JavaBeans & Form

524730-1
2020년 봄학기
4/15/2020
박경신

JavaBeans

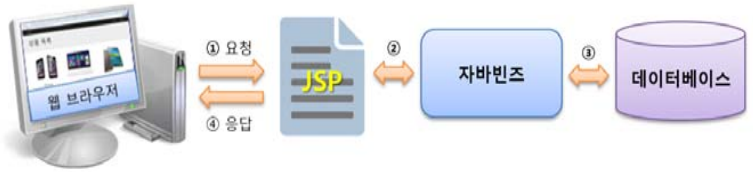
- 자바 빈즈
  - 자바로 작성된 소프트웨어 컴포넌트
  - Sun Microsystems 에서 "자바 빈즈는 빌더 형식의 개발도구에서 가시적으로 조작이 가능하고 또한 재사용이 가능한 소프트웨어 컴포넌트이다."로 정의
  - 자바빈즈는 엔터프라이즈 자바빈즈(EJB)와 많은 유사성이 있으나 혼동하지 말아야 함
  - **<jsp:useBean> 액션태그를 통해 JSP에서 손쉽게 연동**
  - 웹 프로그래밍에서 데이터 표현을 목적으로 사용

EJB

- 엔터프라이즈 자바 빈즈 (EJB)
  - J2EE(Java2 Enterprise Edition)에서의 컴포넌트 모델 (서버측 컴포넌트)
  - 분산 환경을 고려함
  - J2EE 자체는 스펙으로 여러 회사에서 제품을 구현함
  - 기업형 애플리케이션 개발에서 비즈니스 로직 및 데이터 처리를 담당함

JavaBeans

- 자바 빈즈
  - 자바 클래스
  - 멤버 변수와 getter, setter 메서드로 구성됨
  - 특정 기능을 수행하는 별도의 메서드를 가질수 있음.
  - JSP 모델 2에서는 뷰 컴포넌트로만 활용이 권장됨(비즈니스 로직 X)



JavaBeans 작성 규칙

- JavaBeans 클래스 작성 규칙
  - **자바 클래스는 java.io.Serializable 인터페이스를 구현해야 함**
  - **인자가 없는 기본 생성자가 있어야 함**
  - **모든 멤버 변수는 private 접근 지정자로 설정해야 함**
  - **모든 멤버 변수에 대한 getter/setter() 메소드가 존재해야 함**
    - getter() 메소드는 멤버 변수에 저장된 값을 가져올 수 있는 메소드
    - setter() 메소드는 멤버 변수에 값을 저 장할 수 있는 메소드

JavaBeans 클래스

- 자바빈즈 클래스 구성

```
class PersonBean implements java.io.Serializable {
    // 멤버 변수, HTML form 이름, DB 테이블의 컬럼명과 일치
    private String name;
    private int age;

    public PersonBean() { } // 기본 생성자
    public PersonBean(String name, int age) { ..... } // 생성자

    // getter, setter 메서드, 멤버 변수와 일치
    public String getName() {       return name;       }
    public void setName(String name) {      this.name = name;       }
    public int getAge() {       return age;       }
    public void setAge(int age) {      this.age = age;       }
}
```

JavaBeans 클래스

- 자바 빈즈 클래스 구성
  - 멤버변수
    - 클래스외부에서의 접근을 막기 위해 private 로 선언
    - 멤버변수이름, HTML form 이름, DB테이블 컬럼명과 일치하게 작성함
  - getter 메서드
    - 멤버변수의 값을 반환
    - 멤버변수명의 첫글자를 대문자로 해야함(Bean 액션과 연동).
      <jsp:getProperty name="person" property="name" />
    - 내부적으로 **getName()** 메서드를 호출
  - setter 메서드
    - 멤버변수의 값을 설정
    - 멤버변수명의 첫글자를 대문자로 해야함.
      <jsp:setProperty name="person" property="name" value="Park"/>
    - 내부적으로 **setName(request.getParameter("name"))**과 함께 호출

useBean 액션태그

- useBean 액션태그
  - 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="자바빈즈식별자" scope="범위" class="자바빈즈이름"/>

속성	설명
id	자바 빈즈를 식별하기 위한 이름
class	패키지 이름을 포함한 자바 빈즈의 이름
scope	자바 빈즈가 저장되는 영역을 설정 page (기본값), request, session, application 중 하나

useBean 액션태그

- useBean 액션태그 사용 예

```
<jsp:useBean id="person" scope="request" class="dto.PersonBean"/>
```

- **useBean 액션태그 자바 코드 변환 시**

```
PersonBean person = (PersonBean)request.getAttribute("person");
if(person == null) {
    person = new PersonBean();
    request.setAttribute("person", person);
}
```

- scope
  - page : 현재 JSP 페이지 내
  - request : request가 최종 포워딩되는 페이지까지
  - session : 세션을 유지할 때까지
  - application : 웹 애플리케이션이 종료될 때까지

setProperty 액션태그

- **setProperty 액션태그**
  - useBean 액션 태그와 함께 자바빈즈의 setter() 메소드에 접근하여 자바빈즈의 멤버 변수인 프로퍼티의 값을 저장

<jsp:setProperty name="자바빈즈식별자" property= " 프로퍼티이름" value="프로퍼티값" />

  - name - 자바빈 객체의 이름
  - property - 값을 설정할 프로퍼티
  - value - 프로퍼티의 값

<jsp:setProperty name="자바빈즈식별자" property="프로퍼티이름" param="파라미터이름" />

  - param - 프로퍼티의 값으로 사용할 파라미터 이름.

<jsp:setProperty name="자바빈즈식별자 " property="*" />

  - 프로퍼티와 동일한 이름의 파라미터를 이용해서 값을 설정
  - 폼에 입력한 값을 자바 객체에 저장할 때 유용하게 사용

프로퍼티 타입에 따른 값 매핑

프로퍼티의 타입	변환 방법	기본 값
boolean 또는 Boolean	Boolean.valueOf(String)을 값으로 갖는다.	false
byte 또는 Byte	Byte.valueOf(String)을 값으로 갖는다.	(byte) 0
short 또는 Short	Short.valueOf(String)을 값으로 갖는다.	(short) 0
char 또는 Character	입력한 값의 첫 번째 글자를 값으로 갖는다.	(char) 0
int 또는 Integer	Integer.valueOf(String)을 값으로 갖는다.	0
long 또는 Long	Long.valueOf(String)을 값으로 갖는다.	0L
double 또는 Double	Double.valueOf(String)을 값으로 갖는다.	0.0
float 또는 Float	Float.valueOf(String)을 값으로 갖는다.	0.0f

setProperty 액션태그

- setProperty 액션태그 사용 예

```
<jsp:setProperty  name="person"  property="name" value="Park" />
<jsp:setProperty  name="person"  property="age" value="10" />
<jsp:setProperty name="person" property="*" />
```

- setProperty 액션을 스크립트릿으로 대체

```
<%
    person.setName(request.getParameter("name"));
    person.setAge(request.getParameter("age"));
%>
```

setProperty 액션태그

- 폼 페이지로부터 전달되는 요청 파라미터의 값을 자바빈즈의 프로퍼티로 변경하여 값을 저장할 수 있음
  - 요청 파라미터 이름과 자바빈즈의 프로퍼티 이름이 일치하는 경우:

```
<form action="personProcess.jsp" method="post">
   <input name="age" property="10" />
</form>
```

```
<jsp:setProperty name="person" property="age" />
```

setProperty 액션태그

- 요청 파라미터 이름과 자바빈즈의 프로퍼티 이름이 일치하지 않는 경우:

```
<form action="personProcess.jsp" method="post">
   <input name="userAge" property="10" />
</form>
```

```
<jsp:setProperty name="person" property="age" param="userAge" />
```

- 요청 파라미터 이름과 자바빈즈의 프로퍼티 이름이 모두 일치하는 경우:

```
<form action="personProcess.jsp" method="post">
  <input name="name" property="Park" />
  <input name="age" property="10" />
</form>
```

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

getProperty 액션태그

- **getProperty 액션 태그**
  - 프로퍼티의 값을 가져오기위해 사용

<jsp:getProperty name=" 자바빈즈식별자 " property=" 프로퍼티이름" />

  - 사용 예시

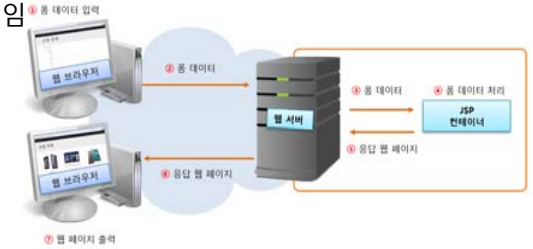
```
이름: <jsp:getProperty  name="person"  property="name" />
나이: <jsp:getProperty  name="person" property="age" />
```

  - getProperty 액션을 **표현식으로 대체**

```
이름: <%= person.getName( ) %>
나이: <%= person.getAge( ) %>
```

Form 처리

- 폼(form)
  - 사용자가 웹 브라우저를 통해 입력된 모든 데이터를 한 번에 웹 서버로 전송하는 양식
  - 전송한 데이터는 웹 서버가 처리하고 처리 결과에 따라 다른 웹 페이지를 보여줌
  - 사용자와 웹 애플리케이션이 상호 작용하는 중요한 기술 중 하나임
  - 사용자가 어떤 내용을 원하는지, 사용자의 요구 사항이 무엇인지 파악할 때 가장 많이 사용하는 웹 애플리케이션의 필수적인 요소임



Form 태그

- 폼(form)을 구성하는 태그 종류

태그	설명
form	폼을 정의하는 태그로 최상위 태그
input	사용자가 입력할 수 있는 태그
select	항목을 선택할 수 있는 태그
textarea	여러 줄을 입력할 수 있는 태그

Form 태그

- form 태그
  - 사용자가 다양한 정보를 입력하고 서로 전달할 때 사용하는 태그
  - 단독으로 쓰이지 않고 사용자가 다양한 정보를 입력하는 양식을 포함하는 최상위 태그
  - 속성을 이용하여 폼 데이터를 전송할 때 어디로 보낼지, 어떤 방식으로 보낼지 설정

```
<form attribute1="value1" [attribute2="value2" ...]>
    // 다양한 입력 양식 태그 <input>, <select>, <textarea>
</form>
```

Form 태그

- form 태그의 속성
  - form 태그의 모든 속성은 필수가 아니라 선택적으로 사용

```
<form action="/nameAction.jsp" method="post">
    <input type="text" name="name" value="K">
</form>
```

속성	설명
action	**폼 데이터를 받아 처리하는 웹페이지의 URL 설정**
method	**폼 데이터가 전송되는 HTTP 방식 설정**
name	폼을 식별하기 위한 이름 설정
target	폼 처리 결과의 응답을 실행할 프레임 설정
enctype	폼을 전송하는 콘텐츠 MIME 유형 설정
accept-charset	폼 전송에 사용할 문자 인코딩 설정

Form의 Input 태그

- Form의 input 태그
  - 사용자가 텍스트 입력이나 선택 등을 다양하게 할 수 있도록 공간을 만드는 태그
  - 종료 태그 없이 단독으로 사용할 수 있음
  - input 태그의 기본 속성
    - type은 text, button, radio, checkbox, password, hidden, file, reset, submit 등 다양한 입력
    - name은 입력 양식을 식별하는 이름 설정
    - value는 입력 양식의 초기값 설정

```
<input type="text" name="firstname" value="K">
```

Form의 Input 태그 종류

- Form의 input type 종류
  - **<input type="button">** 버튼 선택 입력
  - **<input type="checkbox">** 체크박스 버튼 선택 입력
  - <input type="color">
  - <input type="date">
  - <input type="datetime-local">
  - <input type="email">
  - **<input type="file">** 파일 전송 입력
  - **<input type="hidden">** 보이지 않게 숨겨서 값을 전송
  - <input type="image">
  - <input type="month">
  - <input type="number">

Form의 Input 태그 종류

- Form의 input type 종류
  - **<input type="password">** 암호 입력
  - **<input type="radio">** 라디오 버튼 선택 입력
  - <input type="range">
  - **<input type="reset">** 폼에 입력된 값을 모두 초기화
  - <input type="search">
  - **<input type="submit">** 폼에 입력된 값을 모두 서버에 전송
  - <input type="tel">
  - **<input type="text">** 한 줄 텍스트 입력
  - <input type="time">
  - <input type="url">
  - <input type="week">

Form의 Select 태그

- Form의 select 태그
  - 여러 개의 항목이 나타나는 목록 상자에서 항목을 선택하는 태그
  - 시작 태그와 종료 태그가 있으며, 리스트 박스에 여러 항목을 추가 삽입하기 위해 반드시 option 태그를 포함해야 함.

```
<select name="itemselect">
   <option value="item1"> item1 </option>
   <option value="item2"> item2 </option>
   <option value="item3"> item3 </option>
   <option value="item4"> item4 </option>
</select>
```

Form의 Select 태그

- select 태그의 속성
  - name은 입력 양식을 식별하는 이름 설정
  - size는 한 번에 표시할 항목의 개수를 설정
  - multiple은 다중 선택이 가능하도록 함. CTRL-key를 눌러 목록 상자의 항목을 다중 선택함
- option 태그의 속성
  - value는 항목의 값을 설정
  - selected는 해당 항목을 초기값으로 선택
  - disabled는 항목을 비활성화
- optgroup 태그의 속성
  - select 태그 내에 있는 option 태그들을 그룹화하는 데 사용

Form의 Textarea 태그

- Form의 textarea 태그
  - **여러 줄의 텍스트를 입력**할 수 있는 태그
  - 기본 값은<textarea>와 </textarea> 태그 사이에 설정
  - 입력 폼 안에 사용된 태그와 띄어쓰기가 그대로 출력됨

```
<textarea name="이름" cols="너비" rows="높이">
    ... // 생략
</textarea>
```

속성	속성값	설명
name	텍스트	텍스트영역의 이름 설정
cols	**숫자**	**입력할 텍스트영역의 너비(열 크기) 설정**
rows	**숫자**	**입력할 텍스트영역의 높이(행 크기) 설정**
wrap	off	줄 바꿈 안함
	soft	엔터키를 누르지 않아도 끝에서 자동으로 행이 바뀜
	hard	soft와 비슷하며 서버에 전송할 때 캐리지 리턴 문자를 전달

Form & Input 태그


```
<form action="process.jsp" name="member" method="post">
   <p>아이디: <input type="text" name="id">
   <input type="button" value="아이디 중복 검사">
   <p>비밀번호: <input type="password" name="passwd">
   <p>이름: <input type="text" name="name">
   <p>연락처: <input type="text" maxlength="4" name="phone1">
              - <input type="text" maxlength="4" name="phone2">
              - <input type="text" maxlength="4" name="phone3">
   <p>성별: <input type="radio" name="gender" value="MEN" checked>남성
            <input type="radio" name="gender" value="WOMEN">여성
   <p>취미: 독서<input type="checkbox" name="hobby" value="독서" checked>
            운동<input type="checkbox" name="hobby" value="운동">
            영화<input type="checkbox" name="hobby" value="영화">
   <p><input type="submit" value="가입하기">
      <input type="reset" value="다시쓰기">
</form>
```

Form & Input 태그


```
<form action="process.jsp" name="member" method="post">
   // 중간 생략..
   <p>이름: <input type="text" name="name">
    <p>연락처: <select name="phone1">
                 <option value="010">010</option>
                 <option value="011">011</option>
                 <option value="016">016</option>
               </select>
              - <input type="text" maxlength="4" name="phone2">
              - <input type="text" maxlength="4" name="phone3">
   <p>성별: <input type="radio" name="gender" value="MEN" checked>남성
            <input type="radio" name="gender" value="WOMEN">여성
    // 중간 생략..
   <p><input type="submit" value="가입하기">
      <input type="reset" value="다시쓰기">
</form>
```

Form & Input 태그


```
<form action="process.jsp" name="member" method="post">
   // 중간 생략..
   <p>성별: <input type="radio" name="gender" value="MEN" checked>남성
            <input type="radio" name="gender" value="WOMEN">여성
   <p>취미: 독서<input type="checkbox" name="hobby" value="독서" checked>
            운동<input type="checkbox" name="hobby" value="운동">
            영화<input type="checkbox" name="hobby" value="영화">
   <p><textarea name="comment" cols="30" rows="3"
       placeholder="가입인사를 입력해주세요"></textarea>
   <p><input type="submit" value="가입하기">
      <input type="reset" value="다시쓰기">
</form>
```

Form 데이터 처리하기

- 요청 파라미터의 값 받기
  - request 내장 객체는 웹 브라우저가 서버로 보낸 요청에 대한 다양한 정보를 담고 있어 **getParameter()** 메소드를 이용하여 요청 파라미터의 값을 얻을 수 있음

```
String 변수 = request.getParameter("요청 파라미터 이름");
```

Form 데이터 처리하기

- 요청 파라미터의 전체 값 받기
  - 요청 파라미터를 설정하지 않아도 모든 값을 전달받을 수 있음. 또한 텍스트 박스, 라디오 버튼, 드롭다운 박스와 같은 다양한 유형에 대해 한 번에 폼 데이터를 전달받을 수 있음.
  - 폼 데이터의 일괄 처리 메소드

메소드	설명
getParameterNames()	모든 입력 양식의 요청 파라미터 이름을 순서에 상관없이 Enumeration 형태로 전달받음
hasMoreElements()	Enumeration 요소가 있으면 true를 반환하고 그렇지 않으면 false를 반환
nextElement()	Enumeration 요소를 반환

Form 데이터 처리하기

```
<% request.setCharacterEncoding("UTF-8");
<p>아이디: <%= request.getParameter("id") %>
<p>비밀번호: <%= request.getParameter("passwd") %>
<p>이름: <%= request.getParameter("name") %>
<p>연락처: <%= request.getParameter("phone1") %>
          - <%= request.getParameter("phone2") %>
          - <%= request.getParameter("phone3") %>
<p>성별: <%= request.getParameter("gender") %>
<p>취미: <%
request.setCharacterEncoding("UTF-8");
String[] hobby = request.getParameterValues("hobby");
if (hobby != null) {
   for (int i = 0; i < hobby.length; i++) {
      out.println(" " + hobby[i]);
   }
} %>
<p>가입인사: <%= request.getParameter("comment") %>
```

Form 데이터 처리하기

```
<table border="1">
   <tr>
        <th>요청 파라미터 이름</th>
        <th>요청 파라미터 값</th>
   </tr>
   <%
   request.setCharacterEncoding("UTF-8");
   Enumeration<String> paramNames = request.getParameterNames();
   while (paramNames.hasMoreElements()) {
      String paramName = (String) paramNames.nextElement();
      out.print("<tr><td>" + paramName + " </td>\n");
      String paramValue = request.getParameter(paramName);
      out.println("<td> " + paramValue + "</td></tr>\n");
   }
   %>
</table>
```