

# JSP 웹 프로그래밍

액션 태그

상품목록 표시하기

## 액션 태그

---

- ▶ 서버나 클라이언트에게 어떤 행동을 하도록 명령하는 태그
- ▶ JSP 페이지에서 페이지와 페이지 사이를 제어
- ▶ 다른 페이지의 실행 결과 내용을 현재 페이지에 포함
- ▶ 자바 빈즈(JavaBeans) 등의 다양한 기능을 제공
- ▶ XML 형식 `<jsp: ... />`를 사용



# 액션 태그의 종류

액션 태그	형식	설명
forward	<code>&lt;jsp:forward ... /&gt;</code>	다른 페이지로의 이동과 같은 페이지 흐름을 제어합니다.
include	<code>&lt;jsp:include ... /&gt;</code>	외부 페이지의 내용을 포함하거나 페이지를 모듈화합니다.
useBean	<code>&lt;jsp:useBean ... /&gt;</code>	JSP 페이지에 자바빈즈를 설정합니다.
setProperty	<code>&lt;jsp:setProperty ... /&gt;</code>	자바빈즈의 프로퍼티 값을 설정합니다.
getProperty	<code>&lt;jsp:getProperty ... /&gt;</code>	자바빈즈의 프로퍼티 값을 얻어옵니다.
param	<code>&lt;jsp:param ... /&gt;</code>	<code>&lt;jsp:forward&gt;</code> , <code>&lt;jsp:include&gt;</code> , <code>&lt;jsp:plugin&gt;</code> 태그에 인자를 추가합니다.
plugin	<code>&lt;jsp:plugin ... /&gt;</code>	웹 브라우저에 자바 애플릿을 실행합니다. 자바 플러그인에 대한 OBJECT 또는 EMBED 태그를 만드는 브라우저별 코드를 생성합니다.
element	<code>&lt;jsp:element ... /&gt;</code>	동적 XML 요소를 설정합니다.
attribute	<code>&lt;jsp:attribute ... /&gt;</code>	동적으로 정의된 XML 요소의 속성을 설정합니다.
body	<code>&lt;jsp:body ... /&gt;</code>	동적으로 정의된 XML 요소의 몸체를 설정합니다.
text	<code>&lt;jsp:text ... /&gt;</code>	JSP 페이지 및 문서에서 템플릿 텍스트를 작성합니다.



## forward 액션 태그

---

- ▶ 현재 JSP 페이지에서 다른 페이지로 이동하는 태그
- ▶ JSP 컨테이너는 현재 JSP 페이지에서 forward 액션 태그를 만나면
  - ▶ 그 전까지 출력 버퍼에 저장되어 있던 내용을 모두 삭제하고
  - ▶ forward 액션 태그에 설정된 페이지로 프로그램의 제어가 이동

```
<jsp:forward page="파일명" />
```

또는

```
<jsp:forward page="파일명">    </jsp:forward>
```

- ▶ page 속성 값
  - ▶ 현재 JSP 페이지에서 이동할 페이지의 외부 파일명
  - ▶ 외부 파일은 현재 JSP 페이지와 같은 디렉터리에 있으면 파일명만 설정, 그렇지 않으면 전체 URL(또는 상대 경로)을 설정.

# forward 액션 태그의 페이지 흐름 처리 과정

---

```
first.jsp  
<jsp:forward  
page="second.jsp"/>
```

```
second.jsp  
...
```

1. 웹 브라우저에서 웹 서버로 first.jsp를 요청
2. JSP컨테이너는 요청된 first.jsp를 실행
3. first.jsp를 실행하다가 forward액션 태그를 만나면 지금까지 저장된 출력 버퍼의 내용을 삭제하고 프로그램 제어를 page속성에서 설정한 second.jsp로 이동
4. second.jsp를 실행
5. JSP컨테이너는 second.jsp를 실행한 결과를 웹 브라우저에 응답으로 보냄



## forward 액션 태그의 사용 예제

- ▶ forward 액션 태그로 현재 날짜와 시각을 출력하는 페이지로 이동하기

```
forward.jsp ✕
1 <%@ page contentType="text/html; charset=UTF-8"%>
2 <!DOCTYPE html>
3 <html>
4 <head>
5 <title>Action Tag</title>
6 </head>
7 <body>
8     <h3>이 파일은 first.jsp입니다.</h3>
9     <jsp:forward page="second.jsp" />
10    <p>===first.jsp의 페이지===</p>
11 </body>
12 </html>
```

# forward 액션 태그의 사용 예제

second.jsp

```
1 <%@ page contentType="text/html; charset=UTF-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7     <h3>이 파일은 Second.jsp입니다.</h3>
8     Today is :
9     <%=new java.util.Date()%>
10 </body>
11 </html>
```

← → 🚫 📌 http://localhost:8090/JSPTTest/Chapter04/forward.jsp

이 파일은 **Second.jsp**입니다.

Today is : Mon Dec 03 23:43:38 KST 2018

# include 액션 태그

- ▶ include 디렉티브 태그처럼 현재 JSP 페이지의 특정 영역에 외부 파일의 내용을 포함하는 태그
- ▶ 현재 JSP 페이지에 포함할 수 있는 외부 파일
  - ▶ HTML, JSP, 서블릿 페이지 등 

```
<jsp:include page="파일명" flush="false"/>
```
- ▶ page 속성 값
  - ▶ 현재 JSP 페이지 내에 포함할 내용을 가진 외부 파일명
  - ▶ 외부 파일은 현재 JSP 페이지와 같은 디렉터리에 있으면 파일명만 설정, 그렇지 않으면 전체 URL(또는 상대 경로)을 설정
- ▶ flush 속성 값
  - ▶ 설정한 외부 파일로 제어가 이동할 때 현재 JSP 페이지가 지금까지 출력 버퍼에 저장한 결과를 처리, 기본 값은 false
  - ▶ true 로 설정하면 외부 파일로 제어가 이동할 때 현재 JSP 페이지가 지금까지 출력 버퍼에 저장된 내용을 웹 브라우저에 출력하고 출력 버퍼를 비움.



# include 액션 태그의 페이지 흐름 처리 과정

---

```
first.jsp  
<jsp:include="second.jsp"  
flush="false"/>
```

```
second.jsp  
...
```

1. JSP컨테이너는 요청 받은 first.jsp를 처리하고 first.jsp내의 출력 내용이 출력 버퍼에 저장됨
2. include액션 태그를 만나면 하던 작업을 멈추고 프로그램 제어를 second.jsp로 이동
3. second.jsp를 실행하고 second.jsp내의 출력 내용이 출력 버퍼에 저장됨
4. second.jsp의 처리가 끝나면 다시 first.jsp로 프로그램의 제어가 이동, 이동 위치는 include태그 문장의 다음 행이 됨.
5. first.jsp의 나머지 부분을 처리하고, 출력할 내용이 있으면 출력 버퍼에 저장
6. JSP컨테이너는 출력 버퍼의 내용을 웹 브라우저에 응답을 보냄

## include 액션 태그와 include 디렉티브 태그의 차이

---

구분	include 액션 태그	include 디렉티브 태그
처리 시간	요청 시 자원을 포함합니다.	번역 시 자원을 포함합니다.
기능	별도의 파일로 요청 처리 흐름을 이동합니다.	현재 페이지에 삽입합니다.
데이터 전달 방법	request 기본 내장 객체나 param 액션 태그를 이용하여 파라미터를 전달합니다.	페이지 내의 변수를 선언한 후 변수에 값을 저장합니다.
용도	화면 레이아웃의 일부분을 모듈화할 때 주로 사용합니다.	다수의 JSP 웹 페이지에서 공통으로 사용되는 코드나 저작권과 같은 문장을 포함하는 경우에 사용합니다.
기타	동적 페이지에 사용합니다.	정적 페이지에 사용합니다.



# include 액션 태그의 사용 예제

- ▶ include 액션 태그에 현재 날짜와 시각을 출력하는 페이지 포함하기

```
include.jsp
1 <%@ page contentType="text/html; charset=UTF-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7   <h2>include 액션 태그</h2>
8   <jsp:include page="include_date.jsp"/>
9   <p>-----</p>
10 </body>
11 </html>
```

http://localhost:8090/JSPTest/Chapter04/include.jsp

## include 액션 태그

오늘의 날짜 및 시각

Tue Dec 04 00:10:59 KST 2018

# include 액션 태그의 사용 예제

---

include\_date.jsp

```
1 <%@ page contentType="text/html; charset=UTF-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7     <p>오늘의 날짜 및 시각</p>
8     <p>
9         <%=new java.util.Date()%>
10    </p>
11 </body>
12 </html>
```

## param 액션 태그

---

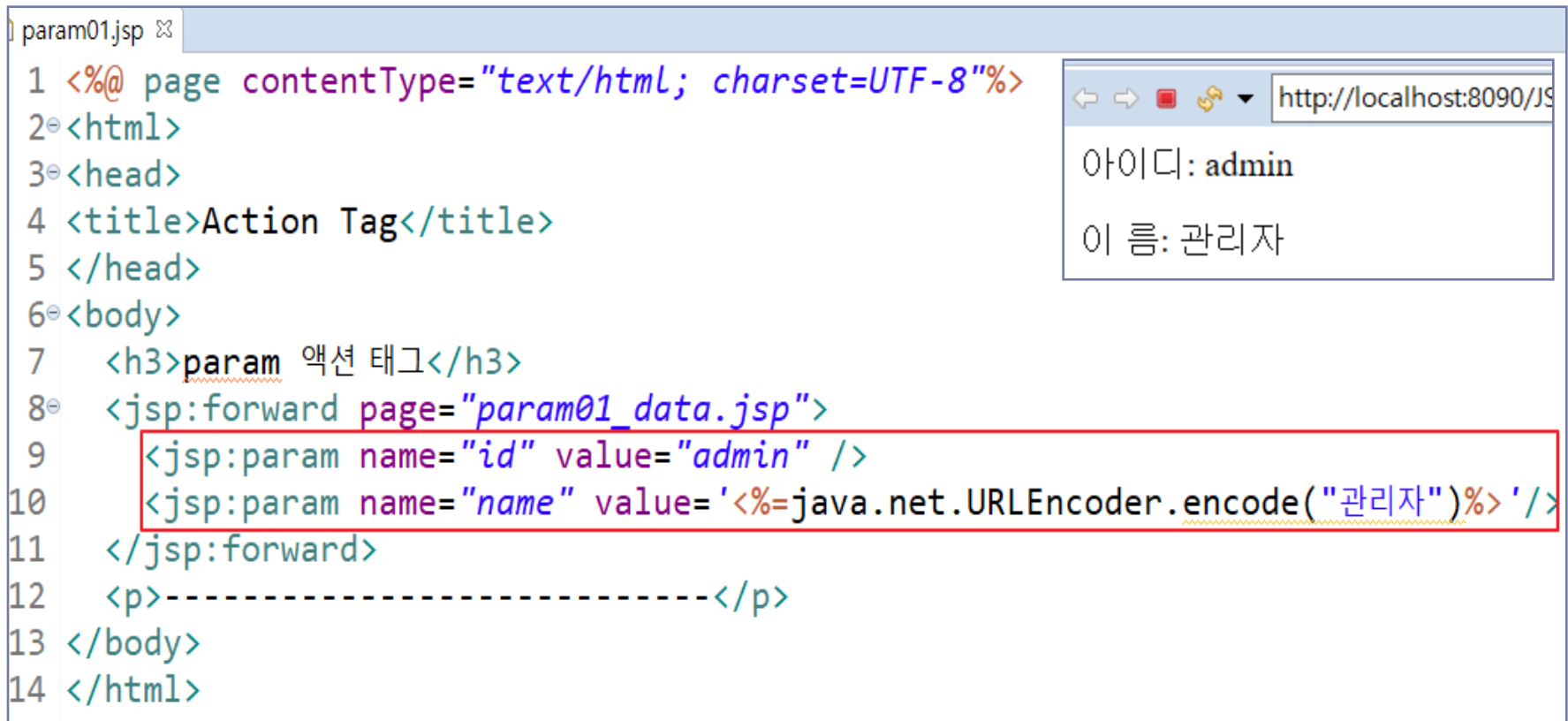
- ▶ 현재 JSP 페이지에서 다른 페이지에 정보를 전달하는 태그
- ▶ 단독으로 사용되지 못하며 <jsp:forward>나 <jsp:include> 태그의 내부에 사용
- ▶ 다른 페이지에 여러 개의 정보를 전송해야 할 때는 다중의 param 액션 태그를 사용
- ▶ param 액션태그 형식

```
<jsp:forward page="파일명" >  
    <jsp:param name="매개변수명1" value="매개변수값1" />  
    [<jsp:param name="매개변수명2" value="매개변수값2" /> ... ]  
</jsp:forward>
```



# param 액션 태그 예제

- ▶ Forward 액션 태그와 param 액션 태그에 아이디와 이름 전달하기



The image shows a JSP editor window with a file named `param01.jsp`. The code contains a JSP page directive, HTML boilerplate, and a JSP forward action tag. Inside the forward tag, two JSP param action tags are used to pass data. The first param tag sets the name 'id' to the value 'admin'. The second param tag sets the name 'name' to the value `<%=java.net.URLEncoder.encode("관리자")%>`. A red box highlights these two param tags. To the right of the editor, a browser window is shown with the URL `http://localhost:8090/JSP`. The browser displays the output of the JSP page, which is '아이디: admin' and '이름: 관리자'.

```
1 <%@ page contentType="text/html; charset=UTF-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7   <h3>param 액션 태그</h3>
8   <jsp:forward page="param01_data.jsp">
9     <jsp:param name="id" value="admin" />
10    <jsp:param name="name" value='<%=java.net.URLEncoder.encode("관리자")%>' />
11  </jsp:forward>
12  <p>-----</p>
13 </body>
14 </html>
```

Browser Output:

아이디: admin  
이름: 관리자

# param 액션 태그 예제

```
param01_data.jsp
1 <%@ page contentType="text/html; charset=UTF-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7   <p>
8     아이디: <%=request.getParameter("id")%></p>
9     <%
10       String name = request.getParameter("name");
11       %>
12   <p>
13     이름: <%=java.net.URLDecoder.decode(name)%></p>
14 </body>
```

# param 액션 태그 예제

- ▶ include 액션 태그와 param 액션 태그에 제목과 현재 날짜 전달하기

param02.jsp

```
1 <%@ page contentType="text/html; charset=" %>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7     <h3>param 액션 태그</h3>
8     <jsp:include page="param02_data.jsp">
9         <jsp:param name="title"
10             value='<%=java.net.URLEncoder.encode("오늘의 날짜와 시각")%>' />
11         <jsp:param name="date"
12             value="<%=java.util.Calendar.getInstance().getTime()%>" />
13     </jsp:include>
14 </body>
15 </html>
```

http://localhost:8090/JSPTest/Chapter04

param 액션 태그

오늘의 날짜와 시각

Today is :Tue Dec 04 11:43:40 KST 2018





# param 액션 태그 예제

---

```
param02_data.jsp ✕
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7     <%
8         String title = request.getParameter("title");
9     %>
10    <h3><%=java.net.URLDecoder.decode(title)%></h3>
11    Today is :<%=request.getParameter("date")%>
12 </body>
13 </html>
```



# 자바빈즈

---

- ▶ 동적 콘텐츠 개발을 위해 자바 코드를 사용하여 자바 클래스로 로직을 작성하는 방법
- ▶ JSP 페이지에서 화면을 표현하기 위한 계산식이나 자료의 처리를 담당하는 자바코드를 따로 분리하여 작성하는 것
- ▶ JSP 페이지는 HTML과 같이 쉽고 간단한 코드만으로 구성



## 자바빈즈를 작성할 때 규칙

---

- ▶ 자바 클래스는 `java.io.Serializable` 인터페이스를 구현해야 함.
- ▶ 인수가 없는 기본 생성자가 있어야 함
- ▶ 모든 멤버 변수인 프로퍼티는 `private` 접근 지정자로 설정해야 함.
- ▶ 모든 멤버 변수인 프로퍼티는 `getter/setter()` 메소드가 존재해야 함.
  - ▶ `getter()` 메소드 - 멤버 변수에 저장된 값을 가져올 수 있는 메소드
  - ▶ `setter()` 메소드 - 멤버 변수에 값을 저장할 수 있는 메소드



## 자바빈즈 액션 태그: useBean 액션 태그

- ▶ JSP 페이지에서 자바빈즈를 사용하기 위해 실제 자바 클래스를 선언하고 초기화하는 태그
- ▶ id 속성과 scope 속성을 바탕으로 자바빈즈의 객체를 검색하고, 객체가 발견되지 않으면 빈 객체를 생성

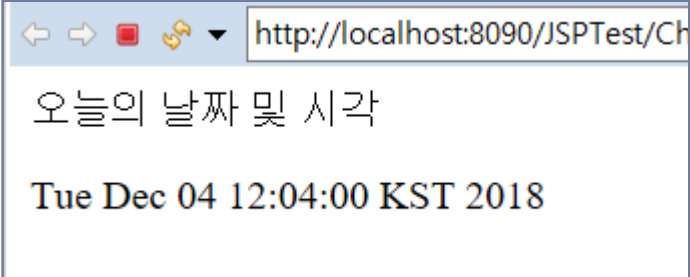
```
<jsp:useBean id= "자바빈즈 식별이름" class= "자바빈즈 이름" scope= "범위"/>
```

속성	설명
id	자바빈즈를 식별하기 위한 이름입니다.
class	패키지 이름을 포함한 자바빈즈 이름입니다. 자바빈즈는 인수가 없는 기존 생성자가 있어야 하며 추상 클래스를 사용할 수 없습니다.
scope	자바빈즈가 저장되는 영역을 설정합니다. page(기본 값), request, session, application 중 하나의 값을 사용합니다.

# 자바빈즈 액션 태그 예제

- ▶ useBean 액션 태그에 Date클래스를 사용하여 현재 날짜와 시각 출력하기

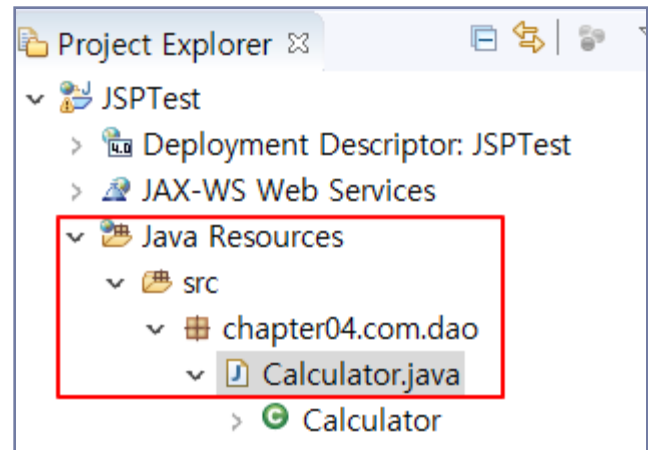
```
*useBean01.jsp ✕
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7   <jsp:useBean id="date" class="java.util.Date" />
8   <p>
9     <%
10       out.println("오늘의 날짜 및 시각");
11     %>
12   </p>
13   <%=date%>
14 </body>
15 </html>
```



## 자바빈즈 액션 태그 예제

- ▶ 자바빈즈 Calculator를 생성하고 useBean액션 태그에 Calculator클래스를 사용하여 숫자 출력하기

```
Calculator.java ✖
1 package chapter04.com.dao;
2
3 public class Calculator {
4     public int process(int n)
5     {
6         return n*n*n;
7     }
8 }
```



# 자바빈즈 액션 태그 예제

useBean02.jsp

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7     <jsp:useBean id="bean" class="chapter04.com.dao.Calculator" />
8     <%
9         int m = bean.process(5);
10        out.print("5의 3제곱: " + m);
11    %>
12 </body>
13 </html>
```

← → 🏠 🔍 http://localhost:8090/JSPTest/

5의 3제곱: 125

# 자바빈즈 액션 태그 예제

- ▶ 자바빈즈 Person을 생성하고 useBean액션 태그에 Person 클래스를 사용하여 아이디와 이름 출력하기

```
Person.java ✖
1 package chapter04.com.dao;
2 public class Person {
3     private int id = 20181204;
4     private String name = "홍길동";
5     public Person() {
6     }
7     public int getId() {
8         return id;
9     }
10    public void setId(int id) {
11        this.id = id;
12    }
13    public String getName() {
14        return name;
15    }
16    public void setName(String name) {
17        this.name = name;
18    };
19 }
```



# 자바빈즈 액션 태그 예제

useBean03.jsp

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7 <jsp:useBean id="person" class="chapter04.com.dao.Person" scope="request" />
8 <p>
9   아이디 :
10 <%=person.getId()%>
11 <p>
12   이 름 :
13 <%=person.getName()%>
14 </body>
15 </html>
```

← → ⏏ 🔄 ▼ http://localhost:8090/

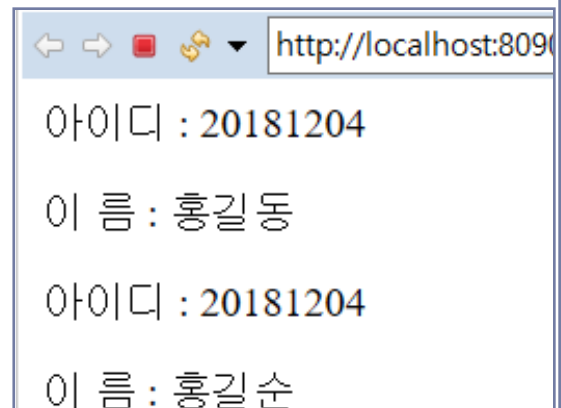
아이디 : 20181204

이 름 : 홍길동

# 자바빈즈 액션 태그 예제

- ▶ useBean 액션 태그에 이전에 생성한 자바빈즈 Person으로 아이디와 이름을 설정하여 출력하기

```
useBean04.jsp
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7 <jsp:useBean id="person" class="chapter04.com.dao.Person" scope="request" />
8 <p>
9   아이디 :
10   <%=person.getId()%>
11 <p>
12   이 름 :
13   <%=person.getName()%>
14   <%
15     person.setId(20181204);
16     person.setName("홍길순");
17   %>
18   <jsp:include page="useBean03.jsp" />
19 </body>
20 </html>
```



## 자바빈즈 액션 태그: setProperty 액션 태그

---

- ▶ 프로퍼티의 값 저장
- ▶ useBean 액션 태그와 함께 자바빈즈의 setter() 메소드에 접근하여 자바빈즈의 멤버 변수인 프로퍼티의 값을 저장하는 태그

```
<jsp:setProperty name="자바빈즈 식별이름" property="프로퍼티 이름" value="값" />
```

- ▶ 폼 페이지로부터 전달되는 요청 파라미터의 값을 직접 저장
- ▶ 자바빈즈의 프로퍼티로 변경하여 값을 저장
- ▶ 모든 자바빈즈 프로퍼티 이름과 동일하게 요청 파라미터를 설정



# 자바빈즈 액션 태그: setProperty사용 예

```
<jsp:setProperty name="member" property="id" value="admin" />
```

속성	설명
name	useBean 태그에 id 속성 값으로 설정된 자바빈즈를 식별하기 위한 이름입니다.
property	자바빈즈의 프로퍼티 이름입니다. 만약 프로퍼티 이름에 '*'를 사용하면 모든 요청 파라미터가 자바빈즈 프로퍼티의 setter() 메소드에 전달됨을 의미합니다.
value	변경할 자바빈즈의 프로퍼티 값입니다. 만약 프로퍼티 값이 null이거나 존재하지 않는 요청 파라미터인 경우에는 setProperty 액션 태그가 무시됩니다.
param	자바빈즈의 프로퍼티 값을 전달하는 요청 파라미터의 이름입니다. param과 value를 동시에 모두 사용할 수 없으며 하나를 선택하여 사용하는 것은 가능합니다.



## 자바빈즈 액션 태그: 프로퍼티 값 출력 예

---

```
<jsp:useBean id="member" class="com.dto.MemberBean" scope="page" />
```

```
<% out.println("아이디 : "+member.getId()); %>
```



# 자바빈즈 액션 태그의 사용법

---

- ▶ 요청 파라미터 이름과 자바빈즈의 프로퍼티 이름이 일치하는 경우:

```
//폼 페이지  
<form action="memberProcess.jsp" method ="post">  
    <input name="id" value="admin" />  
</form>
```

```
//jsp 페이지  
<jsp:setProperty name="member" property="id" />
```



# 자바빈즈 액션 태그의 사용법

---

- ▶ 요청 파라미터 이름과 자바빈즈의 프로퍼티 이름이 일치하지 않는 경우:

```
//폼 페이지  
<form action="memberProcess.jsp" method="post">  
    <input name="userId" value="admin" />  
</form>
```

```
//jsp 페이지  
<jsp:setProperty name="member" property="id" param="userId" />
```



## 자바빈즈 액션 태그의 사용법

- ▶ 요청 파라미터 이름과 자바빈즈의 프로퍼티 이름이 모두 일치하는 경우:

```
//폼 페이지
<form action="memberProcess.jsp" method ="post">
    <input name="id" value="admin" />
    <input name="name" value="관리자" />
</form>
```

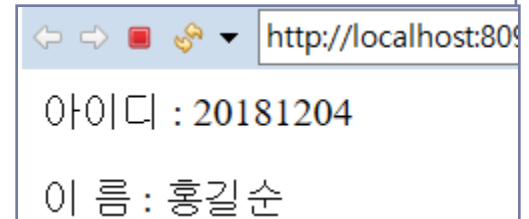
```
//jsp 페이지
<jsp:setProperty name="member" property="*" />
```



# 자바빈즈 액션 태그의 사용 예제

- ▶ **setProperty** 액션 태그에 자바빈즈 Person으로 아이디와 이름을 설정하여 출력하기

```
setProperty.jsp
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7 <jsp:useBean id="person" class="chapter04.com.dao.Person" scope="request" />
8 <jsp:setProperty property="id" name="person" value="20181204" />
9 <jsp:setProperty property="name" name="person" value="홍길순" />
10 <p>
11     아이디 :
12     <%
13         out.println(person.getId());
14     %>
15 <p>
16     이 름 :
17     <%
18         out.println(person.getName());
19     %>
20 </body>
21 </html>
```



## 자바빈즈 액션 태그: getProperty 액션 태그

- ▶ 프로퍼티의 값 가져오기
- ▶ useBean 액션 태그와 함께 자바빈즈의 getter( ) 메소드에 접근하여 자바빈즈의 멤버 변수인 프로퍼티의 값을 가져오는 태그

```
<jsp:getProperty name="자바빈즈 식별이름" property="프로퍼티 이름" />
```

속성	설명
name	useBean 태그에 id 속성 값으로 설정된 자바빈즈를 식별하기 위한 이름입니다.
property	자바빈즈의 프로퍼티 이름입니다. 만약 프로퍼티 이름에 '*'를 사용하면 모든 요청 파라미터가 자바빈즈 프로퍼티의 getter( ) 메소드에 전달됨을 의미합니다.

# 자바빈즈 액션 태그 사용법

---

## ▶ [getProperty 액션 태그 사용 예]

```
<jsp:getProperty name="member" property="name" />
```

## ▶ [자바빈즈 프로퍼티 값 출력 예]

```
<% out.println(member.getName()); %>
```



# 자바빈즈 액션 태그의 사용 예제

- ▶ `getProperty` 액션 태그에 자바빈즈 `Person`을 이용하여 아이디와 이름을 가져와 출력하기

```
getProperty01.jsp
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7   <jsp:useBean id="person" class="chapter04.com.dao.Person" scope="request" />
8   <p>
9     아이디 :<jsp:getProperty property="id" name="person" />
10  <p>
11    이름 :<jsp:getProperty property="name" name="person" />
12 </body>
13 </html>
```

← → ⏏ 🔄 ▼ http://localhost:809

아이디 :20181204

이름 :홍길동

# 자바빈즈 액션 태그의 사용 예제

- ▶ `setProperty`, `getProperty` 액션 태그로 자바빈즈 `Person`을 이용하여 아이디와 이름을 설정 후 출력하기

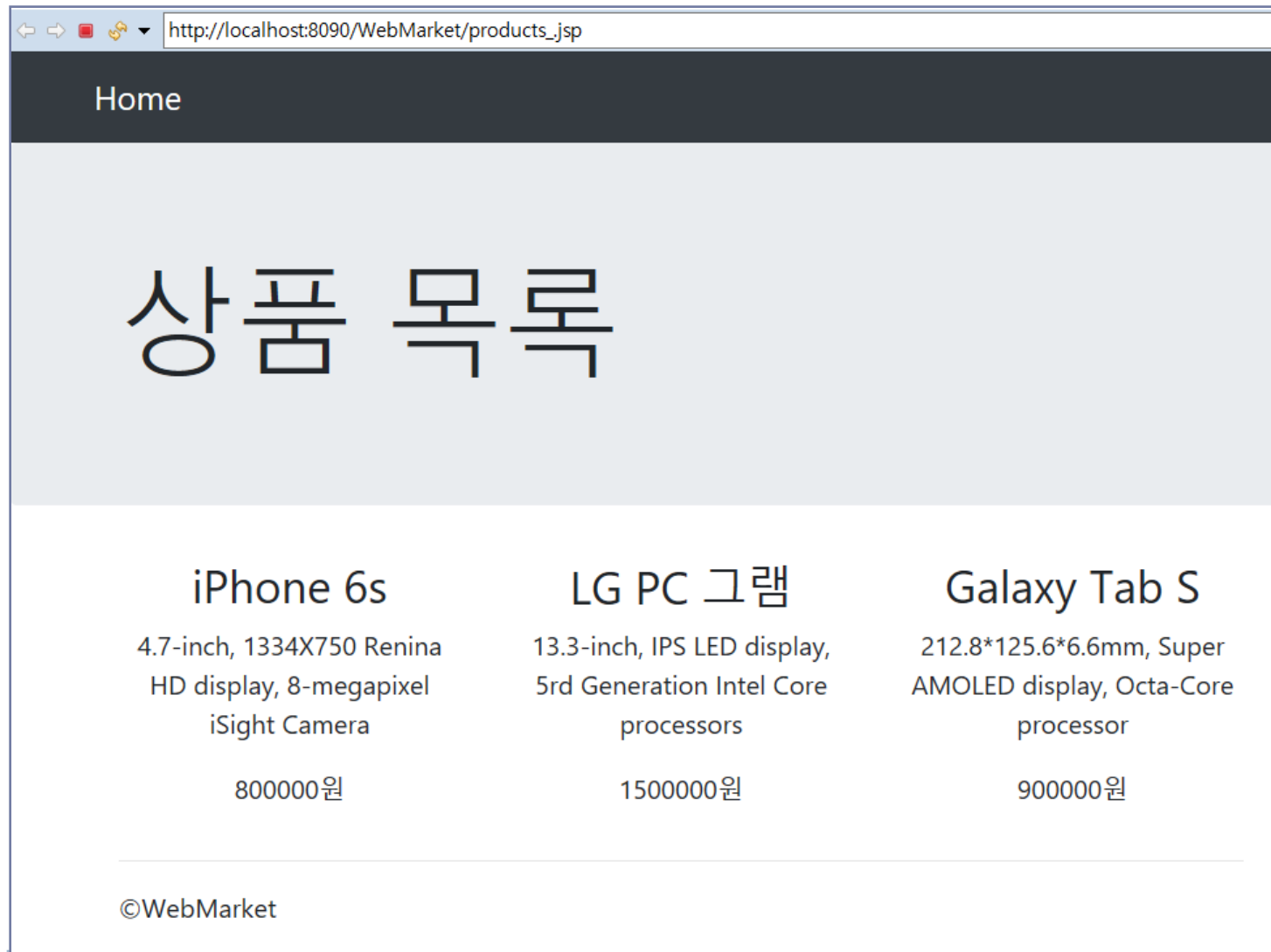
```
getProperty01.jsp
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <html>
3 <head>
4 <title>Action Tag</title>
5 </head>
6 <body>
7   <jsp:useBean id="person" class="chapter04.com.dao.Person" scope="request" />
8   <jsp:setProperty property="id" name="person" value="20181204" />
9   <jsp:setProperty property="name" name="person" value="홍길순" />
10  <p>
11    아이디 :<jsp:getProperty property="id" name="person" />
12  <p>
13    이 름 :<jsp:getProperty property="name" name="person" />
14 </body>
15 </html>
```

http://localhost:8090

아이디 :20181204

이 름 :홍길순

# [웹 쇼핑몰] 상품 목록 표시하기



# 상품 목록 표시하기

- ▶ 상품 클래스 생성하기
- ▶ 멤버 변수 선언하기

```
Product.java ✖
1 package dto;
2
3 import java.io.Serializable;
4
5 public class Product implements Serializable {
6
7     private static final long serialVersionUID = -4274700572038677000L;
8
9     private String productId;    //상품 아이디
10    private String pname;        //상품명
11    private Integer unitPrice;    //상품 가격
12    private String description;  //상품 설명
13    private String manufacturer; //제조사
14    private String category;     //분류
15    private long unitsInStock;   //재고 수
16    private String condition;    //신상품 or 중고품 or 재생품
17 }
```

# 상품 목록 표시하기

---

## ▶ 생성자 작성하기

```
5 public class Product implements Serializable {  
6     public Product() {  
7         super();  
8     }  
9  
10    public Product(String productId, String pname, Integer unitPrice) {  
11        super();  
12        this.productId = productId;  
13        this.pname = pname;  
14        this.unitPrice = unitPrice;  
15    }  
16
```





# 상품 목록 표시하기

## ▶ 모든 멤버 변수의 Setter/Getter( ) 메소드 작성하기:

```
public class Product implements Serializable {  
    //....  
    public String getProductId() {  
        return productId;  
    }  
  
    public void setProductId(String productId) {  
        this.productId = productId;  
    }  
  
    public String getPname() {  
        return pname;  
    }  
  
    public void setPname(String pname) {  
        this.pname = pname;  
    }  
  
    public Integer getUnitPrice() {  
        return unitPrice;  
    }  
  
    public void setUnitPrice(Integer unitPrice) {  
        this.unitPrice = unitPrice;  
    }  
}
```

```
    public String getDescription() {  
        return description;  
    }  
  
    public void setDescription(String description) {  
        this.description = description;  
    }  
  
    public String getManufacturer() {  
        return manufacturer;  
    }  
  
    public void setManufacturer(String manufacturer) {  
        this.manufacturer = manufacturer;  
    }  
  
    public String getCategory() {  
        return category;  
    }  
  
    public void setCategory(String category) {  
        this.category = category;  
    }  
  
    public long getUnitsInStock() {  
        return unitsInStock;  
    }  
  
    public void setUnitsInStock(long unitsInStock) {  
        this.unitsInStock = unitsInStock;  
    }  
}
```

```
    public String getCondition() {  
        return condition;  
    }  
  
    public void setCondition(String condition) {  
        this.condition = condition;  
    }  
  
    public static long getSerialVersionUID() {  
        return serialVersionUID;  
    }  
}
```

# 상품 목록 표시하기

- ▶ [자바빈즈로 사용할 상품 데이터 접근 클래스 만들기]
  - ▶ 자바빈즈로 사용할 클래스, 멤버 변수와 기본 생성자 만들기

```
ProductRepository.java
1 package dao;
2 import java.util.ArrayList;
3 import dto.Product;
4
5 public class ProductRepository {
6     private ArrayList<Product> listOfProducts = new ArrayList<Product>();
7     public ProductRepository()
8     {
9         Product phone = new Product("P1234", "iPhone 6s", 800000);
10        phone.setDescription("4.7-inch, 1334X750 Retina HD display, "
11                               + "8-megapixel iSight Camera");
12        phone.setCategory("Smart Phone");
13        phone.setManufacturer("Apple");
14        phone.setUnitsInStock(1000);
15        phone.setCondition("New");
16
17        Product notebook = new Product("P1235", "LG PC 그램", 1500000);
18        notebook.setDescription("13.3-inch, IPS LED display, "
19                                + "5rd Generation Intel Core processors");
20        notebook.setCategory("Notebook");
21        notebook.setManufacturer("LG");
22        notebook.setUnitsInStock(1000);
23        notebook.setCondition("Refurbished");
24    }
25 }
```

# 상품 목록 표시하기

---

## ▶ ProductRepository.jsp의 생성자

```
20     notebook.setCategory( Notebook );
21     notebook.setManufacturer("LG");
22     notebook.setUnitsInStock(1000);
23     notebook.setCondition("Refurbished");
24
25     Product tablet = new Product("P1236", "Galaxy Tab S", 900000);
26     tablet.setDescription("212.8*125.6*6.6mm, "
27         + "Super AMOLED display, Octa-Core processor");
28     tablet.setCategory("Tablet");
29     tablet.setManufacturer("Samsung");
30     tablet.setUnitsInStock(1000);
31     tablet.setCondition("Old");
32
33     listOfProducts.add(phone);
34     listOfProducts.add(notebook);
35     listOfProducts.add(tablet);
36 }
37 }
```

# 상품 목록 표시하기

---

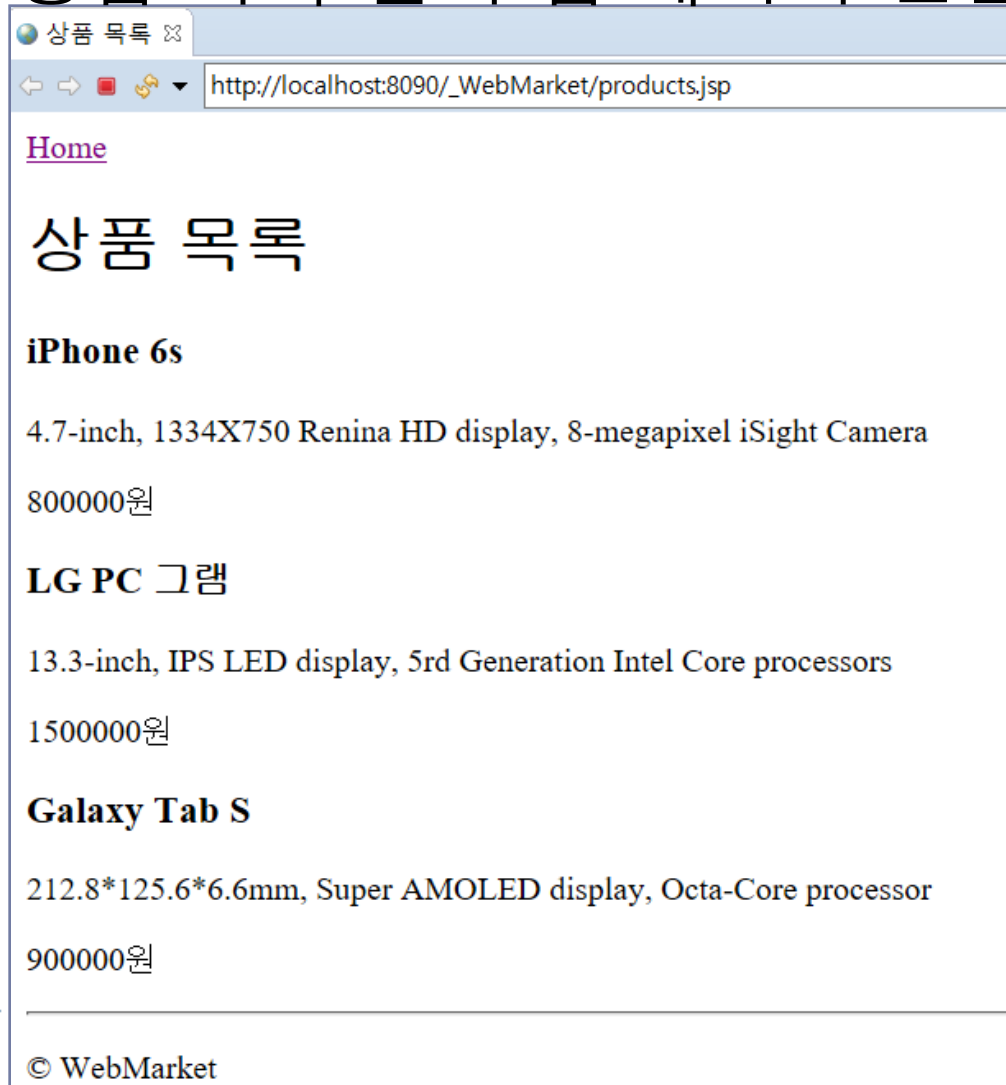
- ▶ 상품 목록을 가져오는 메소드 만들기

```
public class ProductRepository {  
    public ArrayList<Product> getAllProducts()  
    {  
        return listOfProducts;  
    }  
}
```



# 상품 목록 표시하기

## ▶ 상품 목록 출력 웹 페이지 만들기(부트스트랩 적용 전)



# 상품 목록 표시하기

---

products.jsp

```
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <%@ page import="java.util.ArrayList"%>
3 <%@ page import="dto.Product"%>
4 <jsp:useBean id="productDAO" class="dao.ProductRepository"
5   scope="session" />
6 <html>
7 <head>
8 <title>상품 목록</title>
9 </head>
10 <body>
11   <jsp:include page="menu.jsp" />
12   <h1>상품 목록</h1>
```

## 상품 목록 표시하기(products.jsp)

```
13 <%
14     ArrayList<Product> listOfProducts = productDAO.getAllProducts();
15 %>
16 <%
17     for (int i = 0; i < listOfProducts.size(); i++) {
18         Product product = listOfProducts.get(i);
19     %>
20     <h3><%=product.getPname()%></h3>
21     <p><%=product.getDescription()%>
22     <p><%=product.getUnitPrice()%>원
23 <%
24     }
25 %>
26 <hr>
27 <jsp:include page="footer.jsp" />
28 </body>
29 </html>
```



# 상품 목록 표시하기

## ▶ 상품 목록 출력 웹 페이지 만들기(부트스트랩 적용 후)

```
products.jsp
1 <%@ page contentType="text/html; charset=utf-8"%>
2 <%@ page import="java.util.ArrayList"%>
3 <%@ page import="dto.Product"%>
4 <jsp:useBean id="productDAO" class="dao.ProductRepository" scope="session" />
5 <html>
6 <head>
7 <link rel="stylesheet"
8   href="https://maxcdn.bootstrapcdn.com/bootstrap/4.0.0/css/bootstrap.min.css">
9 <title>상품 목록</title>
10 </head>
11 <body>
12 <jsp:include page="menu.jsp" />
13 <div class="jumbotron">
14 <div class="container">
15 <h1 class="display-3">상품 목록</h1>
16 </div>
17 </div>
18 <%
19   ArrayList<Product> listOfProducts = productDAO.getAllProducts();
20 %>
```



# 상품 목록 표시하기

```
22= <div class="container">
23=     <div class="row" align="center">
24=         <%
25=             for (int i = 0; i < listOfProducts.size(); i++) {
26=                 Product product = listOfProducts.get(i);
27=             %>
28=             <div class="col-md-4">
29=                 <h3><%=product.getPname()%></h3>
30=                 <p><%=product.getDescription()%>
31=                 <p><%=product.getUnitPrice()%>원
32=             </div>
33=         <%
34=             }
35=         %>
36=     </div>
37=     <hr>
38= </div>
39= <jsp:include page="footer.jsp"/>
40 </body>
41 </html>
```