

## Chapter 15

# JSTL ( JSP Standard Tag Library )

## 커스텀 태그 ( Custom Tag )

# 커스텀 태그의 개요

## ■ 커스텀 태그(Custom Tag)

- 특정한 기능을 가지는 태그를 직접 생성하는 방법
  - 초기 커스텀 태그는 JSP에서 반복되는 코드를 캡슐화하기 위해 등장하였음
  - 기본적으로 제공되는 태그 외에 사용자가 확장한 태그라는 의미로 커스텀 태그라고 불림
- JSP문서에 존재하는 자바 코드를 제거하기 위해 사용됨
  - 액션 태그나 표현 언어를 사용하더라도 조건식이나 반복문 등의 자바 코드가 사용됨
  - 이러한 자바 코드를 최소화하고자 커스텀 태그를 사용함
- 커스텀 태그(Custom Tag) 종류
  - 사용자 커스텀 태그
    - 개발자가 필요에 의해 만든 태그
  - JSTL(JSP Standard Tag Library)
    - JSP 페이지에서 가장 많이 사용하는 기능을 태그로 제공
    - JSTL 라이브러리를 따로 설치해서 사용함

## 사용자 커스텀 태그

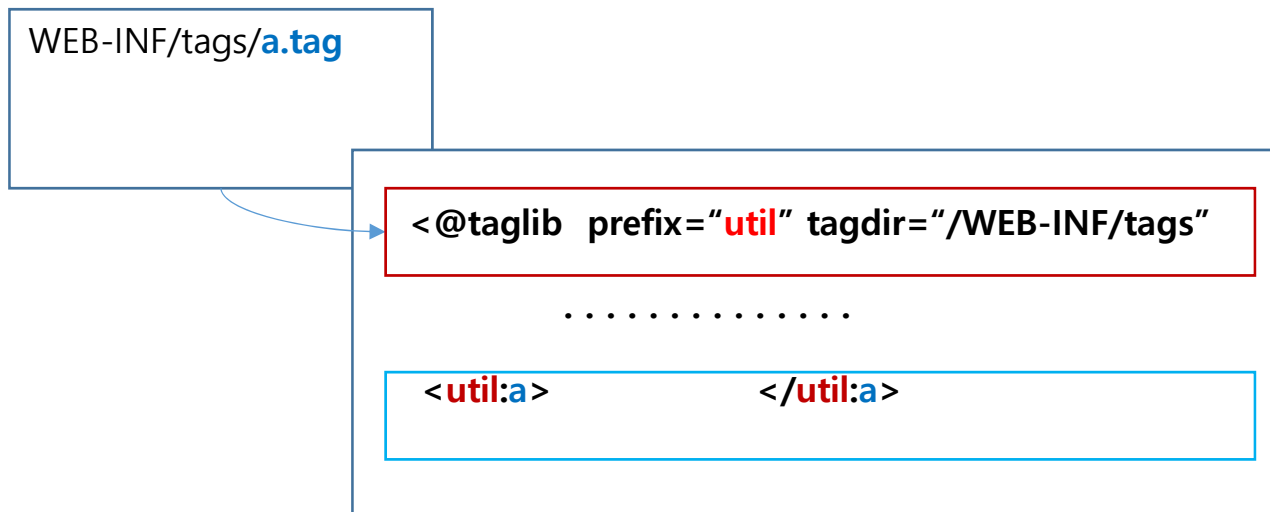
# 사용자 커스텀 태그

## ■ 사용자 커스텀 태그

### • 태그 파일 작성

- ▶ 태그 파일이란 사용자가 정의하고자 하는 태그의 기능을 구현하는 파일을 말함
- ▶ WEB-INF에 tags하는 폴더를 생성하고, tags 폴더 내에 태그 파일을 작성함
- ▶ 태그 파일은 .tag 라는 확장자를 가짐
- ▶ 파일의 이름은 접두어와 함께 커스텀 태그의 이름으로 사용됨

### • 생성된 태그 파일을 JSP 문서에 연결하여 사용 가능



# 사용자 커스텀 태그

## ■ 태그 파일의 구성

- 태그 파일의 설정은 tag 지시어를 사용해 지정함

```
<@tag body-content="바디설정" pageEncoding="문자집합" description="설명" isELIgnored="true|false">
```

### ▶ body-content

- 태그 바디에 대한 설정을 지정하며 empty, scriptless가 주로 사용됨
- empty : 태그의 바디가 없다는 것을 의미
- scriptless : 태그의 바디가 있다는 것을 의미 (기본 값)

### ▶ pageEncoding

- 태그 파일에 포함된 문자열의 문자 집합을 지정함
- 예를 들어, 태그 파일 내에 한글이 존재할 경우 utf-8로 지정해야 함

### ▶ Description

- 태그 파일의 설명을 지정

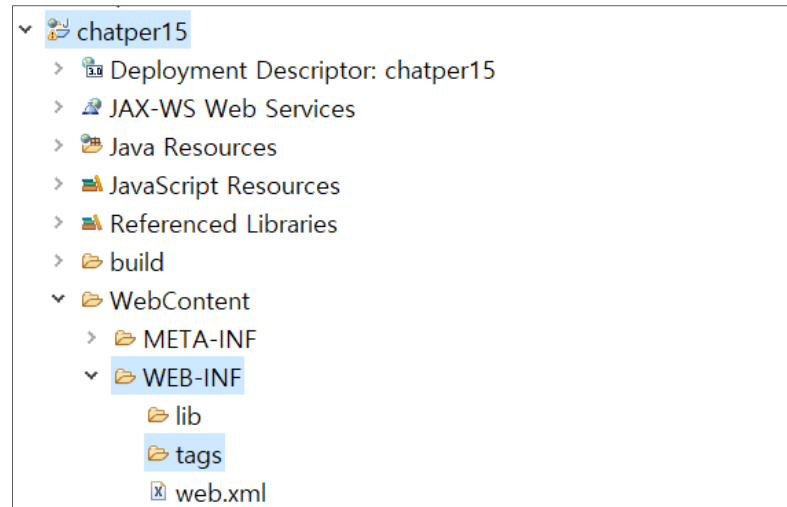
### ▶ isELIgnored

- EL을 사용할지 하지 않을 지를 지정

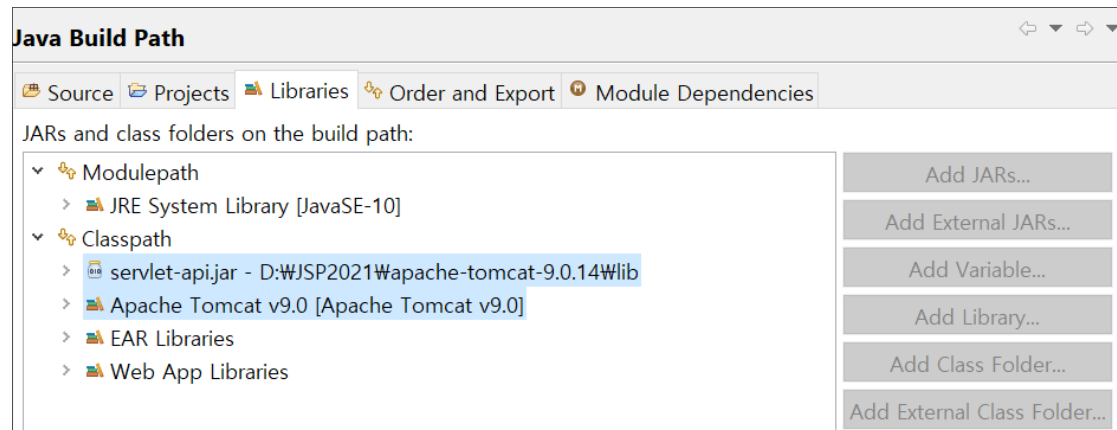
# 사용자 커스텀 태그

## ■ 프로젝트 생성

- 프로젝트 이름 : chapter15
  - ▶ web.xml 파일은 반드시 생성해야 함
  - ▶ WEB-INF에 tags 폴더를 생성



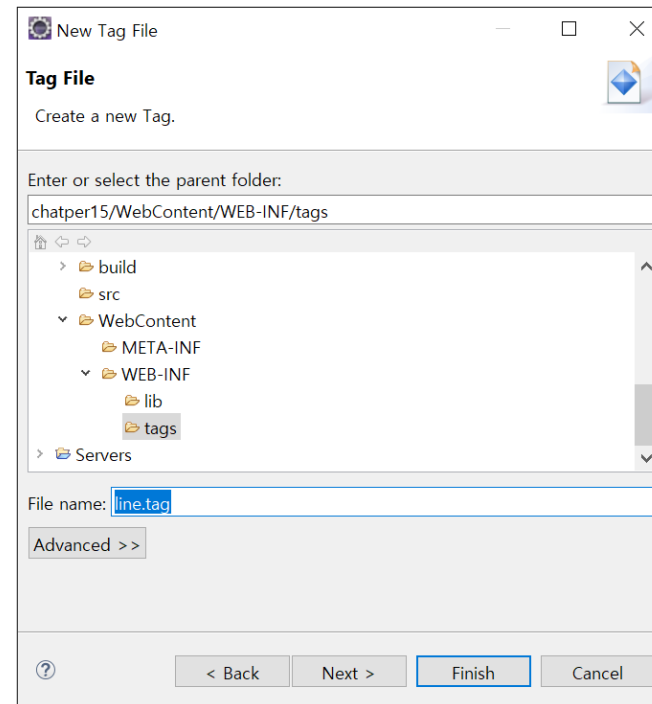
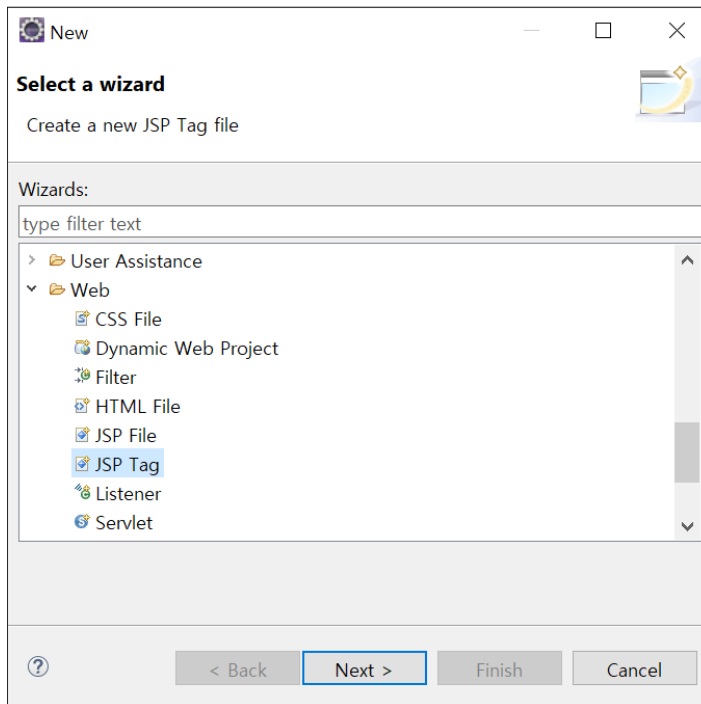
- servlet-api 라이브러리 추가



# 사용자 커스텀 태그

## ■ 태그 파일 생성

- 생성한 tags 폴더를 선택한 후 마우스 오른쪽 버튼으로 단축 메뉴 생성
  - ▶ [New]/ [Other...]/ [Web] 에서 'JSP Tag'를 선택한 후 [Next] 버튼 클릭
- WEB-INF/tags 폴더를 선택한 다음 태그 파일 이름을 지정
  - ▶ 태그 파일의 이름은 line.tag 로 지정





## ■ 태그 파일 작성

- 여러 개의 점선을 출력하는 커스텀 태그를 작성
- 생성하는 태그 사이에는 body가 존재하지 않음
  - ▶ 시작 태그는 존재하지만 종료 태그는 존재하지 않음 구조임
  - ▶ body-content 속성의 값을 empty로 지정해야 함

```
1 <%@ tag language="java" pageEncoding="UTF-8" body-content="empty" %>
2 -----<br>
3
```

# 사용자 커스텀 태그

## ■ 태그 파일을 JSP 문서에 연결

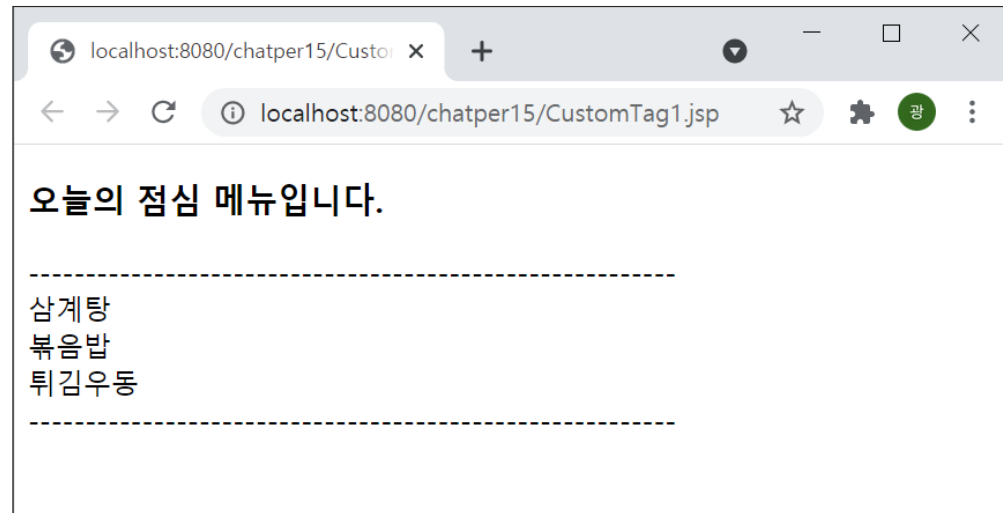
- taglib 지시어를 사용해 태그 파일을 연결

```
<%@taglib prefix="접두어" tagdir="/WEB-INF/tags" %>
```

- ▶ **prefix**
  - 태그의 이름을 구성하는 접두어를 지정
- ▶ **tagdir**
  - 태그 파일이 위치하는 폴더의 경로를 지정
- ▶ 예를 들어, **prefix** 값을 **util**로 지정하고 태그 파일의 이름이 **line.tag**인 경우 커스텀 태그의 표현

```
<util:line />
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@taglib prefix="util" tagdir="/WEB-INF/tags" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12
13   <h3>오늘의 점심 메뉴입니다.</h3>
14
15   <util:line />
16     삼계탕<br>
17     볶음밥<br>
18     튀김우동<br>
19   <util:line/>
20
21 </body>
22 </html>
```



# 사용자 커스텀 태그

## ■ 속성을 가지는 커스텀 태그 생성

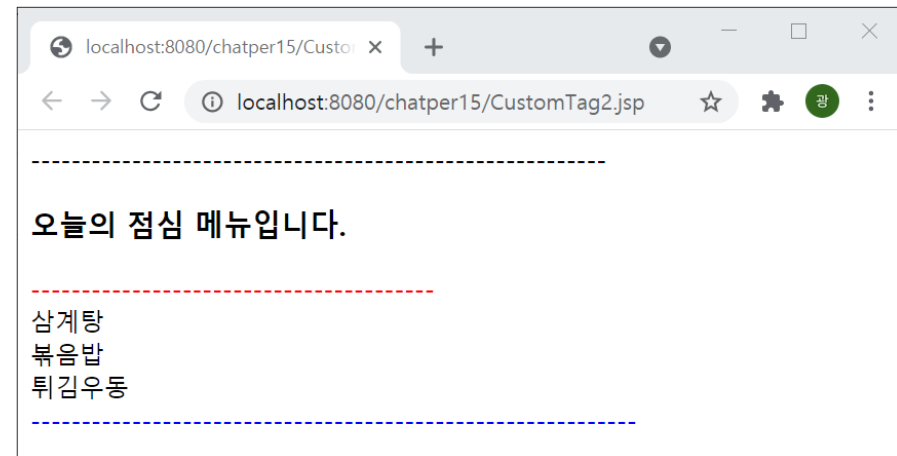
- 태그 파일에 속성을 추가하여 커스텀 태그가 속성을 사용하도록 지정할 수 있음
  - ▶ 예를 들어 ‘ ’ 문자를 연속해 출력하는 경우
    - 출력할 문자의 색상이나 개수를 전달받아 출력할 수 있음
    - ‘ ’ 문자의 색상을 지정할 수 있음
- 속성은 attribute 지시어를 사용해 속성을 지정

```
<%@attribure name="값" [ type="데이터의 클래스 타입" ] [ required="true" ]
```

- ▶ name
  - 속성의 이름을 지정
- ▶ type
  - 전달된 속성 값은 모두 String 타입으로 간주됨
  - 다른 데이터 타입의 속성 값을 전달 받을 경우 클래스 타입을 지정해야 함
- ▶ required
  - 반드시 입력되어야 하는 경우 true 값을 지정

```
1 <%@ tag language="java" pageEncoding="UTF-8" body-content="empty" %>
2
3 <%@attribute name="color" %>
4 <%@attribute name="size" type="java.lang.Integer" required="true" %>
5
6 <font color=${color}>
7
8 <%
9     for(int cnt=0; cnt < size; cnt++) {
10         out.print("-");
11     }
12
13 %>
14
15 </font>
16 <br>
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@taglib prefix="util" tagdir="/WEB-INF/tags" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12
13   <util:line />
14
15   <h3>오늘의 점심 메뉴입니다.</h3>
16
17   <util:line2 color="red" size="40"/>
18   삼계탕<br>
19   볶음밥<br>
20   튀김우동<br>
21   <util:line2 color="blue" size="60"/>
22
23 </body>
24 </html>
```



# 사용자 커스텀 태그

## ■ 바디를 가지는 커스텀 태그 생성

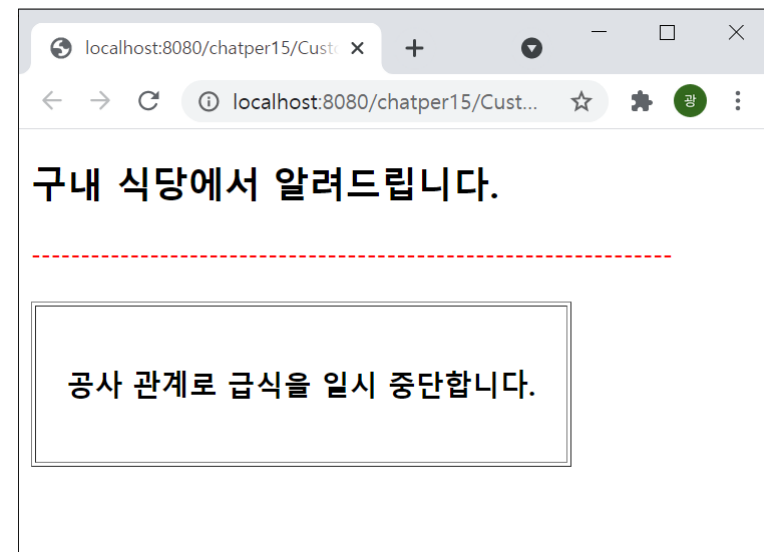
- 커스텀 태그의 시작 태그와 종료 태그 사이에 오는 내용을 태그의 바디(본체)라고 함
  - body-content의 값을 scriptless로 지정해야 함
- 본체를 나타내기 위해서는 doBody 액션 태그를 사용함

```
<jsp:doBody>
```

```
1 <%@ tag language="java" pageEncoding="UTF-8" body-content="scriptless"%>
2
3 <table border=1 cellpadding=20>
4
5     <tr>
6         <td>
7             <jsp:doBody/>
8         </td>
9     </tr>
10
11 </table>
```



```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@taglib prefix="util" tagdir="/WEB-INF/tags" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12
13   <h2>구내 식당에서 알려드립니다.</h2>
14   <util:line2 size="65" color="red"/>
15   <br>
16
17   <util:box>
18     <h3>
19       공사 관계로 급식을 일시 중단합니다.<br>
20     </h3>
21   </util:box>
22
23 </body>
24 </html>
```



**JSTL**

# JSTL의 개요

## ■ JSTL (JSP Standard Tag Library)

- 커스텀 태그 중 가장 많이 사용되는 태그를 표준화하여 라이브러리로 제공
  - JSP 내에서만 사용할 수 있는 커스텀 액션과 함수를 제공
- JSTL의 주요 용도
  - 간단한 프로그램 로직 구현
  - 제어의 이동
  - 날짜, 시간, 숫자의 포맷
  - 문자열을 처리하는 함수 호출

# JSTL의 개요

## ■ JSLT의 사용 예

### • 반복문 처리의 예

- ▶ <h3>안녕하세요</h3> 를 10번 반복해 출력한 구문

```
<c:forEach begin="1" end="10">  
  <h3>안녕하세요</h3>  
</c:forEach>
```

### • 수치 값 출력의 예

- ▶ 3.141592를 3.14로 표현하는 구문

```
<fmt:formatNumber value="3.141592" pattern="#.00" />
```

### • 문자열 처리의 예

- ▶ hello라는 문자열을 대문자로 출력하는 구문

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

# JSTL의 개요

## ■ JSTL에서 사용되는 라이브러리의 종류

속성 값	세부 기능	접두어	관련 URI
코어	변수 지원, 흐름 제어, 반복문 처리, URL 처리	c	<a href="http://java.sun.com/jsp/jstl/core">http://java.sun.com/jsp/jstl/core</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>
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>
데이터베이스	SQL	sql	<a href="http://java.sun.com/jsp/jstl/sql">http://java.sun.com/jsp/jstl/sql</a>
함수	컬렉션 처리, 문자열 처리	fn	<a href="http://java.sun.com/jsp/jstl/functions">http://java.sun.com/jsp/jstl/functions</a>

- 톰캣 컨테이너는 JSTL을 기본적으로 제공하고 있지 않으므로 직접 설치해야 함

# JSTL 라이브러리 다운로드와 설치

## ■ JSTL 라이브러리 다운로드와 설치

### • 다운로드

- ▶ <http://tomcat.apache.org/download-taglibs.cgi> 에서 다운로드함

#### Standard-1.2.5

##### Source Code Distributions

- [Source README](#)
- [zip](#) ([pgp](#), [sha512](#))

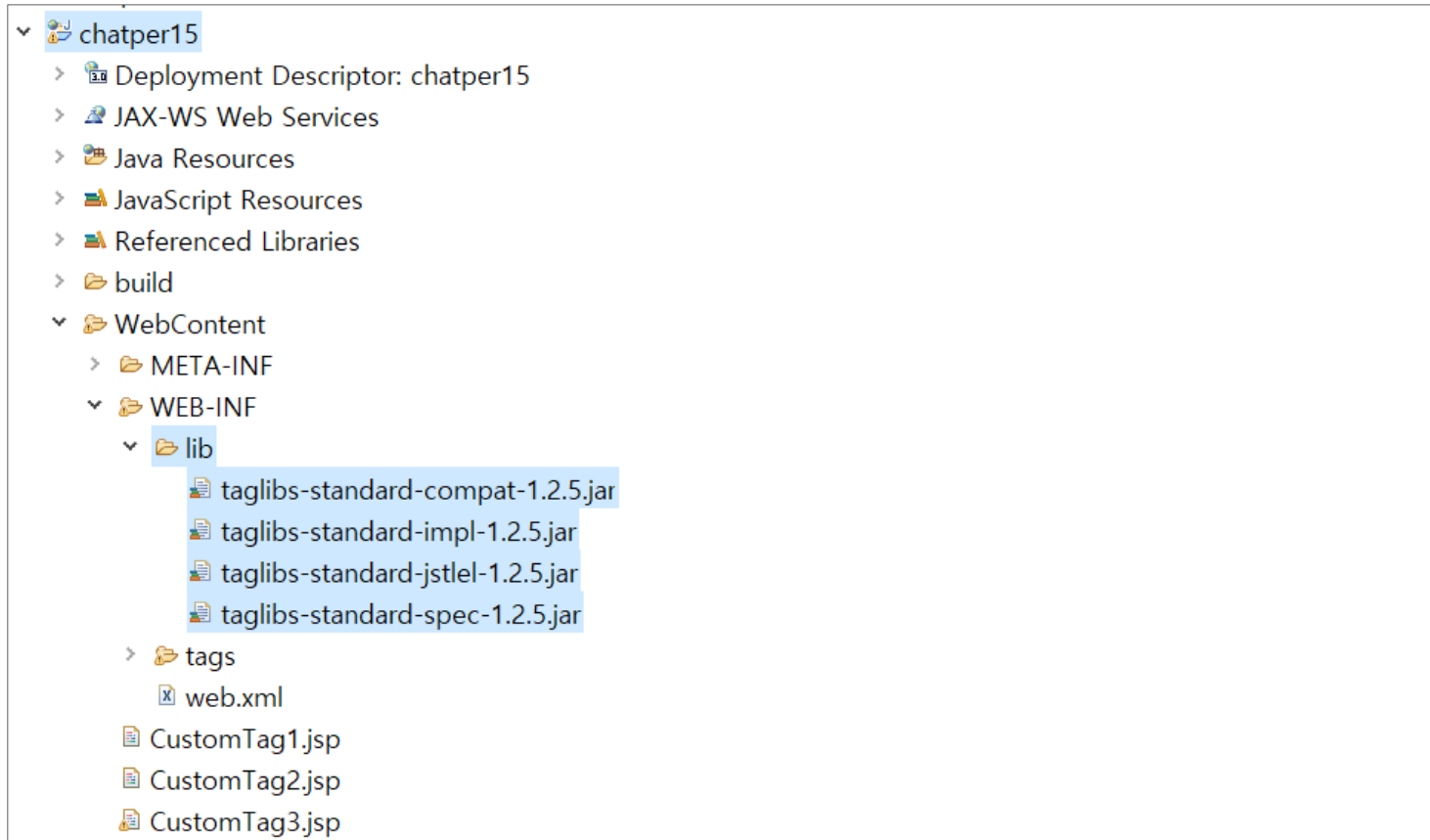
##### Jar Files

- [Binary README](#)
- Impl:
  - [taglibs-standard-impl-1.2.5.jar](#) ([pgp](#), [sha512](#))
- Spec:
  - [taglibs-standard-spec-1.2.5.jar](#) ([pgp](#), [sha512](#))
- EL:
  - [taglibs-standard-jstlel-1.2.5.jar](#) ([pgp](#), [sha512](#))
- Compat:
  - [taglibs-standard-compat-1.2.5.jar](#) ([pgp](#), [sha512](#))

# JSTL 라이브러리 다운로드와 설치

- 설치

- ▶ 다운로드한 파일을 이클립스의 '프로젝트/WebContent/WEB-INF/lib'에 저장



# JSTL 라이브러리 사용 방법

## ■ JSTL 라이브러리 사용 방법

```
<%@taglib prefix="접두어" uri="라이브러리의 관련 URI"%>
```

- taglib 지시어를 이용해서 접두어(prefix)와 라이브러리의 URI 식별자를 연결
- taglib 지시어 는 uri와 prefix라는 두 개의 속성을 사용
  - ▶ prefix : 라이브러리를 나타내는 접두어
  - ▶ URI : 해당 prefix에 대한 URI
- [예] 코어(core) 라이브러리를 사용하는 예

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



## 코어 (core) 라이브러리

# 코어 라이브러리

## ■ 코어 라이브러리의 주요 태그

기능	태그	설명
변수 선언/ 출력/제거	<c:set>	변수를 선언하고 값을 초기화
	<c:out>	변수의 내용을 출력
	<c:remove>	변수를 제거
제어	<c:if>	조건 처리
	<c:choose>	다중 조건 처리
	<c:forEach>	반복 처리
	<c:forEachItem>	구분자를 이용한 토큰 처리
	<c:catch>	예외 처리에 사용
URL 관리	<c:import>	외부 자원 임포트
	<c:uri>	URI 정보 저장
	<c:redirect>	지정한 URL로 이동

## 변수 관련 라이브러리 <c:set>

### ■ <c:set> 태그

- 변수를 선언하고 초기값을 지정하는 태그

```
<c:set var="변수명" value="값" [scope="변수의 영역"] />
```

- 변수 선언에 있어서 Java와의 차이점

- ▶ Java : 변수를 선언할 때 변수의 데이터 타입을 반드시 지정해야 함
- ▶ <c:set> : 변수를 선언할 때 데이터 타입을 지정

- 커스텀 태그는 XML 문법을 따르므로 숫자 데이터라도 인용 부호 내에 표현해야 함

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

- value 값 위치에 EL 식을 쓸 수도 있음

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

## 변수 관련 라이브러리 <c:set>

- <c:set>으로 생성된 변수는 스크립팅 요소에서 사용할 수 없음

▶ EL에는 사용할 수 있지만 표현식에는 사용할 수 없음

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

```
${num} ( O )
```

```
<%= num %> ( X )
```

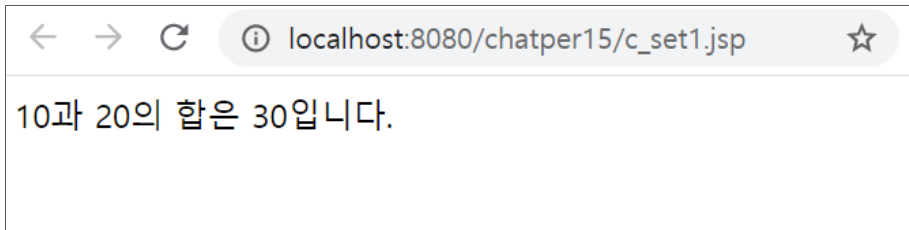
- 스크립팅 요소에서 생성된 변수는 <c:set>에서 사용할 수 있음

▶ <c:set>의 value 속성의 값으로 표현식을 사용할 수 있음

```
<%  
    int num1=100;  
    int num2=200;  
%>
```

```
<c:set var="num" value="<%=num2%>"/> ( O )
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core"%>
5
6 <c:set var="num1" value="10" />
7 <c:set var="num2" value="20" />
8 <c:set var="sum" value="${num1+num2}" />
9
10 <!DOCTYPE html>
11 <html>
12 <head>
13   <meta charset="UTF-8">
14 </head>
15 <body>
16
17   ${num1}과 ${num2}의 합은 ${sum}입니다.
18
19 </body>
20 </html>
```



## 변수 관련 라이브러리 <c:set>

### ■ <c:set>의 활용 예

- 다음의 구문은 모두 동일한 기능을 수행함

```
< a href="http://localhost:8080/chapter14/test.jsp">
```

- 컨텍스트 이름(chapter14)이 변경되면 모든 소스를 변경해야 함

```
< a href="<%=request.getContextPath() %>/test.jsp">
```

- 컨텍스트 이름이 변경되어도 상관없으나 jsp 코드를 사용해야 함

```
< a href="${pageContext.request.contextPath}/test.jsp">
```

- 컨텍스트 이름이 변경되어도 상관없고 EL을 사용하지만 href 속성의 이름이 길다는 단점이 있음

- EL과 <c:set>을 사용하면 긴 속성을 짧게 표현할 수 있음

```
<c:set name="context" value="${pageContext.request.contextPath}" />
```

```
< a href="${context}/test.jsp">
```

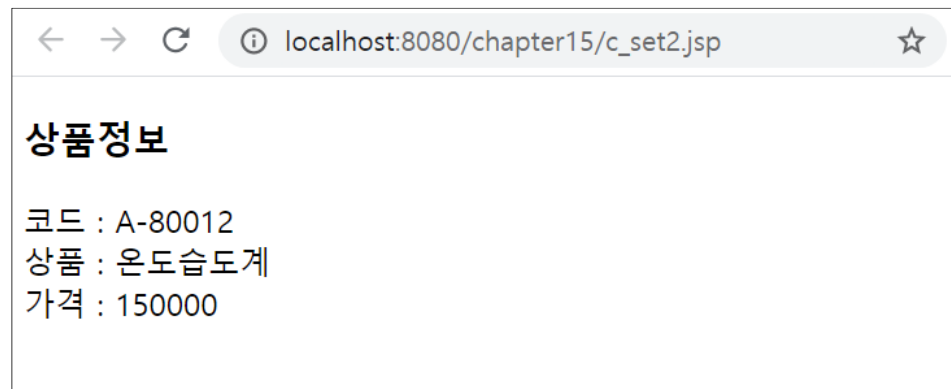
▶ 자주 사용되는 방법임

### ■ <c:set>의 scope 속성

- scope 속성을 사용하면 page 영역 뿐만 아니라 request, session, application 영역의 속성을 지정 가능

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <% request.setCharacterEncoding("utf-8"); %>
7
8 <c:set var="CODE" value="A-80012" scope="request" />
9 <c:set var="NAME" value="온도습도계" scope="request" />
10 <c:set var="PRICE" value="${150000}" scope="request" />
11
12 <jsp:forward page="c_set3.jsp"></jsp:forward>
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3 <!DOCTYPE html>
4 <html>
5 <head>
6     <meta charset="UTF-8">
7 </head>
8 <body>
9
10     <h3>상품정보</h3>
11     코드 : ${CODE} <br>
12     상품 : ${NAME} <br>
13     가격 : ${PRICE} <br>
14
15 </body>
16 </html>
```





## 변수 관련 라이브러리 <c:remove>

### ■ <c:remove> 태그

- <c:set>으로 지정한 변수를 제거하기 위한 태그

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

- var 속성

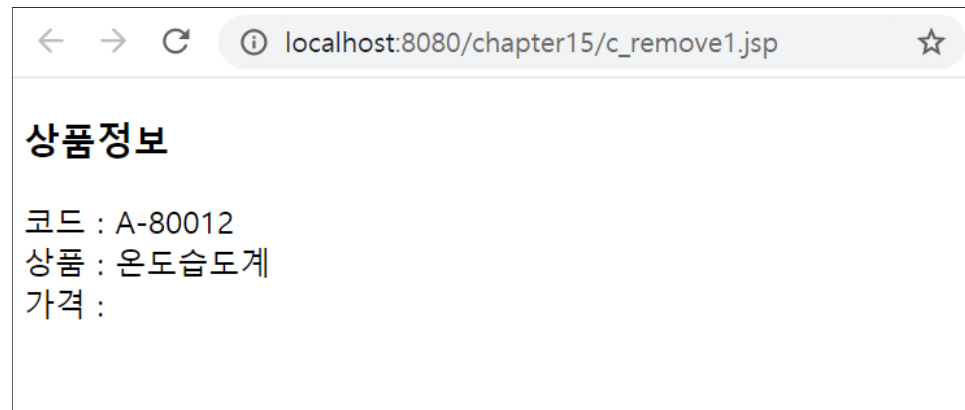
- ▶ 제거하고자 하는 변수의 이름을 지정

- scope 속성

- ▶ 변수가 page, request, session, application 영역에 속성으로 저장된 경우, 해당 영역의 변수를 제거하기 위해 사용
  - [예] scope를 request로 지정한 경우, request 영역의 속성 중 지정한 속성(변수)를 제거
- ▶ scope 속성을 생략할 경우, 모든 영역(page, request, session, application)에 있는 지정된 동일한 이름의 변수를 제거
  - 특정 영역의 변수를 제거하려면 scope 속성에 영역을 지정해야 함

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <% request.setCharacterEncoding("utf-8"); %>
7
8 <c:set var="CODE" value="A-80012" scope="request" />
9 <c:set var="NAME" value="온도습도계" scope="request" />
10 <c:set var="PRICE" value="${150000}" scope="request" />
11
12 <jsp:forward page="c_remove2.jsp"></jsp:forward>
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <c:remove var="PRICE" scope="request"/>
7
8 <!DOCTYPE html>
9 <html>
10 <head>
11   <meta charset="UTF-8">
12 </head>
13 <body>
14
15   <h3>상품정보</h3>
16   코드 : ${CODE} <br>
17   상품 : ${NAME} <br>
18   가격 : ${PRICE} <br>
19
20 </body>
21 </html>
```



## 제어 관련 라이브러리 <c:if>

### ■ <c:if> 태그

- 자바의 if구문과 같은 역할을 수행하는 태그

```
<c:if test="조건식" [var="변수명"] [scope="변수의 영역"] >  
    조건을 만족하면 처리될 문장  
</c:if>
```

- <c:if> 태그의 속성

- ▶ test

- 조건식을 지정하는 속성
- 조건식은 반드시 EL로 기술되어야 하며, 자바 코드나 표현식은 사용할 수 없음

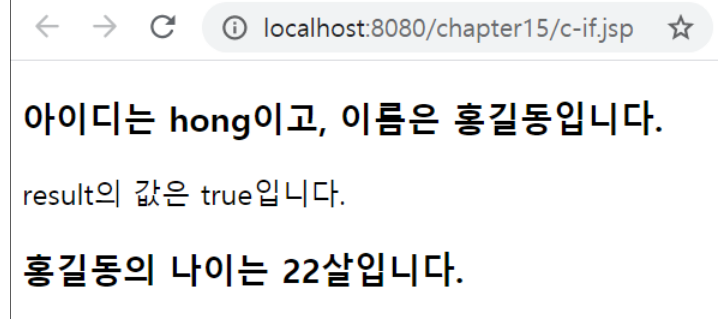
- ▶ var

- test 속성의 결과를 저장하기 위한 변수를 지정
- true나 false가 저장됨

- ▶ scope

- 변수가 저장되는 영역

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <c:set var="id" value="hong"/>
7 <c:set var="name" value="'홍길동'"/>
8 <c:set var="age" value="22"/>
9
10 <!DOCTYPE html>
11 <html>
12 <head>
13   <meta charset="UTF-8">
14 </head>
15 <body>
16
17   <c:if test="${(id=='hong') && (name=='홍길동')}" var="result">
18     <h3>아이디는 ${id}이고, 이름은 ${name }입니다.</h3>
19     result의 값은 ${result}입니다.
20   </c:if>
21
22   <c:if test="${age==22}">
23     <h3>${name }의 나이는 ${age}살입니다.</h3>
24   </c:if>
25
26 </body>
27 </html>
```



## 제어 관련 라이브러리 <c:choose>

### ■ <c:choose> 태그

- 하나 이상의 조건이 존재할 때 사용

<c:choose>

<c:when test="조건식1" >

조건식1이 만족할 경우 수행

</c:when>

<c:when test="조건식2" >

조건식2가 만족할 경우 수행

</c:when>

...

<c:otherwise>

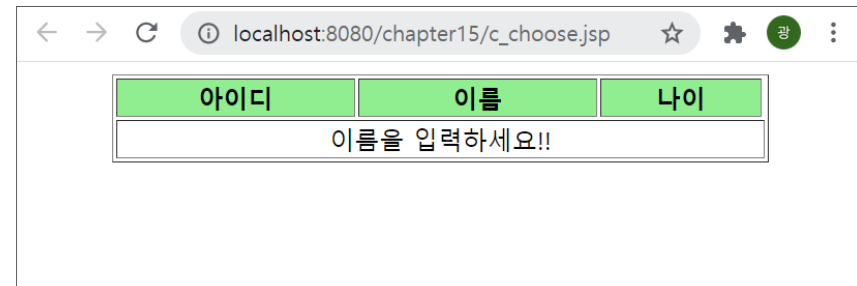
모든 조건이 만족하지 않을 경우 수행

</c:otherwise>

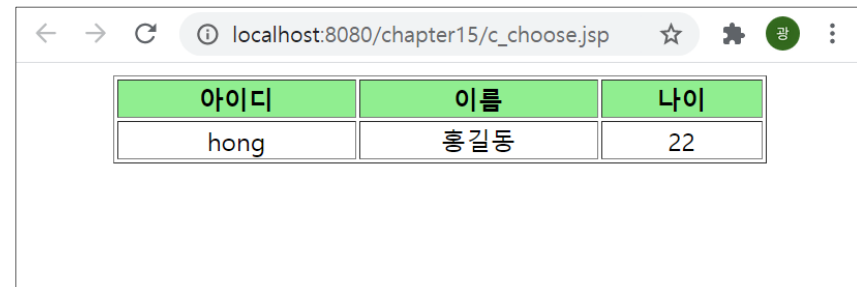
</c:choose>

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <c:set var="id" value="hong"/>
7 <c:set var="name" value="\${ '홍길동' }"/>
8 <c:set var="age" value="\${22}"/>
9
10 <!DOCTYPE html>
11 <html>
12 <head>
13   <meta charset="UTF-8">
14 </head>
15 <body>
16
17   <table align="center" border="1" >
18
19     <tr align="center" bgcolor="lightgreen">
20       <td width=150 ><b>아이디</b></td>
21       <td width=150 ><b>이름</b></td>
22       <td width=100><b>나이</b></td>
23     </tr>
24
```

```
25 <c:choose>
26
27 <c:when test="${empty name}">
28 <tr align="center">
29 <td colspan=5 >이름을 입력하세요!!</td>
30 </tr>
31 </c:when>
32
33 <c:otherwise >
34 <tr align="center">
35 <td>${id}</td>
36 <td>${name}</td>
37 <td>${age}</td>
38 </tr>
39 </c:otherwise>
40
41 </c:choose>
42 </body>
43 </html>
```



아이디	이름	나이
	이름을 입력하세요!!	



아이디	이름	나이
hong	홍길동	22



## 제어 관련 라이브러리 <c:forEach>

### ■ <c:forEach> 태그

- 순환문 역할을 수행하는 태그

```
<c:forEach [ var="변수 이름" ] begin="시작값" end="마지막값 " step="증가값" >  
    반복할 내용  
</c:forEach>
```

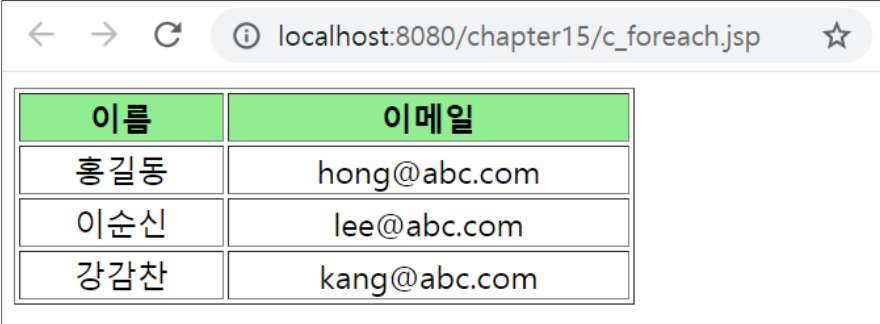
- <c:forEach> 태그

- ▶ var
  - 순환을 수행하는 제어 변수 (생략 가능)
- ▶ begin/ end
  - 순환을 위한 제어 변수의 시작 값/ 종료 값
- ▶ step
  - 매 순환할 때마다 제어 변수의 증가 값 (생략하면 1로 인식됨)

```
1 package chapter15;
2
3 public class MemberBean {
4
5     private String name;
6     private String email;
7
8     public MemberBean() {
9
10    }
11
12    public MemberBean(String name, String email) {
13        this.name = name;
14        this.email = email;
15    }
16
17    public String getName() {
18        return name;
19    }
20    public void setName(String name) {
21        this.name = name;
22    }
23
24    public String getEmail() {
25        return email;
26    }
27    public void setEmail(String email) {
28        this.email = email;
29    }
30 }
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ page import="java.util.*" %>
5 <%@ page import="chapter15.*" %>
6
7 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
8
9 <%
10   List membersList = new ArrayList();
11
12   MemberBean m1 = new MemberBean("홍길동", "hong@abc.com");
13   MemberBean m2 = new MemberBean("이순신", "lee@abc.com");
14   MemberBean m3 = new MemberBean("강감찬", "kang@abc.com");
15
16   membersList.add(m1);
17   membersList.add(m2);
18   membersList.add(m3);
19 %>
20
21 <c:set var="membersList" value="<%= membersList%>" />
22
23 <!DOCTYPE html>
24 <html>
25 <head>
26   <meta charset="UTF-8">
27 </head>
```

```
28
29 <body>
30
31 <table border="1">
32
33 <tr align="center" bgcolor="lightgreen">
34 <td width="100" ><b>이름</b></td>
35 <td width="200"><b>이메일</b></td>
36 </tr>
37
38 <c:forEach var="i" begin="0" end="2" step="1" >
39 <tr align="center">
40 <td>${membersList[i].name}</td>
41 <td>${membersList[i].email}</td>
42 </tr>
43 </c:forEach>
44
45 </table>
46
47 </body>
48 </html>
```



이름	이메일
홍길동	hong@abc.com
이순신	lee@abc.com
강감찬	kang@abc.com

## 제어 관련 라이브러리 <c:forTokens>

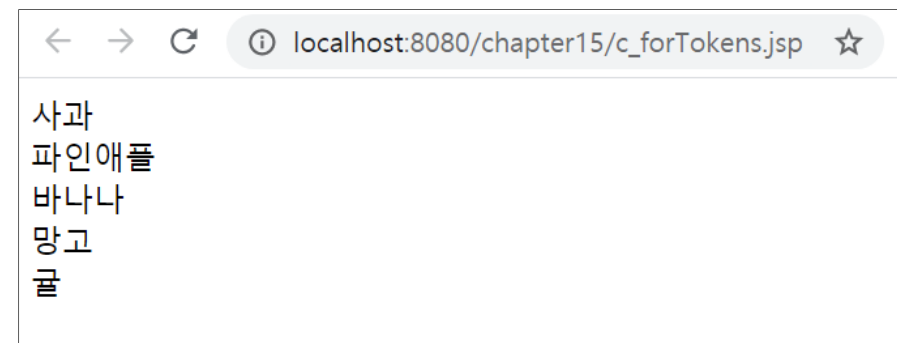
### ■ <c:forTokens> 태그

- 문자열에 포함된 토큰을 분리해 각각의 토큰을 순서대로 출력
  - ▶ 자바에서 for 구문과 java.util.StringTokenizer를 결합한 기능을 제공

```
<c:forTokens var="변수" items="토큰이 포함된 문자열" delims="분리자">  
    출력할 내용  
</c:forTokens>
```

- ▶ var는 분리된 토큰들이 저장되는 변수를 의미

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12
13   <c:set var="fruits" value="사과, 파인애플, 바나나, 망고, 귤" />
14
15   <c:forTokens var="token" items="${fruits}" delims="," >
16     ${token} <br>
17   </c:forTokens>
18
19 </body>
20 </html>
```



## 제어 관련 라이브러리 <c:catch>

### ■ <c:catch> 태그

- 자바의 try ~ catch 구문에서 try 구문과 동일한 기능을 수행
  - ▶ 태그의 이름이 catch이지만 Java에서 try에 해당하는 점에 유의해야 함

```
<c:catch var="변수" >
```

오류가 발생할 가능성이 있는 구문

```
</c:catch>
```

- ▶ var는 exception 객체를 저장할 변수를 의미

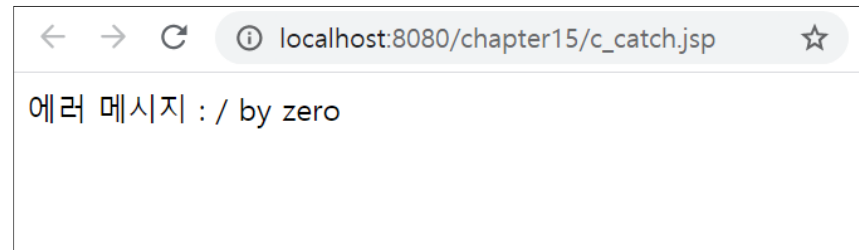
```
<c:catch var="e" >
```

```
<% int result = num1 / num2 %>
```

```
</c:catch>
```

- try ~ catch 구문에서 catch 구문을 지정하는 별도의 구문은 없음
  - ▶ 다른 커스텀 태그를 사용해 직접 지정해야 함

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <%
7     int num1 = 100;
8     int num2 = 0;
9 %>
10
11 <c:catch var="e">
12     <% int result = num1/num2; %>
13     결과는<%=result%>
14 </c:catch>
15
16 <c:if test="${e != null}">
17     에러 메시지 : ${e.message}
18 </c:if>
```





## 제어 관련 라이브러리 <c:out>

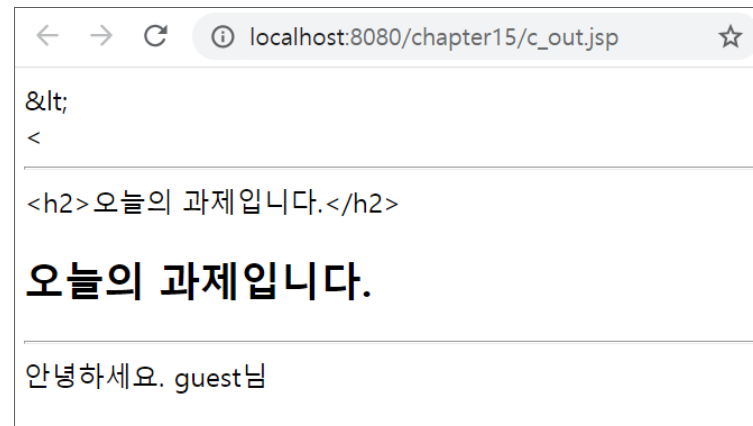
### ■ <c:out> 태그

- 지정한 값을 브라우저에 출력해 주는 기능을 수행

```
<c:out value="출력할 값" [default="기본값"] [escapeXml="true|false" ] />
```

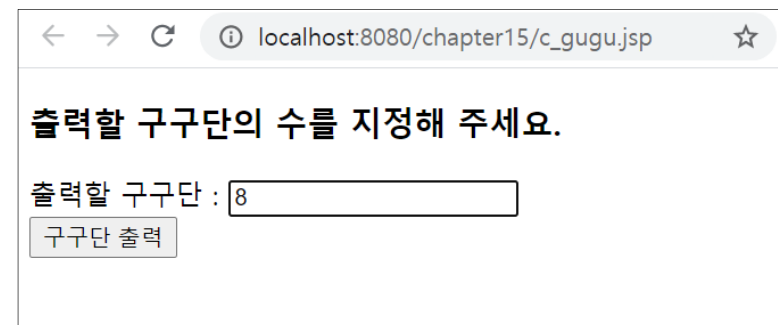
- HTML에서 사용되는 특수 코드를 처리하기 위해서 <c:out> 태그를 사용
  - ▶ <c:out>는 특수 코드(예를 들어, &gt;)를 해당 문자(기호)로 변경할 것인지를 선택할 수 있음
- <c:out> 태그의 속성
  - ▶ value
    - 출력할 값이나 변수를 지정
  - ▶ default
    - value 속성의 값이 없을 때 출력할 기본 값
  - ▶ escapeXml
    - 특수 코드를 해당 문자(기호)로 변환 여부 지정
    - true 일 경우 : value 속성이 특수 코드(또는 HTML 태그)를 가질 경우를 그대로 출력
    - false 일 경우 : value 속성이 특수 코드(또는 HTML 태그)를 가질 경우 변환(적용)하여 출력

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12   <c:out value="&lt;" escapeXml="true" /><br>
13   <c:out value="&lt;" escapeXml="false" /><br><hr>
14
15   <c:out value="<h2>오늘의 과제입니다.</h2>" escapeXml="true" /><br>
16   <c:out value="<h2>오늘의 과제입니다.</h2>" escapeXml="false" /><hr>
17
18   안녕하세요. <c:out value="\${userid}" default="guest" />님
19 </body>
20 </html>
```



## • 구구단 출력의 예

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <!DOCTYPE html>
5 <html>
6 <head>
7   <meta charset="UTF-8">
8 </head>
9 <body>
10
11   <h3>출력할 구구단의 수를 지정해 주세요.</h3>
12
13   <form method=get action="c_gugu_result.jsp">
14     출력할 구구단 : <input type=text name="dan" /> <br>
15     <input type="submit" value="구구단 출력">
16   </form>
17
18 </body>
19 </html>
```



출력할 구구단의 수를 지정해 주세요.

출력할 구구단 :

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <% request.setCharacterEncoding("UTF-8"); %>
6
7 <!DOCTYPE html>
8 <html>
9 <head>
10   <meta charset="UTF-8">
11 </head>
12 <body>
13
14   <c:set var="dan" value="${param.dan }" />
15
16   <c:out value="${dan}" />단 출력 <hr>
17
18   <c:forEach var="i" begin="1" end="9" step="1" >
19     <c:out value="${dan}" /> * <c:out value="${i}" />
20     <c:out value="' ' = '}'" />
21     <c:out value="${i*dan }" /><br>
22   </c:forEach>
23
24 </body>
25 </html>
```

## 8단 출력

8 \* 1 = 8  
8 \* 2 = 16  
8 \* 3 = 24  
8 \* 4 = 32  
8 \* 5 = 40  
8 \* 6 = 48  
8 \* 7 = 56  
8 \* 8 = 64  
8 \* 9 = 72

## URL 관련 라이브러리 <c:redirect>

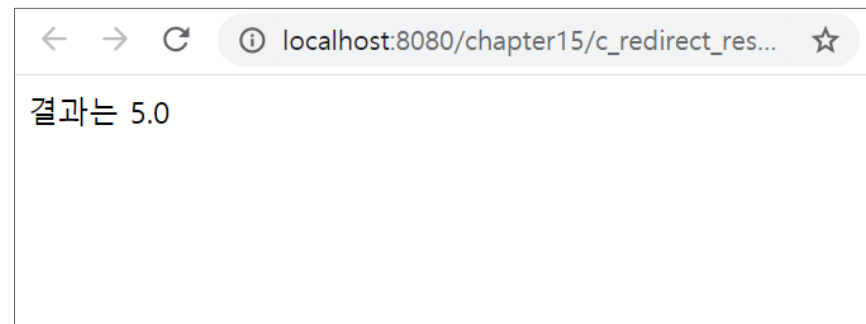
### ■ <c:redirect> 태그

- 지정된 URL로 제어를 이동하기 위한 태그
  - ▶ response.sendRedirect() 와 동일한 기능 수행
- 파라미터를 함께 전달하기 위해 <c:param> 태그를 사용할 수 있음

```
<c:redirect url="이동할 URL">  
  [ <c:param name="파라미터" value="값" /> ]  
  .....  
</c:redirect>
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12
13   <c:redirect url="c_redirect_result.jsp">
14     <c:param name="num1" value="100" />
15     <c:param name="num2" value="20" />
16   </c:redirect>
17
18 </body>
19 </html>
```

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9   <meta charset="UTF-8">
10 </head>
11 <body>
12
13   <c:catch var="e">
14     결과는 ${param.num1 / param.num2}
15   </c:catch>
16
17   <c:if test="${e != null}">
18     에러 메시지 : ${e.message}
19   </c:if>
20
21 </body>
22 </html>
```



# URL 관련 라이브러리 <c:url>

## ■ <c:url> 태그

- URL 정보를 저장하기 위한 태그

- ▶ <c:set> 태그와 유사한 기능을 수행

```
<c:url var="변수" value="URL정보">  
    [ <c:param name="파라미터" value="값" /> ]  
    .....  
</c:url>
```

- <c:set> 태그와의 차이점

- ▶ <c:set> 태그

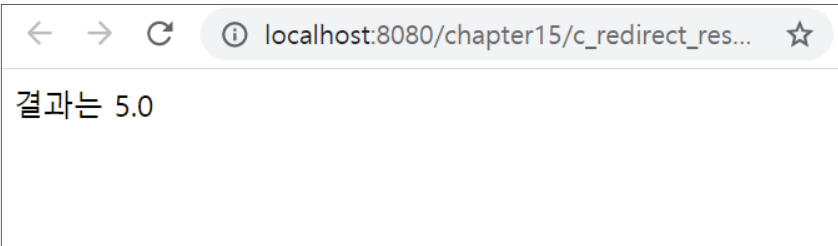
- 일반 변수를 생성하고 데이터를 저장

- ▶ <c:url> 태그

- 변수를 생성하고 URL 정보를 저장 ( 해당 URL로 바로 분기하지 않고 정보만을 저장 )
- URL로 전달할 파라미터 지정 가능 ( <c:param> 태그를 사용 )



```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5
6 <!DOCTYPE html>
7 <html>
8 <head>
9     <meta charset="UTF-8">
10 </head>
11 <body>
12
13     <c:url var="next" value="c_redirect_result.jsp">
14         <c:param name="num1" value="100" />
15         <c:param name="num2" value="20" />
16     </c:url>
17
18     <c:redirect url="${next}" />
19
20 </body>
21 </html>
```



## 포메팅 (formatting) 라이브러리

# 포매팅 라이브러리

## ■ 포매팅 라이브러리

- 숫자나 문자, 시간 데이터 등을 원하는 형태로 표현

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

▶ 접두어로 fmt를 주로 사용

- 포매팅 라이브러리의 기능

- ▶ 날짜의 표시 형식
- ▶ 숫자의 천 단위 소수점 사용 여부
- ▶ 국가별 언어 지정 및 시간 출력 등

# 포메팅 라이브러리

## ■ 포메팅 라이브러리의 주요 태그

태그	설명
<fmt:formatDate>	날짜와 시각 정보의 출력 형식을 지정
<fmt:formatNumber>	숫자 정보의 출력 양식 지정
<fmt:timeZone>	지역 시간대에 따른 날짜 시간 정보 출력
<fmt:setTimeZone>	
<fmt:setlocal>	국가와 언어 별로 통화량, 날짜와 시간 정보를 출력하기

## 포메팅 라이브러리 <c:formatDate>

### ■ <c:formatDate> 태그

- 날짜와 시각의 형식을 지정하는 태그

```
<fmt:formatDate value="날짜객체" type="출력정보" dateStyle="값" timeStyle="값" pattern="패턴" />
```

- <c:formatDate> 태그를 사용하려면 Date 객체를 먼저 생성해야 함
  - ▶ 출력할 날짜와 시각을 Date 클래스 타입의 객체로 넘겨줘야 하기 때문

## 포메팅 라이브러리 <c:formatDate>

- 주요 속성

- ▶ value

- 출력할 날짜와 시간을 가진 객체

- ▶ type

- 출력할 날짜/ 시간 정보의 종류를 지정
    - date : 날짜 정보
    - time : 시간 정보
    - both : 날짜 정보와 시간 정보

- ▶ dateStyle

- 날짜의 출력 형식을 지정
    - full, long, medium, short를 지정할 수 있음

- ▶ timeStyle

- 시간 출력 형식을 지정
    - full, long, medium, short를 지정할 수 있음

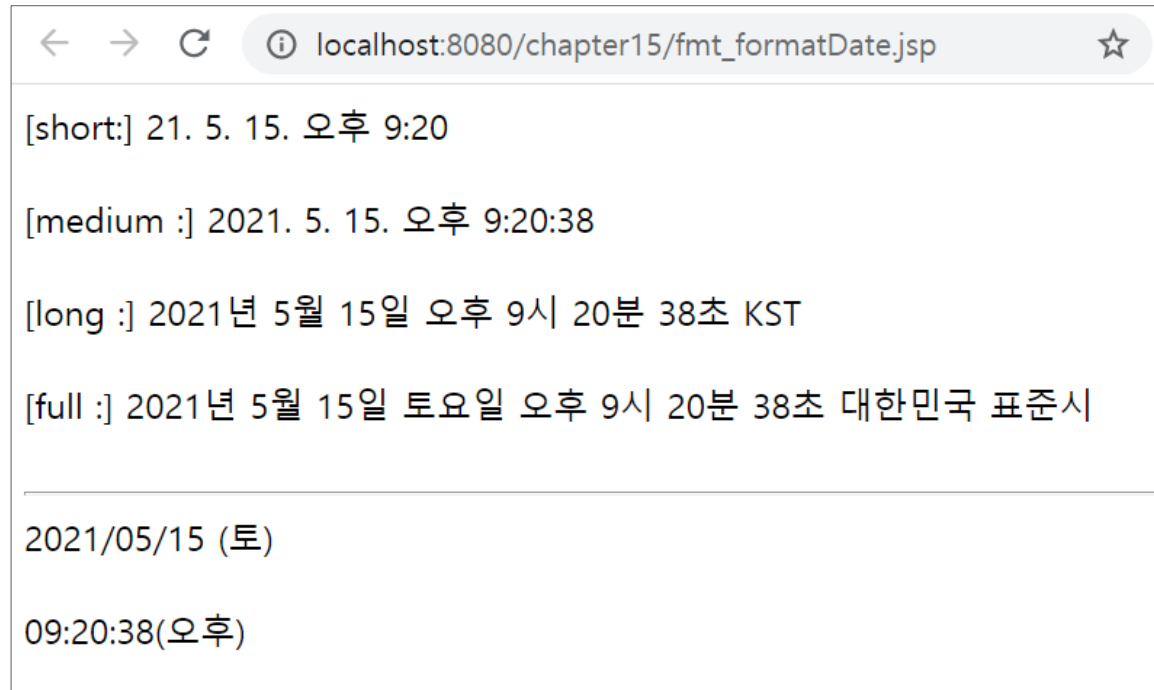
- ▶ pattern

- 직접 출력 형식을 지정
    - 자바 클래스 SimpleDateFormat에 지정된 패턴을 사용

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.util.Date" %>
4
5 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
6 <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
7
8 <c:set var="today" value="<%=new Date() %>" />
9
10 <!DOCTYPE html>
11 <html>
12 <head>
13   <meta charset="UTF-8">
14 </head>
15 <body>
16
17   [short:] <fmt:formatDate value="${today}" type="both" dateStyle="short" timeStyle="short" />
18   <br><br>
19   [medium :] <fmt:formatDate value="${today}" type="both" dateStyle="medium" timeStyle="medium"/>
20   <br><br>
21   [long :] <fmt:formatDate value="${today}" type="both" dateStyle="long" timeStyle="long"/>
22   <br><br>
23   [full :] <fmt:formatDate value="${today}" type="both" dateStyle="full" timeStyle="full"/>
24   <br><br><hr>
25
26   <fmt:formatDate value="${today}" type="date" pattern="yyyy/MM/dd (E)" />
27   <br><br>
28   <fmt:formatDate value="${today}" type="time" pattern="hh:mm:ss(a)" />
29
30 </body>
31 </html>

```



A screenshot of a web browser window displaying the output of the <c:formatDate> tag. The browser's address bar shows the URL 'localhost:8080/chapter15/fmt\_formatDate.jsp'. The page content displays four different date and time formats: '[short:] 21. 5. 15. 오후 9:20', '[medium :] 2021. 5. 15. 오후 9:20:38', '[long :] 2021년 5월 15일 오후 9시 20분 38초 KST', and '[full :] 2021년 5월 15일 토요일 오후 9시 20분 38초 대한민국 표준시'. Below these, there is a horizontal line, followed by the date '2021/05/15 (토)' and the time '09:20:38(오후)'.

```
< > ↻ ⓘ localhost:8080/chapter15/fmt_formatDate.jsp ☆
```

[short:] 21. 5. 15. 오후 9:20

[medium :] 2021. 5. 15. 오후 9:20:38

[long :] 2021년 5월 15일 오후 9시 20분 38초 KST

[full :] 2021년 5월 15일 토요일 오후 9시 20분 38초 대한민국 표준시

---

2021/05/15 (토)

09:20:38(오후)



## 포메팅 라이브러리 <c:formatNumber>

### ■ <c:formatNumber> 태그

- 숫자 형식을 지정하기 위한 태그

```
<fmt:formatNumber value="숫자정보" type="percent|number|currency"
                  groupingUsed="true|false" currencyCode="통화코드"
                  currencySymbol="통화기호" pattern="패턴" />
```

- value

- ▶ 형식을 지정해 출력할 숫자 정보

- type

- ▶ 출력할 숫자 관련 정보의 형식

- percent : 백분율 형식
    - number : 숫자 형식
    - currency : 화폐 통화량 출력

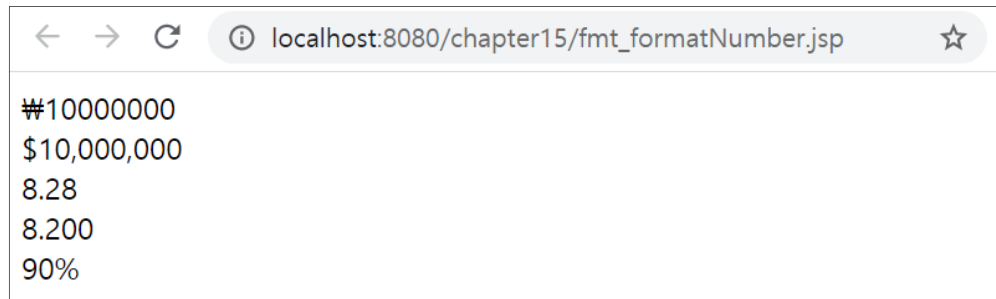
## 포메팅 라이브러리 <c:formatNumber>

- **groupingUsed**
  - ▶ 천단위 콤마(,) 기호로 구분 여부를 지정
    - 기본값은 true
- **currencyCode**
  - ▶ 출력할 통화 코드를 지정
    - 한국 원화는 KRW
- **currencySymbol**
  - ▶ 출력할 통화 기호 지정
- **pattern**
  - ▶ 출력할 숫자 정보의 패턴 지정
  - ▶ 실수 부분의 자릿수 지정 등을 지정

```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
6
7 <!DOCTYPE html>
8 <html>
9 <head>
10   <meta charset="UTF-8">
11 </head>
12 <body>
13
14   <c:set var="price" value="10000000"/>
15
16   <fmt:formatNumber value="${price}" type="currency" currencyCode="KRW" groupingUsed="false" /><br>
17   <fmt:formatNumber value="${price}" type="currency" currencySymbol="$" groupingUsed="true" /><br>
18
19   <fmt:formatNumber value="8.2789" type="number" pattern="#.00" /><br>
20   <fmt:formatNumber value="8.2" type="number" pattern="#.000" /><br>
21
22   <fmt:formatNumber value="0.9" type="percent" /><br>
23
24 </body>
25 </html>

```



## 포메팅 라이브러리 <c:timeZone>, <c:setTimeZone>

### ■ <c:timeZone>와 <c:setTimeZone> 태그

- 시간 대마다 달라지는 시각 정보를 자동으로 출력하는 태그
- <c:timeZone>
  - ▶ 태그 사이에 존재하는 날짜와 시각 정보를 지정한 시간대로 출력

```
<fmt:timeZone value="지역이름">
```

날짜와 시각 표시

```
</fmt:timezone>
```

- <c:setTimeZone>
  - ▶ 태그 아래의 모든 날짜와 시간 관련 코드를 지정한 지역 이름 시간대로 출력
    - 다음 시간 설정이 나타날 때까지 현재 설정이 유지됨

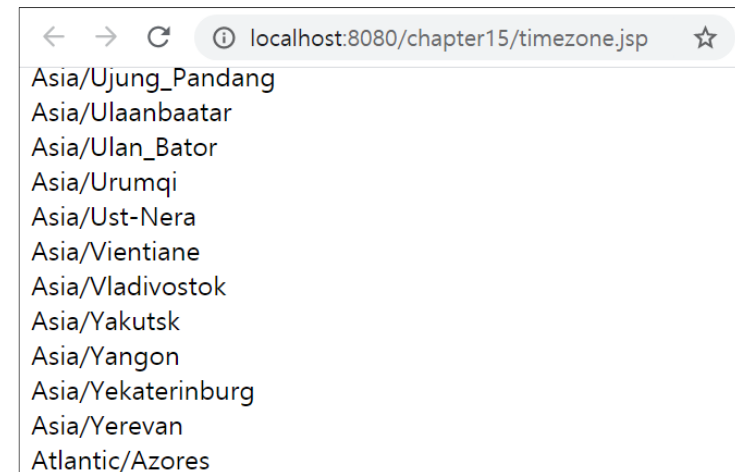
```
<fmt:setTimeZone value="지역이름">
```

## 포메팅 라이브러리 <c:timeZone>, <c:setTimeZone>

### ■ 지역 이름 추출

- java.util.TimeZone 객체의 getAvailableIDs() 메서드를 통해 추출 가능
  - ▶ 지역 이름들을 배열로 반환

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.util.TimeZone" %>
4
5 <!DOCTYPE html>
6 <html>
7 <head>
8   <meta charset="UTF-8">
9 </head>
10 <body>
11
12 <%
13   String arr[] = TimeZone.getAvailableIDs();
14   for(int i=0; i<arr.length; i++) {
15     out.println(arr[i] + "<br>");
16   }
17 %>
18
19 </body>
20 </html>
```

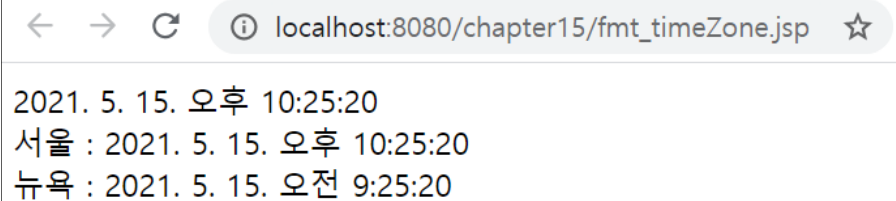


← → ↻ ⓘ localhost:8080/chapter15/timezone.jsp ☆

Asia/Ujung\_Pandang  
Asia/Ulaanbaatar  
Asia/Ulan\_Bator  
Asia/Urumqi  
Asia/Ust-Nera  
Asia/Vientiane  
Asia/Vladivostok  
Asia/Yakutsk  
Asia/Yangon  
Asia/Yekaterinburg  
Asia/Yerevan  
Atlantic/Azores

## 포메팅 라이브러리 <c:timeZone>, <c:setTimeZone>

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.util.*" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
6 <c:set var="today" value="<%=new Date()%>" />
7
8 <!DOCTYPE html>
9 <html>
10 <head>
11   <meta charset="UTF-8">
12 </head>
13 <body>
14   <!-- 기본값은 현재 시스템 설정에 따름 -->
15   <fmt:formatDate value="${today}" type="both"/><br>
16
17   서울 :
18   <fmt:timeZone value="Asia/Seoul">
19     <fmt:formatDate value="${today}" type="both"/><br>
20   </fmt:timeZone>
21
22   뉴욕 :
23   <fmt:timeZone value="America/New_York">
24     <fmt:formatDate value="${today}" type="both"/><br>
25   </fmt:timeZone>
26
27 </body>
28 </html>
```

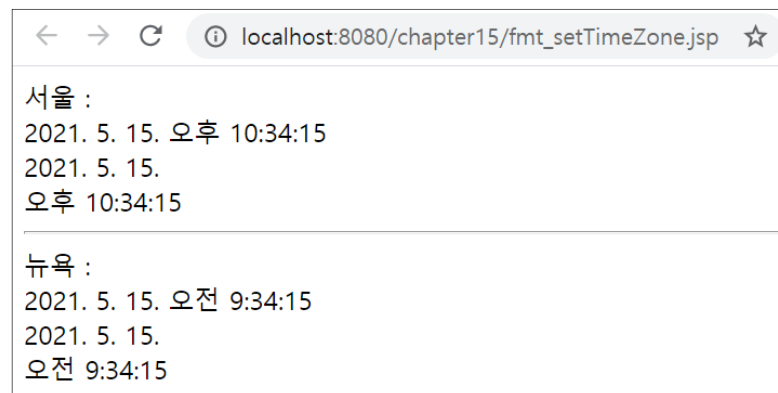


localhost:8080/chapter15/fmt\_timeZone.jsp ☆

2021. 5. 15. 오후 10:25:20  
서울 : 2021. 5. 15. 오후 10:25:20  
뉴욕 : 2021. 5. 15. 오전 9:25:20

## 포메팅 라이브러리 <c:timeZone>, <c:setTimeZone>

```
1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3 <%@ page import="java.util.*" %>
4 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
5 <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
6 <c:set var="today" value="<%=new Date()%>" />
7
8 <!DOCTYPE html>
9 <html>
10 <head>
11   <meta charset="UTF-8">
12 </head>
13 <body>
14
15   서울 : <br>
16   <fmt:setTimeZone value="Asia/Seoul" />
17   <fmt:formatDate value="${today}" type="both"/><br>
18   <fmt:formatDate value="${today}" type="date"/><br>
19   <fmt:formatDate value="${today}" type="time"/><br><hr>
20
21   뉴욕 : <br>
22   <fmt:setTimeZone value="America/New_York" />
23   <fmt:formatDate value="${today}" type="both"/><br>
24   <fmt:formatDate value="${today}" type="date"/><br>
25   <fmt:formatDate value="${today}" type="time"/><br>
26
27 </body>
28 </html>
```



## 포메팅 라이브러리 <c:setLocal>

### ■ <c:setLocal> 태그

- 국가 별로 로케일을 지정하기 위한 태그

```
<fmt:setLocal value="로케일" />
```

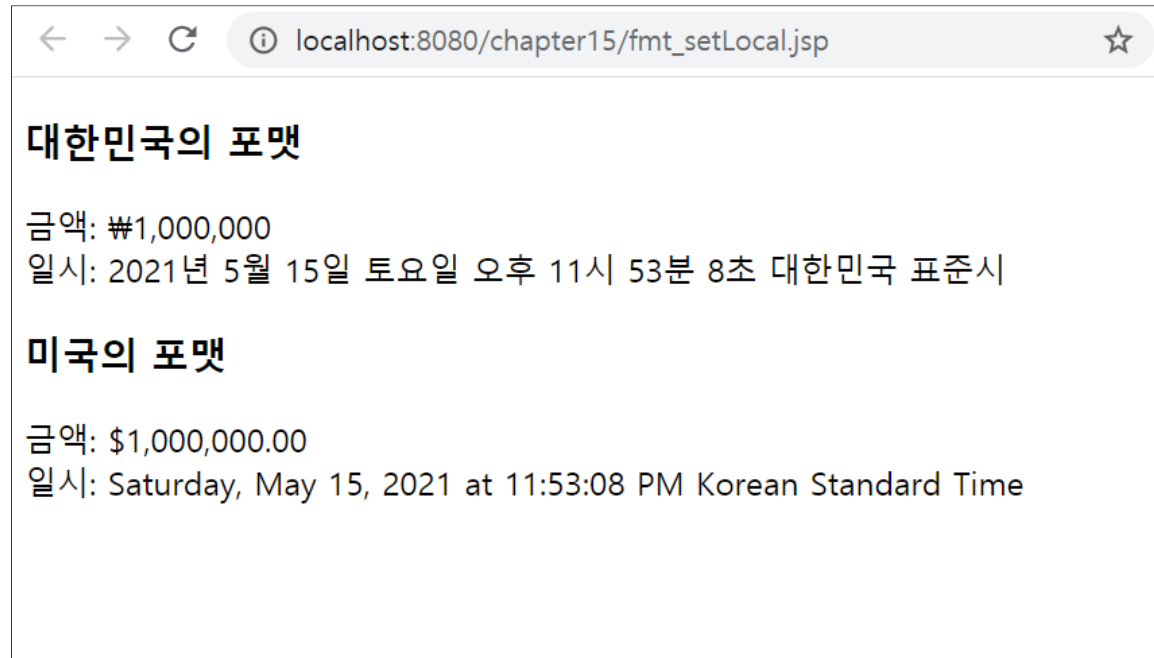
- 국가 별로 통화량 기호를 지정하고, 로케일로 지정한 언어로 날짜와 시간 정보를 출력 가능
  - ▶ 시간대 별로 시간 정보를 출력하는 것이 아니라 로케일로 지정한 언어로 현재 시스템의 시간을 표현
  - ▶ 다음 라인이 아니라 다음 로케일이 지정될때까지 이후에 존재하는 모든 코드에 영향을 미침



```

1 <%@ page language="java" contentType="text/html; charset=UTF-8"
2   pageEncoding="UTF-8"%>
3
4 <%@ page import="java.util.*" %>
5 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
6 <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>
7 <c:set var="myDate" value="<%=new Date()%>" />
8
9 <!DOCTYPE html>
10 <html>
11 <head>
12   <meta charset="UTF-8">
13 </head>
14 <body>
15
16   <H3>대한민국의 포맷</H3>
17     <fmt:setLocale value="ko_KR" />
18     금액: <fmt:formatNumber value="1000000" type="currency" /> <BR>
19     일시: <fmt:formatDate value="\${myDate}" type="both" dateStyle="full" timeStyle="full" />
20     <BR>
21
22   <H3>미국의 포맷</H3>
23     <fmt:setLocale value="en_US" />
24     금액: <fmt:formatNumber value="1000000" type="currency" /> <BR>
25     일시: <fmt:formatDate value="\${myDate}" type="both" dateStyle="full" timeStyle="full" />
26
27 </body>
28 </html>

```



# 함수 라이브러리

## ■ 함수 라이브러리

- 문자열을 처리하는데 주로 사용

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

- ▶ prefix는 주로 fn을 사용

# 함수 라이브러리

- 문자열 관련 메서드

메서드	반환값	내용
fn:contains(A,B)	boolean	문자열 A에 문자열 B가 포함되어 있는지 확인
fn:endsWith(A, B)	boolean	문자열 A의 끝이 B로 끝나는지 확인
fn:indexOf(A, B)	int	문자열 A에서 B가 처음으로 위치하는 인덱스(index)를 반환
fn:length(A)	int	문자열 A의 전체 길이를 반환
fn:replace(A, B, C)	String	문자열 A에서 B까지 해당되는 문자를 찾아 C로 변환
fn:toLowerCase(A)	String	A를 모두 소문자로 변환
fn:toUpperCase(A)	String	A를 모두 대문자로 변환
fn:substring(A, B, C)	String	A에서 인덱스 번호 B에서 C까지 해당하는 문자열을 반환
fn:split(A, B)	String[ ]	A에서 B에서 지정한 문자열로 나누어 배열로 반환
fn:trim(A)	String	문자열 A에서 앞뒤 공백을 제거

**수고하셨습니다**