


## 12장 표준 태그 라이브러리(JSTL)

### 12.1 JSTL이란?

- JSP Standard Tag Library : 널리 사용되는 커스텀 태그를 표준으로 만든 태그 라이브러리
- JSP 페이지에서 많이 사용되는 논리적인 판단, 반복 처리, 포맷 처리를 위한 커스텀 태그를 표준으로 만들어서 정의한다.

```
<%-- 스크립트 코드(스크립트릿, 표현식)과 HTML 코드 --%>
<%
    if (list.size() > 0) {
        for (int i=0; i<list.size(); i++) {
            Data data = (Data) list.get(i);
        }
        <%= data.getTitle() %>
        ...
    }
} else {
    데이터가 없습니다.
}
%>
```



```
<%-- JSTL 태그 --%>
<ctag:if test="!empty ${list}">
    <ctag:foreach varName="data" list="${list}">
        ${data.title}
    </ctag:foreach>
</ctag:if>
<ctag:if test="empty ${list}">
    데이터가 없습니다.
</ctag>
```

#### 12.1.1 JSTL이 제공하는 태그의 종류

라이브러리	주요 기능	접두어	관련 URL
코어	변수 지원 흐름 제어 URL 처리	c	<a href="http://java.sun.com/jsp/jstl/core">http://java.sun.com/jsp/jstl/core</a>
XML	XML 코어 흐름 제어 XML 변환	x	<a href="http://java.sun.com/jsp/jstl/xml">http://java.sun.com/jsp/jstl/xml</a>
국제화	지역 메시지 형식 숫자 및 날짜 형식	fmt	<a href="http://java.sun.com/jsp/jstl/fmt">http://java.sun.com/jsp/jstl/fmt</a>
데이터베이스	SQL	sql	<a href="http://java.sun.com/jsp/jstl/sql">http://java.sun.com/jsp/jstl/sql</a>
함수	컬렉션 처리 String 처리	fn	<a href="http://java.sun.com/jsp/jstl/functions">http://java.sun.com/jsp/jstl/functions</a>

#### 12.1.2 JSTL 라이브러리 받기

- JSTL을 사용하려면 JSTL 1.2 버전을 구현한 jar 파일을 다운로드해야 한다. 다운로드한



JSTL.jar 파일을 WEB-INF/lib 디렉터리에 복사한다.

- <http://repo1.maven.org/maven2/jstl/jstl/1.2/jstl-1.2.jar>

- 다른 방법은 Apache Maven 파일(pom.xml)에 아래와 같이 추가해 놓는다.

```
<dependency>
  <groupId>jstl</groupId>
  <artifactId>jstl</artifactId>
  <version>1.2</version>
</dependency>
```

## 12.2 코어 태그

기능 분류	태그	설명
변수 지원	set	JSP에서 사용할 변수를 설정한다.
	remove	설정된 변수를 제거한다.
흐름 제어	if	조건에 따라 내부 코드를 수행한다.
	choose	다중 조건을 처리할 때 사용된다.
	forEach	컬렉션이나 Map의 각 항목을 처리할 때 사용된다.
	forEachTokens	구분자로 분리된 각각의 토큰을 처리할 때 사용된다.
URL 처리	import	URL을 사용하여 다른 자원의 결과를 삽입한다.
	redirect	지정한 경로로 리다이렉트 한다.
	url	URL을 재작성 한다.
기타 태그	catch	익셉션을 처리할 때 사용한다.
	out	JspWriter에 내용을 출력한다.

- 코어 태그 라이브러리를 사용하려면 JSP 페이지에 다음과 같이 taglib 디렉티브를 추가해야 한다.



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

### 12.2.1 변수 지원 태그

#### (1) <c:set> 태그

- EL 변수를 설정한다.
- 객체의 프로퍼티 값을 설정한다.

```
<%-- 기본형 --%>
<c:set var="변수명" value="값" [scope="영역"] />
<c:set target="대상" property="프로퍼티이름" value="값" />

<%-- 예시 --%>
<c:set var="member" value="<%= member %>" />
<c:set target="member" property="name" value="홍길동" />
```



## (2) <c:remove> 태그

### ■ 변수 삭제

```
<c:remove var="varName" [scope="영역"] />

<c:set var="name" value="최범균" scope="request" />
<c:set var="name" value="최범균" scope="session" />
<c:remove var="name" />
```

## 12.2.2 흐름 제어 태그

### (1) <c:if> 태그

#### ■ 조건이 true일 경우 몸체 내용 실행

```
<c:if test="조건">
    ...
</c:if>
```

[chap12/use\_if\_tag.jsp]

```
<%@ page contentType="text/html; charset=utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<html>
<head>
<title>if 태그</title>
</head>
<body>
    <c:if test="true">
        무조건 수행<br>
    </c:if>

    <c:if test="{param.name == 'bk'}">
        name 파라미터의 값이 ${param.name} 입니다.<br>
    </c:if>

    <c:if test="{18 < param.age}">
        당신의 나이는 18세 이상입니다.
    </c:if>
</body>
</html>
```



### (2) <c:choose>, <c:when>, <c:otherwise> 태그

#### ■ switch - case - default와 동일

```
<c:choose>
    <c:when test="{member.level == 'trial'}">
        ...
    </c:when>
    <c:when test="{member.level == 'regular'}">
```

```

    ...
    </c:when>
    <c:otherwise>
    ...
    </c:otherwise>
</c:choose>

```

[chap12/use\_choose\_tag.jsp]

```

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

    <ul>
        <c:choose>
            <c:when test="${param.name == 'bk'}">
                <li>당신의 이름은 ${param.name} 입니다.
            </c:when>
            <c:when test="${param.age > 20}">
                <li>당신은 20세 이상입니다.
            </c:when>
            <c:otherwise>
                <li>당신은 'bk'가 아니고 20세 이상이 아닙니다.
            </c:otherwise>
        </c:choose>
    </ul>

</body>
</html>

```

### (3) <c:forEach> 태그

- 집합이나 컬렉션 데이터 사용
- 특정 회수 반복
- varStatus 속성 : 루프 정보를 담는 객체를 저장할 변수명을 값으로 갖는다.
  - index : 루프 실행에서 현재 인덱스, count - 루프 실행 회수
  - begin : begin 속성 값, end - end 속성 값, step - step 속성 값
  - first : 현재 실행이 첫 번째 실행인 경우 true
  - last : 현재 실행이 루프의 마지막 실행인 경우 true
  - current : 컬렉션 중 현재 루프에서 사용할 객체

```

<c:forEach var="변수" items="아이템">
    ... ${변수사용} ...
</c:forEach>

<c:forEach var="i" begin="1" end="10" [step="값"]>
    ${i} 사용
</c:forEach>

<c:forEach var="item" items="%= someItemList %" varStatus="status">
    ${status.index + 1} 번째 항목 : ${item.name}

```

```
</c:forEach>
```

[chap12/use\_foreach\_tag.jsp]

```
<%@ page contentType="text/html; charset=utf-8"%>
<%@ page import="java.util.HashMap"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
<%
    HashMap<String, Object> mapData = new HashMap<String, Object>();
    mapData.put("name", "최범균");
    mapData.put("today", new java.util.Date());
%>
<c:set var="intArray" value="<%=new int[] { 1, 2, 3, 4, 5 }%>" />
<c:set var="map" value="<%=mapData%>" />
<html>
<head>
<title>forEach 태그</title>
</head>
<body>

    <h4>1부터 100까지 홀수의 합</h4>
    <c:set var="sum" value="0" />
    <c:forEach var="i" begin="1" end="100" step="2">
        <c:set var="sum" value="{sum + i}" />
    </c:forEach>
    결과 = ${sum}

    <h4>구구단: 4단</h4>
    <ul>
        <c:forEach var="i" begin="1" end="9">
            <li>4 * ${i} = ${4 * i}
        </c:forEach>
    </ul>

    <h4>int형 배열</h4>

    <c:forEach var="i" items="{intArray}" begin="2" end="4"
        varStatus="status">
        ${status.index}-${status.count}-${i} <br />
    </c:forEach>

    <h4>Map</h4>

    <c:forEach var="i" items="{map}">
        ${i.key} = ${i.value}<br>
    </c:forEach>

</body>
</html>
```

#### (4) <c:forTokens> 태그

- java.util.StringTokenizer 클래스와 같은 기능을 제공하는 태그이다.

```
<c:forTokens var="token" items="문자열" delims="구분자">
    ${token}의 사용
</c:forTokens>
```

[chap12/use\_fortokens\_tag.jsp]

```

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

<html>
<head><title>forTokens 태그</title></head>
<body>

    콤마와 점을 구분자로 사용:<br>
    <c:forTokens var="token"
        items="빨강색,주황색,노란색,초록색,파랑색,남색,보라색"
        delims=",." >
        ${token}
    </c:forTokens>

</body>
</html>

```

## 12.2.3 URL 처리 태그

### (1) <c:url> 태그

- 절대 URL과 상대 URL을 알맞게 생성

```

<c:url value="URL" [var="varName"] [scope="영역"]>
    <c:param name="이름" value="값" />
</c:url>

```

[chap12/use\_url\_tag.jsp]

```

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

    <c:url value="http://search.daum.net/search" var="searchUrl">
        <c:param name="w" value="blog" />
        <c:param name="q" value="공원" />
    </c:url>

    <ul>
        <li>${searchUrl}</li>
        <li><c:url value="/use_if_tag.jsp" /></li>
        <li><c:url value="./use_if_tag.jsp" /></li>
    </ul>

</body>
</html>

```

### (2) <c:redirect> 태그

- 지정한 페이지로 리다이렉트

```

<c:redirect value="URL" [context="varName"]>
    <c:param name="이름" value="값" />

```

```
</c:redirect>
```

## 12.2.4. 기타 코어 태그

### (1) <c:out> 태그

- 데이터를 출력

```
<c:out value="value" [escapeXml="(true|false)"] [default="defaultValue"] />
<c:out value="value" [escapeXml="(true|false)"]>
    default value
</c:out>
```

### (2) <c:catch> 태그

- 몸체에서 발생한 예외를 변수에 저장

```
<c:catch var="exName">
...
예외가 발생할 수 있는 코드
...
</c:catch>
${exName} 사용
```

## 12.3 국제화 태그

기능 분류	태그	설명
로케일 지정	setLocale	Locale을 지정한다.
	requestEncoding	요청 파라미터의 캐릭터 인코딩을 지정한다.
메시지 처리	bundle	사용할 번들을 지정
	message	지역에 알맞은 메시지를 출력
	setBundle	리소스 번들을 읽어와 특정 변수에 저장
숫자 및 날짜 포매팅	formatNumber	숫자를 포매팅
	formatDate	Date 객체를 포매팅
	parseDate	문자열로 표시된 날짜를 분석해서 Date 객체로 변환
	parseNumber	문자열로 표시된 날짜를 분석해서 숫자로 변환
	setTimeZone	시간대 정보를 특정 변수에 저장
	timeZone	시간대를 지정

### 12.3.1 로케일 지정 태그

- <fmt:setLocale value="언어코드" scope="범위" />
  - 국제화 태그가 Accept-Language 헤더에서 지정한 언어가 아닌 다른 언어를 사용하도록 지정하는 기능
- <fmt:requestEncoding value="캐릭터셋" />
  - 요청 파라미터의 캐릭터 인코딩을 지정
  - request.setCharacterEncoding("캐릭터셋")과 동일

```

<%@ tablib prefix="fmt" url="http://java.sun.com/jsp/jstl/fmt" %>
<fmt:setLocale value="ko" scope="request" />

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

<!-- 위 코드는 다음 코드와 동일하다. -->
<%
    request.setCharacterEncoding("utf-8");
%>

```

### 12.3.2 예제로 사용할 리소스 번들

[chap12/WEB-INF/classes/resource/message.properties]

```

TITLE = MadVirus's Learning JSP 2.3
GREETING = HI! I'm BK
VISITOR = Your ID is {0}.

```

[chap12/WEB-INF/classes/resource/message\_ko.properties]

```

TITLE = \ucd5c\ubc94\uade0\uc758 JSP 2.3 \ubc30\uc6b0\uae30
GREETING = \uc548\u155\u558\u138\u694. \ucd5c\ubc94\uade0\uc785\u2c8\u2e4.
VISITOR = \ub2f9\uc2e0\uc758 \uc544\u774\u514\u294 {0}\uc785\u2c8\u2e4.

```

### 12.3.3 메시지 처리 태그

- 메시지 처리 태그는 다음과 같이 세 가지가 있다.
  - <fmt:bundle> : 태그 몸체에서 사용할 리소스 번들을 지정한다.
  - <fmt:message> : 메시지를 출력한다.
  - <fmt:setBundle> : 특정 메시지 번들을 사용할 수 있도록 로딩한다.
- <fmt:message> 태그의 메시지 읽는 순서
  - bundle 속성에 지정한 리소스 번들을 사용
  - <fmt:bundle> 태그에 중첩된 경우 <fmt:bundle> 태그에서 설정한 리소스 번들 사용
  - 1과 2가 아닐 경우 기본 리소스 번들 사용. 기본 리소스 번들은 web.xml 파일에서 javax.servlet.jsp.jstl.fmt.localizationContext 컨텍스트 속성을 통해서 설정 가능

```

<fmt:bundle basename="resource.message" [prefix="접두어"]>
    <fmt:message key="GREETING" />
</fmt:bundle>

<fmt:setBundle var="message" basename="resource.message" />
...
<fmt:message bundle="${message}" key="GREETING" />

```

[chap12/use\_message\_tag.jsp]

```

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

```



```

<%-- <fmt:setLocale value="en" /> --%>
<fmt:bundle basename="resource.message">
<fmt:message key="TITLE" var="title"/>
<html>
<head><title>${title}</title></head>
<body>

<fmt:message key="GREETING" />
<br>
<c:if test="${! empty param.id}">
<fmt:message key="VISITOR">
    <fmt:param value="${param.id}" />
</fmt:message>
</c:if>

</body>
</html>
</fmt:bundle>

```

[chap12/use\_message\_tag2.jsp]

```

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

<fmt:setBundle var="message" basename="resource.message" />

<fmt:message bundle="${message}" key="TITLE" var="title"/>
<html>
<head><title>${title}</title></head>
<body>

<fmt:message bundle="${message}" key="GREETING" />
<br>
<c:if test="${! empty param.id}">
<fmt:message bundle="${message}" key="VISITOR">
    <fmt:param value="${param.id}" />
</fmt:message>
</c:if>

</body>
</html>

```

## 12.3.4 숫자 및 날짜 포매팅 처리 태그

### (1) <fmt:formatNumber> 태그

- 숫자를 포매팅
- 주요 속성

속성	표현식/EL	타입	설명
value	사용 가능	String 또는 Number	양식에 맞춰 출력할 숫자
type	사용 가능	String	어떤 양식으로 출력할지를 정한다. number는 숫자형식, percent는 % 형식, currency는 통화형식으로 출력. 기본 값은 number.
pattern	사용 가능	String	직접 숫자가 출력되는 양식을 지정한다. DecimalFormat 클래스에서 정의되어 있는 패턴 사용
var	사용 불가	String	포매팅 한 결과를 저장할 변수 명. var 속성을 사용하지 않으면 결

			과가 곧바로 출력.
scope	사용 불가	String	변수를 저장할 영역. 기본 값은 page 이다.

```
<fmt:formatNumber value="숫자값" [type="값타입"] [pattern="패턴"]
[currentCode="통화코드"] [currencySymbol="통화심볼"]
[groupingUsed="(true|false)"] [var="변수명"] [scope="영역"] />
```

## (2) <fmt:parseNumber> 태그

- 문자열을 숫자 데이터 타입으로 변환

- 주요 속성

속성	표현식/EL	타입	설명
value	사용 가능	String	파싱할 문자열
type	사용 가능	String	value 속성의 문자열 타입을 지정. number, currency, percentage 가 올 수 있다. 기본 값은 number
pattern	사용 가능	String	직접 파싱할 때 사용할 양식을 지정
var	사용 불가	String	파싱한 결과를 저장할 변수 명을 지정
scope	사용 불가	String	변수를 저장할 영역을 지정한다. 기본 값은 page.

```
<fmt:parseNumber value="값" [type="값타입"] [pattern="패턴"]
[parseLocale="통화코드"] [integerOnly="true|false"]
[var="변수명"] [scope="영역"] />
```

## (3) <fmt:formatDate> 태그

- 날짜 정보를 담은 객체(Date)를 포매팅

- 주요 속성

속성	표현식/EL	타입	설명
value	사용 가능	java.util.Date	포매팅할 날짜 및 시간 값
type	사용 가능	String	날짜, 시간 또는 둘 다 포매팅 할 지의 여부를 지정
dateStyle	사용 가능	String	날짜에 대한 포매팅 스타일을 지정
timeStyle	사용 가능	String	시간에 대한 포매팅 스타일을 지정
pattern	사용 가능	String	직접 파싱할 때 사용할 양식을 지정
var	사용 불가	String	파싱한 결과를 저장할 변수 명을 지정
scope	사용 불가	String	변수를 저장할 영역을 지정

```
<fmt:formatDate value="날짜값"
[type="타입"] [dateStyle="날짜스타일"] [timeStyle="시간스타일"]
[pattern="패턴"] [timeZone="타임존"]
[var="변수명"] [scope="영역"] />
```

## (4) <fmt:parseDate> 태그

```
<fmt:parseDate value="2009-03-01 13:00:59" pattern="yyyy-MM-dd HH:mm:ss" var="date" />
${date}
```

## (5) <fmt:timeZone> 태그와 <fmt:setTimeZone> 태그

### ■ 국제화 태그가 사용할 시간대 설정

```
<fmt:timeZone value="Hongkong">
  <!-- 사용하는 시간을 Hongkong 시간대에 맞춘다. -->
  <fmt:formatDate ... />
</fmt:timeZone>
```

## 12.3.5 web.xml 파일에 국제화 관련 태그 기본값 설정하기

### ■ 국제화 관련 컨텍스트 초기화 파라미터

속성 이름	설명
javax.servlet.jsp.jstl.fmt.localizationContext	기본으로 사용할 리소스 번들을 지정한다. 리소스 번들의 basename을 입력한다.
javax.servlet.jsp.jstl.fmt.locale	기본으로 사용할 로케일을 지정한다.
javax.servlet.jsp.jstl.fmt.timeZone	기본으로 사용할 시간대를 지정한다.

## 12.4 함수

### ■ JSTL이 제공하는 EL 함수

함수	설명
length(obj)	obj가 List와 같은 Collection인 경우 저장된 항목의 개수를 리턴하고, obj가 문자열일 경우 문자열의 길이를 리턴한다.
toUpperCase(str)	str을 대문자로 변환한다.
toLowerCase(str)	str을 소문자로 변환한다.
substring(str, idx1, idx2)	str.substring(idx1, idx2)의 결과를 리턴한다. idx2가 -1일 경우 str.substring(idx1)과 동일하다.
trim(str)	str 좌우의 공백문자를 제거한다.
replace(str, src, dest)	str에 있는 src를 dest로 변환한다.
indexOf(str1, str2)	str1에서 str2가 위치한 인덱스를 구한다.
startsWith(str1, str2)	str1이 str2로 시작할 경우 true를, 그렇지 않을 경우 false를 리턴한다.
endsWith(str1, str2)	str1이 str2로 끝나는 경우 true를, 그렇지 않을 경우 false를 리턴한다.
contains(str1, str2)	str1이 str2를 포함하고 있을 경우 true를 리턴한다.
escapeXml(str)	XML의 객체 참조에 해당하는 특수 문자를 처리한다. 예를 들어, '&'는 '&amp;'로 변환한다.

[chap12/use\_function.jsp]

```
<%@ page contentType = "text/html; charset=utf-8" %>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %>
<html>
<head><title>함수 사용</title></head>
<body>
<c:set var="str1" value="Functions <태그>를 사용합니다. " />
<c:set var="str2" value="사용" />
<c:set var="tokens" value="1,2,3,4,5,6,7,8,9,10" />

length(str1) = ${fn:length(str1)} <br>
toUpperCase(str1) = "${fn:toUpperCase(str1)}" <br>
```

```
toLowerCase(str1) = "${fn:toLowerCase(str1)}" <br>
substring(str1, 3, 6) = "${fn:substring(str1, 3, 6)}" <br>
substringAfter(str1, str2) = "${fn:substringAfter(str1, str2)}" <br>
substringBefore(str1, str2) = "${fn:substringBefore(str1, str2)}" <br>
trim(str1) = "${fn:trim(str1)}" <br>
replace(str1, src, dest) = "${fn:replace(str1, " ", "-")}" <br>
indexOf(str1, str2) = "${fn:indexOf(str1, str2)}" <br>
startsWith(str1, str2) = "${fn:startsWith(str1, 'Fun')}" <br>
endsWith(str1, str2) = "${fn:endsWith(str1, "합니다.")}" <br>
contains(str1, str2) = "${fn:contains(str1, str2)}" <br>
containsIgnoreCase(str1, str2) = "${fn:containsIgnoreCase(str1, str2)}" <br>

<c:set var="array" value="${fn:split(tokens, ',')}" />

join(array, "-") = "${fn:join(array, "-")}" <br>
escapeXml(str1) = "${fn:escapeXml(str1)}" <br>

</body>
</html>
```