

01. JSTL의 개념과 구성

1. JSTL이란?

- JSTL(JSP Standard Tag Library)은 커스텀 태그 라이브러리 기술을 이용해서 일반적으로 필요한 기능들을 표준화한 것으로 크게 핵심(CORE), xml, I18N(국제화), 데이터베이스(SQL), 함수(functions) 라이브러리로 구성된다.
- 커스텀 태그 기반이므로 JSTL을 사용하는 방법은 일반적인 커스텀 태그와 같다.
- 다운로드 : <http://tomcat.apache.org/taglibs/standard/>
- taglib 지시어 사용법(core 라이브러리의 경우)

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

[표 11-1] JSTL 라이브러리별 URI 및 prefix

라이브러리	URI	prefix
핵심	http://java.sun.com/jsp/jstl/core	c
XML	http://java.sun.com/jsp/jstl/xml	x
I18N	http://java.sun.com/jsp/jstl/fmt	fmt
데이터베이스	http://java.sun.com/jsp/jstl/sql	sql
함수	http://java.sun.com/jsp/jstl/functions	fn

01. JSTL의 개념과 구성

2. JSTL 구성

- JSTL은 태그가 제공하는 기능의 성격에 따라 5가지 주요 라이브러리로 구분된다.
- 교재에서는 Core(핵심)을 중심으로 살펴본다.

[표 11-2] JSTL 라이브러리

라이브러리	기능	태그	접두어
핵심	General Purpose Actions	• catc • out • remove • set	c
	Conditional Actions	• choose • when • otherwise • if	
	Iterator Actions	• forEach • forTokens	
	URL Related Actions	• import • redirect • url • param	
XML	Core	• out • parse • set	x
	Flow Control	• choose • when • otherwise • forEach • if	
	Transformation	• transform • param	

01. JSTL의 개념과 구성

- I18N 은 다국어 처리와 관련된 기능을 제공하고 sql(데이터베이스)는 간단하게 데이터베이스 관련 작업을 지원하며 functions(함수)는 여러 부가기능들을 제공한다.

라이브러리	기능	태그	접두어
I18N	Locale	• setLocale	fmt
	Message Formatting	• bundle • message • param • setBundle • requestEncoding	
	Number and DateFormatting	• formatNumber • formatDate • parseDate • parseNumber • setTimeZone • timeZone	
데이터베이스	Database Access	• setDataSource • query • dateParam • param • transaction • update • dateParam • param	sql
함수		• ontains • containsIgnoreCase • endsWith • escapeXml • indexOf • join • length • replace • split • startsWithsubstring • substringAfter • substringBefore • toLowerCase • toUpperCase • trim	fn

01. JSTL의 개념과 구성

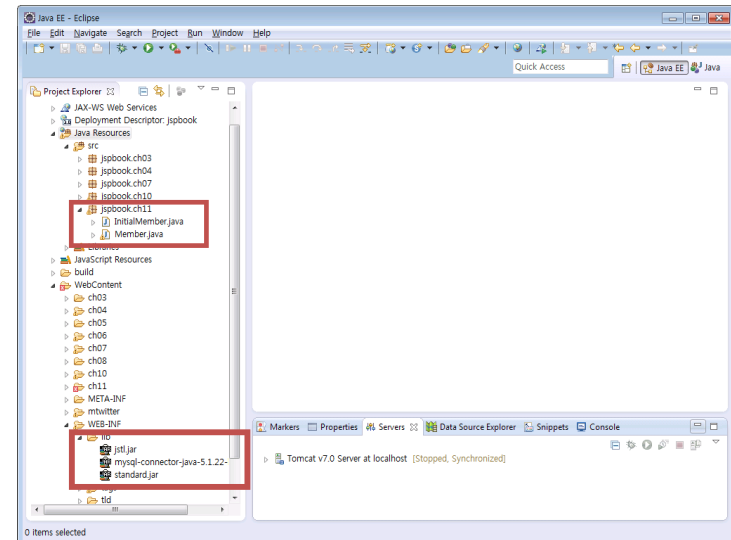
3. JSTL 학습 사전 준비

- JSTL은 규격화된 커스텀 태그 라이브러리를 배우는 것으로 특별히 어려운 점은 없으나 각 태그의 기능을 이해하고 활용하려면 적절한 데이터가 미리 준비되어 있어야 한다.
- 여기서는 톰캣 시작 시 InitialMember 리스너 클래스에서 Member 클래스 객체 10개를 생성하고 샘플 데이터로 초기화하는 작업을 자동으로 수행한다. 리스너 클래스와 관련해서는 13장에서 더욱 자세히 살펴본다.

■ JSTL 설치와 리스너 클래스 복사

❶ tomcat.apache.org/taglibs/standard/에서 1.1.2 버전을 다운로드한 후, 압축을 풀고 [lib] 폴더에서 jstl.jar와 standard.jar 파일을 찾아 jspbook 프로젝트의 [WEB-INF\lib] 폴더로 복사한다.

❷ 'jspbook.ch11' 패키지를 만든 후 예제소스 폴더 [jspbook\src\jspbook\ch11]에서 InitialMember.java, Member.java 파일을 해당 패키지로 드래그 앤 드롭한다.



[그림 11-1] JSTL 설치와 리스너 클래스 소스 복사

01. JSTL의 개념과 구성

■ 핵심 라이브러리 예제 데이터 및 리스너 설명 – 교재 p.476 ~ 477 참고

- 리스너는 톰캣의 특정 이벤트에 동작하는 특수한 목적의 서블릿 프로그램으로 여기서는 톰캣 시작시 ArrayList 를 생성하고 초기 데이터를 등록한 다음 application scope 에 저장한다.
- 저장된 데이터를 이용해 JSTL 실습에 활용 한다.

02. 핵심 라이브러리의 주요 태그

1. 기본 기능 태그

■ <c:out> 태그

- 초기에는 jsp 표현식(<%= %>)을 대체하기 위해 개발 되었으나 표현언어가 JSP에 기본으로 제공 되면서 사용 빈도는 줄었으나 몇몇 옵션은 유용하게 사용할 수 있다.

▪ 사용법

- 태그 바디가 없는 경우

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

- 태그 바디가 있는 경우

```
<c:out value="value" [escapeXml="{true|false}"]>
```

default value (value에 내용이 없을 때 출력될 기본 값)

```
</c:out>
```

[표 11-3] <c:out> 태그 속성 값

속성	필수	기본 값	설명
value	Y	없음	출력될 내용 또는 표현식이다.
default	N	태그 바디에 있는 내용	value 값에 내용이 없는 경우에 출력할 내용으로, 태그 바디 혹은 속성 값 형태로 올 수 있다.
escapeXml	N	true	출력될 내용에 <, >, &, ' 등의 문자를 일반 문자로 변환할 것인지 결정한다. 예를 들어 출력될 내용에 HTML 태그가 포함되어 있다면 이 값을 false로 해야 태그가 반영된 내용이 화면에 보인다. 만일 true로 할 경우 태그가 그대로 화면에 보이게 된다.

02. 핵심 라이브러리의 주요 태그

■ <c:set> 태그

- <c:set> 태그는 변수 값을 설정하거나 객체의 멤버변수 값을 설정할 때 사용한다.

- 사용법

- 해당 범위에 속성 값을 추가하는 경우(바디가 올 수도 있다)

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

- 특정 target 객체에 새로운 속성 값을 설정하는 경우(바디가 올 수도 있다)

```
<c:set value="value" target="target" property="propertyName"/>
```

[표 11-4] <c:set> 태그 속성 값

속성	필수	기본 값	설명
value	N	없음	저장할 변수 값
target	N	없음	값이 저장될 객체 이름
property	N	없음	target 객체의 멤버변수 이름
var	N	없음	값이 저장될 변수 이름
scope	N	page	값이 저장될 범위(page, session, request, application)

02. 핵심 라이브러리의 주요 태그

■ <c:remove> 태그

- <c:remove> 태그는 해당 scope 에 설정된 객체를 제거 한다.
- 사용법

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

[표 11-5] <c:remove> 태그 속성 값

속성	필수	기본 값	설명
var	Y	없음	삭제할 변수 이름
scope	N	모든 범위	삭제할 범위

02. 핵심 라이브러리의 주요 태그

■ <c:catch> 태그

- <c:catch> 태그는 바디에서 실행되는 코드의 예외를 처리한다.
- JSP를 뷰 역할에 충실하게 프로그래밍 한다면 크게 사용할 일은 많지 않다.
- 사용법

```
<c:catch [var="varName"]>  
nested actions  
</c:catch>
```

[표 11-6] <c:catch> 태그 속성 값

속성	필수	기본 값	설명
var	Y	없음	오류 메시지를 저장할 변수 이름

02. 핵심 라이브러리의 주요 태그

2. 조건 처리 태그

■ <c:if> 태그

- <c:if> 태그는 조건에 따라 바디 내용을 처리 한다.
- 자바의 if와 비슷하지만 else문은 지원하지 않는다(단순 조건 체크만 가능).
- 사용법
 - 바디 내용이 없는 경우

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

- 바디 내용이 있는 경우

```
<c:if test="testCondition" [var="varName"] [scope="{page|request|session|application}"]>
```

Body content

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

[표 11-7] <c:if> 태그 속성 값

속성	필수	기본 값	설명
test	Y	없음	검사할 조건
var	N	없음	test 조건의 결과를 저장할 변수(결과는 true 혹은 false)
scope	N	page	변수가 저장될 범위

02. 핵심 라이브러리의 주요 태그

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

- 이들 태그는 함께 사용되며 자바의 if ~ else if 문, switch 문과 유사하다.
- <c:choose> 태그 내에는 <c:when> 태그가 여러 개 올 수 있다.

■ 사용법

```
<c:choose>
  body content (<when> and <otherwise> subtags)
<c:when test="testCondition">
  body content
</c:when>
<c:otherwise>
  conditional block
</c:otherwise>
</c:choose>
```

[표 11-8] <c:when> 태그 속성값

속성	필수	기본 값	설명
test	Y	없음	검사할 조건

02. 핵심 라이브러리의 주요 태그

3. 순환 처리 태그

■ <c:forEach> 태그

- 반복문과 관련된 태그로 자바의 for 문과 유사하다. 가장 중요하고 널리 쓰이는 JSTL 태그 중 하나임.
- 여러 옵션 활용법을 잘 익혀 두어야 한다.

▪ 사용법

- 컬렉션 객체의 크기만큼 반복

```
<c:forEach[var="varName"] items="collection" [varStatus="varStatusName"]  
[begin="begin"] [end="end"] [step="step"]>  
body content  
</c:forEach>
```

- 지정된 횟수 반복

```
<c:forEach [var="varName"] [varStatus="varStatusName"] begin="begin" end="end"  
[step="step"]>  
body content  
</c:forEach>
```

02. 핵심 라이브러리의 주요 태그

- varStatus는 반복 과정에서 프로그램적으로 필요한 여러 정보를 제공함.
- 자바 언어에서의 for와 같은 자유도는 없기 때문에 모든 반복 처리에 <c:forEach>를 사용하기는 어렵고 데이터 생성시 <c:forEach>에서 처리되기 편리한 형태로 가공해주는 것이 필요하다.

[표 11-9] <c:forEach> 속성 값

속성	필수	기본 값	설명
items	N	없음	반복을 위한 데이터를 가진 아이탬의 컬렉션
begin	N	0	반복 시작 번호
end	N	컬렉션의 마지막 값	반복 끝 번호
step	N	1	반복의 증가분
var	N	없음	현재 아이탬이 있는 변수
varStatus	N	없음	반복 상태 값이 있는 변수

02. 핵심 라이브러리의 주요 태그

■ <c:forTokens> 태그

- <c:forTokens>태그는 기본적으로 for문과 유사한 동작을 하지만 문자열을 토큰으로 구분해 처리하는 기능을 제공한다.
- 자바의 StringTokenizer 클래스와 유사하다고 볼 수 있다.
- 토큰은 문자열을 구분하기 위한 캐릭터로 "-", tab, 공백 등 동일한 규칙으로 문자열을 구성해야 한다.
- 예를 들어 전화번호는 "010-1234-1234" 와 같이 표현하며 이때 토큰은 "-" 이 된다.
- **사용법**

```
<c:forTokens items="stringOfTokens" delims="delimiters"  
[var="varName"]  
[varStatus="varStatusName"]  
[begin="begin"] [end="end"] [step="step"]>  
body content  
</c:forTokens>
```

02. 핵심 라이브러리의 주요 태그

- 기본적으로 <c:forEach>와 동일하며 delims를 통해 토큰(구분자)를 지정한다.

[표 11-10] <c:forEach> 속성 값

속성	필수	기본 값	설명
items	N	없음	반복을 위한 데이터를 가진 아이템의 컬렉션
delims	Y	없음	구분자(Delimiter)로 사용할 문자
begin	N	0	반복 시작 번호
end	N	컬렉션의 마지막 값	반복 끝 번호
step	N	1	반복의 증가분
var	N	없음	현재 아이템이 있는 변수
varStatus	N	없음	반복 상태 값이 있는 변수

02. 핵심 라이브러리의 주요 태그

4. URL 관련 태그

■ <c:import> 태그

- 특정 URL 페이지를 현재 페이지에 포함시킨다.
- <jsp:include> 액션과 유사하다.
- **사용법**
 - 포함하고자 하는 자원을 문자열 형태로 포함하는 경우

```
<c:import url="url" [context="context"] [var="varName"]  
[scope="{page|request|session|application}"] [charEncoding="charEncoding"]>  
optional body content for <c:param> subtags  
</c:import>
```

- 포함하고자 하는 자원을 Reader 객체로 포함하는 경우

```
<c:import url="url" [context="context"] varReader="varReaderName"  
[charEncoding="charEncoding"]>  
body content where varReader is consumed by another action  
</c:import>
```


02. 핵심 라이브러리의 주요 태그

- 동적인 페이지를 포함할 때 사용할수는 있으나 성능상에 문제가 발생할 수 있다.
- 포함한 페이지는 <c:out>을 이용해 출력하기 때문에 용량이 큰 페이지의 사용은 자제하는 것이 좋다.

[표 11-11] <c:import> 속성 값

속성	필수	기본 값	설명
url	Y	없음	현재 페이지 내에 포함시킬 URL
context	N	Current application	현재 웹 애플리케이션 컨텍스트 이름
charEncoding	N	ISO-8859-1	현재 페이지 내에 포함시킬 페이지 캐릭터셋을 지정
var	N	Print to page	포함할 페이지의 내용을 가지는 변수 이름
scope	N	page	var의 범위
varReader	N	없음	자원 내용을 읽으려는 java.io.Reader 변수

02. 핵심 라이브러리의 주요 태그

■ <c:url> 태그

- URL Rewriting 즉, 제공된 URL에 파라미터 등을 추가해 프로그램에서 URL 관련 처리(링크 등)를 손 쉽게 할 수 있는 기능을 제공한다.

- **사용법**

- 바디가 없는 경우

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

- 바디가 있는 경우

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

02. 핵심 라이브러리의 주요 태그

- 파라미터는 <c:param> 태그를 이용해 추가할 수 있다.
- scope 지정을 통해 처리하고자 하는 url 객체를 공유할 수 있다.

[표 11-12] <c:url> 속성 값

속성	필수	기본 값	설명
value	Y	없음	기본 URL
context	N	Current application	현재 웹 애플리케이션의 컨텍스트 이름
var	N	Print to page	포함할 페이지의 내용을 가지는 변수 이름
scope	N	page	var의 범위

02. 핵심 라이브러리의 주요 태그

■ <c:redirect> 태그

- response.sendRedirect() 메서드나 <jsp:forward> 액션과 유사하다.
- 지정된 페이지로 사용자 요청을 이동시키며 <c:param>을 통해 파라미터 추가도 가능하다.

▪ 사용법

- 바디가 없는 경우

```
<c:redirect url="value" [context="context"]/>
```

- 바디가 있는 경우

```
<c:redirect url="value" [context="context"]/>
```

```
<c:param> subtags
```

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

[표 11-13] <c:redirect> 속성 값

속성	필수	기본 값	설명
value	Y	없음	기본 URL
context	N	Current application	현재 웹 애플리케이션의 컨텍스트 이름

02. 핵심 라이브러리의 주요 태그

■ <c:param> 태그

- import, url, redirect 와 함께 사용된다.
- url에 파라미터를 GET 방식으로 추가한다.
- 사용법
 - 바디가 없는 경우

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

- 바디 내용을 속성 값으로 사용하는 경우

```
<c:param name="name">
```

```
parameter value
```

```
</c:param>
```

[표 11-14] <c:param> 속성 값

속성	필수	기본 값	설명
url	Y	없음	현재 페이지 내에 포함시킬 URL
context	N	Current application	현재 웹 애플리케이션 컨텍스트 이름