Java Web Programming 입문

(Servlet/JSP)

26일차

오늘의 키워드

- 액션 태그 (Action Tag)
  - Action Tag
  - Java Beans
  - <jsp:useBean>
  - <jsp:setProperty>
  - <jsp:getProperty>
  - <jsp:forward>
  - <jsp:include>
  - <jsp:param>

액션 태그 (Action Tag)

- 액션 태그(Action Tag)
  - 는?
    - JSP Page에 Java Code 삽입
      - <% ~ %>, <%! ~ %>, <%= %>, <%@ ~ %> 이용
    - JSP에서 사용되는 몇몇 기능을
      - XML 기반 Tag로 정의
      - 실행 시 Java Code로 자동 변환
  - 목적
    - JSP Code를 더 빠르고 쉽게 작성하기 위해
      - for Developer

액션 태그 (Action Tag)

- 액션 태그(Action Tag)
  - 종류
    - JSP Page에서 Java Beans 사용

Java 컴포넌트 파일	기능
ConnectionPool.java	데이터베이스 연결을 여러 개 열어두고 관리하여 데이터베이스 연결의 효율을 높임
ConnectionFactory.java	데이터베이스의 종류에 따른 커넥션 연결을 구현해 놓은 컴포넌트
BoardData.java	게시판의 여러 정보를 묶어서 캡슐화된 형태로 사용할 수 있게 함
AdminManager.java	게시판 관리를 위한 컴포넌트로써 새로운 게시판의 생성, 수정, 삭제 및 데이터베이스에 연결해서 질의를 수행하는 역할을 함
BoardManager.java	일반 사용자를 위한 컴포넌트로써 AdminManager 클래스를 상속받고 게시판 글의 읽기, 쓰기, 수정, 삭제, 페이징 등 일반적인 게시판의 기능이 구현됨

    - JSP Page에서 실행제어 관련
      - <jsp:include>
      - <jsp:forward>
      - <jsp:param>

액션 태그 (Action Tag)

- JavaBeans
  - Web Application에서
    - Business Logic을 가지고 있는
      - 어떤 Data의 처리를 담당하는 Class
  - Code적 측면에서 본다면
    - 어떤 Object에 대해
      - Member Variable
      - Member Variable에 값을 넣고 빼는 Member Method
      - 로 이루어진 Class
  - 회원 Java Beans 라면
    - Member Variable
      - 회원의 이름, 나이, 주민등록번호, 직업 등등
    - Member Method
      - 각 Member Variable에 값을 넣고 추출할 수 있는 Method
      - getter(), setter

액션 태그 (Action Tag)

- JavaBeans
  - 결국은 Java 파일
  - 구성
    - Constructor
    - Member Variable
    - Data를 저장하기 위한 set~ 형태의 Member Method
      - setter()
    - Data를 추출하기 위한 get~ 형태의 Member Method
      - getter()

액션 태그 (Action Tag)

- JavaBeans
  - 만들기
    - Package
      - Java Beans는 보통 관련된 녀석들끼리 묶어서 사용

```
package actiontag;
```

    - Class
      - Class와 Constructor 구현
        - Constructor가 필요없으면 만들지 않아도 된다

```
public class ActionTagTest {
    public ActionTagTest() {
        // . . .
    }
    // . . .
}
```

    - Member Variable
      - 일반적으로 소문자로 시작
      - 외부접근을 허용하지 않도록 private

```
private String str = "";
```

액션 태그 (Action Tag)

- JavaBeans
  - 만들기
    - Member Method
      - Java Beans는 Class가 가지고 있는 Member Variable에
        - 값을 넣거나 추출하기 위한 Member Method를 포함
      - 외부 접근이 가능하도록 public로 선언
      - setter : set~ Member Method
        - Member Variable에 값을 집어넣는 Method
        - set + Member Variable Name 형태
        - 흔히 setter 라고 지칭

```
public void setStr(String str) {
    this.str = str;
}
```

      - getter : get~ Member Method
        - Member Variable로부터 값을 추출하는 Method
        - get + Member Variable Name 형태
        - 흔히 getter 라고 지칭

```
public String getStr() {
    return str;
}
```

액션 태그 (Action Tag)

- JavaBeans
  - ActionTagTest_01.java

```java
package actiontag;

public class ActionTagTest {

    private String str = "";
    private String id = "";

    public ActionTagTest() { }

    public void setStr(String str) {
        this.str = str;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getStr() {
        return str;
    }
    public String getContent() {
        return id + " : " + str;
    }
}
```

액션 태그 (Action Tag)

- Scriptlet 으로 Java Beans 사용
  - 어떻게?
    - Scriptlet 내부에 Object를 만든 후 사용
  - Java Beans Class를 Import

```
<%@page import="actiontag.*"%>
```

  - Java Beans Object 생성

```
ActionTagTest acT = new ActionTagTest();
```

  - Java Beans Object 사용

```
acT.setStr("testStr");
acT.setId("testID");
out.print(acT.getStr());
```

액션 태그 (Action Tag)

- Scriptlet 으로 Java Beans 사용
  - UseNewTest.jsp

```
<%@page import="actiontag.*"%>

<%@page contentType="text/html; charset=euc-kr"%>
<%@page import="actiontag.*"%>
<html>
    <head>
        <title>Hello JSP</title>
    </head>
    <body>
        <%
            ActionTagTest acT = new ActionTagTest();
            acT.setStr("testStr");
            acT.setId("testID");
            out.print("<h3>" + acT.getStr() + "</h3>");
        %>
    <h3><%= acT.getContent()%></h3>
    </body>
</html>
```

액션 태그 (Action Tag)

- <jsp:useBean> Action tag
  - Java Beans Object를 생성하기 위한 Action Tag
  - 사용방법

```
<jsp:useBean   id="객체 이름"   class="클래스 이름"   scope="유효 범위" />
```

  - id : Java Beans Object 이름
  - class : 생성할 Java Beans Class

```
actiontag.ActionTagTest acT = new actiontag.ActionTagTest();
```

```
<jsp:useBean   id="acT "   class= "actiontag.ActionTagTest" />
```

  - scope : 생성한 Java Beans Object의 유효범위
    - page
      - 기본 속성, 현재 JSP Page 내에서만 Object 사용가능
    - request
    - session
    - application
      - application 영역에서
        - id 속성에 지정된 이름과 같은 객체가 존재하는지 확인 후
        - 동일한 이름의 Object가 존재하지 않는다면 새 Object를 만든 후
          - application 영역에 저장

액션 태그 (Action Tag)

- <jsp:setProperty> Action Tag
  - Java Beans Object의 setter Method 호출
    - Member Variable 값 설정
  - 사용

```
<jsp:setProperty name="객체 이름"  property="멤버 변수 이름"  value="값" />
```

    - name : 생성된 Java Beans Object의 이름
    - property : Member Variable 이름
    - value : Member Variable 에 저장될 값

```
acT.setStr("Hello");
```

```
<jsp:setProperty name="acT" property="str" value="Hello" />
```

액션 태그 (Action Tag)

- <jsp:getProperty> Action Tag
  - Java Beans Object의 getter Method 호출
    - Member Variable에 할당된 값을 추출
  - 사용

```
<jsp:getProperty    name="객체 이름"    property="멤버 변수 이름" />
```

    - name : Java Beans의 Object
    - property : 값을 추출하려는 Member Variable

```
out.print(acT.getStr());
```

```
<jsp:getProperty    name="acT"    property="str" />
```

액션 태그 (Action Tag)

- Action Tag로 Java Beans 이용
  - UseTagTest.jsp

```
<%@page contentType="text/html; charset=euc-kr"%>

<html>
    <body>
        <jsp:useBean id="acT" class="actiontag.ActionTagTest" />
        <jsp:setProperty name="acT" property="str" value="Hello"/>
        <jsp:setProperty name="acT" property="id" value="JSP"/>
        <h3><jsp:getProperty name="acT" property="str"/></h3>
        <h3><jsp:getProperty name="acT" property="content"/></h3>
    </body>
</html>
```

  - 결과는 동일
    - JSP Page에서 작성된 Action Tag는
      - Servlet에서 동일한 Java Code로 변환되기 때문
    - 어떤 방법을 이용하든 결과는 동일
    - Action Tag와 JSP를 혼합하여 사용해도 동일하게 동작

액션 태그 (Action Tag)

- JSP VS Action Tag
  - Java Beans Object 생성

```
<jsp:useBean id="acT" class="actiontag.ActionTagTest" />
```

```
<%
    actiontag.ActionTagTest acT = new actiontag.ActionTagTest();
%>
```

  - Member Variable에 값 할당

```
<jsp:setProperty name="acT" property="str" value="Hello"/>
<jsp:setProperty name="acT" property="id" value="JSP"/>
```

```
<%
    acT.setStr("JSP");
    acT.setId("jabook");
%>
```

  - Member Variable 값 추출

```
<jsp:getProperty name="acT" property="str"/>
<jsp:getProperty name="acT" property="content"/>
```

```
<%
    out.print(acT.getStr());
%>
<%= acT.getContent()%>
```

액션 태그 (Action Tag)

- Parameter로부터 Java Beans Object 생성
  - Java Beans에 Constructor 추가
    - Java Code를 이용해서 Java Beans를 초기화하면
      - Java Beans Class에
        - 여러 가지 Constructor을 이용하여 초기화 할 수 있음
    - Constructor에서 사용되는
      - Parameter의 유무에 구애 받지 않고
        - 입력된 Parameter를 이용하여
        - 초기화된 Java Beans Object 생성 가능

액션 태그 (Action Tag)

- ActionTagTest_02.java

```
package actiontag;

public class ActionTagTest {

    private String str = "";
    private String id = "";

    public ActionTagTest() {     }

    public ActionTagTest(String id, String str) {
        this.id = id;
        this.str = str;
    }

    public void setStr(String str) {
        this.str = str;
    }
    public void setId(String id) {
        this.id = id;
    }
    public String getStr() {
        return str;
    }
    public String getContent() {
        return id + " : " + str;
    }
}
```

액션 태그 (Action Tag)

- Scriptlet 에서 Java Beans 초기화
  - Client로부터 전달된
    - id, str 값을 지역 변수에 저장한 후
      - Java Beans Class에 Constructor의 Parameter로 사용하여
        - Java Beans Object를 생성

```
<%
    String id = request.getParameter("id");
    String str = request.getParameter("str");
    ActionTagTest acT = new ActionTagTest(id, str);
%>
```

액션 태그 (Action Tag)

- Scriptlet 에서 Java Beans 초기화
  - ParameterNewTest.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>
<%@page import="actiontag.*"%>

<html>
    <head>
        <title>Hello JSP</title>
    </head>
    <body>
        <%
            String id = request.getParameter("id");
            String str = request.getParameter("str");
            ActionTagTest acT = new ActionTagTest(id, str);
            out.print("<h3>" + acT.getStr() + "</h3>");
        %>
        <h3><%= acT.getContent()%></h3>
    </body>
</html>
```

  - 실행
    - http://localhost/MySample/ParameterNewTest.jsp?id=testId&str=testStr

액션 태그 (Action Tag)

- Action Tag를 사용한 Java Beans 초기화
  - Java Code 이용할 때와는 달리
    - Action Tag에서는 반드시
      - Parameter가 없는 Constructor만을 사용해야
        - Java Beans Object 생성이 가능
    - Parameter가 있는 Constructor를 이용해서
      - Java Beans를 초기화하는 것은 불가능
    - 그래서…
      - <jsp:useBean> Action Tag 내에서
        - Java Beans Object 초기화를 위한 Code를 삽입하는 다른 방법 제공
        - Constructor 처럼 Java Beans Object가
          - 최초에 생성될 때 한번만 수행

```
<jsp:useBean id="acT" class="actiontag.ActionTagTest">
    <jsp:setProperty name="acT" property="*" />
</jsp:useBean>
```

        - property 속성 값이 “*”면’
          - Client로 전송되어온 Data 이름을 검사
          - 자동으로 setter을 사용하여 Member Variable 에 값을 할당
            - 일치디는 Method가 없어도 Error가 발생하지 않음

액션 태그 (Action Tag)

- Action Tag를 사용한 Java Beans 초기화
  - ParameterTagTest.jsp

```
<%@page contentType="text/html; charset=euc-kr"%>

<html>
    <body>
        <jsp:useBean id="acT" class="actiontag.ActionTagTest">
            <jsp:setProperty name="acT" property="*"/>
        </jsp:useBean>
        <h3><jsp:getProperty name="acT" property="str"/></h3>
        <h3><jsp:getProperty name="acT" property="content"/></h3>
    </body>
</html>
```

  - http://localhost/MySample/ParameterTagTest.jsp?id=testId2&str=testStr2

액션 태그 (Action Tag)

- <jsp:setProperty> Action Tag의
  - property 속성값이 “*” 로 설정되면

```
<jsp:useBean id="acT" class="actiontag.ActionTagTest">
    <jsp:setProperty name="acT" property="*" />
</jsp:useBean>
```

    - Client로부터 전달된
      - request Object의 Parameter 이름을 검사한 후
      - Parameter 없는 Constructor로 생성하고
      - Parameter 이름에 해당하는 각 setter Method 들을 호출
        - Member Variable에 값을 할당
        - Parameter 이름에 해당하는 setter Method가 없다면

```
<%
    actiontag.ActionTagTest acT = new actiontag.ActionTagTest();
    acT.setId(request.getParameter("id"));
    acT.setStr(request.getParameter("str"));
%>
```

- <jsp:useBean> Action Tag를 사용한 초기화는
  - 반드시 Parameter가 없는 Constructor을 사용
  - Class에 Parameter가 없는 Constructor이 없다면
    - <jsp:useBean> Action Tag 사용 불가

액션 태그 (Action Tag)

- Java Code or Action Tag
  - Java Beans를 활용하는 두 가지 방법

    - Scriptlet 내부에 Java Code 를 이용
      - Java Beans Object 생성 후, 사용
      - 개발자에게 친숙

    - Action tag를 이용
      - <jsp:useBean>, <jsp:setProperty>, <jsp:getProperty>
      - Java Beans Object 생성 후, 사용
      - 디자이너에게 친숙

    - 어떤 방법을 사용하느냐에 대한 내용이 정해져 있지 않다면
      - 만드는 사람이 편한 방법으로
      - 맘대로 하세요~

액션 태그 (Action Tag)

- <jsp:forward> Action Tag
  - 제어권을 가진 Page가 다른 Page를 호출하면서
    - 자기가 가진 모든 권한을 다른 Page로 넘기는 역할
    - pageContext 내장 객체의 forward() Method와 동일한 기능
  - 사용

```
<jsp:forward    path="이동할 페이지의 URL" />
```

    - path 속성을 이용하여 이동할 Page의 URL 기술

```
<jsp:forward    path="ForwardTo.jsp" />
```

  - 동작
    - <jsp;forward> Action tag에 의해 다른 Page가 호출되면
      - 현재 Page의 제어권이 대상 Page로 넘어감
      - 대상 Page가 호출 되면서 내부적으로
        - out.clear() Method가 호출되어
          - 출력 Buffer에 있던 모든 내용이 지워지므로
          - 이전 Page에서의 출력문은 모두 효력을 잃음

액션 태그 (Action Tag)

- ForwardFrom.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>

<html>
    <head>
        <title>jsp:forward 태그</title>
    </head>
    <body>
        <h2>jsp:forward 태그를 이용한 다른 페이지 호출</h2>
        <h3>ForwardFrom.jsp의 시작 입니다.</h3>
        <jsp:forward page="ForwardTo.jsp" />
    <h3>ForwardFrom.jsp의 마지막 입니다.</h3>
    </body>
</html>
```

- ForwardTo.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>

<html>
    <head>
        <title>jsp:forward 태그</title>
    </head>
    <body>
        <h3>ForwardTo.jsp의 시작 입니다.</h3>
        <%=request.getParameter("name")%>
        <h3>ForwardTo.jsp의 마지막 입니다.</h3>
    </body>
</html>
```

- http://localhost/MySample/ForwardFrom.jsp?name=TestName

액션 태그 (Action Tag)

- <jsp:forward> Action Tag
  - 다른 Page로 이동하면서 제어권도 같이 넘겨줌
  - 제어권을 넘기는 순간
    - out.clear() Method가 호출되어
    - 출력 Buffer에 있던 모든 내용이 삭제
  - 다른 Page로 넘어간 제어권은 다시 돌아오지 않음

액션 태그 (Action Tag)

- <jsp:include> Action Tag
  - 다른 Page를 호출해서 실행 후 그 결과를 현재 Page에 포함하는 기능
    - pageContext 내장 객체의 include() Method와 동일한 기능
  - 사용

```
<jsp:include   path="포함할 페이지 URL" />
```

    - path 속성은 현재 Page에 포함될 Page의 URL을 나타냄

```
<jsp:include   path="IncludeTo.jsp" />
```

    - 현재 Page에 지정한 Page를 호출하여 실행한 후 그 결과를 포함
      - <jsp:forward> Action Tag와 마찬가지로
        - 다른 Page를 호출할 때 제어권을 넘겨주지만
        - 실행한 후 다시 제어권을 돌려받음

액션 태그 (Action Tag)

- <jsp:include> Action Tag
  - 동작
    - <jsp:forward>, <jsp:include>
    - 공통점
      - 둘다 호출되는 Page로 제어권이 넘어감
    - 차이점
      - 제어권의 전달
        - <jsp:forward>
          - 제어권을 다른 Page로 넘기면 다시 받아 올 수 없다.
        - <jsp:include>
          - 제어권을 다른 Page로 넘긴 후 다시 자신의 Page로 돌려받는다.
      - 출력
        - <jsp:forward>
          - 다른 Page 호출 시 out.clear() Method가 호출되어 출력 Buffer의 내용이 지워짐
        - <jsp:include>
          - 다른 Page 호출 시 out.flush() Method가 호출되어 이전의 내용이 출력

액션 태그 (Action Tag)

- IncludeFrom.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>
<html>
    <head>
        <title>jsp:include 태그</title>
    </head>
    <body>
        <h2>jsp:include 태그를 이용한 다른 페이지 호출</h2>
        <h3>IncludeFrom.jsp의 시작 입니다.</h3>
        <jsp:include page="IncludeTo.jsp" />
        <h3>IncludeFrom.jsp의 마지막 입니다.</h3>
    </body>
</html>
```

- IncludeTo.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>
<html>
    <head>
        <title>jsp:include 태그</title>
    </head>
    <body>
        <h3>IncludeTo.jsp의 시작 입니다.</h3>
        <%=request.getParameter("name")%>
        <h3>IncludeTo.jsp의 마지막 입니다.</h3>
    </body>
</html>
```

- http://localhost/MySample/ForwardFrom.jsp?name=testName

액션 태그 (Action Tag)

- <jsp:include> Action Tag
  - 제어권이 이동될 Page로 넘어간 후
    - 다시 이전 Page로 돌아옴
  - 이전 Page에서 읽었던 내용들이 모두 출력
    - <jsp:include> 를 만나게되면
      - 출력 Buffer에 있던 내용들이 모두
      - out.flush() Methdo에 의해 출력
- <jsp:forward> VS <jsp:include>
  - <jsp:forward>
    - 다른 Page로 제어권을 넘기면 다시 받아올 수 없음
    - 제어권을 넘기는 순간까지의 출력 Buffer에 있어 들였던 내용을 모두 지움
  - <jsp:include>
    - 다른 Page로 제어권을 넘겨도 제어권을 다시 돌려받음
    - 제어권을 넘기는 순간까지의 출력 Buffer에 읽어들였던 내용을
      - 제어권을 받는 Page로 고스란히 넘겨줌
      - 따라서 읽은 내용 모두 출력이 가능

액션 태그 (Action Tag)

- <jsp:param> Action Tag
  - <jsp:forward>나 <jsp:include>를 사용해서
    - 다른 JSP Page를 호출할 때
      - Parameter를 전달하기 위한 Action Tag
  - 사용

```
<jsp:param   name="매개변수 이름"   value="매개변수의 값" />
```

    - name : 다른 Page로 전달할 때 Parameter의 이름
    - value : 해당 Parameter의 값을 지정

```
<jsp:include path="ParamTo.jsp">
    <jsp:param name="id" value="Micheal"/>
</jsp:include>
```

  - 동작
    - 다른 Page 호출 시 제어권이 넘어가므로
      - request 내장 객체 또한 호출된 Page에서 동일하게 사용
        - 이전 페이지에서 전달된 Parameter 정보도
        - 호출된 Page에 동일하게 전달됨을 의미

액션 태그 (Action Tag)

- <jsp:param> Action Tag
  - 특징
    - <jsp:forward>, <jsp:include>를 이용해서
      - 다른 Page를 호출 할 때
      - 추가적인 정보전달을 위해 <jsp:param> 을 사용하여
      - Parameter 를 전달
    - 만약 전달 시 동일한 이름의 Parameter가 존재한다면
      - <jsp:param>에 의해 지정된 값으로 Update되어 전달
      - <jsp:param>에 의해 정의된 Parameter가 우선됨

액션 태그 (Action Tag)

- ParamFrom.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>
<html>
    <head>
        <title>jsp:param 태그</title>
    </head>
    <body>
        <h2>jsp:param 태그를 이용한 파라미터 전달</h2>
        <h3>ID : <%=request.getParameter("id")%></h3>
        <h3>Str : <%=request.getParameter("str")%></h3>
        <jsp:include page="ParamTo.jsp">
            <jsp:param name="id" value="Micheal"/>
        </jsp:include>
    </body>
</html>
```

- ParamTo.jsp

```jsp
<%@page contentType="text/html; charset=euc-kr"%>
<html>
    <head>
        <title>jsp:param 태그</title>
    </head>
    <body>
        <h3>ParamTo.jsp의 시작 입니다.</h3>
        <h3>ID : <%=request.getParameter("id")%></h3>
        <h3>Str : <%=request.getParameter("str")%></h3>
    </body>
</html>
```

- http://localhost/MySample/ParamFrom.jsp?id=testID&str=Thanks

액션 태그 (Action Tag)

- <jsp:param> Action Tag
  - 이전 Page에서 Parameter 값을 직접 넣어주어도
    - <jsp:include> 내
      - <jsp:param> 를 이용하여
      - Parameter 값을 바꾸어 주면
        - Parameter 가 넘어갈 Page로 갔을 때
        - <jsp:param> 에서 넘기는 값이 우선
        - <jsp:forward> 에서도 동일하게 적용

오늘 숙제

- 집에서 해보자!