JSP Taglib

JSP Tag Library

EL

Expression **L**anguage

스크립트릿이나
자바코드를 이용하여
레이아웃을 구성할 경우
변수를 입력하는 화면이 복잡해져서
코드의 가독성이 줄어들 수 있다.

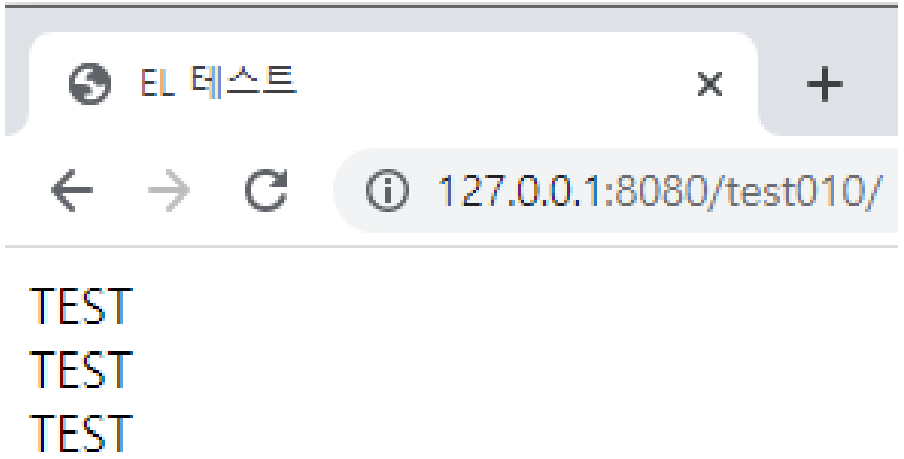
이 경우 EL을 이용하면
코드를 단순화 하여 생산성을 높일 수 있다.

${값}을 이용하여 사용 가능하다
자바스크립트에서의 `${var}`와 같은 기능이다.

EL값 확인

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ page import="java.net.*" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL 테스트</title>
        <link rel="stylesheet" href="test.css">
    </head>
    <body>
        <!-- EL(Expression Language) -->
        <!-- jsp를 표현하는 또 다른 방식 -->
        <%="TEST" %><br><!-- 표현식을 사용하는 방법 -->
        <%out.print("TEST"); %><br><!-- 스크립트릿을 사용하는 방법 -->
        ${"TEST" }<br><!-- EL을 사용하는 방법 -->
    </body>
</html>
```



값을 출력하는 방법은 여러가지가 존재한다.

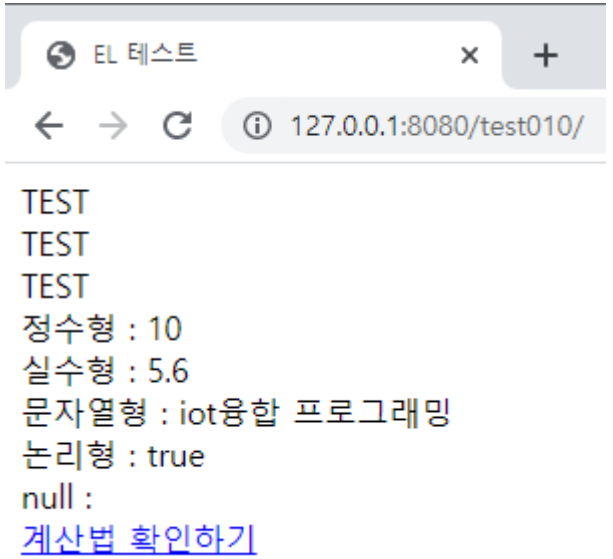
EL의 종류

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ page import="java.net.*" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL 테스트</title>
        <link rel="stylesheet" href="test.css">
    </head>
    <body>
        <!-- EL(Expression Language) -->
        <!-- jsp를 표현하는 또 다른 방식 -->
        <%="TEST" %><br><!-- 표현식을 사용하는 방법 -->
        <%out.print("TEST"); %><br><!-- 스크립트릿을 사용하는 방법 -->
        ${"TEST" }<br><!-- EL을 사용하는 방법 -->

        <!-- EL의 종류 -->
        정수형 : ${10 }<br>
        실수형 : ${5.6 }<br>
        문자열형 : ${"iot융합 프로그래밍" }<br>
        논리형 : ${true }<br>
        null : ${null }<br>
        <!-- null은 값이 표시되지 않는다. -->

        <a href="expr.jsp">계산법 확인하기</a>
    </body>
</html>
```

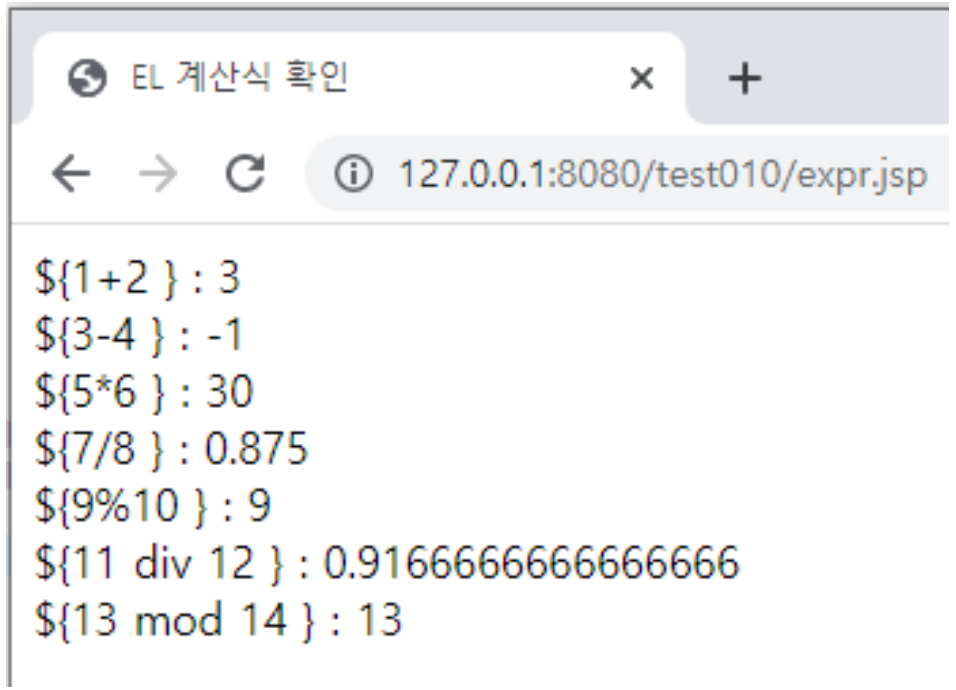


null은 값이 표기되지 않음을 확인

EL의 산술연산

expr.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL 계산식 확인</title>
        <link rel="stylesheet" href="test.css">
    </head>
    <body>
        <!-- 앞에 \를 붙이면 부호까지 표기된다. -->
        <!-- 산술연산 -->
        \${1+2 } : ${1+2 }<br>
        \${3-4 } : ${3-4 }<br>
        \${5*6 } : ${5*6 }<br>
        \${7/8 } : ${7/8 }<br>
        \${9%10 } : ${9%10 }<br>
        \${11 div 12 } : ${11 div 12 }<br><!-- 나누기와 동일 -->
        \${13 mod 14 } : ${13 mod 14 }<br><!-- 나머지와 동일 -->
    </body>
</html>
```



null은 값이 표기되지 않음을 확인

EL의 관계연산

```
<!-- 관계연산 -->
\${1==2 } : ${1==2 }<br>
\${1 eq 2 } : ${1 eq 2 }<br>
\${3!=4 } : ${3!=4 }<br>
\${3 ne 4 } : ${3 ne 4 }<br>
\${5<6 } : ${5<6 }<br>
\${5 lt 6 } : ${5 lt 6 }<br>
\${7>8 } : ${7>8 }<br>
\${7 gt 8 } : ${7 gt 8 }<br>
\${9<=10 } : ${9<=10 }<br>
\${9 le 10 } : ${9 le 10 }<br>
\${11>=12 } : ${11>=12 }<br>
\${11 ge 12 } : ${11 ge 12 }<br>
```

${1==2 } : false
${1 eq 2 } : false
${3!=4 } : true
${3 ne 4 } : true
${5<6 } : true
${5 lt 6 } : true
${7>8 } : false
${7 gt 8 } : false
${9<=10 } : true
${9 le 10 } : true
${11>=12 } : false
${11 ge 12 } : false

영문자로 된 기호표기가 가능함

EL의 논리연산

```
<!-- 논리연산 -->
\${true && false } : ${true && false }<br>
\${true || false } : ${true || false }<br>
\${!true} : ${!true }<br>

<!-- null 확인연산 -->
\${empty null } : ${empty null }<br>
```

${true && false } : false
${true || false } : true
${!true} : false
${empty null } : true

영문자로 된 기호표기가 가능함

EL의 특징

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL의 패러미터 확장</title>
    </head>
    <body>
        <div id="wrap">
            <form method="post" action="test.jsp">
                아이디 : <input type="text" name="user"><br>
                비밀번호 : <input type="password" name="pw"><br>
                <hr>
                좋아하는 계절<br>
                <input type="checkbox" name="season" value="spring">봄
                <input type="checkbox" name="season" value="summer">여름
                <input type="checkbox" name="season" value="fall">가을
                <input type="checkbox" name="season" value="winter">겨울
                <input type="submit" value="로그인!">
            </form>

            <form method="post" action="add.jsp">
                <input type="text" name="first">+
                <input type="text" name="second"><br>
                <input type="submit" value="계산하기!">
            </form>
        </div>
    </body>
</html>
```

test.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL의 패러미터 확인</title>
    </head>
    <body>
        <div id="wrap">
            <%
                request.setCharacterEncoding("utf-8");
                //EL역시도 request 정보를 따라 움직이므로 한글깨짐을 수정하려면
                //다음의 내용이 표기되어 있어야 한다.
            %>
            <h1>${param }</h1>
            <h1>아이디 : ${param.user }</h1>
            <h1>비밀번호 : ${param["pw"] }</h1>
            <!-- 여러개의 값이 오는 경우 처음에 찍힌 값만이 보이는 것을 확인 -->
            <h1>선호하는 계절 : ${param["season"] }</h1>

            <%
                String[] arr=request.getParameterValues("season");
            %>
            <%
                for(int i=0; i<arr.length; i++){
            %>
                    <h1><%=arr[i]%></h1>
            <%
                }
            %>
            ${paramValues.season} <br>
            ${paramValues.season[0]}<br>
            ${paramValues.season[1]} <br>

            기존 방식 : <%=request.getParameter("id")%><br>
            EL 방식 : ${param.id }<br>

            <!-- 자바식은 리터럴로 인하여 String을 별개의 객체로 본다-->
            자바식 방식 : <%=request.getParameter("user")=="admin"%><br>
            <!-- EL은 자바의 연산이 끝난 이후의 값을 기준으로 하므로 같은 리터럴에 해당한다. -->
            EL 방식 : ${param.user=="admin" }<br>
        </div>

    </body>
</html>
```

EL을 통하여 request값을 받을 수 있다.

EL의 계산

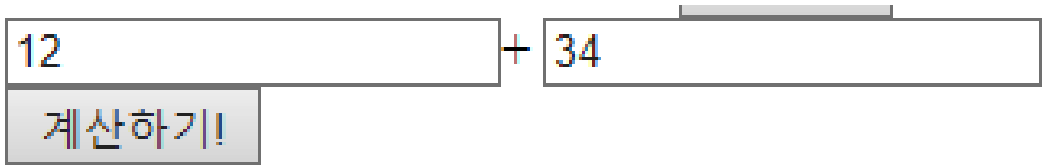
add.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL의 패러미터 특징</title>
    </head>
    <body>
        <div id="wrap">
            <%
                request.setCharacterEncoding("utf-8");
                String first=request.getParameter("first");
                String second=request.getParameter("second");

                String result=first+second;
            %>
            <h1>더한 값 : <%=result %></h1>

            <%
                int fNum=Integer.parseInt(first);
                int sNum=Integer.parseInt(second);
                int rNum=fNum+sNum;
            %>
            <h1>parseInt : <%=rNum %></h1>

            <h1>EL의 방식 : ${param.first+param.second}</h1>
            <h1>기존대로의 처리 : ${param.first}${param.second}</h1>
        </div>
    </body>
</html>
```



더한 값 : **1234**

parseInt : 46

EL의 방식 : 46

기존대로의 처리 : **1234**

int와 String의 계산을 임의로 처리함에 주의

EL의 scope

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>EL의 스코프범위</title>
    </head>
    <body>
        <%
            //scope범위
            //page : 현재 페이지에서의 스코프
            //request : 리퀘스트 단위에서의 스코프
            //session : 세션 단위에서의 스코프
            //application : 어플리케이션 단위에서의 스코프

            pageContext.setAttribute("name", "page scope");
            request.setAttribute("name", "request scope");
            session.setAttribute("name", "session scope");
            application.setAttribute("name", "application scope");
        %>
        <!-- 아무것도 지정하지 않았다면 page scope를 기준으로 데이터를 가져온다. -->
        <h1>기본형 : ${name }</h1>
        <!-- 명시적으로 scope의 범위를 지정하여 사용이 가능하다. -->
        <h1>\${pageScope.name } : ${pageScope.name }</h1>
        <h1>\${requestScope.name } : ${requestScope.name }</h1>
        <h1>\${sessionScope.name } : ${sessionScope.name }</h1>
        <h1>\${applicationScope.name } : ${applicationScope.name }</h1>
    </body>
</html>
```

기본형 : page scope

${pageScope.name } : page scope

${requestScope.name } : request scope

${sessionScope.name } : session scope

${applicationScope.name } : application scope

개발 시 객체와 변수의 scope단위에 주의

scope 처리

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>scope</title>
    </head>
    <body>
        <%
            pageContext.setAttribute("page", "page단위의 데이터");
            session.setAttribute("ses", "session단위의 데이터");
            application.setAttribute("app", "application단위의 데이터");
        %>
        <h1>\${page } : ${page }</h1>
        <h1>\${req } : ${req }</h1>
        <h1>\${ses } : ${ses }</h1>
        <h1>\${app } : ${app }</h1>

        <a href="index.jsp">페이지 단위의 scope</a><br>
        <a href="#" onclick="sub()">리퀘스트 단위의 scope</a><br>
        <a href="session.jsp">세션 단위의 scope</a><br>
        <a href="application.jsp">어플리케이션 단위의 scope</a><br>

        <form id="req" method="get" action="request.jsp">
            <input type="hidden" name="req" value="request단위의 데이터">
        </form>
        <script>
            let req=document.getElementById("req");
            function sub(){
                req.submit();
            }
        </script>
    </body>
</html>
```

${page } : page단위의 데이터

${req } :

${ses } : session단위의 데이터

${app } : application단위의 데이터

페이지 단위의 scope
리퀘스트 단위의 scope
세션 단위의 scope
어플리케이션 단위의 scope

reques로 받은 데이터가 없으므로 null

scope 처리

request.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>리퀘스트 단위 테스트</title>
    </head>
    <body>
        <h1>request로 전송되어온 데이터의 존속 확인</h1>
        <h1>\${page } : ${page }</h1>
        <h1>\${param.req } : ${param.req }</h1>
        <h1>\${ses } : ${ses }</h1>
        <h1>\${app } : ${app }</h1>

        <a href="index.jsp">페이지 단위의 scope</a><br>
        <a href="request.jsp">리퀘스트 단위의 scope</a><br>
        <a href="session.jsp">세션 단위의 scope</a><br>
        <a href="application.jsp">어플리케이션 단위의 scope</a><br>
    </body>
</html>
```

request로 전송되어온 데이터의 존속 확인

${page } :

${param.req } : request단위의 데이터

${ses } : session단위의 데이터

${app } : application단위의 데이터

페이지 단위의 scope
리퀘스트 단위의 scope
세션 단위의 scope
어플리케이션 단위의 scope

page단위는 연결되지 않음을 확인

scope 처리

session.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>세션 단위 테스트</title>
    </head>
    <body>
        <h1>세션정보 존속 중</h1>
        <h1>\${page } : ${page }</h1>
        <h1>\${param.req } : ${param.req }</h1>
        <h1>\${ses } : ${ses }</h1>
        <h1>\${app } : ${app }</h1>
        <%
            session.invalidate();
        %>
        <h1>세션정보 만료 후</h1>
        <h1>\${ses } : ${ses }</h1>

        <a href="index.jsp">페이지 단위의 scope</a><br>
        <a href="request.jsp">리퀘스트 단위의 scope</a><br>
        <a href="session.jsp">세션 단위의 scope</a><br>
        <a href="application.jsp">어플리케이션 단위의 scope</a><br>
    </body>
</html>
```

세션정보 존속 중

${page } :

${param.req } :

${ses } : session단위의 데이터

${app } : application단위의 데이터

세션정보 만료 후

${ses } :

페이지 단위의 scope
리퀘스트 단위의 scope
세션 단위의 scope
어플리케이션 단위의 scope

세션 만료시까지 데이터가 존속함을 확인

scope 처리

application.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>어플리케이션 단위 테스트</title>
    </head>
    <body>
        <h1>application 단위에서는 항상 존속 확인</h1>
        <h1>\${page } : ${page }</h1>
        <h1>\${param.req } : ${param.req }</h1>
        <h1>\${ses } : ${ses }</h1>
        <h1>\${app } : ${app }</h1>
        <%
            application.removeAttribute("app");
        %>
        <h1>application 단위 데이터 삭제</h1>
        <h1>\${app } : ${app }</h1>

        <a href="page.jsp">페이지 단위의 scope</a><br>
        <a href="request.jsp">리퀘스트 단위의 scope</a><br>
        <a href="session.jsp">세션 단위의 scope</a><br>
        <a href="application.jsp">어플리케이션 단위의 scope</a><br>
    </body>
</html>
```

application 단위에서는 항상 존속 확인

${page } :

${param.req } :

${ses } :

${app } : application단위의 데이터

application 단위 데이터 삭제

${app } :

페이지 단위의 scope
리퀘스트 단위의 scope
세션 단위의 scope
어플리케이션 단위의 scope

application은 모든상황, 모든 사용자에게 연동되는 데이터이므로
사용에 주의하여야 한다.

JSTL

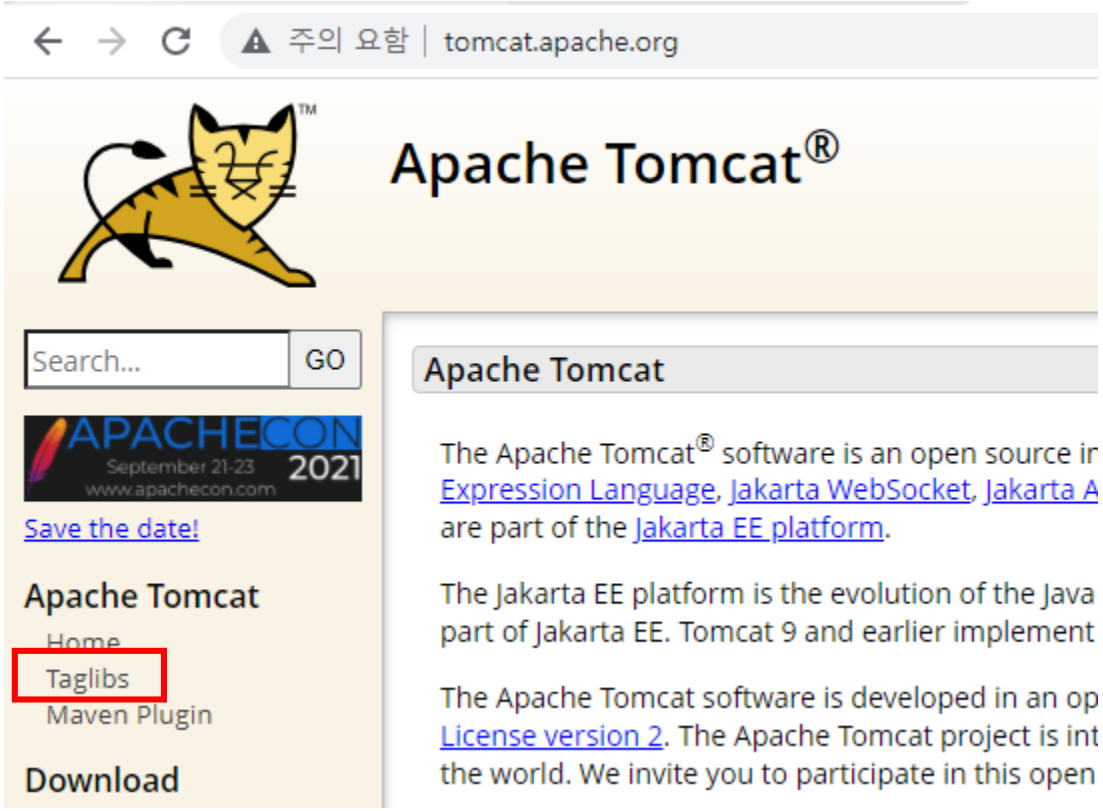
Jsp **S**tandard **T**ag **L**ibrary

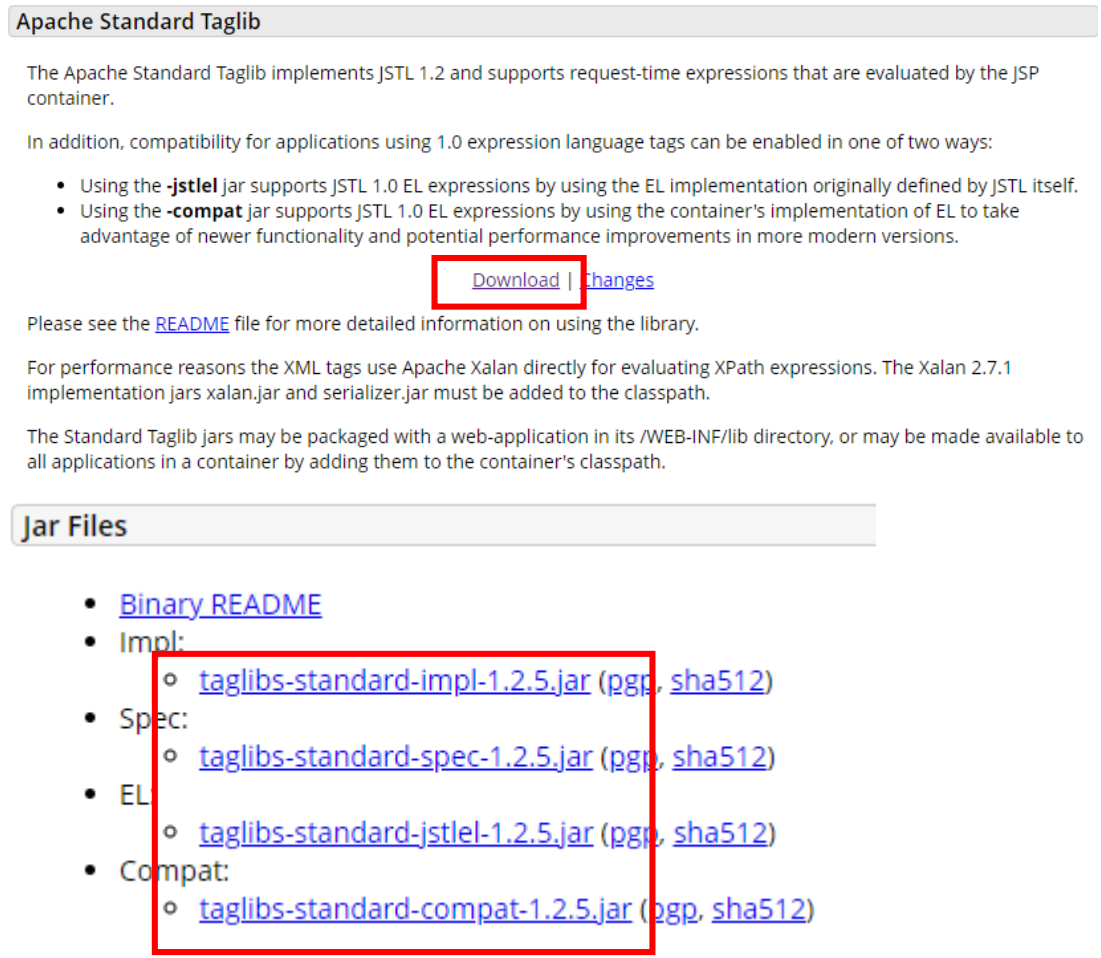
jsp에서 지원하는 기본 액션tag
만으로는
원하는 기능을 구현하기 쉽지 않다.

JSTL을 이용하면
코드의 가독성을 높이고
협업을 편리하게 할 수 있다.

**Tag의 형태로 사용 가능한
라이브러리이다.**

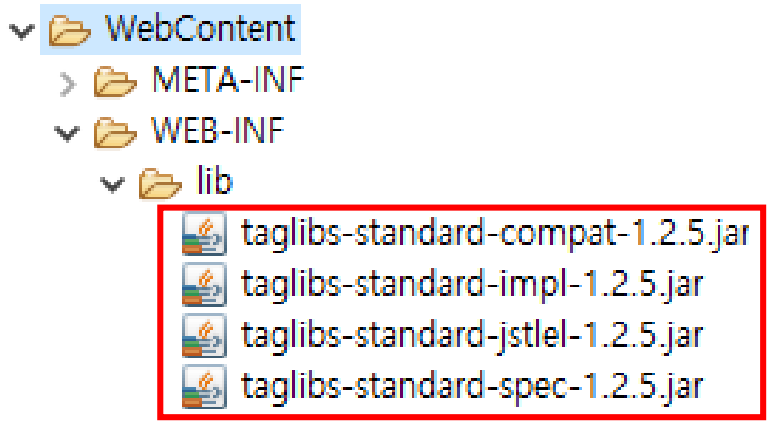
jstl 다운로드





다운로드 받아 필요한 프로젝트에 넣어 준다.

jstl의 사용



jstl출력
EL출력
scriptlet출력
java출력
java로 저장한 값
jstl로 저장한 값
30

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl의 활용</title>
    </head>
    <body>
        <!-- jsp standard tag library -->
        <!-- jstl을 사용하기 위해서는 tag라이브러리를 불러와야 한다. -->
        <!-- taglib은 특정 태그를 사용할 수 있도록 해 주는 기능을 의미한다.  -->
        <!-- uri : uniform resource identifier, 고유식별자 인터넷에 존재하는 각 자료들의 id 고유값 -->
        <!-- url : uniform resource locator, 해당 자료의 위치를 표기하는 방법으로 사이트의 도메인처럼 위치를 의미 -->
        <!-- urn : uniform resource name, 해당 자료의 이름을 표기하는 방법 -->
        <!-- 기본기능 :: c :: http://java.sun.com/jsp/jstl/core -->
        <!-- 형식변경 :: fmt :: http://java.sun.com/jstl/fmt -->
        <!-- DB조작 :: sql :: http://java.sun.com/jstl/sql -->
        <!-- XML조작 :: x :: http://java.sun.com/jstl/xml -->
        <!-- 함수처리 :: fn :: http://java.sun.com/jsp/jstl/fn -->

        <!-- c:out 값을 출력한다. -->
        <c:out value="jstl출력"></c:out><br>
        ${"EL출력" }<br>
        <%="scriptlet출력" %><br>
        <%out.print("java출력"); %><br>

        <!-- c:set 속성에 값을 저장한다 -->
        <%pageContext.setAttribute("test", "java로 저장한 값"); %>
        <c:set var="test2" value="jstl로 저장한 값" scope="page"></c:set>
        ${test }<br>
        ${test2 }<br>
        <c:set var="number">
            30
        </c:set>
        ${number }
    </body>
</html>
```

개발 시 객체와 변수의 scope단위에 주의

el과 jstl

test.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
    <html>
    <head>
        <meta charset="utf-8">
        <title>자바빈과 jstl</title>
    </head>
    <body>
        <%
            String var="자바단위";
            pageContext.setAttribute("page", "페이지 단위");
            request.setAttribute("request", "리퀘스트 단위");
            session.setAttribute("session", "세션 단위");
            application.setAttribute("app", "어플리케이션 단위");
            //스코프 범위로 넘어가지 않으면 출력되지 않음을 확인
        %>
        <h1>EL의 출력</h1>
        \${var } : ${var }<br>
        \${page } : ${page }<br>
        \${request } : ${request }<br>
        \${session } : ${session }<br>
        \${app } : ${app }<br>

        <c:set var="page" value="페이지 단위" scope="page"></c:set>
        <c:set var="request" value="리퀘스트 단위" scope="request"></c:set>
        <c:set var="session" value="세션 단위" scope="session">
        </c:set><c:set var="app" value="어플리케이션 단위" scope="application"></c:set>

        <h1>jstl의 출력</h1>
        <c:out value="${page }"></c:out><br>
        <c:out value="${request }"></c:out><br>
        <c:out value="${session }"></c:out><br>
        <c:out value="${app }"></c:out><br>
    </body>
</html>
```

EL의 출력

${var } :
${page } : 페이지 단위
${request } : 리퀘스트 단위
${session } : 세션 단위
${app } : 어플리케이션 단위

jstl의 출력

페이지 단위
리퀘스트 단위
세션 단위
어플리케이션 단위

표현기법이 조금 상이함에 주의할 것

자바빈과 jstl

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
    <html>
    <head>
        <meta charset="utf-8">
        <title>자바빈과 jstl</title>
    </head>
    <body>
        <h1>자바방식</h1>
        <%
            beans.Student s1=new beans.Student();
            s1.setName("이진선");
            s1.setAge("19");
            s1.setAddress("서울시 강북구");
        %>
        <%=s1 %>
        <h1>jsp액션태그방식</h1>
        <jsp:useBean id="s2" class="beans.Student"></jsp:useBean>
        <jsp:setProperty name="s2" property="name" value="고희선"></jsp:setProperty>
        <jsp:setProperty name="s2" property="age" value="25"></jsp:setProperty>
        <jsp:setProperty name="s2" property="address" value="서울시 중구"></jsp:setProperty>
        <%=s2 %>
        <h1>jstl방식</h1>
        <c:set var="s3" value="<%=new beans.Student() %>"></c:set>
        <c:set target="${s3 }" property="name" value="이영준"></c:set>
        <c:set target="${s3 }" property="age" value="31"></c:set>
        <c:set target="${s3 }" property="address" value="서울시 성북구"></c:set>
        ${s3 }
    </body>
</html>
```

자바방식

<<정보출력>> 이름:이진선,나이:19,주소:서울시 강북구

jsp액션태그방식

<<정보출력>> 이름:고희선,나이:25,주소:서울시 중구

jstl방식

<<정보출력>> 이름:이영준,나이:31,주소:서울시 성북구

자바빈을 jstl과 결합하면 다소 복잡해질 수 있다.

jstl:delete

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
    <html>
    <head>
        <meta charset="utf-8">
        <title>jstl의 활용</title>
    </head>
    <body>
        <c:set var="add" value="${3+5 }"></c:set>
        value="add" : <c:out value="add"></c:out><br>
        value="\${add}" : <c:out value="${add}"></c:out><br>
        <c:remove var="add"></c:remove>
        삭제후 : <c:out value="${add}"></c:out><br>
        <!-- c:set 은 setAttribute의 기능을 수행-->
        <!-- c:remove 은 removeAttribute의 기능을 수행-->
    </body>
</html>
```

value="add" : add
value="${add}" : 8
삭제후 :

자바빈을 jstl과 결합하면 다소 복잡해질 수 있다.

jstl:if

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl if</title>
    </head>
    <body>
        <div id="wrap">
            <form method="post" action="select.jsp">
                색상을 선택하세요
                <select name="color">
                    <option value="1">빨강</option>
                    <option value="2">주황</option>
                    <option value="3">노랑</option>
                    <option value="4">초록</option>
                    <option value="5">파랑</option>
                    <option value="6">남색</option>
                    <option value="7">보라</option>
                </select>
                <input type="submit" value="전송">
            </form>
        </div>
    </body>
</html>
```

select.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>색상 출력</title>
    </head>
    <body>
        <h1>일반 자바방식</h1>
        <%
            request.setCharacterEncoding("utf-8");
            String color=request.getParameter("color");
            int num=Integer.parseInt(color);

            if(num==1){
        %>
                <span style="color:red;">빨강</span>
        <%
            }
        %>


        <h1>jstl을 사용한 경우</h1>
        <c:if test="${param.color==1 }">
            <span style="color:red;">빨강</span>
        </c:if>
    </body>
</html>
```

색상을 선택하세요 빨강 주황 노랑 초록 파랑 남색 보라 전송

일반 자바방식

빨강

jstl을 사용한 경우

빨강

조건문이나 함수를 수행하는데 이용 가능하다.

jstl:if else

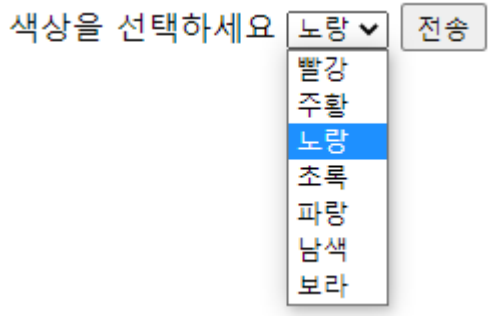
select.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>색상 출력</title>
    </head>
    <body>
        <h1>일반 자바방식</h1>
        <%
            request.setCharacterEncoding("utf-8");
            String color=request.getParameter("color");
            int num=Integer.parseInt(color);

            if(num==1){
        %>
                <span style="color:red;">빨강</span>
        <%
            }else{
        %>
                <span style="color:orange;">주황</span>
        <%
            }
        %>

        <h1>jstl을 사용한 경우</h1>
        <c:if test="${param.color==1 }">
            <span style="color:red;">빨강</span>
        </c:if>

        <h1>c:choose 방식</h1>
        <c:choose>
            <c:when test="${param.color==1 }">
                <span style="color:red;">빨강</span>
            </c:when>
            <c:otherwise>
                <span style="color:orange;">주황</span>
            </c:otherwise>
        </c:choose>
    </body>
</html>
```



일반 자바방식

주황

jstl을 사용한 경우

c:choose 방식

주황

HTML과 유사한 형태로 표현이 가능하다.

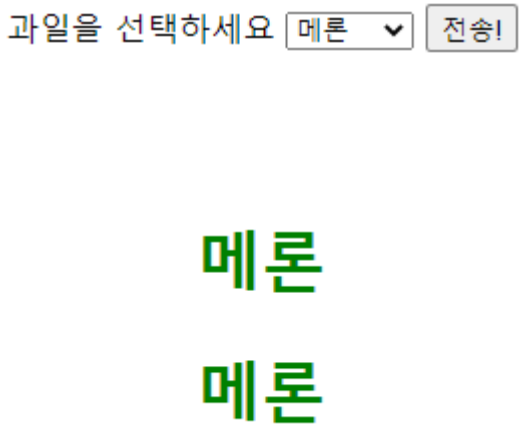
jstl:if else if

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>JSTL if-else if</title>
    </head>
    <body>
        <form method="post" action="select.jsp">
            과일을 선택하세요
            <select name="fruit">
                <option value="1">사과</option>
                <option value="2">메론</option>
                <option value="3">바나나</option>
            </select>
            <input type="submit" value="전송!">
        </form>
    </body>
</html>
```

select.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>다중 조건문</title>
    </head>
    <body>
        <%
            String str=request.getParameter("fruit");
            System.out.println(str);
            if(str.equals("1")){
                out.print("<h1 style='color:red;'>사과</h1>");
            }else if(str.equals("2")){
                out.print("<h1 style='color:green;'>메론</h1>");
            }else if(str.equals("3")){
                out.print("<h1 style='color:yellow;'>바나나</h1>");
            }
        %>
        <c:choose>
            <c:when test="${param.fruit==1 }">
                <h1 style="color:red;">사과</h1>
            </c:when>
            <c:when test="${param.fruit==2 }">
                <h1 style="color:green;">메론</h1>
            </c:when>
            <c:when test="${param.fruit==3 }">
                <h1 style="color:yellow;">바나나</h1>
            </c:when>
        </c:choose>
    </body>
</html>
```



기존의 방식과 비교해보면 큰 차이가 없음을 알 수 있다.

jstl:forEach

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl 반복문</title>
    </head>
    <body>
        <%
            String[] season={"봄","여름","가을","겨울"};

            //for문방식
            for(int i=0; i<season.length; i++){
                out.print("<h3 style='color:red;'>"+season[i]+"</h3>");
            }

            //향상된 for문방식
            for(String str:season){
                out.print("<h3 style='color:blue;'>"+str+"</h3>");
            }
            pageContext.setAttribute("array", season);
        %>

        <!-- jstl forEach방식 -->
        <c:forEach var="str" items="${array }">
            <h3 style='color:green;'>${str }</h3>
        </c:forEach>
    </body>
</html>
```

봄

여름

가을

겨울

봄

여름

가을

겨울

봄

여름

가을

겨울

java에서의 배열for문을 사용하는 방식

jstl:forEach

```
<!-- jstl forEach방식 -->
<!-- varStatus 속성을 이용하면 일반 for문처럼 값을 확인할 수 있다. -->
<c:forEach var="str" items="${array }" varStatus="iter">
    <h3 style='color:green;'>${str }:${iter.index }:${iter.count }:${iter.first }:${iter.last }</h3>
</c:forEach>
```

봄:0:1:true:false

여름:1:2:false:false

가을:2:3:false:false

겨울:3:4:false:true

status 패러미터를 사용할 수 있다.

jstl:forEach if혼합

```
<!-- jstl forEach방식 -->
<!-- varStatus 속성을 이용하면 일반 for문처럼 값을 확인할 수 있다. -->
<c:forEach var="str" items="${array }" varStatus="iter">
    <c:if test="${iter.first }"><div style="border:10px solid red;"></c:if>
    <h3 style='color:green;'>${str }:${iter.index }:${iter.count }:${iter.first }:${iter.last }</h3>
    <c:if test="${iter.last }"></div></c:if>
</c:forEach>
```

봄:0:1:true:false

여름:1:2:false:false

가을:2:3:false:false

겨울:3:4:false:true

간결한 형태의 제어문을 태그단위에서 사용가능

jstl:for

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl 반복문</title>
    </head>
    <body>
        <%
            //for문방식
            for(int i=0; i<=10; i++){
                out.print("<h3 style='color:red;'>"+i+"</h3>");
            }
        %>

        <!-- jstl for방식 -->
        <c:forEach var="i" begin="0" end="10" varStatus="status">
            <h3 style='color:blue;'>${i }</h3>
        </c:forEach>
    </body>
</html>
```

0 0
1 1
2 2
3 3
4 4
5 5
6 6
7 7
8 8
9 9
10 10

forEach가 아닌 for를 사용할 수도 있다.

jstl:for 반복조절

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl 반복문</title>
    </head>
    <body>
        <%
            //for문방식
            for(int i=0; i<=10; i+=2){
                out.print("<h3 style='color:red;'>"+i+"</h3>");
            }
        %>

        <!-- jstl for방식 -->
        <c:forEach var="i" begin="0" end="10" step="2" varStatus="status">
            <h3 style='color:blue;'>${i }:${status.index }:${status.count }</h3>
        </c:forEach>
    </body>
</html>
```

0

2

4

6

8

10

0:0:1

2:2:2

4:4:3

6:6:4

8:8:5

10:10:6

for와 같은 횟수조작이 가능하다.

jstl:forTokens

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl 토큰처리</title>
    </head>
    <body>
        <%
            String str="봄,여름,가을,겨울";
            pageContext.setAttribute("season", str);
        %>

        <!-- 토큰을 통하여 해당 데이터를 분리해 낼 수 있다. -->
        <c:forTokens var="data" items="${season }" delims=",">
            ${data }<br>
        </c:forTokens>
    </body>
</html>
```

봄
여름
가을
겨울

특정 문자를 잘라낼 수있다.

jstl:forTokens

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl 토큰처리</title>
    </head>
    <body>
        <%
            String str="봄,여름,가을,겨울";
            pageContext.setAttribute("season", str);

            String msg="iot,융합.프로그래밍?학과";
            pageContext.setAttribute("iot", msg);
        %>

        <!-- 토큰을 통하여 해당 데이터를 분리해 낼 수 있다. -->
        <c:forTokens var="data" items="${season }" delims=",">
            ${data }<br>
        </c:forTokens>

        <c:forTokens var="data" items="${iot }" delims=",.?">
            ${data }<br>
        </c:forTokens>
    </body>
</html>
```

봄
여름
가을
겨울
iot
융합
프로그래밍
학과

구분해야 하는 문자가 다양한 경우 여러 개 사용가능

jstl:import

```jsp
<!-- 다른 페이지를 import해오는 방법 -->
<!-- 변수처럼 저장해서 재사용이 가능하다. -->
<c:import url="test.jsp" var="test" scope="application" charEncoding="utf-8"></c:import>
${test }

<a href="app.jsp">다른 페이지로 이동</a>
```

test.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<h1>이 내용은 test.jsp에 적힌 내용입니다.</h1>
```

이 내용은 test.jsp에 적힌 내용입니다.

다른 페이지로 이동

app.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>scope에 저장된 내용을 확인</title>
    </head>
    <body>
        ${applicationScope.test }
    </body>
</html>
```

이 내용은 test.jsp에 적힌 내용입니다.

scope 단위에 페이지를 저장할 수 있어 편리

jstl:fmt

index.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ page import="java.util.*" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl 포멧사용</title>
    </head>
    <body>
        <!-- 주의!! taglib부분의 jsp가 누락되면 fmt는 jsp를 기반으로 동작하지 않으므로 스크립트릿 및 EL을 사용할 수 없다. -->
        <c:set var="num" value="1000000" scope="page"></c:set>
        fmt:number : <fmt:formatNumber value="${num }" type="number"></fmt:formatNumber><br>
        fmt:number  구분기호 없음 : <fmt:formatNumber value="${num }" type="number" groupingUsed="false"></fmt:formatNumber><br>
        fmt:number  %표기 : <fmt:formatNumber value="0.5" type="percent"></fmt:formatNumber><br>
        fmt:number  단위표기 : <fmt:formatNumber value="5000" type="currency"></fmt:formatNumber><br>
        fmt:number  단위변경 : <fmt:formatNumber value="5000" type="currency" currencySymbol="$"></fmt:formatNumber><br>
        fmt:number  패턴표기 : <fmt:formatNumber value="1234.5678" pattern="000000,000.000000#"></fmt:formatNumber><br>
        <!-- 0.0# : 0은 자릿수 #은 빈공간인 경우 작성하지 않겠다는 의미 -->
        <!-- 0,00 : 두자리마다 ,로 구별을 하겠다는 의미, 소수 자릿수를 표기하지 않으면 소수는 나오지 않는다. -->
        <!-- 0,0.00# : 소수 셋째짜리까지 표기하며 매 숫자마다 ,를 적겠다는 패턴 -->
        <!-- 0,000.0# : 소수 둘째자리까지 표기하며 천단위 구분기호를 적겠다는 패턴 -->
        <!-- 000000,000.0000000 : 천단위 구분기호를 사용하며 9자리를 확보하고 빈 공간은 0으로 표기한다. 소수는 7자리까지 표기하며 빈 공간은 0으로 든다. -->

        <c:set var="date" value="<%=new java.util.Date() %>" scope="page"></c:set>
        \${date } : ${date }<br>
        fmt:formatDate 기본방식 : <fmt:formatDate value="${date }"></fmt:formatDate><br>
        fmt:formatDate time : <fmt:formatDate value="${date }" type="time"></fmt:formatDate><br>
        fmt:formatDate both : <fmt:formatDate value="${date }" type="both"></fmt:formatDate><br>
        fmt:formatDate 서식지정 : <fmt:formatDate value="${date }" pattern="yyyy년 MM월 dd일 hh시 mm분 ss초"></fmt:formatDate><br>
    </body>
</html>
```

fmt:number : 1,000,000
fmt:number 구분기호 없음 : 1000000
fmt:number %표기 : 50%
fmt:number 단위표기 : ₩5,000
fmt:number 단위변경 : $5,000
fmt:number 패턴표기 : 000,001,234.567800
${date } : Wed Jul 07 19:21:02 KST 2021
fmt:formatDate 기본방식 : 2021. 7. 7
fmt:formatDate time : 오후 7:21:02
fmt:formatDate both : 2021. 7. 7 오후 7:21:02
fmt:formatDate 서식지정 : 21년 07월 07일 07시 21분 02초

표현 서식을 지정할 수 있어 매우 편리하다.

jstl 기본언어설정

locale.jsp

```jsp
<%@ page language="java" contentType="text/html; charset=utf-8"
    pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
<%@ page import="java.util.*" %>
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>jstl Locale 설정</title>
    </head>
    <body>
        <c:set var="date" value="<%=new java.util.Date() %>" scope="page"></c:set>
        톰캣 서버의 기본 Locale 확인 : <%=response.getLocale() %><br>

        <hr>
        <fmt:setLocale value="ko_kr"></fmt:setLocale>
        로케일을 한국어로 설정 한 후 locale확인 : <%=response.getLocale() %><br>
        통화 : <fmt:formatNumber value="5000" type="currency"></fmt:formatNumber><br>
        날짜 : <fmt:formatDate value="${date }"></fmt:formatDate><br>

        <hr>
        <fmt:setLocale value="ja_jp"></fmt:setLocale>
        로케일을 일본어로 설정 한 후 locale확인 : <%=response.getLocale() %><br>
        통화 : <fmt:formatNumber value="5000" type="currency"></fmt:formatNumber><br>
        날짜 : <fmt:formatDate value="${date }"></fmt:formatDate><br>

        <hr>
        <fmt:setLocale value="en_us"></fmt:setLocale>
        로케일을 영어로 설정 한 후 locale확인 : <%=response.getLocale() %><br>
        통화 : <fmt:formatNumber value="5000" type="currency"></fmt:formatNumber><br>
        날짜 : <fmt:formatDate value="${date }"></fmt:formatDate><br>
    </body>
</html>
```

톰캣 서버의 기본 Locale 확인 : ko_KR

로케일을 한국어로 설정 한 후 locale확인 : ko_KR
통화 : ₩5,000
날짜 : 2021. 7. 7

로케일을 일본어로 설정 한 후 locale확인 : ja_JP
통화 : ￥5,000
날짜 : 2021/07/07

로케일을 영어로 설정 한 후 locale확인 : en_US
통화 : $5,000.00
날짜 : Jul 7, 2021

표현 서식을 지정할 수 있어 매우 편리하다.