07장. 익스프레션 언어

강사 강태광

EL(Expression Language)의 개요

1.EL(Expression Language)
- 표현언어를 의미
- jsp스크립트를 대신하여속성값들을 좀 더 편리하게 출력하기위해 제공되는언어
- 예를들어<%=hello%>라는코드를EL로표현할때는${hello}로표현된다

2.표현식

test 변수를 표현할 때 → ${test}

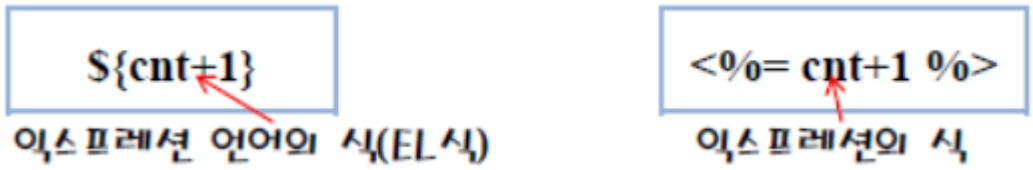
---

hello 객체의test 변수를표현하면다음과같다

${hello.test} 나 ${hello['test']}

익스프레션 언어란

1.익스프레션 언어(**expression language)란 식(expression)을 중심으로 코드를 기술하는 언어이다.**
2.연산자와 피연산자의 조합을 ${와 }로 둘러싸서 표현한다.



- 위에 사용된 **cnt 데이터 이름의 의미는 서로 다르다. 익스프레션에서 사용된 cnt는 자바 프로그래밍 언어의 변수 이름이며, EL 식에서 사용된 cnt는 애트리뷰트의 이름으로 해석.**
- 애트리뷰트란 **setAttribute, getAttribute, removeAttribute 메서드를 통해 저장되고, 관리되는 데이터를 의미한다.**

익스프레션 언어 예시

```
<%
    int sum = 0;
    for (int cnt = 1; cnt <= 100; cnt++)
    sum += cnt;
    request.setAttribute( "RESULT ", new Integer(sum));
    RequestDispatcher dispatcher =
    request.getRequestDispatcher( "HundredResult.jsp ");
    dispatcher.forward(request, response);
%>
```

덧셈의 결과를 애트리뷰트로 저장합니다

호출

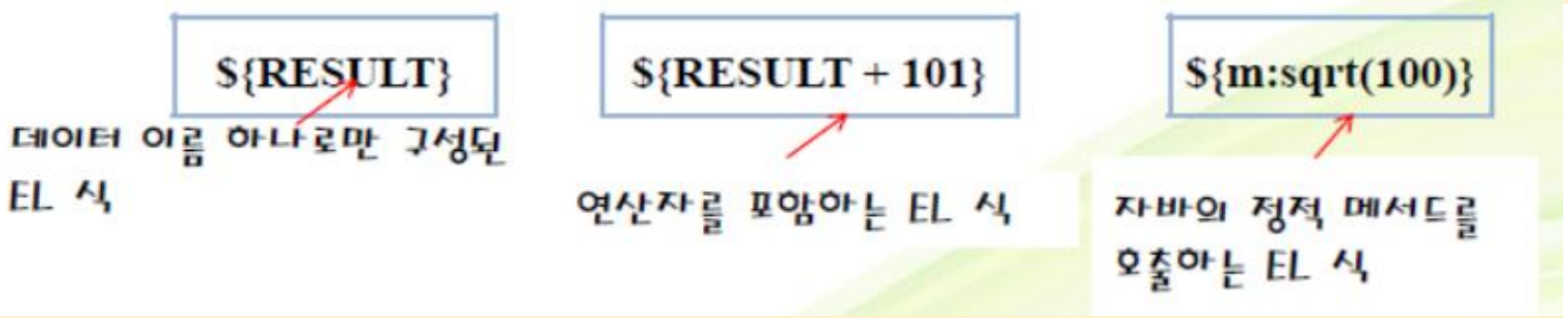
```
<%@page contentType= "text/html; charset=euc-kr "%>
<HTML>
<HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
<BODY>
1부터 100까지 더한 결과는? ${RESULT}
</BODY>
</HTML>
```

Attribute 값을 가져다가 출력합니다

애트리뷰트 값을 출력하는 EL 식

익스프레션 언어의 기초 문법

1. 익스프레션 언어의 유일한 목적은 식을 계산해서 그 결과를 출력하는 것이므로 다음과 같은 하나의 문법으로 표현할 수 있으며 이 문법을 **EL 식이라고 부른다**
2. 식'위치에는 데이터 이름 하나로만 구성된 식이 들어갈 수도 있고, 연산자를 포함하는 식이 들어 갈 수도 있으며, 자바의 정적 메서드를 호출하는 식이 들어갈 수도 있다



JSP/서블릿 기술에서 사용되는 네 종류의 애트리뷰트

애트리뷰트의 종류	호출할 때 사용하는 내장 변수	메서드의 소속
page 애트리뷰트	pageContext 내장 변수	javax.servlet.jsp.JspContext 클래스
request 애트리뷰트	request 내장 변수	javax.servlet.ServletRequest 인터페이스
session 애트리뷰트	session 내장 변수	javax.servlet.http.HttpSession 인터페이스
application 애트리뷰트	application 내장 변수	javax.servlet.ServletContext 인터페이스

EL 출력 예시(Request)

1부터 100까지의 합을 구하는 JSP 페이지 (hundred.jsp)

```
<%
    int sum = 0;
    for (int cnt = 1; cnt <= 100; cnt++)
    sum += cnt;
    // request 데이터 영역에  애트리뷰트를 저장
    request.setAttribute( "RESULT ", new Integer(sum));
    RequestDispatcher dispatcher =
    request.getRequestDispatcher( "HundredResult.jsp ");
    dispatcher.forward(request, response);
%>
```

1부터 100까지의 합을 출력하는 JSP 페이지(HundredResult.jsp)

```
<%@page contentType= "text/html; charset=euc-kr "%>
<HTML>
    <HEAD><TITLE>1부터 100까지의 합</TITLE></HEAD>
    <BODY>
      // request 데이터 영역에 있는 애트리뷰트 값을 가져다가 출력
        1부터 100까지 더한  결과는? ${RESULT}
    </BODY>
</HTML>
```

EL 출력 예시(page)

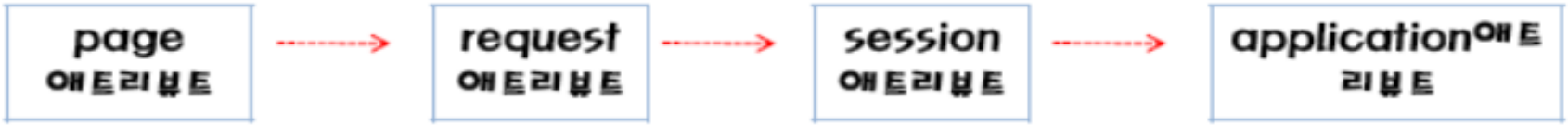
1부터 1000까지의 합을 구하는 JSP 페이지 (Thousand.jsp)

```
<%
    int sum = 0;
    for (int cnt = 1; cnt <= 1000; cnt++)
    sum += cnt;
    // page 데이터 영역에  애트리뷰트를 저장
    pageContext.setAttribute( "RESULT ", new Integer(sum));
%>
```

```
<HTML>
    <HEAD><TITLE>1부터 1000까지의 합</TITLE></HEAD>
    <BODY>
      // page 데이터 영역에 있는 애트리뷰트 값을 가져다가 출력
        1부터 1000까지 더한  결과는? ${RESULT}
    </BODY>
</HTML>
```

EL 식 출력 범위

1.EL 식에 있는 데이터 이름을 해석하는 순서는 사용 범위가 좁은 애트리뷰트 부터 점점 더 사용 범위가 넓은 애트리뷰트 순으로 진행 된다.



2.순서에 상관없이 특정한 종류의 애트리뷰트를 짚어서 출력하고 싶을 때는 다음과 같이 표시하면 된다.

```
${pageScope.SUM}          // page 애트리뷰트임을 표시
${requestScope.SUM}       // request 애트리뷰트임을 표시
${sessionScope.SUM}        // session 애트리뷰트임을 표시
${applicationScope.SUM}    // application 애트리뷰트임을 표시
```

익스프레션 언어의 내장 객체

내장 객체 이름	표현하는 데이터	객체의 타입
pageScope	page 애트리뷰트의 집합	Map
requestScope	request 애트리뷰트의 집합	Map
sessionScope	session 애트리뷰트의 집합	Map
applicationScope	application 애트리뷰트의 집합	Map
param	웹 브라우저로부터 입력된 데이터의 집합	Map
paramValues	웹 브라우저로부터 입력된 데이터의 집합 (똑같은 이름의 데이터가 여럿일 때 사용)	Map
header	HTTP 요청 메시지에 있는 HTTP 헤더의 집합	Map
headerValues	HTTP 요청 메시지에 있는 HTTP 헤더의 집합 (똑같은 이름의 HTTP 헤더가 여럿일 때 사용)	Map
cookie	웹 브라우저로부터 전송된 쿠키의 집합	Map
initParam	웹 애플리케이션의 초기화 파라미터의 집합	Map
pageContext	JSP 페이지의 환경 정보의 집합	PageContext

EL 내장 객체

1.param은 웹 브라우저에서 <FORM> 엘리먼트를 통해 입력된 데이터를 가져올 때 사용하는 내장 객체이다.
2. param 객체의 사용 방법은 두 가지

- param 뒤에 마침표를 찍고 해당 데이터 이름을 쓰는 방법
  ${param.NUM}
- param 뒤에 대괄호를 치고, 그 안에 작은따옴표나 큰 따옴표로 묶은 데이터 이름을 쓰는 방법
  ${param["COLOR "]}

3. <FORM> 엘리먼트를 통해 똑같은 이름의 데이터가 여러 개 입력되는 경우도 있는데, 그럴 때는 paramValuse 내장 객체를 사용하면 된다

① 객체 이름 뒤에 마침표를 찍고, 그 다음에 데이터 이름을 쓰고, 그 다음에 데이터 값의 인덱스를 대괄호로 묶어서 표시
${paramValues.ANIMAL[0]}

② 객체 이름 뒤에 두 개의 대괄호를 치고 그 안에 각각 따옴표로 묶은 데이터 이름과 인덱스를 쓰는 것
${paramValues["ANIMAL "][1]}

☞ EL 식의 인덱스가 0부터 시작하므로, 첫 번째 데이터 값을 가져오기 위해서는 인덱스를 0이라고 써야 하고, 두 번째 데이터 값을 가져오기 위해서는 1이라고 표기

EL 내장 객체 예시(쿠키저장 및 출력)

쿠키 데이터 값을 저장하는 JSP 페이지 CookieDataWriter.jsp

```
<%@page contentType= "text/html; charset=euc-kr " %>
<%
        Cookie cookie = new Cookie( "NAME ", "John ");
        response.addCookie(cookie);
%>
<HTML>
<HEAD><TITLE>쿠키 데이터 저장 프로그램</TITLE></HEAD>
<BODY>
      쿠키 값이 설정되었습니다.
</BODY></HTML>
```

쿠키 데이터 값을 출력하는 JSP 페이지 CookieDataReader.jsp

```
<%@page contentType= "text/html; charset=euc-kr " %>
<HTML>
      <HEAD><TITLE>쿠키 데이터 출력 프로그램</TITLE></HEAD>
      <BODY>
            // 쿠키 데이터의 값을 가져와서 출력
             NAME 쿠키 데이터의 값은? ${cookie.NAME.value}
      </BODY>
</HTML>
```

EL 내장 객체 예시(초기화 파라미터 출력)

1. initParam은 웹 애플리케이션의 초기화 파라미터 값을 가져다가 출력할 때 사용하는 내장 객체.
   initParam 객체의 이름 뒤에 마침표나 대괄호를 이용해서 해당 초기화 파라미터의 이름을 표시

2. 웹 애플리케이션의 초기화 파라미터 이름
   ${initParam.DB_NAME}
   ${initParam.["DB_NAME "]}

web.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app id="WebApp_ID" version="2.4" xmlns="http://java.sun.com/xml/ns/j2ee"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-
app_2_4.xsd">
	<welcome-file-list>
		<welcome-file>index.html</welcome-file>
		<welcome-file>index.htm</welcome-file>
		<welcome-file>index.jsp</welcome-file>
	</welcome-file-list>
	<context-param>
		<param-name>DB_NAME</param-name>
		<param-value>Oracle</param-value>
	</context-param>
</web-app>
```

EL 내장 객체 **pageContext**

1. pageContext 내장 객체는 JSP 페이지의 주변 환경에 대한 정보를 제공.
   이 내장 객체의 사용 방법은 다소 독특.
2. pageContext 내장 객체의 타입은 PageContext라고 되어 있는데 이것은 java.servlet.jsp 패키지에 속하는 클래스 이름이다. 이 객체를 이용하면 PageContext 클래스에 속하는 get으로 시작하는 이름의 메서드를 호출.
3. pageContext 클래스에는 8개의 get-메서드가 있으며, EL 식을 이용해서 이 메서드들을 호출할 때는 메서드 이름 제일 앞에 있는 get이라는 단어를 떼고, 그 다음에 있는 첫 문자를 소문자로 고친 이름을 사용.

```
${pageContext.request}              // getRequest 메서드를 가리키는 단어
${pageContext[ "request "]}         // getRequest 메서드를 가리키는 단어
```

4. 이 EL 식은 getRequest 메서드의 리턴값을 출력하는데, 그 값은 JSP 페이지의 request 내장 변수의 값과 동일한 객체이므로 사용자에게 아무 의미도 없는 참조 값만 출력

```
// getRequestURI 메서드의 리턴값을 가져오는 단어
${pageContext.request.requestURI}
${pageContext.["request "][ "requestURI "]}
${pageContext.request[ "requestURI "]}
${pageContext[ "request "].requestURI}
```

EL 연산자1

1. EL 연산자의종류와쓰임
   1) EL 연산자

연산자	설명
.	빈 또는 맵에 접근하기 위한 연산자이다.
[]	배열 또는 리스트에 접근하기 위한 연산자이다.
()	연산할 때 우선 순위를 주려고 할 때 사용한다.
x ? a : b	x의 조건이 만족하면 a를 리턴하고, 만족하지 않으면 b를 리턴한다.
Empty	값이 NULL일 경우 true를 반환한다.

2) 산술연산자

산술 연산자	설명
+	더하기 연산자
-	빼기 연산자
*	곱하기 연산자
/ 또는 div	나누기 연산자
% 또는 mod	나머지 연산자

EL 연산자2

3) 논리 연산자

논리 연산자	설명
&& 또는 and	두 항의 내용을 모두 만족할 경우 true, 그렇지 않으면 false를 반환한다.
|| 또는 or	두 항의 내용 중 하나라도 만족하면 true, 그렇지 않으면 false를 반환한다.
! 또는 not	값이 만족하지 않으면 true, 만족하면 false를 반환한다. 즉, true는 false 로 false는 true 로 변경해주는 연산자이다.

4) 비교 연산자

비교 연산자	설명
== 또는 eq	두 항의 값이 같으면 true, 그렇지 않으면 false를 반환한다.
!= 또는 ne	두 항의 값이 다르면 true, 그렇지 않으면 false를 반환한다.
< 또는 lt	'보다 작다'라는 의미를 갖고, 왼쪽 항이 오른쪽 항보다 작으면 true를 반환한다.
> 또는 gt	'보다 크다'라는 의미를 갖고, 왼쪽 항이 오른쪽 항보다 크면 true를 반환한다.
<= 또는 le	'같거나 작다'라는 의미를 갖고, 왼쪽 항이 오른쪽 항보다 같거나 작으면 true를 반환한다.
>= 또는 ge	'같거나 크다'라는 의미를 갖고, 왼쪽 항이 오른쪽 항보다 같거나 크면 true를 반환한다.

EL 연산자3

5) 연산자의 우선순위(연산자의 우선순위를 바꾸는 괄호 연산자)
①여러 개의 연산자가 포함된 수학식에는 왼쪽에서부터 오른쪽으로 계산되는 것이 순서이지만, 가감승제 연산자가 뒤섞여 있을 때는 곱셈, 나눗셈이 덧셈, 뺄셈보다 먼저 계산된다.
익스프레션 언어의 연산자에도 마찬가지로 우선순위가 있다.
② **EL 식 안에 여러 개의 연산자가 있으면 왼쪽부터 오른쪽으로 순서대로 처리되지만, 우선순위가 다른 연산자가 섞여 있으면 높은 우선순위의 연사자가 먼저 처리된다**

${(2 + 3) * 4} **//** 덧셈이 먼저 수행됩니다.

연산자의 우선순위를 바꾸는 괄호 연산자
익스프레션 언어의 연산자 우선 순위

우선순위	연 산 자
↑ 높음	[ ]
	( )
	-(부호) ! not empty
	* / % div mod
	+ -
	< > <= >= it ht le ge
	== != eq ne
	&& or
	|| or
↓ 낮음	? :

EL 연산자4

6) 대괄호 연산자와 마침표 연산자

Winners.jsp

```
<%@page contentType= "text/html; charset=UTF-8 "%>
<%
        String winners[] = new String[3];
        winners[0] = "이수현 ';
        winners[1] = "정세훈 ';
        winners[2] = "김진희 ';
        request.setAttribute( "WINNERS ", winners);  // 배열을 애트리뷰트 형태로 저장
        RequestDispatcher dispatcher = request.getRequestDispatcher( "WinnersView.jsp ");
        dispatcher.forward(request, response);
%>
```

WinnersView**.jsp**

```
<%@page contentType= "text/html; charset=UTF-8 "%>
<HTML>
        <HEAD><TITLE>우승자 명단</TITLE></HEAD>
        <BODY>
                <H3>우승자 명단</H3>
                1등. ${WINNERS[0]} <BR>
                2등. ${WINNERS[1]} <BR>
                3등. ${WINNERS[2]} <BR>
        </BODY>
</HTML>
```

EL 연산자5

7) Java.util.Map
  자바의 표준 라이브러리에 있는 인터페이스 이름이며, 이 인터페이스에는 여러 데이터 항목을 <이름, 값> 쌍으로 저장해서 관리할 수 있는 메서드

```
// 이름이 String 타입이고, 값이 Integer 타입인 데이터를 저장할 수 있는 HashMap 객체생성
HashMap<String, Integer> map = new HashMap<String, Integer>();
// HashMap 객체에 3개의 데이터를 저장
map.put( "Edgar ", new Integer(95));
map.put( "Thomas ", new Integer(100));
map.put( "John ", new Integer(75));
```

Map 객체에 저장핚 데이터 값을 가져오기 위해서는 대괄호 연산자나 마침표 연산자를 이용해서 데이터의 이름을 지정

```
${MAP[ "John "]}
${MAP.John}
```

EL 연산자 Map 예시

mapCreate.jsp

```
 <%@page contentType="text/html; charset=euc-kr"%>
<%@page import="java.util.*" %>
<%
        HashMap<String, String> map = new HashMap<String, String>();
        map.put("Park", "목동");
        map.put("Jasica", "크라이스 처치");
        map.put("Susan", "시드니");
        request.setAttribute("ADDRESS", map);
        RequestDispatcher dispatcher = request.getRequestDispatcher("mapView.jsp?NAME=Park");
        dispatcher.forward(request, response);
%>
```

오출

mapView.jsp

```
<%@ page language="java" contentType="text/html; charset=EUC-KR"
pageEncoding="EUC-KR"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
 <html>
        <head> <meta http-equiv="Content-Type" content="text/html; charset=EUC-KR">
                <title>Insert title here</title>
        </head>
        <body>
                ${param.NAME}의 주소는? ${ADDRESS[param.NAME]} <p>
                ${ADDRESS.Jasica} <p> ${ADDRESS.Susan} <p>
        </body>
</html>
```

EL 연산자예시

Operators.jsp

```
<%@page contentType= "text/html; charset=UTF-8 " %>
<HTML>
	<HEAD><TITLE>익스프레션 언어 연산자 연습</TITLE></HEAD>
	<BODY>
		X = ${param.NUM1}, Y = ${param.NUM2} <BR><BR>
		X + Y = ${param.NUM1 + param.NUM2} <BR>
		X - Y = ${param.NUM1 - param.NUM2} <BR>
		X * Y = ${param.NUM1 * param.NUM2} <BR>
		X / Y = ${param.NUM1 / param.NUM2} <BR>
		X % Y = ${param.NUM1 % param.NUM2} <BR><BR>
		X가 더 큽니까? ${param.NUM1 - param.NUM2 > 0} <BR>
		Y가 더 큽니까? ${param.NUM1 - param.NUM2 < 0} <BR><BR>
		X와 Y가 모두 양수입니까? ${(param.NUM1 > 0) && (param.NUM2 > 0)} <BR><BR>
		X와 Y가 같습니까? ${param.NUM1 == param.NUM2? "예 ' : "아니오 '} <BR><BR>
	</BODY>
</HTML>
```