

Anyframe Simpleweb VO Plugin



Version 1.0.1

저작권 © 2007-2011 삼성SDS

본 문서의 저작권은 삼성SDS에 있으며 Anyframe 오픈소스 커뮤니티 활동의 목적하에서 자유로운 이용이 가능합니다. 본 문서를 복제, 배포할 경우에는 저작권자를 명시하여 주시기 바라며 본 문서를 변경하실 경우에는 원문과 변경된 내용을 표시하여 주시기 바랍니다. 원문과 변경된 문서에 대한 상업적 용도의 활용은 허용되지 않습니다. 본 문서에 오류가 있다고 판단될 경우 이슈로 등록해 주시면 적절한 조치를 취하도록 하겠습니다.

I. Introduction	1
II. Controller	2
1. SimpleServiceController	3
III. VO 기반의 Simpleweb 개발	4
2. Step 1. JSP 호출	5
3. Step 2. JSP 개발	7

I.Introduction

Simpleweb VO Plugin은 Spring MVC 기반의 웹 어플리케이션을 개발할 때 개발자가 웹 개발을 보다 쉽고 간단하게 할 수 있도록, Transfer Object로 VO(Value Object) 객체를 사용하는 공통 Controller 클래스 및 태그 라이브러리들을 사용하여 작성된 샘플 코드와 이를 활용하는데 필요한 참조 라이브러리들로 구성되어 있다.

Installation

Command 창에서 다음과 같이 명령어를 입력하여 simpleweb-vo plugin을 설치한다.

```
mvn anyframe:install -Dname=simpleweb-vo
```

installed(mvn anyframe:installed) 혹은 jetty:run(mvn clean jetty:run) command를 이용하여 설치 결과를 확인해볼 수 있다.

Dependent Plugins

Plugin Name	Version Range
Simpleweb [http://dev.anyframejava.org/docs/anyframe/plugin/optional/simpleweb/1.0.1/reference/htmlsingle/simpleweb.html]	2.0.0 > *

II.Controller

이번 장에서는 Transfer Object를 VO로 사용할 경우 사용하게 될 Anyframe에서 제공하는 Controller 클래스에 대해 알아보도록 한다.

1.SimpleServiceController

SimpleServiceController는 VO 타입의 객체로 Transfer Object를 사용할 때의 데이터 바인딩을 지원해줄 AbstractServiceController를 확장한 Controller 클래스이다. HttpServletRequest의 Parameter로 넘어온 값을 메소드 정보의 argument 타입의 객체(여기선 VO객체가 됨)로 바인딩 해주며 호출하려고 하는 비즈니스 서비스 메소드의 argument의 타입이나 return 타입을 가지고 추가 작업을 해주기도 한다. (ex> 비즈니스 서비스의 메소드 argument 타입이 org.anyframe.datatype.SearchVO 타입일 경우 pageIndex 값을 Request로부터 얻어내어 설정해준다.) 데이터 바인딩 방법을 바꾸거나 특정 작업을 추가 할 때, 또는 AbstractServiceController의 기능을 변경하고 싶을 때는 SimpleServiceController를 확장하여 구현하면 된다. 다음은 Simpleweb-VO Plugin 설치로 추가된 Spring 속성 정의 파일 simpleweb-vo-servlet.xml 파일의 일부이다.

```
<!-- simple direct service controller -->
<bean name="/simple.do" class="org.anyframe.simpleweb.controller.SimpleServiceController">
  <property name="beanMethodsRepo" ref="beanMethodsRepo" />
</bean>
```

위와 같이 정의하면 "/simple.do" 요청을 SimpleServiceController에서 처리하게 된다.

III.VO 기반의 Simpleweb 개발

이번 장에서는 Simpleweb VO Plugin (simpleweb-vo) 설치시 생성되는 Sample Application을 중심으로 VO 기반의 Simpleweb을 개발하는 방법에 대해 설명하도록 한다.

2.Step 1. JSP 호출

simpleweb plugin (simpleweb-vo)를 설치하면 left layout에 SimpleWeb-VO Sample이 추가 정의된다. 이 메뉴를 클릭하면 Simpleweb을 이용한 Movie Finder List를 출력하도록 한다. 이 때, Anyframe 의 link tag를 사용하며 다음은 simpleweb plugin (simpleweb-vo) 설치로 추가 정의된 left.jsp 파일의 일부이다.

```
<simpleweb:link id="voMovie" service="simplewebVoMovieFinder.getPagingList(search)"
  layout="voLayout"
  tiles="body:/WEB-INF/jsp/simpleweb-vo/moviefinder/movie/list.jsp">
<simpleweb:model name="search"
  type="${package}.simpleweb.vo.moviefinder.service.MovieSearchVO" />
<simpleweb:setProperty name="title">Simpleweb-VO Sample</simpleweb:setProperty>
</simpleweb:link>
```

위의 tag에서 볼 수 있듯이 SimpleWeb-VO Sample을 텍스트로 가지는 "voMovie"가 id인 링크(a href..)가 생성된다. 호출하게 될 비즈니스 서비스는 service로 정의된 "simplewebVoMovieFinder"가 bean id인 클래스가 되며 메소드는 getPagingList(search) 메소드가 된다. 메소드의 argument는 MethodInfo를 통해 메소드의 argument의 타입을 알아낸 후 Request의 Parameter를 바인딩 시켜 객체를 셋팅하게 된다. 하지만 이때 argument의 타입이 Primitive 타입인 경우에는 Request의 해당 키 값을 찾아 셋팅해 준다. 다시 말해서 MethodInfo를 통해 알아낸 getPagingList(search)라는 메소드의 argument는 하나이고 org.anyframe.datatype.SearchVO 타입이라면 SearchVO타입의 객체를 생성하여 Request에 있는 Parameter들을 생성된 객체에 셋팅해주게 될 것이다. 하지만 getPagingList(search)라는 메소드의 argument는 하나이고 Primitive 타입이라면 request에서 "search"라는 Parameter를 찾아 셋팅해 주게 된다. 만약 argument가 여러 개일 경우 modelAttribute에 해당하는 객체가 첫 번째 argument로 정의되어 있어야 하며 나머지 argument들은 Primitive 타입만 가능하다. 요청을 처리한 후 출력할 페이지의 Tiles Definition은 "voLayout"이며 put-attribute의 name이 "body"인 요소의 value를 "/WEB-INF/jsp/simpleweb-vo/moviefinder/movie/list.jsp"로 셋팅하게 될 것이다. 다음은 이와 관련된 simpleweb plugin (simpleweb-vo) 설치로 추가 정의된 Tiles Definition 파일인 tilesviews.xml의 일부이다.

```
<!-- simpleweb-vo-tiles-definition-START -->
  <definition name="voLayout" template="/WEB-INF/jsp/simpleweb-vo/layout/standard.jsp">
    <put-attribute name="top" value="/sample/layouts/top.jsp" />
    <put-attribute name="left" value="/sample/layouts/left.jsp" />
    <put-attribute name="body" expression="${requestScope.body}" />
  </definition>
<!-- simpleweb-vo-tiles-definition-END -->
```



Tiles Definition의 expression

Tiles를 사용할 때 layout 내에 추가될 페이지에 대해서는 value 혹은 expression(Tiles 버전 2.1.X 이상) attribute를 이용하여 정의할 수 있다. value를 사용하면 정적으로 넣어줄 페이지를 xml 파일내에 일일이 정의하게 되며 expression을 사용하면 Spring EL을 사용하여 Scope에 있는 값을 꺼내어 셋팅할 수 있게 된다. 다시 말해서 \${requestScope.body}라고 정의 할 경우 request attribute에 있는 body 값을 찾아 셋팅해 준다.

요청 후 출력해줄 페이지인 list.jsp에서는 Spring의 Form tag를 사용하며 이 페이지를 출력하기 위해서는 반드시 modelAttribute 값을 셋팅해 줘야 한다. 그러므로 model tag를 사용하여 "search"라는 modelAttribute에 "\${package}.simpleweb.vo.moviefinder.service.MovieSearchVO" 타입의 객체를 생성해 주도록 한다.



Partial Rendering

Anyframe에서는 Spring JS 기반의 Partial Rendering 기능을 제공한다. 먼저 Partial Rendering을 요청하는 link tag나 submit tag에서 fragment 속성과 render 속성을 설정해 줘야 한다. 설정 예는 아래와 같다.

```
<simpleweb:link id="voMovie"
  service="simplewebVoMovieFinder.getPagingList(search)" layout="voLayout"
  tiles="body:/WEB-INF/jsp/simpleweb-vo/moviefinder/movie/
  list.jsp" render="partial">
  ...중략... >
```

위와 같이 설정하게 되면 Tiles Definition의 put-attribute가 "body"인 요소에 대해서만 페이지를 Refresh하게 된다. 이 때, Refresh 하는 부분이 JSP 페이지의 div 영역이 됨을 주의하도록 한다. 다시 말해서 Refresh 되게될 페이지 영역의 div id와 Refresh되어 들어가질 페이지 영역의 div id가 일치해야한다. 자세한 설명을 위해 아래의 standard.jsp의 코드를 보도록 하자.

```
<td width="100%" height="100%" align="left" valign="top" style="padding:0 20px
  0 20px">
  <div id="body">
    <tiles:insertAttribute name="body"/>
  </div>
</td>
```

위의 코드를 포함하는 standard.jsp는 Tiles의 Layout 페이지 이며 코드에서 볼수 있듯이 div id가 "body"인 부분을 포함하고 있다. 이 부분을 Partial Rendering 시켜주기 위해서는 대상 영역에 대한 div id를 "body"로 일치 시켜줘야 한다. 다음 코드는 body 영역에 Partial Rendering 되게될 페이지인 list.jsp 페이지의 일부이다.

```
<!-- list.jsp start -->
<%@ include file="/sample/common/taglibs.jsp"%>
<div id="body">
  ...중략...
</div>
<!-- list.jsp end -->
```

3.Step 2. JSP 개발

위에서 설명한 것과 같이 link tag와 submit tag를 사용하여 JSP 페이지에 진입한 후 본격적으로 JSP를 개발해 보도록 한다. 먼저 페이지가 form 요소를 가질 경우 Spring의 form tag를 사용하여 구현한다. 이 때, modelAttribute의 명은 model tag로 인해 셋팅 하였을 때는 model tag의 name이 되며 어떠한 비즈니스 서비스를 수행한 후 리턴 된 VO의 값을 사용하고 싶을 때는 VO 클래스의 Camelcased명으로 정의해 준다.

```
<form:form modelAttribute="search" method="post" id="searchForm" ...
```

다음으로 form 내부에 Anyframe의 submit tag나 link tag 사용시 필요한 hiddenDiv를 설정해 준다.

```
<div id="hiddenDiv"></div>
```

form의 필드에 대해서는 마찬가지로 Spring의 form tag로 구현하며 필요에 따라 Anyframe validate tag를 사용하여 JavaScript Validation을 적용하도록 한다. 또한, 다른 페이지로의 이동이나 Submit이 발생할 시 Anyframe link tag와 submit tag를 사용하여 구현하도록 한다.

페이지에서 리스트를 포함하고 있을 경우 리스트에서 발생하는 링크에 대해서는 속도 저하의 가능성이 있어 직접 loop안에 Anyframe Tag Library 작성을 가이드하지 않는다. 이에 Anyframe pagenavigator tag에서 partial 속성과 fragment 및 popup 속성을 설정할 수 있도록 지원한다.

```
<c:forEach var="result" items="${resultList}" varStatus="status">

  <tr class="board" onMouseOver="this.style.backgroundColor = '#e4eaff';return true;";
  onMouseOut="this.style.backgroundColor = ''; return true;";>
    <td class="underline">${result.genre.name}</td>
    <td class="underline">
      <a class="linkClass" id="getMovieId${status.count}" href="simple.do?
service=simplewebVoMovieService.get(movieId)... ">
        ${result.title}
      </a>
    </td>
    <td align="left">${result.director}</td>
    <td align="left">${result.actors}</td>
    <td align="center">${result.releaseDate}</td>
  </tr>
</c:forEach>

<anyframe:pagenavigator linkUrl="javascript:document.searchForm.submit();"
pages="${resultPage}" formName="searchForm"
render="partial" linkFragment="body" linkClass="linkClass" linkPopup="true"/>
```

위의 코드에서 볼 수 있듯이 link 대상이 되는 a tag에 linkClass를 정의하고 pagenavigator의 linkClass에 같은 값을 정의한다. 또한, link tag와 submit tag를 사용할 때와 마찬가지로 render, linkFragment, linkPopup 속성을 부여하도록 한다. 목록 조회 페이지에서 Anyframe의 PageNavigator Tag를 사용하여 페이지 네비게이션 바를 출력하지 않는다면, 아래와 같은 script 구문을 목록 조회 페이지 하단에 추가하도록 한다.

```
<script type="text/javascript">
  dojo.query("." + linkClass).forEach(function(element) {
    Spring.addDecoration(new Spring.AjaxEventDecoration({
      elementId: element.id,
      event: "onclick",
      popup: true,
      params: {fragments: body} }));
  });
</script>
```

```
</script>
```

목록 조회 페이지에서 목록 중 하나를 선택하여 상세 조회 페이지로 이동하고자 할때 Popup 화면으로 페이지를 띄우고, Partial Rendering 기능을 적용하고자 하는 경우 위와 같이 작성하면 된다.