

08장. 표준액션 / Java Bean

표준 액션의 개요

1. 액션의 종류는 크게 둘로 나뉘어 진다.
2. 표준 액션(**standard action**)은 JSP 페이지에서 바로 사용할 수 있다.
 커스텀 액션(**custom action**)은 별도의 라이브러리를 설치해야만 사용할 수 있다.
2. 표준 액션과 커스텀 액션은 태그 안에 사용하는 접두어(**prefix**)가 다르기 때문에 외형상으로 쉽게 구분 할 수 있다. 표준 액션에는 모든 태그의 이름 앞에 jsp라는 접두어가 붙고, 커스텀 액션에는 그 밖의 접두어가 붙는다

```
<jsp:include page= "sub.jsp" />
```

표준 액션임을 표시하는 접두어

```
<c:set var= "cnt " value= "0" />
```

커스텀 액션 중 하나임을 표시하는 접두어

JSP 페이지의 모듈화에 사용되는 표준 액션

1. <jsp:include> 표준 액션의 사용 방법

- <jsp:include>는 JSP 페이지에서 다른 웹 자원(JSP 페이지, HTML 문서 등)을 포함시키고자 할 때 사용하는 표준 액션이다.
- 이 표준 액션에는 포함할 웹 자원의 URL을 지정하는 page 애트리뷰트를 써야 한다.

```
<jsp:include page= "Copyright.html" />
```


↑
Copyright.html을 include하는 표준 액션

3. 액션 태그는 XML 문법을 따르므로 단독 태그일 경우에는 위와 같이 '/>'로 끝나도록 만들어야 한다.
4. JSP 페이지 안에 위와 같은 액션 태그가 있으면, 웹 컨테이너는 JSP 페이지를 처리할 때 이 태그의 위치에 Copyright.html의 내용을 대신 출력한다

<jsp:include> 표준 액션의 사용 방법

BookInfo.jsp

```
<%@page contentType= "text/html; charset=euc-kr "%>
<HTML>
<HEAD><TITLE>책 소개</TITLE></HEAD>
<BODY>
<H3>책 소개</H3>
제목: Java 프로그래밍 <BR>
저자: 홍길동 <BR>
페이지수: 908 <BR><BR>
<b><jsp:include page= "Copyright.html "/>
</BODY>
</HTML>
```



Copyright.html 문서를 include합니다

Copyright.html

```
<font size=2>@홍길동과 춘향이는 어울리나 ?</font>
```

예시문 참조

includeTest.jsp

includeTest2.jsp

<jsp:forward> 표준 액션의 사용 방법

1. <jsp:forward>는 JSP 페이지에서 다른 JSP 페이지로 제어를 넘기고자 할 때 사용하는 표준 액션이다.
2. <jsp:include>와 마찬가지로 page 애트리뷰트를 이용해서 해당 JSP 페이지의 URL을 지정해야 한다

```
<jsp:forward page= "Next.jsp" />
```

Next.jsp로 실행의 제어를 넘기는 표준 액션

<jsp:forward> 표준 액션의 사용 예시

Hundred.jsp

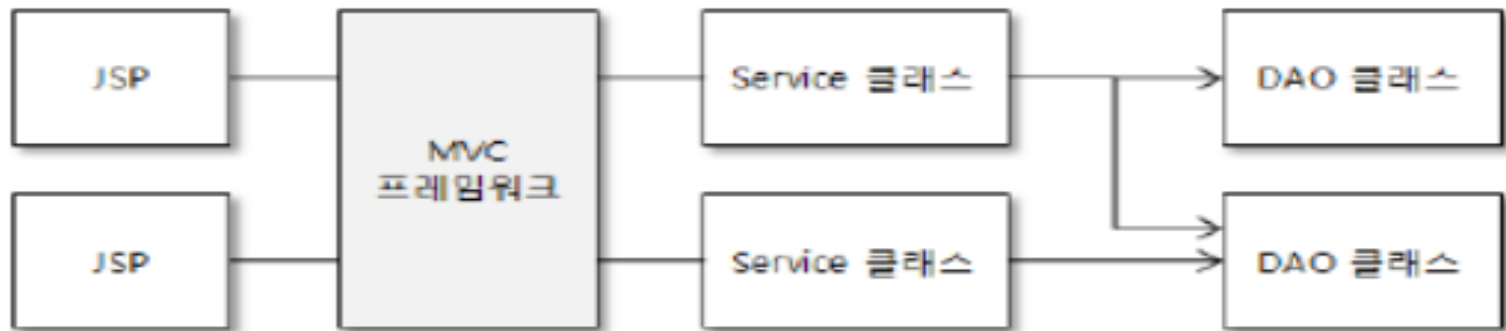
```
<%  
    int sum = 0;  
    for (int cnt = 1; cnt <= 100; cnt++)  
        sum += cnt;  
    request.setAttribute( "RESULT ", new Integer(sum));  
%>  
<jsp:forward page= "HundredResult.jsp " />
```

실행제어
넘김

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

참조: forward.html
foward2.jsp
nameResult.jsp

일반적인 웹 애플리케이션 구조

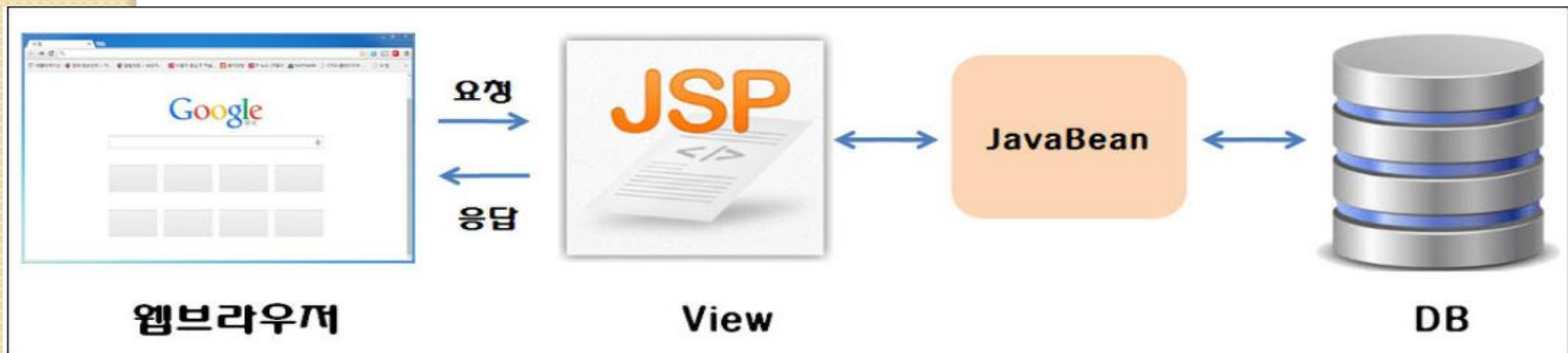


1. Service 클래스 - 사용자의 요청을 처리하는 기능을 제공하는 클래스로 DAO 클래스를 이용해서 DB 연동을 처리
2. DAO 클래스 - DB와 관련된 CRUD 작업을 처리(SQL 쿼리를 실행)
3. JSP(뷰) - Service 클래스가 실행한 결과를 화면에 출력해주거나 Service가 기능을 수행하는 데 필요한 데이터를 전달
4. MVC 프레임워크 - 사용자의 요청을 Service에 전달하고 Service의 실행 결과를 JSP와 같은 뷰에 전달을 수행하는 것으로 스프링 MVC나 스트러츠와 같은 프레임워크가 MVC 프레임워크에 해당하며 서블릿으로 직접 구현 가능
5. Service 클래스나 Dao 클래스의 경우 특별한 경우가 아니면 동시에 여러 개의 객체를 생성해서 사용 하는 경우가 없기 때문에 Singleton 패턴을 이용하는 것이 좋습니다.

자바빈(JavaBean)의 개요

1. 현재까지 작성해온 JSP 페이지의 문제점

- 1) 첫 번째로 JSP 페이지에 화면표출부분과 로직들이 혼재함으로 해서 JSP 페이지를 이해하기 어려워진다는 점
디자인과 협업이 어려워짐
- 2) 두 번째로는 JSP 페이지에 화면표출부분과 로직들이 혼재한 형태의 코드는 재사용하기가 어려움
 - 반복적인 일을 피하기 위해서는 JSP 페이지 내에 있는 반복적인 코드를 따로 작성하여 재사용할 필요가 있음
- 3) JSP 페이지와 로직의 분리해서 로직을 모듈화하는 작업이 필요 => 자바빈의 사용이 필요
- 4) JSP 페이지의 주용 기능 중 하나는 데이터를 보여주는 기능.
흔하게 볼 수 있는 게시판 예로 들면 글 목록 보기, 글쓰기, 글 읽기 등의 기능이 이에 해당.
그런데 이런 데이터를 보여주는 기능과 단순히 화면을 출력하는 부분이 하나의 JSP에 뒤섞여 있으면 문제가 생김. 기능을 확장하거나 코드를 재사용하기가 상당히 어려움.
그렇기에 JSP에서는 데이터를 자바빈(JavaBean)이라는 클래스에 담아서 값을 보여줌



자바빈(JavaBean)의 개요

❖ 이제부터 우리가 작성할 형태



2. 자바빈 사용하는 목적

- 1) JSP 페이지의 로직 부분을 분리해서 코드를 재사용함으로써 프로그램의 효율을 높이는 것
- 2) 현재 모든 프로그래밍에서 모듈화(컴포넌트(component)화)가 대세
 - 프로그램의 모듈화는 한 번 잘 작성해 놓은 코드를 재사용하므로 프로그램의 작성기간이 단축되고, 이미 실시스템에 올렸던 코드를 사용하므로 코드의 안정성이 보장되어 유지 보수에도 좋음

자바빈(JavaBean)의 개요

1. 자바빈 작성

- 자바빈은 클래스이므로 기존의 자바 클래스를 작성하는 방법과 동일.
자바빈의 경우 데이터를 담을 프로퍼티(멤버변수)와 데이터를 가져오거나 세팅하는 기능을 하는 메서드로 구성
- 자바빈은 자바의 클래스이므로 자바의 클래스를 만드는 것과 같은 규칙을 갖는다
- 자바의 클래스를 만들 때는 포함될 패키지명(package문), 해당 클래스를 생성할 때 필요한 패키지명을 포함한 클래스명(import문) 이 필요(생략이 가능)
그러나 class문은 생략할 수 없다.
자바파일에는 1개이상의 클래스를 포함해야함

```
package ch08;
public class Person {
    private String name;
    private String gender;
    private int age;

    public void setName(String name) {
        this.name = name;
    }
    public String getName() {
        return name;
    }
    public String getGender() {
        return gender;
    }
    public void setGender(String gender)
    {
        this.gender = gender;
    }
    public int getAge() {
        return age;
    }
    public void setAge(int age) {
        this.age = age;
    }
}
```

자바빈(JavaBean) 만들기

1. 자바빈의 클래스를 선언

1) 접근제어자 [키워드] class 클래스명{ }

- 접근 제어자는 public, private, default(접근제어자가 없는 형태)가 올 수 있는데, 자바 빈을 작성할 때는 접근제어의 강도가 가장 약한 public을 주로 사용
- 웹에서는 불특정 다수의 접근을 허용해야 하기 때문에 누구나 접근할 수 있는 public을 사용. 키워드는 자바빈에서는 사용 안함.

2. 자바에서 클래스명은 첫 글자는 대문자로 시작하고 나머지는 소문자를 사용한다. 또한 단어가 여러개 합쳐질 때는 시작되는 단어의 시작은 대문자로 시작한다.

- public class UtilClass{ }

3. 자바빈의 프로퍼티(멤버변수)를 선언

- 프로퍼티(property)는 값을 저장하기 위한 필드로 접근제어자를 private로 선언해서 작성

- private String userid;

자바빈(JavaBean) 만들기

4. 자바빈의 메소드의 선언

- 프로퍼티에 접근하기 위한 getXxx(), setXxx()메소드는 접근제어자를 public로 선언해서 작성

```
public void setId(String id){ this. id=id; }  
public String getId(){ return id; }
```
- 데이터 저장소의 역할을 하는 프로퍼티에 값을 저장할때 setXxx()메소드를 사용
- 저장된 값을 사용할때 getXxx()메소드를 사용
- 이때 Xxx는 프로퍼티명으로 첫글자는 대문자로 작성. 하나의 프로퍼티당 하나의 setXxx()메소드와 getXxx()메소드가 존재

1)setXxx()메소드 작성방법

- 프로퍼티에 값을 저장해야 하므로 파라미터로부터 값을 받아오는 setId (String id)와 같은 형태 값을 저장만하는 메소드이므로 리턴타입이 void.
- this.id=id;
- 넘어온 id파라미터의 값을 id프로퍼티에 저장하는 부분으로 this.id가 프로퍼티이다. 프로퍼티명과 파라미터명이 같으면 프로그램에 혼란을 주기 때문에 프로퍼티명앞에는 this가 붙는다.
- this는 자기 자신의 클래스를 가리키는 레퍼런스

2) getXxx()메소드 작성방법

- 저장된 프로퍼티명을 사용하는 메소드이므로 getId()와 같이 파라미터가 필요없다.
- 저장된 값을 사용해야 하기 때문에 반드시 리턴타입을 기술.
- String getId()는 리턴타입이 String이라는 것.
- void이외의 리턴타입이 기술되면 해당 메소드의 마지막에 return문을 반드시 기술해야 한다. 기술 안하면 에러가 발생.

자바빈과 <jsp:useBean>액션태그의 연동

5. 자바빈을 사용하기 위한 3가지의 액션태그

- 자바빈 객체를 생성하기 위한 <jsp:useBean>액션태그
- 자바빈 객체의 프로퍼티값을 저장하기 위해 사용되는 <jsp:setProperty> 액션태그
- 자바빈 객체에서 저장된 프로퍼티값을 사용하기 위해 사용되는 <jsp:getProperty>액션태그

자바 빈 관련 액션태그	내용
<jsp:useBean id="..." class="..." scope="..." />	자바빈 객체를 생성.
<jsp:setProperty name="..." property="..." value="..." />	생성된 자바빈 객체에 프로퍼티 값을 저장.
<jsp:getProperty name="..." property="..." />	생성된 자바빈 객체에서 지정된 프로퍼티값을 가져옴.(사용함)

자바빈과 <jsp:useBean>액션태그의 연동

6. <jsp:useBean>액션태그

- 1) <jsp:useBean>액션태그는 자바빈 객체를 생성
- 2) 사용하는 방법
 - <jsp:useBean id="빈이름" class="자바빈 클래스 이름" scope="범위" />
- 3) <태그의 속성>
 - id : JSP페이지에서 자바빈 객체에 접근 할 때 사용하는 이름이다.
 - class : 패키지 이름을 포함한 자바빈 클래스의 완전한 이름을 입력
 - scope : 자바빈 객체가 저장될 영역을 지정.
 - page, request, session, application 중 하나를 값으로 갖는다. 기본값은 page
- 4) scope 속성을 사용할 경우 속성의 값에 따라 서로 다른 기본 객체
 - page : pageContext 기본 객체
 - request : request 기본 객체
 - session : session 기본 객체
 - application : application 기본 객체

<jsp:useBean>액션태그

1. <jsp:setProperty>액션태그

- 1) <jsp:setProperty>액션태그는 자바빈 객체의 프로퍼티 값을 저장하기 위해 사용된다.
- 2) 사용하는 방법은 다음과 같다.
 - ① <jsp:setProperty>액션태그의 속성에 대한 설명은 다음과 같다.
 - ② name속성은 자바빈 객체의 이름을 명시하는 곳이다. 필수 속성으로 생략이 불가능하다.
- 3) <jsp:setProperty name= "빈 이름" property="프로퍼티 이름" value="프로퍼티에 저장할 값 " />
- 4) property속성은 프로퍼티명을 기술하는 곳이다. 필수 속성으로 생략이 불가능하다.
value속성은 프로퍼티에 저장할 값을 기술하는 곳이다. 생략 가능하다.

2. <jsp:getProperty>액션태그

- 1) <jsp:getProperty>액션태그는 자바빈 객체에서 저장된 프로퍼티값을 사용하기 위해 사용.
- 2) 사용하는 방법은 다음과 같다.
- 3) name속성은 자바빈 객체의 이름을 명시하는 곳이다. 필수 속성으로 생략이 불가능.
- 4) property속성은 프로퍼티명을 기술하는 곳이다. 필수 속성으로 생략이 불가능.
- 5) <jsp:getProperty name= "빈 이름" property="프로퍼티 이름" />

자바빈의 호출에 사용되는 표준 액션 사용예시 1

PersonallInfo.java

```
package mall;
public class PersonallInfo {
    private String name; // 이름
    private char gender; // 성별
    private int age; // 나이

    public void setName(String name) { this.name = name; }
    public String getName() { return name; }
    public void setGender(char gender) { this.gender = gender; }
    public char getGender() { return gender; }
    public void setAge(int age) {
        this.age = age;
    }
    public int getAge() {
        return age;
    }
}
```

이 클래스에 있는 **get-메서드**와 **set-메서드**는 자바빈의 내부 데이터를 인고 쓰는 기능을 제공
이런 메서드를 통해 읽고 쓸 수 있는 데이터를 자바빈의 프로퍼티(property) 라고 함

자바빈의 호출에 사용되는 표준 액션 사용예시 2

1. 자바빈 관련 표준 액션의 기초 사용 방법

- 자바 프로그램에서 클래스를 사용하기 위해서는 클래스의 객체를 만들어야 하며, **JavaBean** 클래스의 경우도 마찬가지로

```
ProductInfo obj = new ProductInfo();
```

<jsp:useBean> 표준 액션을 이용하면 자바 코드를 작성하지 않고도 자바빈 객체를 만들 수 있다.

<jsp:useBean>에는 기본적으로 class와 id라는 두 개의 애트리뷰트를 써야 한다

```
<jsp:useBean id= "obj " class= "mall.PersonalInfo " />
```

2. 자바빈 객체를 만든 다음에는 set-메서드를 이용해서 객체의 프로퍼티 값을 설정할 수 있다.



obj.setAge(27);

The diagram shows the code `obj.setAge(27);` with red arrows pointing to its components: `obj` is labeled '변수 이름' (variable name), `.setAge` is labeled '메서드 이름' (method name), and `27` is labeled '프로퍼티 값' (property value).

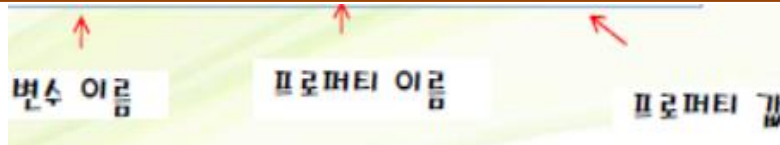
변수 이름 메서드 이름 프로퍼티 값

3. <jsp:setProperty> 표준 액션을 이용하면 set-메서드를 직접 호출하지 않고도 프로퍼티 값을 설정할 수 있다.

4. 이 액션에는 name, property, value라는 세 개의 애트리뷰트를 써야 한다.

이 중 name 애트리뷰트에는 자바빈 객체가 들어 있는 변수 이름을 지정해야 하고, property와 value 애트리뷰트에는 각각 프로퍼티의 이름과 값을 지정해야 한다.

```
<jsp:setProperty name= "obj " property= "age " value= "27 " />
```



The diagram shows the JSP tag `<jsp:setProperty name= "obj " property= "age " value= "27 " />` with red arrows pointing to its attributes: `name= "obj "` is labeled '변수 이름' (variable name), `property= "age "` is labeled '프로퍼티 이름' (property name), and `value= "27 "` is labeled '프로퍼티 값' (property value).

변수 이름 프로퍼티 이름 프로퍼티 값

자바빈의 호출에 사용되는 표준 액션 사용예시 3

5.get-메서드

- 자바빈 객체의 프로퍼티 값을 가져올려면 다음과 같이 **get-메서드**를 이용하면 됨

```
int age = obj.getAge();
```

변수 이름

메서드 이름

6.<jsp:getProperty> 표준 액션을 사용하면 get-메서드를 직접 호출하지 않고도 프로퍼티 값을 가져올 수 있다.

이 액션에는 name과 property라는 두 개의 애트리뷰트를 써야 한다.

name 애트리뷰트에는 자바빈 객체가 들어 있는 변수의 이름을 써야 하고, property 애트리뷰트에는 프로퍼티의 이름을 써야 한다.

```
<jsp:getProperty name="obj" property="age" />
```

변수 이름

프로퍼티 이름

자바빈 관련 표준 액션의 기초 사용 방법

CustomerInfo.jsp

```
<%@page contentType= "text/html; charset=euc-kr "%>
<HTML>
  <HEAD> <TITLE>회원 정보</TITLE> </HEAD>
  <BODY>
```

자바빈 객체를 생성합니다.

```
<jsp:useBean class= "mall.PersonallInfo " id= "pinfo " />
<jsp:setProperty name= "pinfo " property= "name " value= "김연희 " />
<jsp:setProperty name= "pinfo " property= "gender " value= "여 " />
<jsp:setProperty name= "pinfo " property= "age " value= "29 " />
이름: <jsp:getProperty name= "pinfo " property= "name " /> <BR>
성별: <jsp:getProperty name= "pinfo " property= "gender " /> <BR>
나이: <jsp:getProperty name= "pinfo " property= "age " />
```

```
</BODY>
```

```
</HTML>
```

자바빈 객체로부터 프로퍼티 값을 가져다가 출력합니다