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	설명
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 저장 한 Map 객체.
sessionScope	session 기본 객체에 저장된 속성의 <속성, 값> 매핑을 저장 한 Map 객체.
applicationScope	application 기본 객체에 저장된 속성의 <속성, 값> 매핑을 저 장한 Map 객체.
param	요청 파라미터의 <파라미터이름, 값> 매핑을 저장한 Map 객 체.
paramValues	요청 파라미터의 <파라미터이름, 값배열> 매핑을 저장한 Ma p 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>

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

JSTL

JSTL

- JSP Standard Tag Library - 널리 사용되는 커스텀 태그를 표준으로 만든 태그 라이브러리

- JSTL 태그 종류

라이브러리	하위 기능	접두어	관련URI
코어	변수지원 흐름 제어 URL 처리	c	http://java.sun.com/jsp/jstl/core
XML	XML 코어 흐름 제어 XML 변환	x	http://java.sun.com/jsp/jstl/xml
국제화	지역 메시지 형식 숫자 및 날짜 형식	fmt	http://java.sun.com/jsp/jstl/fmt
데이터베이스	SQL	sql	http://java.sun.com/jsp/jstl/sql
함수	콜렉션 처리 String 처리	fn	http://java.sun.com/jsp/jstl/functions

JSTL 1.2 관련 jar 파일 필요

- 아래 사이트에서 jstl-1.2.jar 파일 다운로드
  - https://maven-repository.dev.java.net/repository/jstl/jars/
- jstl-1.2.jar 파일을 WEB-INF/lib에 복사

코어 태그 라이브러리 종류

기능분류	태그	설명
변수 지원	set	JSP에서 사용될 변수를 설정한다.
	remove	설정한 변수를 제거한다.
흐름 제어	if	조건에 따라 내부 코드를 수행한다.
	choose	다중 조건을 처리할 때 사용된다.
	forEach	콜렉션이나 Map의 각 항목을 처리할 때 사용된다.
	forTokens	구분자로 분리된 각각의 토큰을 처리할 때 사용된다.
URL 처리	import	URL을 사용하여 다른 자원의 결과를 삽입한다.
	redirect	지정한 경로로 리다이렉트 한다.
	url	URL을 재작성 한다.
기타 태그	catch	예외 처리에 사용된다.
	out	JspWriter에 내용을 알맞게 처리한 후 출력한다.

변수 지원 태그

- 변수 설정
  - EL 변수 값 설정 (생성 또는 변경)
    - <c:set var="변수명" value="값" [scope="영역"] />
    - <c:set var="변수명" value="값" [scope="영역"]>값</c:set>
  - 특정 EL 변수의 프로퍼티 값 설정
    - <c:set target="대상" property="프로퍼티이름" value="값" />
    - <c:set target="대상" property="프로퍼티이름">값</c:set>

- 변수 삭제
  - <c:remove var="varName" [scope="영역"] />
    - scope 미지정시 모든 영역의 변수 삭제

흐름 제어

- if - 조건이 true일 경우 몸체 내용 실행

```
<c:if test="조건">
...
</c:if>
```

- choose - when - otherwise
  - swich - case - default와 동일

```
<c:choose>
  <c:when test="${member.level == 'trial'}" >
    ...
  </c:when>
  <c:when test="${member.level == 'regular'}" >
    ...
  </c:when>
  <c:otherwise>
    ...
  </c:otherwise>
</c:choose>
```

반복 처리

- forEach
  - 집합이나 콜렉션 데이터 사용

```
<c:forEach var="변수" items="아이템">
  … ${변수사용} …
</c:forEach>
```

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

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

index – 루프 실행에서 현재 인덱스, count – 루프 실행 회수 be
gin – begin 속성 값, end – end 속성 값, step – step 속성 값 fir
st – 현재 실행이 첫 번째 실행인 경우 true
last – 현재 실행이 루프의 마지막 실행인 경우 true
current – 콜렉션 중 현재 루프에서 사용할 객체

URL 관련 태그

- import - 외부/내부 페이지를 현재 위치에 삽입

```
<c:import url="URL" [var="변수명"] [scope="영역"] [charEncoding="캐릭터셋"]>
    <c:param name="파라미터이름" value="값" />
    ...
</c:import>
```

  - 상대 URL import 시 <jsp:include>와 동일하게 동작

- url - 절대 URL과 상대 URL을 알맞게 생성

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

  - 웹 컨텍스트 내에서 절대 경로 사용시 컨텍스트 경로 자동 추가

- redirect - 지정한 페이지로 리다이렉트

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

기타 코어 태그

- out - 데이터를 출력

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

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

  - escapeXml 속성이 true일 경우 다음과 같이 특수 문자 처리
    - < → < , > → >
    - & → & , ' → ' , " → "

- catch - 몸체에서 발생한 예외를 변수에 저장

```
<c:catch var="exName">
…
예외가 발생할 수 있는 코드
…
</c:catch>
${exName} 사용
```

국제화 태그

기능분류	태그	설명
로케일 지정	setLocale	Locale을 지정한다.
	requestEncoding	요청 파라미터의 캐릭터 인코딩을 지정
메시지 처리	bundle	사용할 번들을 지정
	message	지역에 알맞은 메시지를 출력
	setBundle	리소스 번들을 읽어와 특정 변수에 저장
숫자 및 날짜 포맷팅	formatNumber	숫자를 포맷팅
	formatDate	Date 객체를 포맷팅
	parseDate	문자열로 표시된 날짜를 분석해서 Date 객체로 변환
	parseNumber	문자열로 표시된 날짜를 분석해서 숫자로 변환
	setTimeZone	시간대 정보를 특정 변수에 저장
	timeZone	시간대를 지정

로케일 지정 및 요청 파라미터 인코딩 지정

- <fmt:setLocale value="언어코드" scope="범위" />
  - 국제화 태그가 Accept-Language 헤더에서 지정한 언어가 아닌 다른 언어를 사용하도록 지정하는 기능

- <fmt:requestEncoding value="캐릭터셋" />
  - 요청 파라미터의 캐릭터 인코딩을 지정
  - request.setCharacterEncoding("캐릭터셋")과 동일

<fmt:message> 태그

- 리소스 번들 범위에서 메시지 읽기

```
<fmt:bundle basename="resource.message" [prefix="접두어"]>
    <fmt:message key="GREETING" />
</fmt:bundle>
```

- 지정한 번들에서 메시지 읽기

```
<fmt:setBundle var="message" basename="resource.message" />
…
<fmt:message bundle="${message}" key="GREETING" />
```

- <fmt:message> 태그의 메시지 읽는 순서
  - bundle 속성에 지정한 리소스 번들을 사용
  - <fmt:bundle> 태그에 중첩된 경우 <fmt:bundle> 태그에서 설정한 리소스 번들 사용
  - 1과 2가 아닐 경우 기본 리소스 번들 사용. 기본 리소스 번들은 web.xml 파일에서 javax.servlet.jsp.jstl.fmt.localizationContext 콘텍스트 속성을 통해서 설정 가능

formatNumber 태그

- 숫자를 포맷팅

```
<fmt:formatNumber value="숫자값" [type="값타입"] [pattern="패턴"]
    [currencyCode="통화코드"] [currencySymbol="통화심볼"]  [groupin
    gUsed="(true|false)"] [var="변수명"] [scope="영역"] />
```

- 주요 속성

속성	표현식/EL	타입	설명
value	사용 가능	String 또는 Number	양식에 맞춰 출력할 숫자
type	사용 가능	String	어떤 양식으로 출력할지를 정한다. number는 숫자형식, percent는 % 형식, currency는 통화형식으로 출력. 기본 값은 number.
pattern	사용 가능	String	직접 숫자가 출력되는 양식을 지정한다. DecimalFormat 클래스에서 정의되어 있는 패턴 사용
var	사용 불가	String	포맷팅 한 결과를 저장할 변수 명. var 속성을 사용하지 않으면 결과가 곧바로 출력.
scope	사용 불가	String	변수를 저장할 영역. 기본 값은 page 이다.

parseNumber 태그

- 문자열을 숫자 데이터 타입으로 변환

```
<fmt:parseNumber value="값" [type="값타입"] [pattern="패턴"]
    [parseLocale="통화코드"] [integerOnly="true|false"]  [var="변
    수명"] [scope="영역"] />
```

- 주요 속성

속성	표현식/EL	타입	설명
value	사용 가능	String	파싱할 문자열
type	사용 가능	String	value 속성의 문자열 타입을 지정. number, currency, percentage 가 올 수 있다. 기본 값은 number
pattern	사용 가능	String	직접 파싱할 때 사용할 양식을 지정
var	사용 불가	String	파싱한 결과를 저장할 변수 명을 지정
scope	사용 불가	String	변수를 저장할 영역을 지정한다. 기본 값은 page.

formatDate 태그

- 날짜 정보를 담은 객체(Date)를 포맷팅

```
<fmt:formatDate value="날짜값"
    [type="타입"] [dateStyle="날짜스타일"] [timeStyle="시간스타일"]
    [pattern="패턴"] [timeZone="타임존"]
    [var="변수명"] [scope="영역"] />
```

- 주요 속성

속성	표현식/EL	타입	설명
value	사용 가능	java.util.Date	포맷팅할 날짜 및 시간 값
type	사용 가능	String	날짜, 시간 또는 둘 다 포맷팅 할 지의 여부를 지정
dateStyle	사용 가능	String	날짜에 대한 포맷팅 스타일을 지정
timeStyle	사용 가능	String	시간에 대한 포맷팅 스타일을 지정
pattern	사용 가능	String	직접 파싱할 때 사용할 양식을 지정
var	사용 불가	String	파싱한 결과를 저장할 변수 명을 지정
scope	사용 불가	String	변수를 저장할 영역을 지정

timeZone과 setTimeZone

- 국제화 태그가 사용할 시간대 설정

```
<fmt:timeZone value="Hongkong">
  <!-- 사용하는 시간을 Hongkong 시간대에 맞춘다. -->
  <fmt:formatDate ... />
</fmt:timeZone>
```

web.xml, 국제화 태그 콘텍스트 속성

속성 이름	설명
javax.servlet.jsp.jstl.fmt.localizationContext	기본으로 사용할 리소드 번들을 지정한다. 리소스 번들의 basename을 입력한다.
javax.servlet.jsp.jstl.fmt.locale	기본으로 사용할 로케일을 지정한다.
javax.servlet.jsp.jstl.fmt.timeZone	기본으로 사용할 시간대를 지정한다.

JSTL이 제공하는 주요 EL 함수

함수	설명
length(obj)	obj가 List와 같은 Collection인 경우 저장된 항목의 개수를 리턴하고, obj가 문자열일 경우 문자열의 길이를 리턴한다.
toUpperCase(str)	str을 대문자로 변환한다.
toLowerCase(str)	str을 소문자로 변환한다.
substring(str, idx1, idx2)	str.substring(idx1, idx2)의 결과를 리턴한다. idx2가 -1일 경우 str.substring(idx1)과 동일하다.
trim(str)	str 좌우의 공백문자를 제거한다.
replace(str, src, dest)	str에 있는 src를 dest로 변환한다.
indexOf(str1, str2)	str1에서 str2가 위치한 인덱스를 구한다.
startsWith(str1, str2)	str1이 str2로 시작할 경우 true를, 그렇지 않을 경우 false를 리턴한다.
endsWith(str1, str2)	str1이 str2로 끝나는 경우 true를, 그렇지 않을 경우 false를 리턴한다.
contains(str1, str2)	str1이 str2를 포함하고 있을 경우 true를 리턴한다.
escapeXml(str)	XML의 객체 참조에 해당하는 특수 문자를 처리한다. 예를 들어, '&'는 '&'로 변환한다.