

# JSP 프로그래밍

(재)대덕인재개발원

## 16 표준 태그 라이브러리(JSTL)

---

- ▶ 1. JSTL이란
- ▶ 2. 코어 태그
- ▶ 3. 국제화 태그
- ▶ 4. 함수



## 16.1 JSTL

### ▶ 개요

- ▶ JSP는 개발자가 직접<jsp:include>와 같은 태그를 작성할 수 있는 기능을 제공하는데 이를 커스텀 태그라고 한다.
- ▶ 커스텀 태그 중에서 많이 사용되는 것들을 모아서 JSTL(JSP Standard Tag Library)이라는 규약을 만들었다.
- ▶ 다운로드 : <https://maven-repository.dev.java.net/repository/jstl/jars/>

```
<%  
  if(list.size() > 0){  
    for(int i = 0; i < list.size(); i++){  
      Data data = (Data)list.get(i);  
    %>  
    <%= data.getTitle() %>  
    %>  
  }  
}else{  
  %>  
  데이터가 없습니다.  
  <%  
  }  
%>
```



```
<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:if>
```

## 16.1.1 JSTL이 제공하는 태그의 종류

### ▶ JSTL이 제공하는 태그의 종류

라이브러리	하위 기능	접두어	관련 URI
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>

\* 접두어는 JSP 페이지에서 커스텀 태그를 호출할 때 사용되는데, 위 표에서 보여준 접두어는 일반적으로 사용되는 것들이다.

\* 관련 URI는 JSTL이 제공하는 커스텀 태그를 구분해 주는 식별자로서 이 식별자를 통해서 JSP페이지에서 사용할 커스텀 태그 라이브러리를 선택할 수 있다.

## 16.2 코어 태그

### ▶ 코어 태그 라이브러리

기능 분류	태 그	설 명
변수 지원	set	JSP에서 사용될 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	예외 처리에 사용된다.
	out	JspWriter에 내용을 알맞게 처리한 후 출력한다.

## 16.2.1 변수 지원 태그

---

### ▶ <c:set> 태그

- ▶ <c:set> 태그는 EL 변수의 값이나 EL 변수의 프로퍼티의 값을 지정할 때 사용된다. Page, request, session, application 의 네 영역에 저장되어 있는 객체를 <jsp:useBean> 태그와 <jsp:setProperty> 태그를 사용하여 JSP 코드 내에서 사용될 변수를 지정하고 변수 프로퍼티의 값을 지정할 수 있는 것과 마찬가지로 <c:set> 태그는 JSTL의 태그에서 사용될 변수의 값을 처리할 때 사용된다.

```
<c:set var="varName" value="varValue" [scope="영역"] />
```

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

- ▶ var : 값을 지정할 EL 변수의 이름
- ▶ value : 변수의 값을 지정한다. 표현식, EL, 정적인 텍스트를 사용하여 값을 지정할 수 있다.
- ▶ scope : 변수를 지정할 영역을 지정한다. 값은 page, request, session, application 중 하나가 온다. 지정하지 않을 경우 기본값은 page이다.



## 16.2.1 변수 지원 태그(계속)

---

### ▶ <c:set> 태그로 객체의 프로퍼티 값 설정하기

- ▶ **target** : 프로퍼티 값을 설정할 대상 객체. 표현식(<%= 변수 %>)이나 EL 변수(\${varName})를 지정한다. 대상 객체는 자바빈 객체나 Map이어야 한다.
- ▶ **property** : 설정할 프로퍼티의 이름. **target**이 자바빈 객체인 경우 프로퍼티 이름에 해당하는 setter 메서드를 제공해 주어야 한다. 예를 들어, 프로퍼티 이름이 **name**인 경우 **target** 객체는 **setName()** 메서드를 제공해야 한다. Map인 경우 **Map.put(프로퍼티이름, 값)**을 이용해서 값을 추가한다.
- ▶ **value** : 설정할 프로퍼티의 값.
- ▶ \*\* **target** 대상이 EL 변수인 경우 **target** 속성의 값을 **\${member}**와 같이 EL을 이용해서 설정해 주어야 한다.

```
<c:set target="대상" property="프로퍼티이름" value="값" />
```

```
<c:set target="대상" property="프로퍼티이름">값</c:set>
```

---



## 16.2.1 변수 지원 태그(계속)

---

### ▶ <c:set> 태그

- ▶ <c:set> 태그를 실행하면 EL 변수인 varName에 varValue를 값으로 할당한다. 이는 내부적으로 `pageContext.setAttribute(varName, varValue, scope)`를 호출해서 지정한 영역의 속성으로 설정하게 된다.

<%-- value 속성 사용 예 --%>

```
<c:set var="name" value="이산" />
```

```
<c:set var="name" value="<%= m.getFirstName() %>" scope="request" />
```

```
<c:set var="name" value="${m.lastName} ${m.firstName}" />
```

<%-- 태그의 몸체를 값으로 사용하는 예 --%>

```
<c:set var="name">이산</c:set>
```

```
<c:set var="name"><%= m.getLastName() %><%= m.getFirstName() %> </c:set>
```

```
<c:set var="name">${m.lastName} ${m.firstName} </c:set>
```





## 16.2.1 변수 지원 태그(계속)

---

### ▶ <c:set> 태그의 속성 설명 요약

속 성	표현식/EL	타 입	설 명
var	사용 불가	String	EL 변수 이름
value	사용 가능	Object	변수에 할당할 값
scope	사용 불가	String	변수를 생성할 영역. 기본값은 page
target	사용 가능	Object	프로퍼티 값을 설정할 객체 지정
property	사용 가능	String	프로퍼티 이름



## 16.2.1 변수 지원 태그(계속)

### ▶ <c:remove> 태그

- ▶ <c:remove> 태그는 <c:set> 태그로 지정한 변수를 삭제할 때 사용된다.
- ▶ 삭제할 변수의 scope를 지정하지 않으면 동일한 이름으로 저장된 모든 영역의 변수를 모두 삭제한다.

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

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

속 성	표현식/EL	타 입	설 명
var	사용 불가	String	삭제할 EL 변수 이름
scope	사용 불가	String	삭제할 변수가 포함된 영역

## 16.2.2 흐름 제어 태그

---

- ▶ <c:if> 태그

- ▶ <c:if> 태그는 자바 언어의 if 블록과 비슷한 기능을 제공한다.

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



## 16.2.2 흐름 제어 태그

### ▶ <c:if> 태그

```
<%@ page language="java" 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 == 'san'}">
name 패러미터의 값이 ${param.name} 입니다.<br>
</c:if>

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

---

속 성	표현식/EL	타 입	설 명
test	사용가능	boolean	검사 조건
var	사용 불가	String	검사 조건의 계산 결과를 저장한 EL 변수
scope	사용 불가	String	검사 조건 결과 값을 저장할 영역



## 16.2.2 흐름 제어 태그

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

- ▶ 자바의 switch 와 if-else 블록을 혼합한 형태로서 다수의 조건문을 수행.

```
<%@ page language="java" 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>
<c:choose>
  <c:when test="${param.name == 'san'}" >
    당신의 이름은 ${param.name} 입니다.
  </c:when>
  <c:when test="${param.age > 20}" >
    당신은 20세 이상입니다.
  </c:when>
  <c:otherwise>
    넌 뭐니?
  </c:otherwise>
</c:choose>
</body>
</html>
```



```
// 자바 if-else 문
if(name == "san"){
  ...
}else if(age > 20){
  ...
}else{
  ...
}
```

## 16.2.2 흐름 제어 태그

### ▶ <c:forEach> 태그

- ▶ <c:forEach> 태그는 배열, Collection 또는 Map 에 저장되어 있는 값들을 순차적으로 처리할 때 사용할 수 있는 태그로서, 자바의 for, do-while 등을 대신해서 사용.
- ▶ items 속성 : Map, 배열, Collection이 올 수 있다.

```
<c:forEach var="변수" items="아이템">
  <tr>
    <td align="right" bgcolor="ffffff">
      ${변수}
    </td>
  </tr>
</c:forEach>
```

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

## 16.2.2 흐름 제어 태그

---

### ▶ <c:forEach> 태그

- ▶ begin, end, step 속성 : 범위 지정을 통해 자바의 for구문과 같은 효과를 낼 수 있다.
- ▶ varStatus 속성 : List와 같은 컬렉션이나 배열 이용시 현재 항목의 인덱스 값을 구할 수 있다.(index, count, begin, end, step, first, last, current)

```
<c:forEach var="i" items="${intArray}" begin="2" end="4" step="2">  
    [ ${i} ]  
</c:forEach>
```

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





## 16.2.2 흐름 제어 태그

### ▶ <c:forEach> 태그의 varStatus 속성

속 성	설 명
index	루프 실행에서 현재 인덱스
count	루프 실행 횟수
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



## 16.2.2 흐름 제어 태그

### ▶ <c:forEach> 태그의 속성 설명 요약

속 성	표현식/EL	타 입	설 명
var	사용 불가	String	몸체에서 사용할 EL 변수 이름
items	사용 가능	Collection, Iterator, Enumeration, Map, 배열	반복 처리할 데이터
varStatus	사용 불가	String	루프 상태를 저장할 EL 변수 이름(LoopTagStatus)
begin	사용 가능	int	시작 인덱스 값
end	사용 가능	int	종료 인덱스 값
step	사용 가능	int	인덱스 증분 값



## 16.2.2 흐름 제어 태그

### ▶ <c:forTokens> 태그

- ▶ <c:forTokens> 태그는 java.util.StringTokenizer 클래스와 같은 기능을 제공하는 태그이다.
- ▶ <c:forEach> 태그와 동일하게 begin, end, step, varStatus 속성을 제공한다.

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

```
<c:forTokens var="color" items="red,green,blue" delims=",">  
    ${color}  
</c:forTokens>
```



## 16.2.3 URL 처리 태그

- ▶ URL 관련 태그는 내부/외부 자원의 삽입(<c:import> 태그), URL 생성 (<c:url> 태그), 리다이렉트 처리(<c:redirect> 태그)의 세 가지 기능을 제공한다.
- ▶ <c:import> 태그
  - ▶ <c:import> 태그는 특정 URL의 결과를 읽어와 현재 위치에 삽입하거나 EL 변수에 저장할 때 사용된다.
  - ▶ <jsp:include>는 동일한 웹 어플리케이션 내에 위치한 자원을 포함해 주는 기능이라면, <c:import> 태그는 동일한 웹 어플리케이션 뿐만 아니라 외부의 다른 자원을 읽어와 포함시킬 수 있도록 해준다.

```
<c:import url="URL" [var="변수명"] [scope="영역"] [charEncoding="캐릭터셋"]>  
</c:import>
```

## 16.2.3 URL 처리 태그

### ▶ <c:import> 태그 사용 예

```
<%-- URL로 부터 읽어온 내용을 rss 변수에 저장--%>
<c:import url="http://javacan.tistory.com/rss" var="rss" />
...
${rss} <%-- <jsp:include>와 달리 EL 변수에 보관한 뒤 필요에 따라 처리할 수 있다.--%>

<%-- 현재 위치에 읽어온 내용 삽입--%>
<c:import url="http://www.naver.com" />

<%-- 절대 URL 및 상대 URL을 사용할 수 있다.--%>
<c:import url="graphData.jsp" var="data" />

<%-- 요청 패러미터를 추가할 수 있다.--%>
<c:import url="http://www.naver.com/?name=san" />

<c:import url="http://www.naver.com/" >
  <c:param name="name" value="san" />
  <c:param name="age" value="20" />
</c:import>
```

## 16.2.3 URL 처리 태그

---

### ▶ <c:import> 태그의 속성 설명 요약

속 성	표현식/EL	타 입	설 명
url	사용 가능	String	읽어올 URL
var	사용 불가	String	읽어올 결과를 저장할 변수 이름. 변수 사용시 바로 화면에 출력하지 않고, 변수에 담아 놓는다.
scope	사용 불가	String	변수를 저장할 영역
charEncoding	사용 가능	String	결과를 읽어올 때 사용할 캐릭터 인코딩



## 16.2.3 URL 처리 태그

### ▶ <c:url> 태그

- ▶ <c:url>태그는 URL을 생성해 주는 기능을 제공한다.

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

```
<%@ page contentType="text/html; charset=utf-8" %>  
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>  
  
... 중략  
<%-- 패러미터 넘기기 --%>  
<c:url value="http://search.daum.net/search" var="searchUrl" >  
  <c:param name="w" value="blog" />  
  <c:param name="q" value="공원" />  
</c:url>  
  
${searchUrl}<br>  
<c:url value="/use_if_tag.jsp" /><br>  
<c:url value="./use_if_tag.jsp" /><br>  
... 중략
```

## 16.2.3 URL 처리 태그

### ▶ <c:redirect>태그

- ▶ <c:redirect>태그는 response.sendRedirect() 처럼 지정한 페이지로 리다이렉트시켜 주는 기능을 제공한다.
- ▶ 다른 컨텍스트 경로에 포함된 URL로 리다이렉트하고 싶다면 context 속성에 다른 컨텍스트 경로를 적어주면 된다.

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

```
<c:redirect url="/use_c_set.jsp" />  
= /chapter6/use_c_set.jsp
```

```
<c:redirect url="/viewToday.jsp" context="/otherApp" />  
= / otherApp chapter5/viewToday.jsp
```





## 16.2.4 기타 코어 태그

### ▶ <c:out>태그

- ▶ <c:out>태그는 JspWriter에 데이터를 출력할 때 사용되는 태그이다.

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

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

속 성	설 명
value	JspWriter에 출력할 값을 나타낸다. 일반적으로 value 속성의 값은 String과 같은 문자열이다. 만약 value의 값이 java.io.Reader의 한 종류라면 out태그는 Reader로 부터 데이터를 읽어와 JspWriter에 값을 출력한다.
escapeXml	이 속성의 값이 true일 경우 아래와 같이 문자를 변경한다. 생략할 수 있으며, 생략할 경우 기본값은 true이다. <: &lt; , >: &gt; , &: &amp; , ' : &#039; , " : &#034;
default	value 속성에서 지정한 값이 존재하지 않을 때 사용될 값을 지정한다.

## 16.3 국제화 태그

### ▶ 국제화 태그

- ▶ 국제화 태그는 특정 지역에 따라서 알맞은 메시지를 출력해 주고 싶은 경우에 사용된다.
- ▶ JSP 페이지에서 다양한 언어를 지원할 수 있도록 해주는 태그가 국제화 태그이다.

기능 분류	태 그	설 명
로케일 지정	setLocale	Locale을 지정한다
	requestEncoding	요청 패러미터의 캐릭터 인코딩을 지정한다.
메시지 처리	bundle	사용할 번들을 지정한다.
	message	지역에 알맞은 메시지를 출력한다.
	setBundle	리소스 번들을 읽어와 특정 변수에 저장한다.
숫자 및 날짜 포매팅	formatNumber	숫자를 포매팅한다.
	formatDate	Date 객체를 포매팅한다.
	parseDate	문자열로 표시된 날짜를 분석해서 Date 객체로 변환
	parseNumber	문자열로 표시된 날짜를 분석해서 숫자로 변환
	setTimeZone	시간대 정보를 특정 변수에 저장한다.
	timeZone	시간대를 지정한다.

## 16.3.1 로케일 지정 태그

---

### ▶ <fmt:setLocale> 태그

- ▶ 국제화 태그들이 사용할 로케일을 지정한다.
- ▶ 웹 브라우저는 Accept-Language 헤더를 사용해서 수용 가능한 언어 목록을 전송하는데, JSTL의 국제화 태그들은 이 헤더의 값을 사용해서 언어별로 알맞은 처리를 하게 된다.
- ▶ 메시지를 출력해 주는 message 태그는 Accept-Language 헤더의 값이 'ko'일 경우 한글 메시지를 우선순위로 처리하며, 'en'일 경우 영문 메시지를 우선순위로 처리한다.
- ▶ <fmt:setLocale>태그는 국제화 태그가 Accept-Language 헤더에서 지정한 언어가 아닌 다른 언어를 사용하도록 지정하는 기능을 제공한다.

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



## 16.3.3 메시지 처리 태그

- ▶ `<fmt:bundle>` : 태그 몸체에서 사용할 리소스 번들을 지정한다.
- ▶ `<fmt:message>` : 메시지를 출력한다.
- ▶ `<fmt:setBundle>` : 특정 메시지 번들을 사용할 수 있도록 로딩한다.

```
#WEB-INF/classes/resource/message.properties
```

```
TITLE = JSP 2.1 Programming  
GREETING = Hi! I'm SAN  
VISITOR = Your ID is {0}.
```

```
#WEB-INF/classes/resource/message_ko.properties.src
```

```
TITLE = JSP 2.1 프로그래밍  
GREETING = 안녕하세요. 이산입니다.  
VISITOR = 당신의 아이디는 {0} 입니다.
```

```
// 한글 코드의 경우 유니코드 숫자로 변환이 필요  
C:\...\resource>native2ascii message_ko.properties.src message_ko.properties
```

### 16.3.3.1 <fmt:bundle> 태그와 <fmt:message> 태그

- ▶ <fmt:bundle> 태그는 사용할 메시지 번들을 지정하며 <fmt:message> 태그와 함께 사용된다.

```
<fmt:bundle basename="resource.error" prefix="SYSTEM_">  
  <fmt:message key="001" />  
</fmt:bundle>
```

```
<fmt:bundle basename="resource.error">  
  <fmt:message key="SYSTEM_001" />  
</fmt:bundle>
```

- ▶ <fmt:bundle>태그의 속성 설명 요약

속 성	표현식/EL	타 입	설 명
basename	사용 가능	String	사용할 리소스 번들의 이름
prefix	사용 가능	String	bundle 태그의 내부에서 사용되는 message 태그의 key 속성의 값 앞에 자동으로 붙게 될 문자열




---

속 성	표현식/EL	타 입	설 명
basename	사용 가능	String	사용할 리소스 번들의 이름
var	사용 불가	String	리소스 번들을 저장할 EL 변수 명
scope	사용 불가	String	EL 변수를 저장할 영역

속 성	표현식/EL	타 입	설 명
value	사용 가능	String	Locale 을 언어코드_국가코드의 형식으로 지정하며, 국가 코드는 생략이 가능하다. 유효하지 않은 로케일 문자가 지정된 경우, JVM 의 기본 로케일이나 web.xml 에 설정된 기본값으로 설정됨.
scope	사용 불가	String	지정한 로케일이 영향을 미치는 범위를 지정. 기본 page.

---



속 성	표현식 /EL	타 입	설 명
value	사용 가 능	String	파싱할 날짜 및 시간을 표현한 문자열
type	사용 가 능	String	날짜, 시간 혹은 둘다 파싱 할지 여부. date, time, both
dateStyle	사용 가 능	String	미리 결정된 날짜 형식 지정. default, short, medium, long, full
timeStyle	사용 가 능	String	미리 결정된 시간 형식 지정. default, short, medium, long, full
pattern	사용 가 능	String	java.text.DateFormat 에 정의된 패턴에 따라 형식 지정
timeZone	사용 가 능	String java.util.TimeZone	시간대 변경을 위한 속성으로, setTimeZone 태그에서 생 성된 TimeZone 객체나 혹은 timeZoneId 를 사용하여 시 간대를 변경.
parseLocale	사용 가 능	String java.util.Locale	파싱할 때 사용할 로케일 지정
var	사용 불 가	String	포매팅한 결과를 저장할 EL 변수
scope	사용 불 가	String	EL 변수를 저장할 영역



### 16.3.3.1 <fmt:bundle> 태그와 <fmt:message> 태그

- ▶ <fmt:message> 태그는 지정한 리소스 번들로 부터 메시지를 읽어와 출력한다.

속 성	설 명
key	읽어올 메시지의 키 값
var	메시지를 저장할 변수 명
scope	변수가 저장되는 영역 지정
bundle	<fmt:setBundle> 태그를 사용해서 로딩한 번들로 부터 메시지를 읽어올 때 사용

- ▶ 변경 가능한 요소를 제공하는 메시지
  - ▶ 메시지 중에는 {0}, {1}, {2}와 같이 변경 가능한 요소를 제공하는 메시지가 존재할 수도 있다.
  - ▶ VISITOR = Your ID is {0}.
  - ▶ 이때 {0}이나 {1} 부분에 들어갈 값을 지정하기 위해서 <fmt:message> 태그에 <fmt:param> 태그를 사용할 수 있다.

```
<fmt:message key="VISITOR">  
  <fmt:param value="{id}" /> <!-- {0}에 들어간다. -->  
</fmt:message>
```



## 16.4 함수

- ▶ JSTL은 표현 언어에서 사용할 수 있는 함수를 제공한다.

속 성	설 명
length(obj)	obj가 List와 같은 Collection인 경우 저장된 항목의 개수를 리턴하고, Obj가 문자열일 경우 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.
substringAfter(str1, str2)	str1에서 str1에 포함되어 있는 str2 이후의 문자열을 구한다.
substringBefore(str1, str2)	str1에서 str1에 포함되어 있는 str2 이전의 문자열을 구한다.
trim(str)	str 좌우의 공백 문자를 제거한다.
replace(str, src, dest)	str에 있는 src를 dest로 변환한다.

## 16.4 함수

- ▶ JSTL은 표현 언어에서 사용할 수 있는 함수를 제공한다.

속 성	설 명
<code>indexOf(str1, str2)</code>	<code>str1</code> 에서 <code>str2</code> 가 위치한 인덱스를 구한다.
<code>startsWith(str1, str2)</code>	<code>str1</code> 이 <code>str2</code> 로 시작할 경우 <code>true</code> 를, 그렇지 않을 경우 <code>false</code> 를 리턴한다.
<code>endsWith(str1, str2)</code>	<code>str1</code> 이 <code>str2</code> 로 끝나는 경우 <code>true</code> 를, 그렇지 않을 경우 <code>false</code> 를 리턴한다.
<code>contains(str1, str2)</code>	<code>str1</code> 이 <code>str2</code> 를 포함하고 있을 경우 <code>true</code> 를 리턴한다.
<code>containsIgnoreCase(str1, str2)</code>	대소문자 구분없이 <code>str1</code> 이 <code>str2</code> 를 포함하고 있을 경우 <code>true</code> 를 리턴한다.
<code>split(str1, str2)</code>	<code>str2</code> 로 명시한 글자를 기준으로 <code>str1</code> 을 분리해서 배열로 리턴한다.
<code>join(array, str2)</code>	<code>array</code> 에 저장된 문자열을 합친다. 이때 각 문자열 사이에는 <code>str2</code> 이 붙는다.
<code>escapeXml(str)</code>	XML의 객체 참조에 해당하는 특수 문자를 처리한다. 예를 들어, <code>'&amp;'</code> 는 <code>'&amp;amp;'</code> 로 변환한다.

## 16.4 함수

- ▶ JSTL은 표현 언어에서 사용할 수 있는 함수를 제공한다.

```
<%@ 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="Function <태그>를 사용합니다." />
<c:set var="str2" value="사용" />

length(str1) = ${fn:length(str1)}<br>
toUpperCase(str1) = ${fn:toUpperCase(str1)}<br>
substring(str1, 3, 6) = ${fn:substring(str1, 3, 6)}<br>
contains(str1, str2) = ${fn:contains(str1, str2)}<br>

</body>
</html>
```