표현 언어
(Expression Language)

표현 언어

- Expression Language
- JSP에서 사용가능한 새로운 스크립트 언어
- EL의 주요 기능
  - JSP의 네 가지 기본 객체가 제공하는 영역의 속성 사용
  - 집합 객체에 대한 접근 방법 제공
  - 수치 연산, 관계 연산, 논리 연산자 제공
  - 자바 클래스 메서드 호출 기능 제공
  - 표현언어만의 기본 객체 제공
- 간단한 구문 때문에 표현식 대신 사용

구문

- 기본 문법
  - ${expr}, #{expr}
  - 사용예
    - <jsp:include page="/module/${skin.id}/header.jsp" />
    - ${sessionScope.member.id}님 환영합니다.
  - ${expr}은 표현식이 실행되는 시점에 바로 값 계산
  - #{expr}은 값이 실제로 필요한 시점에 값 계산
    - JSP 템플릿 텍스트에서는 사용 불가

- 스크립트 요소(스크립트릿, 표현식, 선언부)를 제외한 나머지 부분에서 사용

EL에서 기본 객체

기본 객체	설 명
pageContext	JSP의 page 기본 객체와 동일하다.
pageScope	pageContext 기본 객체에 저장된 속성의 <속성, 값> 매핑을 저장한 Map 객체.
requestScope	request 기본 객체에 저장된 속성의 <속성, 값> 매핑을 저장한 Map 객체.
sessionScope	session 기본 객체에 저장된 속성의 <속성, 값> 매핑을 저장한 Map 객체.
applicationScope	application 기본 객체에 저장된 속성의 <속성, 값> 매핑을 저장한 Map 객체.
param	요청 파라미터의 <파라미터이름, 값> 매핑을 저장한 Map 객체.
paramValues	요청 파라미터의 <파라미터이름, 값배열> 매핑을 저장한 Map 객체.
header	요청 정보의 <헤더이름, 값> 매핑을 저장한 Map 객체.
headerValues	요청 정보의 <헤더이름, 값 배열> 매핑을 저장한 Map 객체.
cookie	<쿠키 이름, Cookie> 매핑을 저장한 Map 객체.
initParam	초기화 파라미터의 <이름, 값> 매핑을 저장한 Map 객체.

EL 데이터 타입

- 불리언(Boolean) 타입 - true 와 false
- 정수타입 - 0~9로 이루어진 정수 값.
- 실수타입 - 0~9로 이루어져 있으며, 소수점('.')을 사용할 수 있고, 3.24e3과 같이 지수형으로 표현 가능
- 문자열 타입 - 따옴표(' 또는 ")로 둘러싼 문자열.
  - 작은 따옴표 사용시, 값에 포함된 작은 따옴표는 ₩'로 입력
  - ₩ 기호 자체는 ₩₩ 로 표시한다.
- 널 타입 - null

EL에서 객체에 접근

- ${<표현1>.<표현2>} 형식 사용
- 처리 과정
  1. <표현1>을 <값1>로 변환한다.
  2. <값1>이 null이면 null을 리턴한다.
  3. <값1>이 null이 아닐 경우 <표현2>를 <값2>로 변환한다.
     1. <값2>가 null이면 null을 리턴한다.
  4. <값1>이 Map, List, 배열인 경우
     1. <값1>이 Map이면
        1. <값1>.containsKey(<값2>)가 false이면 null을 리턴한다.
        2. 그렇지 않으면 <값1>.get(<값2>)를 리턴한다.
     2. <값1>이 List나 배열이면
        1. <값2>가 정수 값인지 검사한다. (정수 값이 아닐 경우 에러 발생)
        2. <값1>.get(<값2>) 또는 Array.get(<값1>, <값2>)를 리턴한다.
        3. 위 코드가 예외를 발생하면 에러를 발생한다.
  5. <값1>이 다른 객체이면
     1. <값2>를 문자열로 변환한다.
     2. <값1>이 이름이 <값2>이고 읽기 가능한 프로퍼티를 포함하고 있다면 프로퍼티의 값을 리턴한다.
     3. 그렇지 않을 경우 에러를 발생한다.

연산자

- 수치 연산자
  - +, -, *, / 또는 div, % 또는 mod
- 비교 연산자
  - == 또는 eq, != 또는 ne
  - < 또는 lt, <= 또는 le, > 또는 gt, >= 또는 ge
- 논리 연산자
  - && 또는 and
  - || 또는 or
  - ! 또는 not
- empty 연산자
  - empty <값>
    - 값이 null이면, true
    - 값이 빈 문자열("")이면, true
    - 값의 길이가 0인 배열이나 콜렉션이면 true
    - 이 외의 경우에는 false
- 비교 선택 연산자
  - <수식> ? <값1> : <값2>

EL에서 클래스 메서드 호출하기

- 클래스의 static 메서드를 EL에서 호출 가능

```
public class DateUtil {
    public static String format(Date date) {
        SimpleDateFormat formatter = new SimpleDateFormat("yyyy-MM-dd");
        return formatter.format(date);
    }
}
```

- EL에서 호출하려면 다음의 작업 필요
  - EL 함수 정의한 TLD 파일 작성
  - web.xml 파일에 TLD 파일 지정
  - JSP 코드에서 TLD에 정의한 함수 실행

EL 함수를 정의한 TLD 파일

```
<?xml version="1.0" encoding="euc-kr" ?>
<taglib ... version="2.1">
    <description>EL에서 함수실행</description>
    <tlib-version>1.0</tlib-version>
    <short-name>ELfunctions</short-name>
    <function>
        <description>Date 객체 포맷팅</description>
        <name>dateFormat</name> <!-- EL에서 사용될 함수명 -->
        <function-class>kame.chap15.DateUtil</function-class>
        <function-signature>
            java.lang.String format(java.util.Date)
        </function-signature>
    </function>
</taglib>
```

web.xml 파일에 TLD 파일 지정

```
<web-app ... version="3.1">

    <jsp-config>
        <taglib>
            <taglib-uri>
                /WEB-INF/tlds/el-functions.tld
            </taglib-uri>
            <taglib-location>
                /WEB-INF/tlds/el-functions.tld
            </taglib-location>
        </taglib>
    </jsp-config>

</web-app>
```

JSP에서 사용될 URI

TLD 파일 경로

JSP에서 EL 함수 호출

```
<%@ taglib prefix="elfunc" uri="/WEB-INF/tlds/el-functions.tld" %>
<%
    java.util.Date today = new java.util.Date();
    request.setAttribute("today", today);
%>
<html>
<head><title>EL 함수 호출</title></head>
<body>

오늘은 <b>${elfunc:dateFormat(today) }</b> 입니다.
```

prefix:TLD에 정의된 함수명()

EL의 용법

- request나 session 속성으로 전달한 값을 출력
- 액션 태그나 커스텀 태그의 속성 값
  - <jsp:include page="/lo/${layout.module}.jsp" flush="true" />
- 함수 호출
  - 코드의 간결함 및 가독성 향상

EL 비활성화

- web.xml 파일에 비활성화 옵션 설정

```
<jsp-config>
    <jsp-property-group>
        <url-pattern>/oldversion/*</url-pattern>
        <el-ignored>true</el-ignored>
    </jsp-property-group>
</jsp-config>
```

* <deferred-syntax-allowed-as-literal>를 이용한 #{expr}을 문자열로 처리

- page 디렉티브에서 설정
  - isELIgnored 속성을 true로 하면 EL 무시
  - deferredSyntaxAllowedAsLiteral 속성을 true로 하면 #{expr}을 문자열로 처리
- web.xml 파일 버전에 따라 자동 비활성화
  - 서블릿 2.3 버전 : EL 지원하지 않음
  - 서블릿 2.4 버전 : ${expr} 형식의 EL 지원