

2025년 상반기 K-디지털 트레이닝

EL, JSTL

[KB] IT's Your Life



EL 개요

- EL은 데이터를 출력하기 위한 언어
- 문법이 직관적, 사용이 용이
- JSP에서 변수를 출력할 때 사용
- 처리 가능한 데이터형
 - 프리미티브
 - Map, List, 배열, 자바빈 등

☑ EL 내장 객체

내장 객체	설명
pageScope	page 영역에 존재하는 변수 참조 시 사용
requestScope	request 영역에 존재하는 변수 참조 시 사용
sessionScope	session 영역에 존재하는 변수 참조 시 사용
applicationScope	application 영역에 존재하는 변수 참조 시 사용
param	파라미터 값을 참조 시 사용
paramValues	파라미터 배열 값을 참조 시 사용
header	헤더 정보 값을 참조 시 사용
headerValues	헤더 배열 정보 값을 참조 시 사용
cookie	쿠키 정보를 참조 시 사용
initParam	context 초기화 파라미터 참조 시 사용
pageContext	pageContext 참조 시 사용

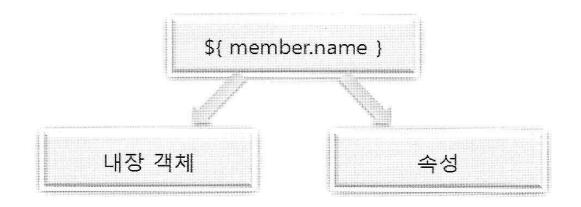
☑ EL 연산자

연산자	설명
	자바빈 또는 Map에 접근할 때 사용
	배열 또는 List에 접근할 때 사용
()	우선 순위 연산자
empty	값이 null인지 판단하는 연산자로서 true 리턴
+, -, *, /, %	산술 연산자자 및 나머지 연산자
&&, , !	논리 연산자
==, >, >=, <, <=, !=	비교 연산자

☑ EL 기본 문법

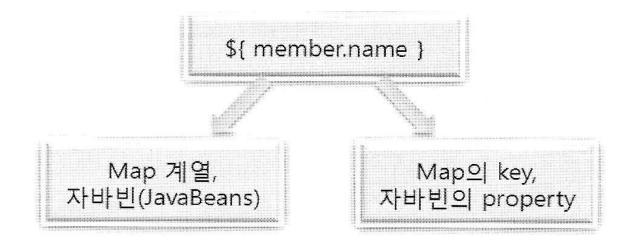
\${표현식}

○ 내장 객체이거나 scope에 저장된 속성을 지정

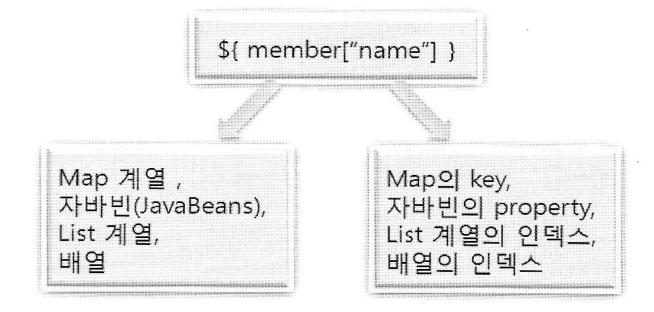


- <%= %>과의 차이
 - Scope의 속성 값이 아닌 JSP 변수 및 표현식을 출력

- Map 계열이거나 자바빈이 지정된 경우
 - 두 번째 값은 Map의 키 값이거나 자바빈의 프로퍼티



○ [] 배열 표기법



✓ login_form.jsp

```
<body>
<form action="login" method="get">
   <fieldset>
      <legend>로그인 폼</legend>
      <
             <label for="userid">아이디</label>
             <input type="text" id="userid" name="userid">
         <
             <label for="passwd">비밀번호</label>
            <input type="password" id="passwd" name="passwd">
         <
            <input type="submit" value="전송">
         </fieldset>
</form>
</body>
```

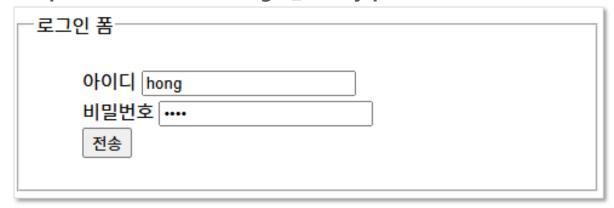
LoginServlet.java

```
@WebServlet("/login")
public class LoginServlet extends HttpServlet {
    protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
        String userid = req.getParameter("userid");
        String passwd = req.getParameter("passwd");
        req.setAttribute("userid", userid);
        req.setAttribute("userid", userid);
        req.setAttribute("passwd", passwd);
        req.getRequestDispatcher("login.jsp").forward(req, res);
    }
}
```

🗹 login.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
   <title>Title</title>
</head>
<body>
   <h1>EL 실습</h1>
   사용자 아이디: ${userid}<br>
   사용자 비밀번호: ${passwd}<br>
</body>
</html>
```

http://localhost:8080/login_form.jsp



http://localhost:8080/login?userid=hong&passwd=1234

EL 실습

사용자 아이디: hong 사용자 비밀번호: 1234

Scope

o page scope

- 한 페이지(servlet 또는 jsp 파일) 내에서만 가능
- 일종의 지역 변수

o request scope

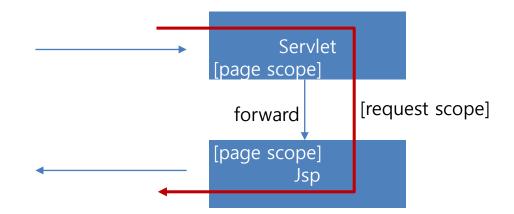
- request 객체에 저장되는 속성
- 접속이 끊길 때까지 유지

session scope

- session 객체에 저장되는 속성
- 세션이 만기되기 전 까지 유지

application scope

- application contex에 저장되는 속성
- 어플리케이션이 종료할 때까지 유



Page

JSP가 처리되는 동안 생성된 JSP에서 사용

Request

Client의 요청을 처리하는 동안. Forward,Include되는 자원들과 공유

Session

Session이 유지 되는 동안 여러 자원간의 공유. Client의 재 요구시 계속 사용 가능

Application

컨테이너가 작동하는 동안 동일한 Web Application에 속하는 모든 자원간의 공유

🗸 스코프 객체

스코프 명	EL(jsp)	Servlet
page scope	pageScope	지역변수
request scope	requestScope	HttpServletRequest
session scope	sessionScope	HttpSession
application scope	applicationScope	ServletContext

☑ EL에서 스코프 객체를 지정하지 않은 경우 속성 찾는 순서

o 예) \${username}

page → request → session → application

특정 스코프에서 속성이 발견되면 그 값을 출력

Member.java

```
package org.scoula.ex05.domain;
                                                                public void setName(String name) {
public class Member {
                                                                    this.name = name;
    private String name;
    private String userid;
                                                                public String getUserid() {
    public Member() {
                                                                    return userid;
    public Member(String name, String userid) {
                                                                public void setUserid(String userid) {
        this.name = name;
                                                                    this.userid = userid;
        this.userid = userid;
    public String getName() {
        return name;
```

ScopeServlet.java

```
@WebServlet("/scope")
public class ScopeServlet extends HttpServlet {
    ServletContext sc;
    @Override
    public void init(ServletConfig config) throws ServletException {
        sc = config.getServletContext();
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
        sc.setAttribute("scopeName", "applicationScope 값"); // Application Scope
        HttpSession session = req.getSession(); // Session Scope
        session.setAttribute("scopeName", "sessionScope 값");
        req.setAttribute("scopeName", "requestScope 값"); // Request Scope
        Member member = new Member("홍길동", "hong");
        req.setAttribute("member", member);
        req.getRequestDispatcher("scope.jsp").forward(req, res);
```

scope.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
   <title>Title</title>
</head>
<body>
   <h1>scope 데이터 보기 </h1>
   pageScope의 속성값은 : ${pageScope.scopeName}<br>
   requestScope의 속성값은 : ${requestScope.scopeName}<br>
   sessionScope의 속성값은 : ${sessionScope.scopeName}<br>
   applicationScope의 속성값은 : ${applicationScope.scopeName}<br>
   scopeName 자동 찾기: ${scopeName} <br>
   member: ${member.name}(${member.userid})<br>
</body>
</html>
</html>
```

http://localhost:8080/scope

scope 데이터 보기

pageScope의 속성값은 :

requestScope의 속성값은 : requestScope 값 sessionScope의 속성값은 : sessionScope 값

applicationScope의 속성값은 : applicationScope 값

scopeName 자동 찾기: requestScope 값

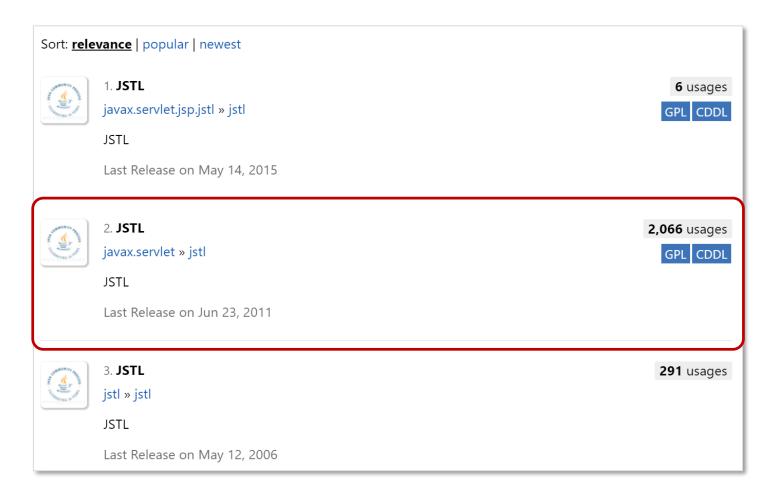
member: 홍길동(hong)

💟 JSTL 개요와 환경 설정

- 액션 태그를 사용자가 직접 제작 가능 → 커스텀 태그
- 커스텀 태그들 중에서 자주 사용되는 태그들을 묶어서 아파치 그룹에서 제공하는 것 → JSTL
- o EL과 JSTL을 함께 사용

Jstl 다운로드

mvnrepository 에서 jstl 검색



build.gradle

```
dependencies {
    compileOnly('javax.servlet:javax.servlet-api:4.0.1')
    implementation 'javax.servlet:jstl:1.2'
    testImplementation("org.junit.jupiter:junit-jupiter-api:${junitVersion}")
    testRuntimeOnly("org.junit.jupiter:junit-jupiter-engine:${junitVersion}")
```

→ 추가 후 Sync!!

taglibs

https://tomcat.apache.org/taglibs.html



Apache Tomcat[®]





G0



Apache Tomcat

Home Taglibs Maven Plugin

Download

Which version? Tomcat 11 (alpha)

Apache Taglibs

Apache Taglibs provides open source implementations of Tag Libraries for use with Java Server Pages (JSPs). In particular, it hosts the Apache Standard Taglib, an open source implementation of the Java Standard Tag Library (JSTL) specification.

Apache Standard Taglib

The Apache Standard Taglib implements JSTL 1.2 and supports request-time expressions that are evaluated by the JSP container.

In addition, compatibility for applications using 1.0 expression language tags can be enabled in one of two ways:

- Using the -jstlel jar supports JSTL 1.0 EL expressions by using the EL implementation originally defined by JSTL itself.
- Using the -compat jar supports JSTL 1.0 EL expressions by using the container's implementation of EL to take advantage of newer functionality and potential performance improvements in more modern versions.

Download | Changes

Please see the README file for more detailed information on using the library.

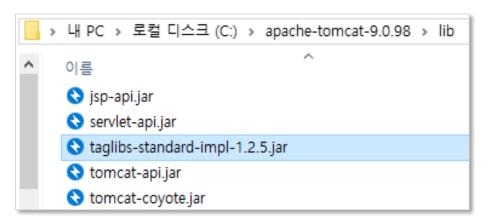
taglibs



- taglibs-standard-impl-1.2.5.jar 배치
 - C:₩apache-tomcat-9.0.89₩lib 모든 프로젝트에 적용

또는

- WEB-INFO/lib 이번 프로젝트에만 사용



🥑 JSTL 라이브러리

라이브러리	URI	Prefix	예제
Core	http://java.sun.com/jsp/jstl/core	С	<c:tag></c:tag>
I18N formatting	http://java.sun.com/jsp/jstl/fmt	fmt	<fmt:tag></fmt:tag>
Functions	http://java.sun.com/jsp/jstl/functions	fn	fn:function()
SQL	http://java.sun.com/jsp/jstl/sql	sql	<sql:tag></sql:tag>
XML processing	http://java.sun.com/jsp/jstl/xml	Х	<x:tag></x:tag>

taglib 지정자로 설정

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

JSTL Core 라이브러리

- 기본적이고 핵심적인 기능들을 구현해 놓은 라이브러리
- 문자열 출력, if문, for문과 같은 제어문 기능 포함

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

✓ JSTL Core 라이브러리에 포함된 태그 목록

태그	설명	사용예
out	지정된 값을 출력할 때 사용	<c:out></c:out>
set	JSP의 setAttribute(name, key) 기능과 동일. scope에 따른 바인딩 처리가 가능	<c:set></c:set>
remove	JSP의 removeAttribute(name) 기능과 동일. scope에 따른 속성 제거가 가능	<c:remove></c:remove>

☑ JSTL Core 라이브러리에 포함된 태그 목록

태그	설명	사용예
if	조건 처리를 사용할 때 사용한다.	<c:if></c:if>
forEach	반복 처리를 하고자 할 때 사용한다.	<c:foreach></c:foreach>
choose	자바의 switch문과 비슷하다.	<c:choose></c:choose>
when	choose의 서브 태그로 사용한다. 조건을 만족한 경우에 사용된다.	<c:when></c:when>
otherwise	choose의 서브 태그로 사용한다. 조건을 만족하지 못한 경우에 사용된다.	<c:otherwise></c:otherwise>
url	URL을 생성하는 기능이다.	<c:url></c:url>
forTokens	자바의 StringTokenizer 클래스 기능이다.	<c:fortokens></c:fortokens>

🗸 <c:set> 태그

- 지정된 scope에 데이터를 바인딩하는 태그
- JSP의 setAttribute(name, value) 메서드와 동일한 기능 제공

```
<c:set var="변수명" value="변수값"
    target="객체" property="객체의 프로퍼티"
    scope="scope값" />
```

🧿 <c:out> 태그

○ 지정된 값을 웹 브라우저에 출력하는 태그

```
<c:out value="출력값" default="기본값" escapeXml="true|false" />
```

🧿 <c:remote> 태그

○ 지정된 scope에 설정한 변수 값을 제거하는 태그

```
<c:remove var="변수명" scope="scope값" />
```


- 조건 처리를 할 때 사용하는 태그
- o 자바의 if문과 동일

```
<c:if test="조건식" var="변수명" scope="scope값">
     문장
</c:if>
```

- 조건식이 참인 경우 출력
- var가 지정되어 있다면 조건식 검사 결과를 변수에 저장
- else는 없음 → 부정표현으로 if 추가

- ♡ <c:choose>, <c:when>, <c:otherwise> 태그
 - 자바의 switch문과 동일한 기능을 제공하는 태그

○ 반복 처리시 사용하는 태그, for문과 비슷

```
<c:forEach var="변수명" items="객체명"
  begin="시작인덱스" end="끝인덱스" step="증가값"
  varStatus="other변수">
  문장
</c:forEach>
```

- items 속성: 배열이나 List 형태의 반복할 객체 지정
- var: 반복문에서 사용할 아이템 변수

for(PhoneInfo pi: list) 의 형태와 동일

■ varStatus : 반복문의 상태에 대한 참조 변수

🧿 <c:forEach> 태그

o varStatus 속성은 루프 정보를 담는 LoopTagStatus 객체를 이용

- index 루프 실행에서 현재 인덱스
- count 루프 실행 회수
- begin begin 속성의 값
- end end 속성의 값
- step step 속성의 값
- first 현재 실행이 첫 번째 실행인 경우 true
- last 현재 실행이 루프의 마지막 실행인 경우 true
- current 콜렉션 중 현재 루프에서 사용할 객체

o 현재 사용항목의 인덱스 값은 varStatus 속성을 이용

```
<c:forEach var="item" items="<%=someItemList%>"
vasrStatus="status">

${status.index +1}번째 항목: ${item.name}
</c:forEach>
```

JstlServlet.java

```
@WebServlet("/jstl_ex")
public class JstlServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest req, HttpServletResponse res) throws ServletException, IOException {
        List<Member> members = new ArrayList<>();
        for (int i = 0; i < 10; i++) {
            Member member = new Member("홍길동_" + i, "hong_" + i);
            members.add(member);
        req.setAttribute("members", members);
        req.setAttribute("role", "ADMIN");
        req.getRequestDispatcher("jstl ex.jsp")
           .forward(reg, res);
```

istl_ex.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<html>
<head>
   <title>Title</title>
</head>
<body>
   <h1>JSTL 테스트</h1>
   <c:if test="${role == 'ADMIN'}">관리자</c:if>
   <c:if test="${role != 'ADMIN'}">일반회원</c:if>
   <c:forEach var="member" items="${members}" varStatus="state">
          >
              ${state.index}
              ${member.name}
              ${member.userid}
          </c:forEach>
   <br>
</body>
</html>
```

http://localhost:8080/jstl_ex

JSTL 테스트

관리자 0 홍길동_0 hong_0 1 홍길동_1 hong_1 2 홍길동_2 hong_2 3 홍길동_3 hong_3 4 홍길동_4 hong_4 5 홍길동_5 hong_5 6 홍길동_6 hong_6 7 홍길동_7 hong_7 8 홍길동_8 hong_8 9 홍길동_9 hong_9

🦁 JSTL formatting 라이브러리

- 국제화/ 지역화 및 데이터 포맷과 관련된 기능 제공
- 국제화/지역화 → 다국어 처리
- 데이터 포맷 → 날짜와 숫자와 관련된 기능

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

JSTL formatting 라이브러리

태그	설명	사용예
requestEncoding	setCharacterEncoding(enc) 메서드와 동일한 기능	<fmt:requestencoding></fmt:requestencoding>
setLocale	다국어 페이지 사용 시 언어 지정	<fmt:setlocale></fmt:setlocale>
timeZone	지정한 지경 값으로 시간을 설정	<fmt:timezone></fmt:timezone>
setBundle	*.properties 확장자의 리소스 번들 파일 접근 시 사용	<fmt:setbundle></fmt:setbundle>
message	번들에서 설정한 값 사용	<fmt:message></fmt:message>
formatNumber	숫자형식의 포맷 지정	<fmt:formatnumber></fmt:formatnumber>
parseNumber	문자열을 숫자로 변환	<fmt:parsenumber></fmt:parsenumber>
formatDate	날짜형식의 포맷 지정	<fmt:formatdate></fmt:formatdate>
parseDdate	문자열을 날짜로 변환	<fmt:parsedate></fmt:parsedate>

😕 <fmt:formatNumber> 태그

수치 데이터를 특정 포맷으로 설정 시 사용되는 태그

```
<fmt:formatNumber value="값" typ="타입" pattern="패턴" var="값" scope="값"
    currencySymbole="값" minIntegerDigits="값" maxIntegerDigits="값"
    minFractionDigits="값" maxFractionDigits="값" />
```

- value : 실제 수치 값을 지정
- type: number, currency, percent 값 중에서 설정
- pattern : 사용자가 지정한 형식 패턴을 설정
- currentcySymbol : 통화 기호 지정
- maxIntegerDigits : 정수의 최대자리 수를 지정
- minIntegerDigits: 정수의 최소 자릿수를 지정
- maxFractionDigits : 소수점 이하 최대 자리수를 지정
- minFractionDigits: 소수점 이하 최소 자릿 수를 지정

✓ <fmt:formatDate> 택그

날짜 데이터를 특정 포맷으로 설정 시 사용

```
<fmt:formatDate value="값" typ="타입" var="값" scope="값"
dateStyle="날짜 스타일" timeStyle="시간스타일" />
```

- value : 실제 날짜와 시간을 설정
- type: time, date, both 중 하나
- dateStyle: 미리 정의된 날짜 스타일 형식
- timeStyle: 미리 정의된 시간 스타일 형식
- pattern: 사용자가 지정한 형식 스타일
 - "YYYY-MM-dd HH:mm:ss" HH: 24시간제
 - "YYYY-MM-dd a hh:mm:ss" a: 오전/오후, hh:12시간제,

JstlServlet.java

```
@WebServlet("/jstl_ex")
public class JstlServlet extends HttpServlet {
    protected void doGet(HttpServletRequest reg, HttpServletResponse res)
            throws ServletException, IOException {
        List<Member> members = new ArrayList<>();
        for (int i = 0; i < 10; i++) {
            Member member = new Member("홍길동_" + i, "hong_" + i);
            members.add(member);
        req.setAttribute("members", members);
        req.setAttribute("role", "ADMIN");
        req.setAttribute("today", new Date());
        req.getRequestDispatcher("jstl_ex.jsp")
           .forward(reg, res);
```

istl_ex.jsp

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>
<html>
<head>
    <title>Title</title>
                                                                                             7 홍길동_7 hong_7
</head>
                                                                                             8 홍길동_8 hong_8
<body>
                                                                                             9 홍길동_9 hong_9
    <h1>JSTL 테스트</h1>
                                                                                             Mon Jan 13 12:40:07 KST 2025
    <br>
                                                                                             2025, 1, 13,
                                                                                             오후 12:40:07
    ${today}<br>
    <fmt:formatDate value="${today}" type="date"/><br>
                                                                                             2025. 1. 13. 오후 12:40:07
    <fmt:formatDate value="${today}" type="time"/><br>
    <fmt:formatDate value="${today}" type="both"/><br>
                                                                                             2025년 1월 13일 오후 12:40
    <fmt:formatDate value="${today}" type="both" dateStyle="short" timeStyle="long"/><br>
                                                                                             2025-01-13 12:40:07
    <fmt:formatDate value="${today}" type="both" dateStyle="long" timeStyle="short"/><br>
                                                                                             2025-01-13 오후 12:40:07
    <fmt:formatDate value="${today}" pattern="YYYY-MM-dd HH:mm:ss"/><br>
    <fmt:formatDate value="${today}" pattern="YYYY-MM-dd a hh:mm:ss"/><br>
</body>
</html>
```

http://localhost:8080/jstl ex

```
25. 1. 13. 오후 12시 40분 7초 KST
```