들을 태그로 간략화할 수 있음
- jsp 스크립트 코드들이 사라져서 가독성이 좋아짐
- 로직 부분의 **jsp 코드를 태그로 대체**시켜서 간략화
- jsp 페이지의 로직을 담당하는 부분인 **if, for, while, 데이터베이스 처리 등과 관련된 표준 커스텀 태그를 제공**
- 모든 태그는 시작 태그와 종료 태그의 쌍으로 이루어져야 함

```
<c:if test="$${empty mem.email}">    <!-- 시작 태그 -->
    <input type="text" name="email" size="40" >
</c:if>    <!-- 종료 태그 -->
```

```
<c:set var="name" value="홍길동" scope = "page" />    <!-- 단독 태그 -->
```



JSTL(Java Server Pages Standard Tag Library) 을 사용하기 위한 환경 설정

- <http://jstl.java.net> 에 접속하여, [Download JSTL] 클릭
- [JSTL API], [JSTL Implementation] 을 각각 클릭
 - javax.servlet.jsp.jstl-api-1.2.1.jar 를 다운로드
 - [JSTL Implementation]를 클릭하고, javax.servlet-jsp.jstl-1.2.1.jar 를 다운로드
- 다운로드 받은 두 개의 파일을 이클립스 프로젝트의 /WEB-INF/lib 폴더에 복사
- 태그 라이브러리를 지정
 - `<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>`
 - `<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt"%>`
 - `<%@ taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml"%>`
 - `<%@ taglib prefix="sql" uri="http://java.sun.com/jsp/jstl/sql"%>`
 - `<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"%>`

Java EE 8 Technologies

Java Platform, Enterprise Edition 8 (Java EE 8)	JSR 366
Web Application Technologies	
Java API for WebSocket 1.1	JSR 356
Java API for JSON Binding 1.0	JSR 367
Java API for JSON Processing 1.1	JSR 374
Java Servlet 4.0	JSR 369
JavaServer Faces 2.3	JSR 372
Expression Language 3.0	JSR 341
JavaServer Pages 2.3	JSR 245
Standard Tag Library for JavaServer Pages (JSTL) 1.2	JSR 52



JSTL이 제공하는 태그의 종류

- JSTL 1.2 라이브러리
 - jsp 2.3 규약은 JSTL 1.2 사용
 - JSTL 1.2 라이브러리들은 다음 지정자를 사용해야 함

라이브러리	URI	Prefix(접두어)	예시
Core (코어)	http://java.sun.com/jsp/jstl/core	c	<c:tagname...>
XML processing	http://java.sun.com/jsp/jstl/xml	x	<x:tagname...>
l18N capable formatting	http://java.sun.com/jsp/jstl/fmt	fmt	<fmt:tagname...>
Database access(SQL)	http://java.sun.com/jsp/jstl/sql	sql	<sql:tagname...>
Functions(함수)	http://java.sun.com/jsp/jstl/functions	fn	fn:functionName(...)



(1) JSTL core

- core(코어) - 변수 선언, 삭제 등 변수와 관련된 작업과 if 문, for문 등과 같은 제어문, URL 처리 등에 사용
- core 태그 라이브러리를 사용하려면 jsp 페이지에 `<%@ taglib>` 디렉티브를 작성해야 함

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

- prefix 속성 : uri 속성에 명시된 값 대신에 해당 페이지에서 prefix 속성값으로 명시된 값을 사용하겠다는 것
 - 태그의 시작에 c를 사용하겠다는 것



(1) JSTL core

태 그	설 명
catch	body 위치에서 실행되는 코드의 예외를 잡아내는 역할 사용 시에는 <c:catch>태그로 사용됨
choose	자바의 switch 문과 같지만, 조건에 문자열 비교도 가능 한 개 이상의 <when>과 한 개의 <otherwise>서브 태그를 가짐 <c:choose>태그로 사용됨
if	조건문을 사용할 때. <c:if>
import	웹 어플리케이션 내부의 자원과 http, ftp와 같은 외부에 있는 자 원을 가져옴. 자원을 자유롭게 가공, 편집 가능 <c:import>
foreach	객체 전체에 걸쳐 반복 실행 할 때 사용 <c:forEach>
forTokens	자바의 StringTokenizer 클래스를 사용하는 것과 같음 <c:forTokens>
out	JSP의 표현식을 대체하는 것으로 가장 많이 사용됨 <c:out>
otherwise	<choose>의 서브 태그로 <when>태그 다음에 표시되는 것으로 조건을 만족하지 못한 경우에 사용.<c:otherwise>



(1) JSTL core

태 그	설 명
param	<import>태그의 URL 뒤에 파라미터로 붙여서 사용할 수 있음
redirect	response.sendRedirect() 를 대체하는 태그로, 지정한 다른 페이지로 이동. <c:redirect>
remove	JSP의 removeAttribute()와 같은 역할을 함.(page request session application) 범위의 변수(속성)를 제거. <c:remove>
set	JSP의 setAttribute()와 같은 역할. (page request session application) 범위의 변수(속성)를 설정. <c:set>
url	쿼리 파라미터로부터 URL을 생성함. <c:url>
when	<choose>의 서브 태그로 조건을 만족한 경우에 사용 <c:when>



(1) JSTL core

- JSTL core 의 기능별 분류
 - 표현 언어 지원 기능
 - `<c:catch>`, `<c:out>`, `<c:remove>`, `<c:set>`
 - 흐름 제어 기능
 - `<c:choose>`, `<c:when>`, `<c:otherwise>`
 - `<c:forEach>`, `<c:forTokens>`, `<c:if>`
 - URL 관리 기능
 - `<c:import>`, `<c:param>`,
 - `<c:redirect>`, `<c:param>`,
 - `<c:url>`, `<c:param>`



<c:set>태그

- jsp의 setAttribute() 와 같은 역할을 하며 (page|request|session|application), 범위의 변수(속성)를 설정함

```
<c:set var="varName" value="value" target="targetObjectName"
property="propertyName" scope="{page|request|session|application}" />
```

- <c:set> 태그의 속성
 - var 속성 : 속성값으로 변수명을 갖는다.
 - value 속성 : var 속성의 속성값으로 지정한 변수의 값을 갖는다.
 - target 속성 : 속성값으로 자바빈 객체명이나 Map 객체명이 온다.
 - property 속성 : target 속성은 속성값으로 자바빈 객체나 Map 객체의 값을 설정할 프로퍼티 명이 옴
 - scope 속성 : 변수(속성)의 공유 범위(유효기간)로 page, request, session, application 속성값 중 하나가 온다. 생략시 기본값으로 page가 설정됨



<c:set>태그

- name 변수의 값으로 hong 을 설정
- name 변수의 공유 범위는 현 페이지 내에서만

```
<c:set var="name" value="hong" />  
<c:set var="age" value="20" />
```

- sum 변수의 값으로 price변수를 설정, 공유 범위는 같은 request 내에서만

```
<c:set var="sum" value="${price}" scope="request" />
```

- person 객체의 address 프로퍼티의 값을 seoul 로 설정

```
<c:set value="seoul" target="person" property="address" />
```

<c:out> 태그

- JSP의 표현식을 대체하는 것으로 가장 많이 사용
- 화면에 해당 변수값을 출력

```
<c:out var="varName" default="defaultValue" escapeXml="{true|false}" />
```

■ <c:out>태그의 속성

- var 속성 : 속성값으로 변수명을 갖는다.
- default 속성 : 기본값을 설정하는 부분
- escapeXml 속성 : 속성값으로 true 또는 false 값을 가짐
 - 생략시 기본값은 true, true로 설정시 escapeXml 속성은 값 중에 포함된 < > & ' " 문자들을 각각 < > & ' " 로 출력
- tel 변수가 가진 값을 화면에 표시, tel 값이 null일 경우 공백으로 출력됨

```
<c:out value="${tel}" />
```

```
${tel}
```

```
<%=tel %>
```

```
out.println(tel);
```



<c:remove> 태그

- jsp 의 removeAttribute()와 같은 역할을 함
- (page|request|session|application), 범위의 변수(속성)를 제거함

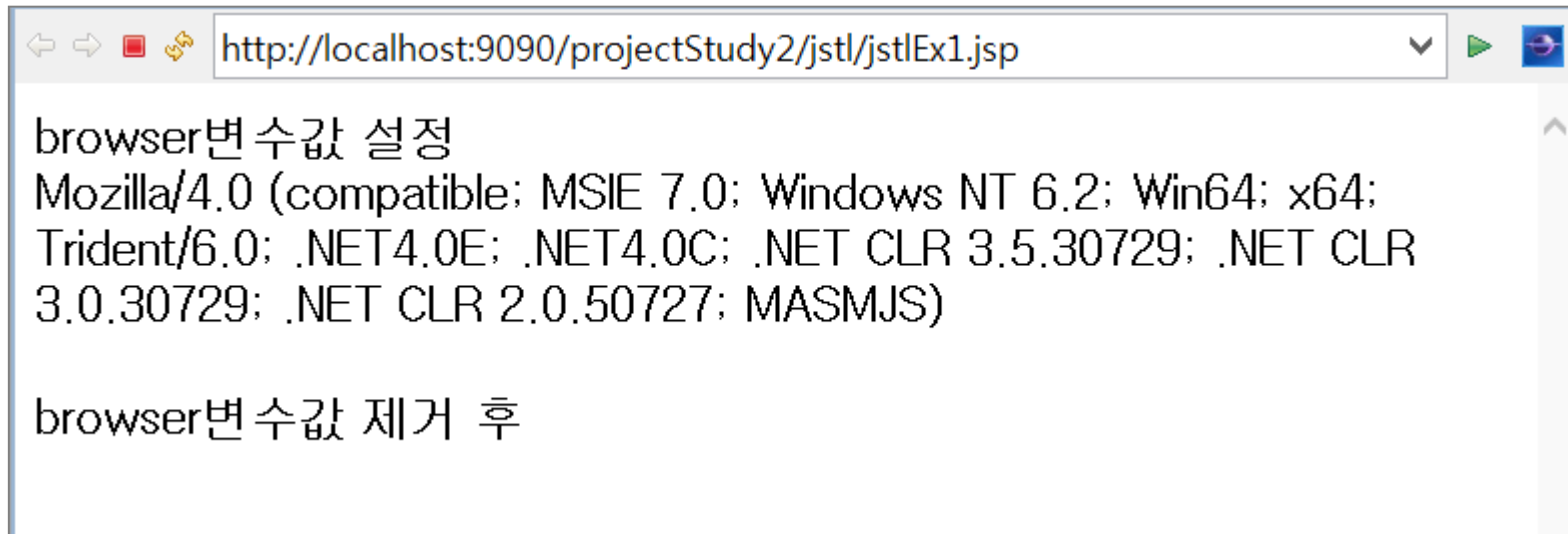
```
<c:remove var="varName" scope="{page|request|session|application}" />
```

- <c:remove>태그의 속성
 - var 속성 : 속성값으로 변수명을 갖는다.
 - scope 속성 : 변수(속성)의 공유 범위(유효기간)로 속성값으로 page, request, session, application 중 하나가 온다. 생략시 기본값으로 page가 설정됨
 - 변수를 제거할 때 scope이 맞지 않으면 제거되지 않음
- scope이 page인 tel 변수의 값을 제거

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

예제-set, out, remove

- JSTL 에서 변수를 하나 설정하고, 화면에 출력 후 다시 해당 변수를 제거하는 예제

A screenshot of a web browser window. The address bar shows the URL 'http://localhost:9090/projectStudy2/jstl/jstlEx1.jsp'. The main content area displays two lines of text: 'browser변수값 설정' followed by a multi-line string of browser and system information, and 'browser변수값 제거 후' below it.

```
browser변수값 설정  
Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; Win64; x64;  
Trident/6.0; .NET4.0E; .NET4.0C; .NET CLR 3.5.30729; .NET CLR  
3.0.30729; .NET CLR 2.0.50727; MASMJS)  
  
browser변수값 제거 후
```




예제 1

```
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<h3>jstl core library 연습</h3>
<c:set var="browser" value="${header['user-agent']}"></c:set>
<h3>el 표현식 이용</h3>
브라우저 정보 : ${browser } <br>
```

```
<h3>jstl 이용</h3>
브라우저 정보: <c:out value="${browser}"></c:out>
```

```
<h3>browser 변수 제거 후</h3>
<c:remove var="browser"/>
브라우저 정보 : <c:out value="${browser }"></c:out>
```



<c:catch>태그

- body 위치에서 실행되는 코드의 예외를 잡아내는 역할

```
<c:catch var="errMsg" />
```

- <c:catch> 태그의 속성
 - var 속성 : 속성값으로 변수명을 갖는다. 이때 에러가 발생하면 속성값으로 지정한 변수에 에러 메시지가 들어감



<c:if>태그

- 조건문을 사용할 때 씬

```
<c:if test="condition" var="varName"  
      scope="{page|request|session|application}" />
```

- <c:if> 태그의 속성

- test 속성 : 속성값에는 조건 판별식이 들어감
- var 속성 : 속성값으로 변수명을 갖는다. if문의 결과값이 var 속성의 값으로 들어감
- scope 속성 : var 속성에서 지정한 변수의 공유 범위를 나타내는 것으로 생략시 page가 기본값

- 조건식 country 변수의 값이 Korea 이면 문장 출력

```
<c:if test="${country == 'Korea'}" />  
    한국입니다.  
</c:if>
```



<c:choose>, <c:when>, <c:otherwise> 태그

- 자바의 switch 문과 같지만, 조건에서 문자열 비교가 가능
- 하나 이상의 <when>과 하나의 <otherwise>서브 태그를 가짐

```
<c:choose>  
  <c:when test="조건">body의 내용</c:when> <!-- 조건 만족시-->  
  <c:otherwise> body의 내용</c:otherwise><!-- 조건 만족 못했을 때-->  
</c:choose>
```

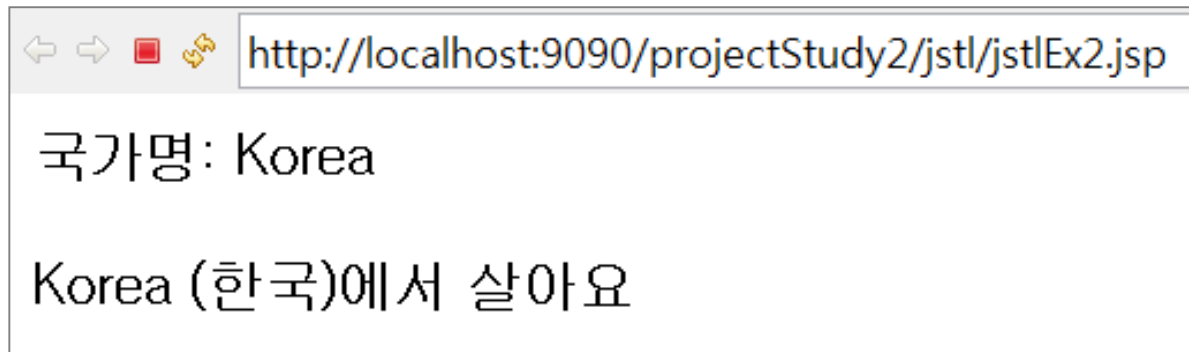
■ <c:when>태그

```
<c:when test="조건">
```

- <c:when>태그의 속성
 - test 속성 : 속성값에는 조건 판별식이 들어감

예제-if, choose, when, otherwise

- JSTL 에서 변수를 하나 설정하고, 그 변수를 <c:if>와 <c:choose>태그에서 조건 판별식으로 사용하는 예제





예제2

```
<%@ page language="java" contentType="text/html; charset=utf-8"    pageEncoding="utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<h3>jstl if, choose when otherwise 연습</h3>
<c:set var="country" value="canada"></c:set>
<h3>if 이용</h3>
<c:if test="${!empty country}">
    ${country } 에서 살아요
</c:if>
<c:if test="${empty country }">
    변수에 값이 없습니다
</c:if>
<h3>choose 이용</h3>
<c:choose>
    <c:when test="${country=='korea' }">
        ${country }(한국)에서 살아요
    </c:when>
    <c:when test="${country=='canada' }">
        ${country }(캐나다)에서 살아요
    </c:when>
    <c:otherwise>
        그외의 나라에서 살아요
    </c:otherwise>
</c:choose>
```



<c:forEach>태그

- 객체 전체에 걸쳐 반복 실행할 때 사용

```
<c:forEach items="condition" begin="begin" end="end" step="step"  
    var="varName" varStatus="varStatus" />
```

- 속성
 - items : 속성값에는 반복할 객체명이 들어감
 - var : 속성값으로 변수명을 갖는다
 - begin : 시작 값
 - end : 마지막 값
 - step : 증가값
 - varStatus : 별도의 변수를 줄 때 사용
- 1~100 까지의 숫자 중에서 2로 나눈 나머지를 출력하는 경우

```
<c:forEach var="k" begin="1" end="100">  
    <c:out value="${k % 2}" />  
</c:forEach>
```

<c:forEach>태그

- CustomerList(컬렉션) 객체로부터 반복해서 값을 추출하는데 추출된 값은 var의 속성인 customerVo 변수에 저장됨
- 나중에 customerVo 변수에 저장된 값을 화면에 표출하거나 다른 식에서 참조할 수 있음

```
<c:forEach var="customerVo" items="${customerList}" >  
    Customer : <c:out value="${customerVo.name}" />  
</c:forEach>
```

```
<%for(int i=0;i<alist.size();i++){  
    PdVO vo = alist.get(i);%>  
    가격 : <%=vo.getPrice()%>  
<%}%>  
<%for(PdVO vo : alist){%>  
    가격: <%=vo.getPrice()%>  
<%}%>  
  
<c:forEach var="vo" items="alist">  
    가격 : ${vo.price}  
</c:forEach>
```


예제-forEach

```
http://localhost:9090/projectStudy2/jstl/jstlEx3.jsp

Header 정보:

param: cache-control
values: no-cache

param: connection
values: Keep-Alive

param: host
values: localhost:9090

param: accept-language
values: ko-KR

param: accept
values: image/jpeg, application/x-ms-application, image/gif, application/xaml+xml, image/pjpeg, application/x-ms-xbap, */*

param: user-agent
values: Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 6.2; Win64; x64; Trident/6.0; .NET4.0E; .NET4.0C; .NET CLR 3.5.3.0.30729; .NET CLR 2.0.50727; MASMJS)

param: accept-encoding
values: gzip, deflate
```

```
<%@ page language="java" contentType="text/html; charset=utf-8"    pageEncoding="utf-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

jstl foreach 연습

1~100까지 짝수의 합 구하기

1~100까지 숫자중에서 짝수 :

```
<c:set var="sum" value="0"></c:set>
```

```
<c:forEach var="i" begin="1" end="100">
```

```
<c:if test="$i%2==0">
```

 $\{i\}$

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

</c:if>

</c:forEach>

1~100까지 짝수의 합 : $\{\text{sum}\}$

<h3>foreach 연습2</h3>

```
<c:forEach var="head" items="${headerValues }">
  header 키(key) : ${head.key} <br>
  header 값(value) :
  <c:forEach var="val" items="${head.value }">
    ${val }
  </c:forEach>
<br><br>
</c:forEach>
```

<c:forEach> 를 사용해서 header로부터 파라미터명과 값을 출력하는 예제



예제

[1] 기존 방식


```
<% Enumeration en = request.getHeaderNames();
    while(en.hasMoreElements()){
        String key =(String)en.nextElement();
        String value = request.getHeader(key);
        out.print(key + " : " + value + "<br><br>");
    }

    int sum=0;
    for(int i=1;i<=10;i++){
        if(i%2==0){
            out.println(i + " ");
            sum+=i;
        }
    }
    out.print("<br>1~10까지의 합 :"+ sum + "<Br>");
%>
```



실습

- 배송비 구하기
 - `sum=0`
 - `totalPrice = 25000`
 - `delivery=0`
 - `totalPrice`가 30000 미만이면 `delivery` 는 3000
 - `sum`은 `totalPrice + delivery`
 - `totalPrice`, `delivery`, `sum` 출력



<c:forTokens>태그

- 자바의 StringTokenizer 클래스를 사용하는 것과 같음
- 문자열을 주어진 구분자로 분할함

```
<c:forTokens items="condition" delims="delimiter" begin="begin"  
            end="end" step="step" var="varName" varStatus="varStatus" />
```

- 속성
 - items : 속성값에는 반복할 객체명이 들어감
 - delims : 문자열을 구분할 구분자가 들어감
 - begin : 반복할 시작 값
 - end : 마지막 값
 - step : 증가값
 - var : 속성값으로 변수명을 갖는다
 - varStatus : 별도의 변수를 줄 때 사용
- “red, yellow, black” 문자열을 “,”로 구분해서 하나씩 출력

```
<c:forTokens items="red, yellow, black" delims="," var="color" />  
    <c:out value="${color}" />  
</c:forTokens>
```

예제-forTokens

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

```
<html>
```

```
<head>
```

```
  <title>JSTL core 예제 - forTokens</title>
```

```
</head>
```

```
<body>
```

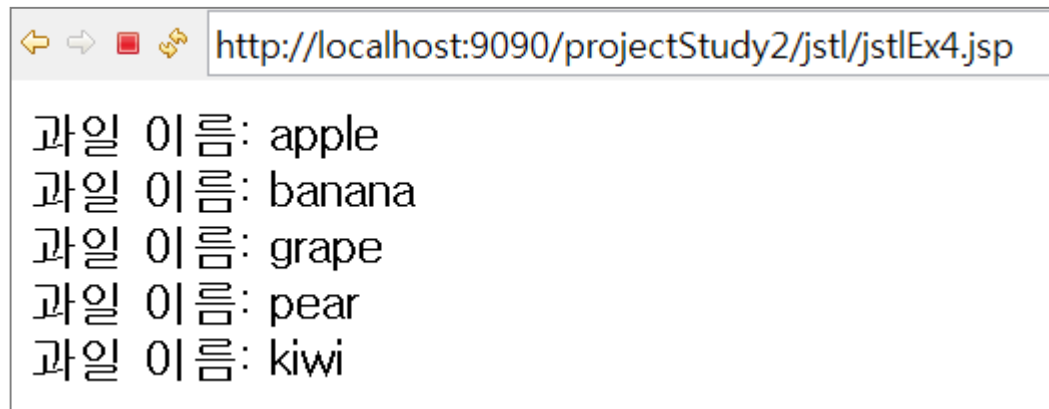
```
<c:forTokens var="fruit" items="apple, banana, grape, pear, kiwi" delims=", ">
```

```
  과일 이름: <c:out value="${fruit}"/><br>
```

```
</c:forTokens>
```

```
</body>
```

```
</html>
```





<c:import>태그

- 웹 어플리케이션 내부의 자원 접근은 물론이고, http, ftp 와 같은 외부에 있는 자원도 가져옴

```
<c:import url="url" var="varName" varReader="varReader"  
charEncoding="charEncoding" scope="{page|request|session|application}" />
```

- 속성
 - url : 읽어올 URL이 들어감
 - var : 읽어올 데이터를 저장할 변수명
 - scope : var 속성에서 지정한 변수의 공유 범위를 나타내는 것으로 생략시 page가 기본값
 - varReader : 리소스의 내용을 Reader 객체로 읽어올 때 사용
 - charEncoding : 읽어온 데이터의 캐릭터셋을 지정

예제-import



http://localhost:9090/projectStudy2/jstl/jstlEx5.jsp

현재 날짜는 2013년01월16일 입니다.

- <c:import>를 사용해서 외부의 페이지를 출력하는 예제

```
<%@ page contentType = "text/html; charset=utf-8" %>
```

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

```
<html>
```

```
<head><title>JSTL core 예제 - import</title></head>
```

```
<body>
```

```
    <c:import url="date1.jsp" var="today"></c:import>
```

```
    <h1>jstl - import </h1>
```

```
    ${today }
```

```
    <hr>
```

```
    <h2>date1.jsp 한 번 더 import</h2>
```

```
    <c:import url="date1.jsp">
```

```
        <c:param name="id" value="hong"></c:param>
```

```
    </c:import>
```

```
</body>
```

```
</html>
```

```
<!-- date1.jsp?id=hong -->
```

```
date1.jsp
```

```
<%
```

```
    Date today = new Date();
```

```
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
```

```
    String nowStr = sdf.format(today);
```

```
%>
```

```
오늘 날짜는 <%=nowStr %>입니다.<br>
```

```
파라미터 id => ${param.id }
```



<c:redirect>태그

- response.sendRedirect() 를 대체하는 태그로 지정한 다른 페이지로 이동함

```
<c:redirect url="url" />
```

- 속성
 - url : 이동할 url 이 들어감
- url 속성의 값에 기술된 jstlEx1.jsp 페이지로 이동

```
<c:redirect url="jstlEx1.jsp" />
```



<c:url>태그

- 쿼리 파라미터로부터 URL을 생성함

```
<c:url var="varName" value="value"  
      scope="{page|request|session|application}" />
```

- 속성

- var : 생성한 URL이 저장될 변수명을 기술함
- value : 생성할 URL이 들어감
- scope : var 속성에서 지정한 변수의 공유 범위를 나타내는 것으로 생략시 page가 기본값

- /customers/register를 URL로 생성해서 registrationURL 변수에 넣는다. 이때 파라미터로 name과 country 값이 들어감
 - /customers/register?name=xxx&country=xxx 로 표시됨

```
<c:url var="registrationURL" value="/customers/register" >  
  <c:param name="name" value="${param.name}" />  
  <c:param name="country" value="${param.country}" />  
</c:url>
```

예제 4

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<link rel="stylesheet" type="text/css" href='<c:url value="/board/css/mainstyle.css"></c:url>'>

<h3>jstl url 연습</h3>
<a href='<c:url value="/elTest/elTest01.jsp" />'>
    elTest01.jsp 페이지로 이동하기
</a>
<br><br>
<img src='<c:url value="/board/images/new.gif" />' border="0">

<br><br>
컨텍스트 패스 :      ${pageContext.request.contextPath}
<br><br>
<img src ="${pageContext.request.contextPath}/board/images/new.gif" border="0">
<br>

```



(3) JSTL fmt

■ JSTL fmt

- JSTL 국제화, 지역화 태그로 다국어 문서를 처리할 때 유용하며 날짜와 숫자 형식을 다룰 때 사용됨

태그	설명
<code>requestEncoding</code>	<code>request.setCharacterEncoding()</code> 와 같은 역할
<code>setLocale</code>	다국어를 지원하는 페이지를 만들기 위해 사용할 경우 ResourceBundle로 불러오는 *.properties 파일들과 연계되어서 사용함
<code>timeZone</code>	타임 존을 적용할 때 사용됨
<code>setTimeZone</code>	특정 scope의 타임 존을 설정할 때 사용됨
<code>bundle</code>	properties 확장자를 사용하는 자원 파일을 읽어오는 역할을 함
<code>setBundle</code>	페이지 전체에서 사용할 수 있는 번들을 지정하는 데 사용됨

(3) JSTL fmt

태그	설명
message	번들 태그에서 정한 값들을 가져옴
param	<fmt:message> 태그의 서브 태그로 <fmt:message>태그에서 설정하지 않은 값을 채워줌
formatNumber	숫자 형식을 표현할 때 사용됨
parseNumber	문자열로부터 수치를 파싱해냄. 즉, 문자열을 숫자로 변환시에 사용됨
formatDate	날짜형식을 표현할 때 사용됨
parseDate	문자열에서 날짜를 파싱해 냄. 문자열을 날짜로 변환시에 사용됨

■ JSTL fmt 기능별 분류

- Locale 설정 : <fmt:setLocale>, <fmt:requestEncoding>
- 메시지 처리 : <fmt:bundle>, <fmt:message>, <fmt:param>, <fmt:setBundle>
- 숫자 날짜 형식 : <fmt:formatNumber>, <fmt:formatDate>, <fmt:parseDate>, <fmt:parseNumber>, <fmt:setTimeZone>, <fmt:timeZone>



<fmt:formatNumber>태그

- 숫자 형식을 표현할 때 사용

```
<fmt:formatNumber value="numericValue" type="{number|currency|percent}"  
  pattern="customPattern" currencyCode="currencyCode"  
  currencySymbol="currencySymbol" groupingUsed="{true|false}"  
  maxIntegerDigits="maxIntegerDigits" minIntegerDigits="minIntegerDigits"  
  maxFractionDigits="maxFractionDigits" minFractionDigits="minFractionDigits"  
  var="varName" scope="{page|request|session|application}" />
```

- 속성

- value : Number 로 형식화될 수치 지정
- type : 숫자, 통화, 퍼센트 중 어느 것으로 표시할 것인지 지정
- pattern : 사용자가 지정한 형식 패턴 지정
- currencySymbol : 통화기호 지정. 통화 형식일 때만 적용



<fmt:formatNumber>태그

- groupingUsed : 출력에 그룹 분리기호를 포함할지 여부 지정
- maxIntegerDigits : 출력에서 Integer 최대 자릿수 지정
- minIntegerDigits : 출력에서 Integer 최소 자릿수 지정
- maxFractionDigits : 출력에서 소수점 이하 최대 자릿수 지정
- minFractionDigits : 출력에서 소수점 이하 최소 자릿수 지정
- var : 출력 결과 문자열을 담는 scope에 해당하는 변수명 지정
- scope : var의 scope 지정



<fmt:parseNumber>태그

- 문자열에서 수치로 파싱

```
<fmt:parseNumber value="numericValue" type="{number|currency|percent}"  
  pattern="customPattern" parseLocale = "parseLocale"  
  IntegerOnly="{true|false}"  
  var="varName" scope="{page|request|session|application}" />
```

- 속성

- value : Number 를 파싱할 수치 지정
- type : 숫자, 통화, 퍼센트 중 어느 것으로 표시할 것인지 지정
- pattern : 사용자가 지정한 형식 패턴 지정
- parseLocale : 파싱작업의 기본 형식 패턴(숫자, 통화, 퍼센트 각각)을 제공하는 Locale로 지정
- integerOnly : 주어진 값에서 integer 부분만 파싱할지 여부를 지정
- var : 파싱 결과를 저장할 scope에 해당하는 변수명 지정
- scope : var의 scope 지정

<fmt:formatDate>태그

- 날짜 형식을 표현할 때 사용

```
<fmt:formatDate value="date" type="{time|date|both}"  
dateStyle="{default|short|medium|long|full}"  
timeStyle="{default|short|medium|long|full}" pattern="customPattern"  
timeZone="timeZone"  
var="varName" scope="{page|request|session|application}" />
```

■ 속성

- value : 형식화될 Date와 time 지정
- type : 형식화할 데이터가 시간, 날짜, 모두의 세 형식 중 하나를 지정
- dateStyle : type 속성을 생략하거나 date 또는 both 일때 사용하는 것으로 미리 정의된 날짜 형식 지정
- timeStyle : type 속성이 time 또는 both 일때 사용하는 것으로 미리 정의된 시간 형식 지정
- pattern : 사용자 지정 형식 스타일 지정
- timeZone : java.util.TimeZone 형식으로 된 시간에 나타날 타임 존 지정
- var : 출력 결과를 문자열로 담는 scope에 해당하는 변수명 지정
- scope : var의 scope 지정



<fmt:parseDate>태그

- 문자열에서 날짜로 파싱할 때 사용

```
<fmt:parseDate value="dateString" type="{time|date|both}"
dateStyle="{default|short|medium|long|full}"
timeStyle="{default|short|medium|long|full}" pattern="customPattern"
timeZone="timeZone" parseLocale="parseLocale"
var="varName" scope="{page|request|session|application}" />
```

- 속성

- value : 파싱할 Date와 time 지정
- type : 파싱할 데이터가 시간, 날짜, 모두의 세 형식 중 하나를 지정
- dateStyle : type 속성을 생략하거나 date 또는 both 일때 사용하는 것으로 미리 정의된 날짜 형식 지정
- timeStyle : type 속성이 time 또는 both 일때 사용하는 것으로 미리 정의된 시간 형식 지정
- pattern : 사용자 지정 형식 스타일 지정
- timeZone : 형식화 시간에 나타날 타임 존 지정
- parseLocale : 파싱하는 동안 적용될 미리 정의된 형식 스타일의 Locale 지정
- var : 파싱 결과를 담는 scope에 해당하는 변수명 지정
- scope : var의 scope 지정



예제

- <fmt:formatNumber>, <fmt:formatDate>를 사용해서 숫자와 날짜 형식을 지정하는 예제

```
<%@ page contentType = "text/html; charset=utf-8" %>
```

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

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

```
number : <fmt:formatNumber value="12345.678" type="number"/><br>
```

```
currency: <fmt:formatNumber value="12345.678" type="currency" currencySymbol="₩"/><br>
```

```
percent : <fmt:formatNumber value="12345.678" type="percent"/><br>
```

```
pattern=".0" : <fmt:formatNumber value="12345.678" pattern=".0"/> <p>
```

```
pattern="#,##0" : <fmt:formatNumber value="12345.678" pattern="#,##0"/> <p>
```

```
<c:set var="now" value="<%= new java.util.Date() %>" />
```

```
<c:out value="${now}"/><br>
```

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

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

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

```
pattern : <fmt:formatDate value="${now}" pattern="yyyy-MM-dd HH:mm:ss" />
```

```
dateStyle - long : <fmt:formatDate value="${now}" type="both" dateStyle="long"/><br>
```

number : 12,345.678
currency: ₩ 12,345.68
percent : 1,234,568%
pattern=".0" :12345.7

pattern="#,##0" :12,346

Sun Mar 31 16:42:45 KST 2013
date : 2013. 3. 31
time : 오후 4:42:45
both : 2013. 3. 31 오후 4:42:45
pattern : 2013-03-31 16:42:45



<fmt:requestEncoding>태그

- request.setCharacterEncoding()와 같은 역할

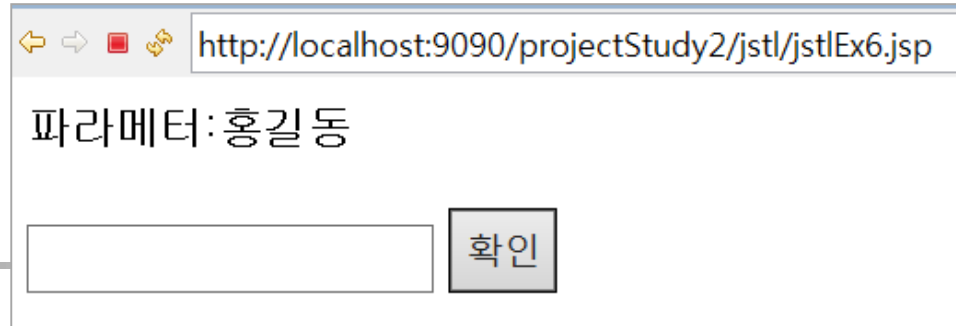
```
<fmt:requestEncoding value="charsetName" />
```

- 속성

- value : 속성값으로 인코딩 값을 기술함
- utf-8 로 인코딩하기

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

예제



파라미터: 홍길동

- `<fmt:requestEncoding>`태그를 사용해서 파라미터의 인코딩을 설정하는 예제

```
<%@ page contentType = "text/html; charset=utf-8" %>
```

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

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

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

```
<html>
```

```
<head><title>JSTL fmt 예제 - requestEncoding</title></head>
```

```
<body>
```

```
파라미터:<c:out value="${param.id}"/>
```

```
request.getParameter("id");
```

```
<form method="post" action="jstlEx6.jsp">
```

```
<input type="text" name="id">
```

```
<input type="submit" value="확인">
```

```
</form>
```

```
</body>
```

```
</html>
```



예제5

```
<%@ page language="java" contentType="text/html; charset=utf-8" pageEncoding="utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<% //사용자가 입력한 정보(post방식의 파라미터) 읽어오기
    //request.setCharacterEncoding("utf-8");
%>
<fmt:requestEncoding value="utf-8"/>
```

```
<h3>el 표현식을 이용해서 파라미터 읽어오기</h3>
<form name="frm1" method="post" action="jstlTest05.jsp">
    이름 : <input type="text" name="name" value="${param.name }">
    아이디 : <input type="text" name="id" value="${param.id }">
    <input type="submit" value="전송"><br>
</form>
```

```
<hr>
<h3>el 표현식 이용</h3>
사용자가 입력한 값(이름) : ${param.name}<br>
아이디 : ${param.id }<br>
```

```
<h3>기존 방식</h3>
이름 : <%=request.getParameter("name")%><br>
아이디 : <%=request.getParameter("id")%>
```




<fmt:setTimeZone>, <fmt:timeZone>태그

- <fmt:setTimeZone> 태그는 특정 스코프의 시간대(타임 존)를 설정

```
<fmt:setTimeZone value="timeZone"
var="varName" scope="{page|request|session|application}" />
```

- 속성

- value : 설정할 시간대를 지정
- var : 시간대 결과를 저장
- scope : var의 scope 지정

- <fmt:timeZone>태그는 타임 존을 적용할 때 사용

```
<fmt:timeZone value="timeZone">
    body 내용
</fmt:timeZone>
```

- value 속성 : 적용할 시간대를 지정

예제



http://localhost:9090/projectStudy2/jstl/jstlEx9.jsp

Korea KST : 2013년 1월 17일 목요일 오후 3시 57분 24초 KST
UK GMT : 2013년 1월 17일 목요일 오전 6시 57분 24초 GMT

- <fmt:timeZone>을 사용해서 시간대별로 시간을 처리하는 예제

```
<%@ page contentType = "text/html; charset=utf-8" %>
```

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

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

```
<html>
```

```
<head><title>JSTL fmt 예제 - timeZone</title></head>
```

```
<body>
```

```
<c:set var="now" value="<%= new java.util.Date() %>" />
```

```
Korea KST : <fmt:formatDate value="${now}" type="both" dateStyle="full" timeStyle="full"/><br>
```

```
UK GMT : <fmt:timeZone value="GMT">
```

```
    <fmt:formatDate value="${now}" type="both" dateStyle="full" timeStyle="full"/>
```

```
</fmt:timeZone>
```

value 값을 GMT 로 설정해서 영국의 시간대를 표시

```
</body>
```

```
</html>
```



<fmt:setLocale>태그

- 다국어를 지원하는 페이지를 만들기 위해 사용할 경우 ResourceBundle로 불러오는 *.properties 파일들과 연계되어서 사용함

```
<fmt:setLocale value="locale" variant="variant"  
[scope="{page|request|session|application}"] />
```

- 속성
 - value : 로케일 값이 들어가며, 두 글자로 된 언어 코드를 반드시 지정해 주어야 함. 언어 코드와 국가 코드는 "-", "_"로 연결함
 - 한글의 경우 ko_kr로 지정할 수 있음
 - variant : 다양한 브라우저의 스펙 등을 기술
 - scope : var 속성에서 지정한 변수의 공유 범위를 나타내는 것, 생략시 page가 기본값.
- 로케일을 한국어로 설정하기

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

언어코드-국가코드

국가	언어	코드	언어 값
오스트레일리아	영어	en-au	3081
오스트리아	독일어	de-at	3079
벨기에(네덜란드어)	네덜란드어	nl-be	2067
벨기에(프랑스어)	프랑스어	fr-be	2060
브라질	포르투갈어	pt-br	1046
캐나다(영어)	영어	en-ca	4105
캐나다(프랑스어)	프랑스어	fr-ca	3084
중국	중국어 간체	zh-cn	2052
체코	체코	cs-cz	1029
덴마크	덴마크어	da-dk	1030
핀란드	핀란드어	fi-fi	1035
프랑스	프랑스어	fr-fr	1036
독일	독일어	de-de	1031
영국	영어	en-gb	2057
그리스	그리스어	el-gr	1032
홍콩	중국어 번체	zh-hk	3076
헝가리	헝가리어	hu-hu	1038
인도	영어	en-in	16393
이스라엘	히브리어	he-il	1037
이탈리아	이탈리아어	it-it	1040
일본	일본어	ja-jp	1041
한국	한국어	ko-kr	1042

국가	언어	코드	언어 값
라틴 아메리카	스페인어	es-la	58378
말레이시아	영어	en-my	17417
멕시코	스페인어	es-mx	2058
네덜란드	네덜란드어	nl-nl	1043
뉴질랜드	영어	en-nz	5129
노르웨이	노르웨이어(북말)	nb-no	1044
노르웨이	노르웨이어(니노르스크)	nn-no	2068
폴란드	폴란드어	pl-pl	1045
포르투갈	포르투갈어	pt-pt	2070
러시아	러시아어	ru-ru	1049
사우디아라비아	아랍어	ar-sa	1025
사우디아라비아	영어	En-xa	61449
싱가포르	영어	en-sg	18441
슬로바키아	슬로바키아어	sk-sk	1051
슬로베니아	슬로베니아어	sl-si	1060
남아프리카	영어	en-za	7177
스페인	스페인어	es-es	3082
스웨덴	스웨덴어	sv-se	1053
스위스(프랑스어)	프랑스어	fr-ch	4108
스위스(독일어)	독일어	de-ch	2055
대만	중국어 번체	zh-tw	1028
터키	터키어	tr-tr	1055
미국(영어)	영어	en-us	1033
미국(스페인어)	스페인어	es-us	21514



<fmt:bundle>태그

- properties 확장자를 사용하는 자원 파일을 읽어오는 역할

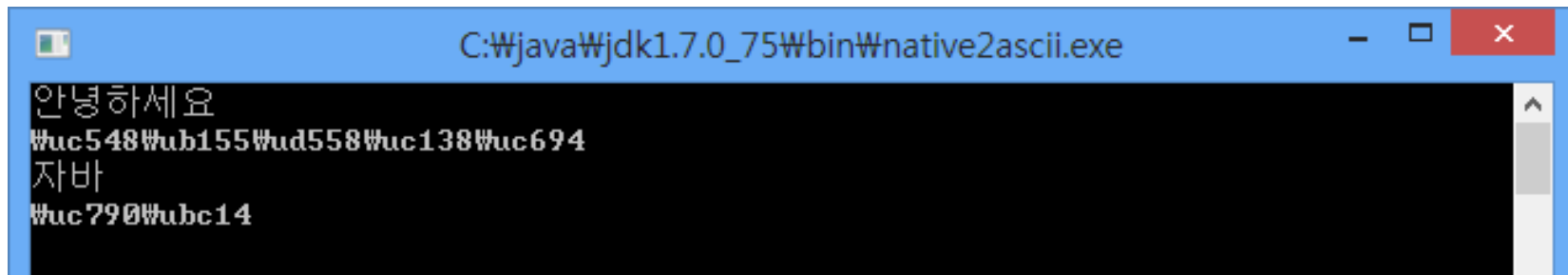
```
<fmt:bundle basename="basename" prefix="prefix" >  
    body내용  
</fmt:bundle>
```

- 속성

- basename : properties 파일명이 기술됨
 - properties 파일은 보통 WEB-INF\classes 폴더 안에 위치하며 디렉토리의 깊이에 따라서 패키지 형식의 이름을 가짐
 - testBundle.properties 파일이 bundle 폴더 안에 있다면 basename의 속성값은 bundle.testBundle로 기술함
 - 로케일이 ko 라면 testBundle_ko.properties 파일을 읽어오게 됨. 로케일이 맞지 않는 경우 testBundle.properties 처럼 언어 코드가 붙지 않은 파일을 읽어옴
- prefix : key 이름이 공통적인 부분을 지정해서 body에서 표현되는 key를 단축시킴

<fmt:bundle>태그

- properties 파일의 내용을 한글로 직접 입력할 수 없기 때문에 한글에 해당하는 유니코드로 변환해서 입력해야 함
- 한글의 유니코드 변환은 j2sdk의 WbinWnative2ascii.exe를 이용
 - C:WjavaWjdk1.7.0_75WbinWnative2ascii.exe



```
C:WjavaWjdk1.7.0_75WbinWnative2ascii.exe
안녕하세요
Wuc548Wub155Wud558Wuc138Wuc694
자바
Wuc790Wubc14
```



<fmt:message>태그

- 번들 태그에서 정한 값들을 가져옴

```
<fmt:message key="messageKey" bundle="resourceBundle"  
    var="varName" scope="{page|request|session|application}" />
```

- 속성

- key : 읽어올 메시지의 key 값이 기술됨
 - bundle : setBundle 태그를 사용해서 로딩한 번들을 읽어올 때 사용
 - var : 메시지를 저장할 변수명 기술
 - scope : 변수가 저장되는 공유 범위를 지정
- key 속성의 속성값이 message에 해당되는 값을 .properties 파일로부터 읽어와 var의 속성에 기술된 msg 변수에 저장

```
<fmt:message key="message" var="msg" />
```




예제

- `<fmt:bundle>` 태그와 `<fmt:message>` 를 사용해서 .properties 파일로부터 설정한 값을 출력하는 예제
- 톰캣홈\webapps\study\WEB-INF\classes 폴더에 하위 폴더로 bundle 폴더 생성
- testBundle.properties
 - name=HAHAHA.
 - message=JSTL is fun.
- testBundle_ko.properties
 - name=\ud558\ud558\ud558.
 - message=JSTL \uc7ac\ubb8\uc788\ub2e4.

하하하
JSTL 재미있다

예제

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
```

```
<fmt:bundle basename="bundle.testBundle">
```

로케일에 따라 WEB-INF\classes\bundle\testBundle.properties 또는 testBundle_ko.properties 가 읽혀짐

```
<html>
```

```
<head><title>JSTL fmt 예제 - bundle , message</title></head>
```

```
<body>
```

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

properties 파일에 있는 name 의 값을 읽어서 화면에 출력

```
<p>
```

```
<fmt:message key="message" var="msg"/>
```

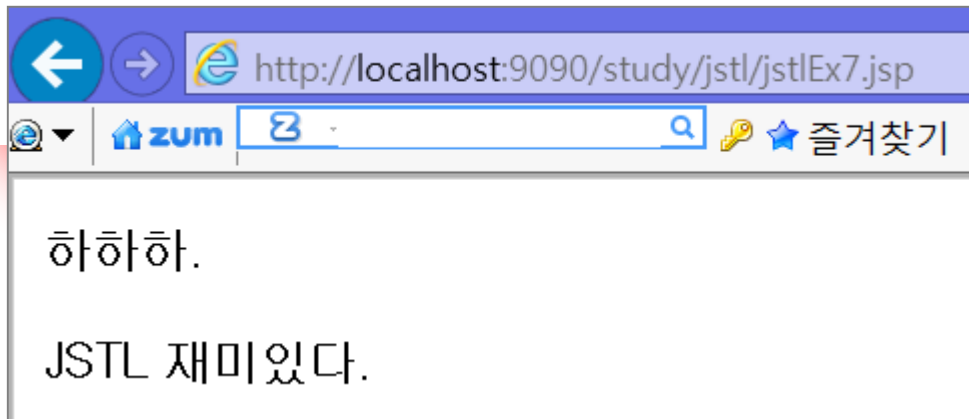
properties 파일에 있는 message 의 값을 읽어와 msg 변수에 넣는다

```
<c:out value="${msg}"/>
```

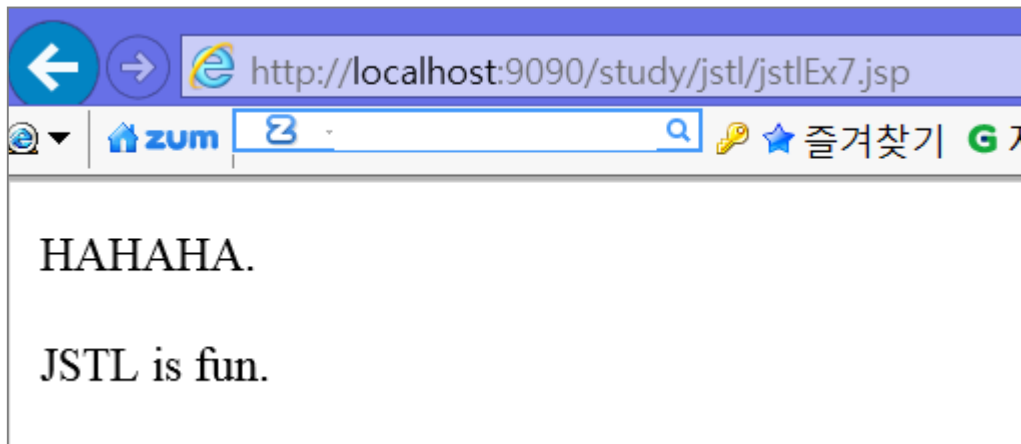
```
</body>
```

```
</html>
```

```
</fmt:bundle>
```



기본 로케일이 한국어인 `ko`로 되어 있어서 `testBundle_ko.properties` 파일이 읽혀짐



로케일을 영어(en)으로 바꾸어서 `testBundle.properties` 파일이 읽히도록 한다.



<fmt:setBundle>태그

- 페이지 전체에서 사용할 수 있는 번들을 지정할 때 사용

```
<fmt:setBundle basename="basename" var="varName"  
    scope="{page|request|session|application}" />
```

- 속성
 - basename : properties 파일명이 기술됨
 - var : 메시지를 저장할 변수명 기술
 - scope : 변수가 저장되는 공유 범위를 지정

(5) JSTL functions

- JSTL에서 제공하는 각종 함수를 사용해서 문자열이나 컬렉션들을 처리

태그	설명
<code>boolean contains(String, substring)</code>	String이 substring을 포함하면 true값을 리턴
<code>boolean containsIgnoreCase(String, substring)</code>	대소문자에 관계없이 String이 substring을 포함하면 true 값을 리턴
<code>boolean endsWith(String, substring)</code>	String이 suffix로 끝나면 true 값 리턴
<code>String escapeXml(String)</code>	String에 XML과 HTML에서 <>&' " 문자들을 각각 < > & ' "로 바꿔준 뒤 문자열을 리턴함
<code>int indexOf(String, substring)</code>	String에서 substring이 처음으로 나타나는 인덱스를 리턴
<code>String join(String[], separator)</code>	배열 요소들을 separator를 구분자로 하여 모두 연결해서 리턴

(5) JSTL functions

태그	설명
<code>int length(item)</code>	item 이 배열이나 컬렉션이면 요소의 개수를, 문자열이면 문자의 개수를 리턴
<code>String replace(String, before, after)</code>	String내에 있는 before 문자열을 after 문자열로 모두 바꿔서 리턴
<code>String[] split(String, separator)</code>	String 내의 문자열을 separator에 따라 잘라내서 잘려진 문자열들을 배열로 구성해서 리턴
<code>boolean startsWith(String, prefix)</code>	String이 prefix로 시작하면 true 리턴
<code>String substring(String, begin, end)</code>	String 에서 begin 인덱스에서 시작해서 end 인덱스에 끝나는 부분의 문자열을 리턴
<code>String substringAfter(String, substring)</code>	String에서 substring이 나타나는 이후의 부분에 있는 문자열을 리턴
<code>String substringBefore(String, substring)</code>	String에서 substring이 나타나는 이전의 부분에 있는 문자열을 리턴
<code>String toLowerCase(String)</code>	String을 모두 소문자로 바꿔 리턴
<code>String toUpperCase(String)</code>	String을 모두 대문자로 바꿔 리턴
<code>String trim(String)</code>	String 앞뒤의 공백을 모두 제거하여 리턴



(5) JSTL functions

```
<%@ 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="str" value="abcd"></c:set>
```

```
${fn:replace(str, "a", "A") } <br>
```

```
${fn:replace("abcd", "a", "A") }
```



예제 7

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

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

```
<h3>jstl functions 연습</h3>
```

```
abcd => Abcd로 바꾸기
```

```
<hr>
```

```
<c:set var="str" value="abcd"></c:set>
```

```
str변수의 값 : ${str } <br>
```

```
replace()함수 사용 후 : ${fn:replace(str, "a", "A") }<br>
```

```
<h4>str변수에서 b의 위치 구하기</h4>
```

```
${fn:indexOf(str, "b") }
```

```
<c:set var="birth" value="1990-07-15"></c:set>
```

```
<h4>birth변수에서 월만 잘라내기</h4>
```

```
${fn:substring(birth, 5, 7) } <br>
```

```
<h4>split()함수 이용</h4>
```

```
<c:set var="result1" value='${fn:split(birth, "-") }'></c:set>
```

```
${result1[0] } 년 ${result1[1] } 월 ${result1[2] } 일
```




예제 7

<h3>함수의 결과를 변수에 넣기</h3>

```
<c:set var="result" value='${fn:replace(str, "a", "A") }'></c:set>
${result}
```



예

```
<input type="text" name="userid" value="${cookie.ck_userid.value}">
```

```
<input type="checkbox" name="chkId"  
    <c:if test="${!empty cookie.ck_userid}">checked</c:if>>  
아이디 저장하기
```

```
<script type="text/javascript">  
    alert("${msg}");  
    location.href="${url}";  
</script>
```

```
<c:if test="${mode=='redirect'}">  
    <script type="text/javascript">  
        alert("${msg}");  
        location.href="${url}";  
    </script>  
</c:if>
```

예

```
<c:if test="${empty sessionScope.userId }">
    <script type="text/javascript">
        alert("먼저 로그인하세요!!!");
        location.href="/herbmall/login/login.do";
    </script>
</c:if>
```

```
request.setAttribute("cartList", list);
request.setAttribute("c_totalPrice",CartService.TOTAL_PRICE);
request.setAttribute("c_deliveryPrice",CartService.DELIVERY_PRICE);
```

```
<c:set var="cartTotalPrice" value="0"/>
<c:set var="totalPrice" value="0"/>
<c:set var="delivery" value="0"/>
<c:set var="sum" value="0"/>

<c:forEach var="cart" items="${cartList }">
    <tr>
        <td align=center>
            
            ${cart.productName} </td>
```



예

```
<td align=right>
    <fmt:formatNumber value="${cart.sellPrice }" pattern="#,##0"/> 원
    </td>

    <td align=center>
        <a href='<c:url value="/shop/cart/cartDelete.do?cartNo=${cart.cartNo}"/>'>삭제</a>
    </td>
</tr> <c:set var = "totalPrice" value="${totalPrice + cart.money }"/>
</c:forEach>

<c:if test="${totalPrice<c_totalPrice}">
    <c:set var = "delivery" value="${c_deliveryPrice}"/>
</c:if>
<c:set var = "sum" value="${totalPrice + delivery}"/>
```



예

```
<a href="<c:url value='/pd/pdList.do' /> ">목록</a>
```

```
<form name="frm1" method="post" action="<c:url value='/pd/pdWrite.do' />" >  
  <input type="text" name="userid" value="{sessionScope.userid}">
```

```
<select name="searchField">  
  <option value="title"  
    <c:if test="{vo.searchField=='title'}">  
      selected  
    </c:if>  
  >제목</option>
```

```
<input type="text" name="id" value="{param.id}">
```



예제 - pdDetail.jsp

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

```
<%
```

```
    //저장된 결과 읽어오기
```

```
    //PdBean bean =(PdBean)request.getAttribute("bean");
```

```
    //DecimalFormat df= new DecimalFormat("#,###");
```

```
%>
```

```
<h3>상품 상세보기2</h3>
```

```
번 호 : ${bean.no}<br>
```

```
상품명 : ${bean.pdName}<br>
```

```
가 격 : <fmt:formatNumber value="${bean.price }" pattern="#,##0"/>원<br>
```

```
등록일 : ${bean.regdate}<br>
```

```
<br>
```

```
<a href="<c:url value='/pd/pdEditForm.do?no=${bean.no}' />">[수정]</a> |
```

```
<a href="<c:url value='/pd/pdList.do' />">[목록]</a><br>
```



예제-pdList.jsp

```
<%@page import="java.text.DecimalFormat"%>
<%@page import="java.text.SimpleDateFormat"%>
<%@page import="com.pd.model.PdBean"%>
<%@page import="java.util.ArrayList"%>
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<% /*      //결과 읽어오기
    ArrayList<PdBean> alist
        =(ArrayList<PdBean>)request.getAttribute("alist");
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
    DecimalFormat df=new DecimalFormat("#,###");
*/ %>
<h3>상품 목록2</h3>
<table width="500" border="1">
    <tr>
        <th>번호</th>
        <th>상품명</th>
        <th>가격</th>
        <th>등록일</th>
    </tr>
```



예제 - pdList.jsp

```
<c:if test="${empty alist}">
    <tr><td colspan="4">상품이 없습니다</td></tr>
</c:if>
<c:if test="${!empty alist}">
    <!-- 반복문 시작 -->
    <c:forEach var="bean" items="${alist }">

        <tr align="center">
            <td>${bean.no}</td>
            <td><a href="<c:url value='/pd/pdDetail.do?no=${bean.no}' />"
                ${bean.pdName}
            </a>
        </td>
            <td align="right">
                <fmt:formatNumber value="${bean.price }" pattern="#,##0"/>원
            </td>
            <td>
                <%-- <fmt:formatDate value="${bean.regdate}" dateStyle="long"/> --%>
                <fmt:formatDate value="${bean.regdate}" pattern="yyyy-MM-dd"/>
            </td>
        </tr>
    </c:forEach>
</c:if>
```




예제 - pdList.jsp

```
</c:forEach>
<!-- 반복문 끝 -->
</c:if>
</table>

<br><br>
<a href="<c:url value='/pd/pdWriteForm.do' />">[등록]</a>
```

예제-게시판 list.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
<link rel="stylesheet" type="text/css"
      href='<c:url value="/board/css/mainstyle.css"/>'>
<h3>공지사항</h3>
<c:if test="${!empty param.keyword}">
    검색어 : ${param.keyword}, ${totalRecord }건이 검색되었습니다.
</c:if>
<table width="600" border="0" cellpadding="0" cellspacing="0">
  <tr><td>
    <table width="100%" border="0" cellpadding="0" cellspacing="0" class="box2">
      <tr>
        <th width="50" height="20">번호</th>
        <th width="300" height="20">제목</th> ... </tr>
      <!--게시판 내용 반복문 시작 -->
        <c:set var="curPos" value="${pb.curPos}"></c:set>
        <c:set var="num" value="${pb.num}"></c:set>
        <c:forEach var="i" begin="1" end="${pb.pageSize }">
          <c:if test="${num}>=1 }">
            <c:set var="bean" value="${alist[curPos] }"></c:set>
            <c:set var="curPos" value="${curPos+1 }"></c:set>
```

```

<tr onMouseOver
    ="this.style.backgroundColor='lightblue';this.style.cursor='hand'"
        onMouseOut="this.style.backgroundColor=''">
    <td>${num } <c:set var="num" value="${num-1 }"></c:set>
    </td>
    <td> <!--제목이 30자 이상인 경우 처리 -->
    <a href='<c:url value="/board/countUpdate.do?no=${bean.no}"></c:url>'>
        <c:if test="${fn:length(bean.title)>30 }">
            ${fn:substring(bean.title, 0, 30) }....
        </c:if>
        <c:if test="${fn:length(bean.title)<=30 }">
            ${bean.title}
        </c:if>
    </a>
    <!-- new 이미지 처리 -->
    <c:if test="${bean.newImgTerm <= 24}">
        <img src='images/new.gif' border='0'>
    </c:if>
    </td>
    <td>
    <c:if test="${!empty bean.email}">
        <a href=mailto:${bean.email}>${bean.name}</a>
    </c:if>
    <c:if test="${empty bean.email}">
        ${bean.name}
    </c:if>
    </td>

```

```

        <td><fmt:formatDate value="${bean.regdate }"
                        type="date" pattern="yyyy-MM-dd"/></td>
        <td>${bean.readcount}</td>
    </c:if>
</c:forEach>
<!--반복처리 끝 -->
</table>
</td> </tr>
<tr>
<td height="30" align="center">
    <!-- 페이지 번호 추가 -->
    <c:if test="${pb.firstPage>1 }">
        <a href='<c:url value="/board/list.do?currentPage=${pb.firstPage-1}"></c:url>'>
            <img src='<c:url value="/board/images/first.JPG"></c:url>' border="0"> </a>
    </c:if>
    <!-- [1][2][3][4][5][6][7][8][9][10] -->
    <c:forEach var="i" begin="${pb.firstPage }" end="${pb.lastPage }">
        <c:if test="${i<=pb.totalPage}">
            <c:if test="${i==pb.currentPage }">
                <span style="color:blue;font-weight:bold">${i }</span>
            </c:if>
            <c:if test="${i!=pb.currentPage }">
                <a href="/board/list.do?currentPage=${i}">[${i }]</a>
            </c:if>
        </c:if>
    </c:forEach>
</td>

```

```

<c:if test="${pb.lastPage<pb.totalPage }">
    <a href="/board/list.do?currentPage=${pb.lastPage+1}">
        
    </a>
</c:if>

```

```

</td>

```

```

</tr>

```

```

<tr>    <td align="right" >

```

```

    <form name="frmSearch" method="post" action='<c:url value="/board/list.do"></c:url>'>

```

```

        <table width="260" border="0" cellpadding="0" cellspacing="0">

```

```

        <tr>

```

```

            <td width="70">

```

```

            <!-- option 태그의 value를 테이블의 컬럼명과 동일하게 주자 -->

```

```

                <select name="category">

```

```

                    <option value="title">제목</option>

```

```

                    <option value="content">내용</option>

```

```

                    <option value="name">작성자</option>

```

```

                </select>

```

```

            </td>

```

```

.....,

```

```

<tr>

```

```

    <td align="right">

```

```

        <a href='<c:url value="/board/writeForm.do"></c:url>' >글쓰기</a>

```

```

    </td>

```

```

</tr>

```

```

</table>

```

```

<h2>글 상세보기</h2>
<div class="divForm">
    <div class="firstDiv">
        <span class="sp1">제목</span> <span>${boardVO.title}</span>
    </div>
    <div>
        <span class="sp1">작성자</span> <span>${boardVO.name}</span>
    </div>
    <div>
        <span class="sp1">등록일</span> <span>${boardVO.regdate}</span>
    </div>
    <div>
        <span class="sp1">조회수</span> <span>${boardVO.readcount}</span>
    </div>
    <div class="lastDiv">
        <p class="content">${boardVO.content}</p>
    </div>
    <div class="center">
        <a href="<c:url value='/board/edit.do?no=${param.no}'/>">수정</a> |
        <a href="<c:url value='/board/delete.do?no=${param.no}'/>">삭제</a> |
        <a href="<c:url value='/board/list.do'/">목록</a>
    </div>
</div>
    <% pageContext.setAttribute("newLine", "WrWn"); %>
    ${fn:replace(boardVO.content, newLine, "<br>")}

```

```

<div class="divForm text_width">
<form name="frmEdit" method="post" action="<c:url value='/board/edit.do' />">
    <!-- hidden 필드-->
    <input type="hidden" name="no" value="${param.no}" />
    <fieldset>
    <legend>글 수정</legend>
        <div class="firstDiv">            <label for="title">제목</label>
            <input type="text" id="title" name="title" value="${boardVO.title}" />
        </div>
        <div>            <label for="name">작성자</label>
            <input type="text" id="name" name="name" value="${boardVO.name}" />
        </div>
        <div>            <label for="pwd">비밀번호</label><input type="password" id="pwd" name="pwd" />
        </div>
        <div>            <label for="email">이메일</label>
            <input type="text" id="email" name="email" value="${boardVO.email}" />
        </div>
        <div> <label for="content">내용</label>
            <textarea id="content" name="content" rows="12" cols="40">${boardVO.content}</textarea>
        </div>
        <div class="center">
            <input type="submit" id="ed_submit" value="수정"/>
            <input type="button" value="글목록" onclick="location.href='<c:url value='/board/list.do' />'" />
        </div>
    </fieldset>
</form> </div>

```

delete.jsp

```
<div class="divForm">
  <form name="frmDelete" method="post" action="<c:url value='/board/delete.do' />" >
    <input type="hidden" name="no" value="${param.no}" />
    <fieldset>
      <legend>글 삭제</legend>
    </div>
    <span class="sp">${param.no} 번 글을 삭제하시겠습니까?</span>
  </div>
  <div>
    <label for="pwd">비밀번호</label>
    <input type="password" id="pwd" name="pwd" />
  </div>
  <div class="center">
    <input type="submit" id="del_submit" value="삭제" />
    <input type="button" value="글목록"
      OnClick="location.href='<c:url value='/board/list.do' />'" />
  </div>
</fieldset>
</form>
</div>
```