

Anyframe Simpleweb jQuery Plugin



Version 1.0.2

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. SimpleJSONController	3
2. SimpleJSONTreeController	4
3. FileUploadController	5
III. jQuery를 이용한 Simpleweb 개발	6
4. jqgrid	8
5. quickpager	10
6. autocomplete	12
7. jstree	13
8. ui-tab	15
9. dropdown	16
10. uploadify	17

I.Introduction

Simpleweb jQuery Plugin은 Spring MVC 기반의 웹 어플리케이션을 개발할 때 개발자가 웹 개발을 보다 쉽고 간단하게 할 수 있도록, JSON 형태의 데이터를 이용한 공통 Controller 클래스 및 태그 라이브러리들을 사용하여 작성된 샘플 코드와 이를 활용하는데 필요한 참조 라이브러리들로 구성되어 있다. 여기에는 jQuery [<http://jquery.com/>]와의 연계 방안에 대한 샘플도 함께 포함되어 있다. Plugin 샘플 코드에 활용된 jQuery 컴포넌트들로는 jqgrid, quickpager, autocomplete, jstree, ui-tab, dropdown, uploadify 등이 있다.

Installation

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

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

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 jQuery와 연계하기 위해, JSON 객체를 리턴하는 경우 사용하게 될 Anyframe에서 제공하는 Controller 클래스에 대해 알아보도록 한다.

1.SimpleJSONController

SimpleJSONController는 JSON 객체를 Datatype으로 사용하는 jQuery의 Component를 지원하기 위해 개발되었다. 현재 SimpleServiceController를 상속받아 구현되어 있으므로 Service Layer에서의 Transfer Object로는 VO 객체를 사용하도록 한다. 물론, jQuery를 연계하기 위해 SimpleJSONController를 사용할 때 Service Layer의 변경이나 추가 코딩없이 기존의 비즈니스 서비스를 호출하여 사용할 수 있다. 현재 simpleweb plugin에서는 jQuery의 Grid, Autocomplete, UI, Tree, Combobox 컴포넌트 예제를 제공하고 있으며 SimpleJSONController는 주로 Grid(jqgrid plugin), Autocomplete(autocomplete plugin) 기능에 대한 데이터를 셋팅해 주는 역할을 하고 있다. jQuery의 Grid 컴포넌트(jqgrid plugin)에서 Grid를 그릴 때 비즈니스 서비스 호출후 리턴되는 Page 객체를 바로 받을 수 있는 것이 아닌 Grid에서 인식할 수 있는 Key와 Value 쌍의 Map 형태로 Model 객체에 셋팅해줘야 한다. 다음은 SimpleJSONController 클래스의 일부이다.

```
if (methodInfo.getOutputParam()[0].getName().equals(
    ANYFRAME_COMMON_PAGE)) {
    //....
    jsonModel.put("page", ((Page) result).getCurrentPage() + "");
    jsonModel.put("total", ((Page) result).getMaxPage() + "");
    jsonModel.put("records", ((Page) result).getTotalCount());
    jsonModel.put("rows", ((Page) result).getList());
}
```

위의 코드에서 볼 수 있듯이 비즈니스 서비스 수행후 Return 값이 org.anyframe.pagination.Page 타입일 경우 jQuery의 Grid에서 인식 할 수 있는 key값으로 jsonModel 객체를 셋팅해 주고있다. Autocomplete 컴포넌트를 사용할 때도 마찬가지로 출력해줄 데이터를 Autocomplete 컴포넌트에서 인식할 수 있는 key값의 데이터로 셋팅해 주는 부분이 필요하다.

```
jsonModel.put("autoData", resultValue);
```

이렇게 생성된 jsonModel 객체는 Controller의 Model 객체로 셋팅되어 View로 전달되게 된다. 위의 코드는 이미 SimpleJSONController에 구현되어 있으므로 개발자가 jQuery의 Grid, AutoComplete의 기능을 사용할 때에는 추가 코딩없이 사용할 수 있다. 이 외의 특정 Key 값의 데이터를 필요로 하는 또 다른 컴포넌트를 도입하거나 추가 기능이 필요하게 되면 SimpleJSONController를 확장하여 사용하면 된다. 다음은 simpleweb-jquery plugin 설치로 추가된 Spring 속성 정의 파일 simpleweb-jquery-servlet.xml 파일의 일부이다.

```
<bean name="/simplejson.do" class="org.anyframe.simpleweb.controller.SimpleJSONController">
    <property name="beanMethodsRepo" ref="beanMethodsRepo" />
</bean>
```

위와 같이 정의하면 "/simplejson.do" 요청을 SimpleJSONController가 처리하게 된다.

2.SimpleJSONTreeController

SimpleJSONTreeController는 jQuery의 Tree 컴포넌트(jstree)를 지원하기 위한 공통 Controller이다. SimpleJSONTreeController는 SimpleJSONController를 상속받아 구현되어 있다. 기본적으로 Tree에서 필요한 노드 정보를 셋팅하기 위해 Tree 컴포넌트에서 인식 할 수 있는 형태의 도메인 객체를 사용해야한다. 이는 Anyframe에서 제공하는 Attributes, Data, JSTreeNode 도메인 클래스를 사용하도록 하고 추가적인 Attribute의 설정이 필요할 경우 해당 클래스를 확장하여 추가하도록 한다. SimpleJSONTreeController는 기본적으로 사용자가 클릭하는 노드의 ID를 가지고 비즈니를 서비스를 호출하고 그 ID에 따라 알맞은 Tree에 출력할 데이터를 셋팅해 주게 된다. 이 때, Tree의 데이터를 셋팅하는 것은 화면 시나리오에 따라 달라질 수 있으므로 개발자는 반드시 SimpleJSONTreeController 클래스를 상속 받고 setTreeData() 메소드를 오버라이드 하여 구현해야한다. 다음은 simpleweb-jquery plugin 설치로 추가된 SimpleJSONTreeController를 확장하여 구현한 MovieTreeController.java 파일의 일부이다.

```
public class MovieTreeController extends SimpleJSONTreeController {

    @Override
    protected void setTreeData(ArrayList<JSTreeNode> listNode, List jsTreeList,
        String id) throws Exception {
        JSTreeNode node;
        Attributes attr;
        String data;
        if (id.equals("0")) {
            //...중략...
            // id가 0일 경우 Tree 데이터 셋팅하는 로직 구현
        }
        else {
            //...중략...
            // id가 0이 아닐 경우 Tree 데이터 셋팅하는 로직 구현
        }
    }
}
```

3.FileUploadController

jQuery의 uploadify 컴포넌트를 사용하여 지정된 파일을 특정 경로에 업로드 하고 파일의 정보를 셋팅하여 Model 객체로 리턴한다. 다음은 FileUploadController 코드의 일부이다.

```
public class FileUploadController extends AbstractController{
    protected IdGenService idGenService;

    public void setIdGenerationService(IdGenService idGenService) {
        this.idGenService = idGenService;
    }

    private String uploadPath = "/upload/";

    public void setUploadPath(String uploadPath){
        this.uploadPath = uploadPath;
    }

    @Override
    protected ModelAndView handleRequestInternal(HttpServletRequest request,
        HttpServletResponse response) throws Exception {

        MultipartFile file = (MultipartFile)
        ((MultipartHttpServletRequest)request).getFile("fileData");
        String fileDir = request.getParameter("fileDir");

        String destDir = request.getSession().getServletContext().getRealPath(
            uploadPath);

        if (fileDir == null) {
            fileDir = idGenService.getNextStringId();
            FileUtils.forceMkdir(new File(destDir + "/" + fileDir));
        }

        file.transferTo(new File(destDir + "/" + fileDir, file.getOriginalFilename()));

        Map<String, Object> result = new HashMap<String, Object>();
        result.put("fileNm", file.getOriginalFilename());
        result.put("fileSize", file.getSize());
        result.put("fileDir", fileDir);

        ModelAndView mav = new ModelAndView("jsonView");
        mav.addObject("uploadResult", result);

        return mav;
    }
}
```

위의 코드에서 볼수 있듯이 "fileData"를 MultipartFile 타입의 객체로 받고 Request Parameter의 "fileDir"를 받아 파일 업로드 로직을 수행한 후 "fileNm", "fileSize", "fileDir"의 데이터를 셋팅한 후 화면에 그 결과를 리턴한다.

III.jQuery를 이용한 Simpleweb 개발

이번 장에서는 Simpleweb jQuery Plugin (simpleweb-jquery) 설치시 생성되는 Sample Application을 중심으로 Simpleweb과 jQuery를 연계하는 방법에 대해 설명하도록 한다. jQuery를 사용할 경우 화면에 출력되게 될 데이터를 JSON 객체형태로 리턴하게 된다. 기본적으로 simpleweb-jquery plugin 설치시 생성되는 Sample Application에서 제공하는 jQuery의 컴포넌트는 jqgrid, jstree, autocomplete, dropdown, jquery-ui, quickpager, uploadify 가 있다. jQuery 관련 파일(js, css)은 [프로젝트]/src/main/webapp/simpleweb-jquery/jquery 폴더 하위에 모두 존재한다.

이러한 jQuery 관련 js나 css 파일은 simpleweb-jquery에서 사용하는 standard.jsp의 상단에 정의되어 있다. 다음은 simpleweb plugin 설치로 추가된 standard.jsp 파일의 일부이다.

```
<!-- for jquery -->
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/jquery-1.6.2.min.js'/>"></script>
<link rel="stylesheet" href="<c:url value='/simpleweb-jquery/css/jquery.css'/>" type="text/css" />

<!-- jquery ui, jqGrid -->
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/jqgrid/i18n/grid.locale-en.js'/>"></script>
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/jquery-ui/jquery-ui-1.8.16.custom.min.js'/>"></script>
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/jqgrid/jquery.jqGrid.min.js'/>"></script>
<link href="<c:url value='/simpleweb-jquery/jquery/jqgrid/ui.jqgrid.css'/>" rel="stylesheet" type="text/css" />

<!-- jquery form -->
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/form/jquery.form.js'/>"></script>

<!-- jquery:jstree-1.0.RC3 -->
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/jstree/jquery.jstree.js'/>"></script>

<!-- jquery tab -->
<link href="<c:url value='/simpleweb-jquery/jquery/jquery-ui/smoothness/jquery-ui-1.8.16.custom.css'/>" rel="stylesheet" type="text/css" />

<!-- jquery uploadify -->
<link type="text/css" href="<c:url value='/simpleweb-jquery/jquery/uploadify/uploadify.css'/>" rel="stylesheet">
<script src="<c:url value='/simpleweb-jquery/jquery/uploadify/swfobject.js'/>" type="text/javascript"></script>
<script src="<c:url value='/simpleweb-jquery/jquery/uploadify/jquery.uploadify.v2.1.4.min.js'/>" type="text/javascript"></script>

<!-- jquery image dropdown -->
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/dropdown/msdropdown/js/jquery.dd.js'/>"></script>
<link href="<c:url value='/simpleweb-jquery/jquery/dropdown/msdropdown/dd.css'/>" rel="stylesheet" type="text/css" />

<!-- quick pager -->
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/quickpager/quickpager.mod.jquery.js'/>"></script>
<link rel="stylesheet" href="<c:url value='/simpleweb-jquery/jquery/quickpager/pagination.css'/>" type="text/css" />

<!-- validator -->
```

```
<script type="text/javascript" src="<c:url value='/simpleweb-jquery/jquery/validate/
jquery.validate.js' />"></script>
```

위에서 import되어 있는 js 파일과 css 파일 중에서 필요한 컴포넌트에 대한 파일만 정의하도록 한다. jQuery에서는 이외에도 여러가지 다양한 plugin 형태의 컴포넌트를 제공하고 있으며 필요에 따라 연계하여 사용하면 된다. 기본적으로 JSON 타입의 객체를 사용한다 할지라도 기본적인 Anyframe Tag Library 사용법은 VO나 Map을 사용할 때와 동일하다. 위에서 언급한 기본적인 jQuery의 컴포넌트 연계 방법에 대해 알아보도록 하자.

4.jqgrid

jqgrid는 리스트 출력을 위한 jquery plugin이다. 다음은 simpleweb-jquery plugin 설치로 추가된 리스트를 출력하는 list.jsp 파일의 일부이다.

```
jQuery("#grid").jqGrid(  
  url: "<c:url value='/simplejson.do?'  
  layout=jqueryLayout&service=simplewebjQueryMovieFinder.getPagingList(searchvo)  
  &viewName=jsonView' />",  
  mtype: 'POST',  
  datatype: "json",  
  //column 이름  
  colNames: [ '<anyframe:message code="movie.genre" />', 'id', '<anyframe:message  
  code="movie.title" />'...],  
  jsonReader: {repeatitems: false},  
  //각 column에 들어갈 데이터  
  colModel: [  
    {name: 'genre.name', index: 'genre.name', align: 'center'}, {key: true, name: '  
    'movieId', hidden: true}...],  
    width: 780, height: 69, scroll: false, forceFit:true, multiselect: true, viewrecords :  
    true,  
    //출력 매줄 row의 갯수  
    rowNum: 3,  
    sortable: true,  
  
    //error 처리  
    loadError: function(xhr,st,err) {  
      alert('<anyframe:message code="error.moviefinderimpl.getpaginglist" />');  
    },  
  
    //cell double click 이벤트 처리  
    ondblclickRow: function(rowId) {  
      rowData = jQuery("#grid").getRowData(rowId);  
      jQuery("#getLink").attr("href",  
        "<c:url value='/simplejsoncommon.do?'  
      layout=jqueryLayout&service=simplewebjQueryMovieService.get(movieId)&  
      tiles=body:/WEB-INF/jsp/simpleweb-jquery/moviefinder/movie/form.jsp&  
      initdataService=simplewebjQueryGenreService.getDropDownGenreList()&  
      initdataResult=genreList&movieId=' />" + rowData.movieId);  
      document.getElementById("getLink").focus();  
    }  
  });
```

jqgrid에서는 리스트 출력을 위한 url을 "url" 속성을 통해서 정의하고 있는데 이 때, Get 방식으로 요청을 보내기 때문에 SimpleWeb 공통 Controller에서 필요한 속성을 link tag나 submit tag없이 parameter로 셋팅해서 보내줘야 한다. 이 때, layout, service, tiles는 기존 tag에서 작성했을 때와 동일하게 작성한다. 단, init tag의 valuelist로 key, value 쌍으로 정의했던 초기화 데이터 셋팅에 대해서는 key에 해당하는 값을 "initdataResult", value 값을 "initdataService"로 보내준다. JSON 객체를 사용하는 경우 리턴되는 viewName은 반드시 "jsonView"로 정의한다.

위의 코드에서는 ondblclickRow 라는 callback 함수를 사용하여 cell double click 이벤트를 처리하고 있다. Simpleweb-jQuery Sample Application에서는 해당 cell의 상세 정보를 popup으로 출력하는 로직을 구현하고 있다. jQuery 함수 내에서 Spring JS의 함수를 바로 호출할 수는 없으므로 "getLink"라는 임의의 link를 만들고 href 속성과 이벤트 발생시 popup을 출력할 수 있도록 되어있다. 다음은, JSP에 작성된 getLink와 Spring JS 코드의 일부이다.

```
<a id="getLink" name="getLink"></a>  
<script type="text/javascript">  
  Spring.addDecoration(new Spring.AjaxEventDecoration({
```

```

    elementId: getLink,
    event: "onfocus",
    popup: true,
    params: {fragments:"body"}
  });
</script>

```

이렇게 jqgrid에 대한 Script 코드가 완성 되면 리스트를 출력 해줄 부분에 아래와 같이 table을 이용해서 리스트를 출력해 줄 수 있도록 한다.

```
<table id="grid" class="scroll" cellpadding="0" cellspacing="0"><tr><td/></tr></table>
```

위와 같이 jqgrid로 구현된 리스트의 모습은 아래와 같다.

<input type="checkbox"/>	Genre	Title	Director	Actors	Release Date
<input type="checkbox"/>	Adventure	Alice in Wonderland	Tim Burton	Johnny Depp	2010-03-04
<input type="checkbox"/>	Sci-Fi	Avatar	James Cameron	Sigourney Weaver	2010-12-16
<input type="checkbox"/>	Action	Green Zone	Paul Greengrass	Yigal Naor	2010-03-25

※ jqgrid를 사용하여 리스트를 작성할 때 너무 많은 양의 데이터를 한꺼번에 출력하려고 하면 리스트를 출력하는데 있어서 많은 시간이 걸리거나 브라우저가 멈추는 현상이 발생할 수 있다. 이에 한번에 출력하는 데이터의 건수는 100개 이내로 하며 데이터가 많을 경우 pager를 이용해 paging 처리 할 것을 권고한다.

5.quickpager

jqgrid는 기본적으로 Paging 처리를 위한 Pager를 제공하고 있다. Anyframe에서는 pagenavigator와 유사한 Pager 출력을 위해 quickpager를 확장하여 사용하고 있다. quickpager를 사용하기 위해서는 리스트 Script내의 loadComplete 함수 안에 paging 정보를 셋팅 해주고 search 버튼을 클릭하는 이벤트를 발생 시키도록 한다. 관련 jQuery 코드는 아래와 같다.

```
jQuery("#grid").jqGrid(
{
  중략...
  loadComplete : function(xhr) {
    $("#pagination").quickPager( {
      pageSize: "3",
      pageUnit: "3",
      //page index를 전달할 값의 id
      pageIndexId: "pageIndex",
      //search botton의 id
      searchButtonId: "searchMovies",
      currentPage: jQuery("#grid").getGridParam("page"),
      totalCount: jQuery("#grid").getGridParam("records"),
      searchUrl: "#"
    });
  }
  ...중략...
});
$("#searchMovies").click( function() {
  jQuery("#grid").setGridParam({page:$("#pageIndex").val()});
  jQuery("#grid").setPostData({searchKeyword:$("#searchkeyword").val(), nowPlayingCondition:
$("#nowPlayingCondition").val()});
  jQuery("#grid").setGridParam({
    url:"<c:url value='/simplejson.do?
layout=jqueryLayout&service=simplewebjQueryMovieFinder.getPagingList(searchvo)&viewName=jsonView' /
>"}
    .trigger("reloadGrid");
});
```

위와 같이 Script 코드가 작성 되면 pagenavigator 출력 부분에 아래와 같이 div 영역을 표시해준다.

```
<div id=".boardNavigation">
<input type="hidden" id="pageIndex" name="pageIndex" value="1" />
<div id="pagination" class="pagination"></div>
</div>
```

위와 같이 정의한 quickpager는 아래와 같은 pagenavigator를 출력하게 된다.

Genre	Title	Director	Actors	Release Date
Adventure	Alice in Wonderland	Tim Burton	Johnny Depp	2010-03-04
Sci-Fi	Avatar	James Cameron	Sigourney Weaver	2010-12-16
Action	Green Zone	Paul Greengrass	Yigal Naor	2010-03-25

1 | 2 | 3 | Next > | Next End >>



jqgrid에서 제공하는 pager

jqgrid에서도 paging 처리를 위한 간편한 pager를 제공한다. 구현 방법은 아래와 같다.

```
//jqgrid 속성 설정 내에 정의
pager : jQuery('#pager')

<!-- JSP 내의 pager 출력 부분에 정의 -->
```

```
<div id="pager" class="scroll" style="text-align: center;"></div>
```

다음 그림은 pager 적용 모습이다.

<input type="checkbox"/>	Genre	Title	Director	Actors	Release Date
<input type="checkbox"/>	Adventure	Alice in Wonderland	Tim Burton	Johnny Depp	2010-03-04
<input type="checkbox"/>	Sci-Fi	Avatar	James Cameron	Sigourney Weaver	2010-12-16
<input type="checkbox"/>	Action	Green Zone	Paul Greengrass	Yigal Naor	2010-03-25

Page 1 of 8

View 1 - 3 of 23

6.autocomplete

autocomplete plugin은 사용자가 입력한 prefix를 가지고 자동 완성 기능을 제공하는 plugin이다.

```
$("#searchKeyword").autocomplete(  
    "<:url value='/simplejson.do?  
layout=jqueryLayout&service=simplewebjQueryMovieService.getMovieTitleList(q  
&viewName=jsonView' />", {  
    //속성 정의  
    width : 200,  
    scrollHeight : 200,  
    selectFirst:true,  
    mustMatch:true,  
    matchCase:true,  
    autoFill:true,  
    scroll: true  
});
```

위의 Script 코드에서 볼 수 있듯이 마찬가지로 Simpleweb 공통 Controller를 사용하고 있으며 자동 완성 기능을 위해 호출해야할 비즈니스 서비스를 정의하고 있다. 이 때, autocomplete plugin은 입력된 값을 "q"라는 key로 Request Parameter로 보내기 때문에 argument의 이름은 반드시 "q"로 정의해준다.

다음은 autocomplete이 적용된 input box의 모습이다.



7.jstree

jstree는 Tree를 출력해주는 컴포넌트이다. 기본적인 구현 방법은 아래와 같다.

```
$("#tree").jstree({
    'plugins' : ["themes", "json_data", "ui", "types", "contextmenu", "crrm"],
    'themes' : {...},
    'json_data' : {
        'ajax':{
            "url" : "<c:url value='/simplejquerytree.do' />",
            "data" : function(n){ return {id: id, searchkeyword : return_id};},
            "success" : function(data){
                return data.JSTreeNodeList;
            },
            ...
        }
    }
})

<div id="tree" class="demo" style="overflow: auto; height: 445px; width: 280px; border: 1px solid #C9CFDD;">
<span>Movie Tree</span>
</div>
```

jstree 컴포넌트도 url을 가지고 페이지 로드시 Tree를 출력한다. 이 때도 마찬가지로 Simpleweb 기능을 이용하여 service를 지정해주고 viewName을 "jsonView"로 명시해 준다. 위에서 호출하는 "/simplejquerytree.do"의 bean 정의는 아래와 같다.

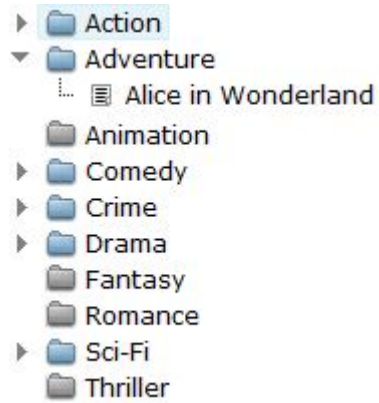
```
<bean name="/simplejquerytree.do"
class="${package}.simpleweb.jquery.moviefinder.web.MovieTreeController">
<property name="beanMethodsRepo" ref="beanMethodsRepo" />
</bean>
```

MovieTreeController는 내부적으로 SimpleJSONTreeController를 상속받고 있으며 setTreeData() 메소드를 오버라이드 하고있다. jstree를 이용한 Tree 컴포넌트를 출력하려고 할때 Controller 작성 방법에 대해서는 **SimpleJSONTreeController** [http://dev.anyframejava.org/docs/anyframe/plugin/optional/simpleweb-jquery/1.0.2/reference/htmlsingle/simpleweb-jquery.html#simpleweb_jquery_controller_simplejsontreecontroller] 부분을 참고한다. 또한, jstree는 Tree 출력을 위해 여러가지 bind 메소드를 제공한다. 다음은 bind 메소드 정의 예이다.

```
$("#tree").jstree({
    ...중략...
}).bind("remove.jstree", function(e, data) {
    if(confirm("Are you sure you want to delete this movie?")){
        data.rslt.obj.each(function() {
            $.ajax({
                async : false,
                type : 'POST',
                url : "<c:url value='/simplejson.do' />",
                layout=jqueryLayout&service=simplewebjQueryMovieService.remove(movieId)&viewName=jsonView' />";
            data : {
                "movieId" : $(data.rslt.obj).attr("id")
            },
            error : function() {
                $.jstree.rollback(data.rlbk);
            }
        });
    }
});
```

```
});  
});  
}else{  
  $.jstree.rollback(data.rlbk);  
}  
}).bind("select_node.jstree", function (e, data) {  
  ...종략...  
});
```

다음은 jstree를 이용하여 Tree를 출력한 화면이다.



8.ui-tab

jQuery의 ui plugin을 사용하면 간단하게 tab을 적용할 수 있다.

```
<script type="text/javascript">
$(function() {
  $("#tabs").tabs();
  $("#tabs").hide();
  $("#tabs").show();
})
</script>

<div id="tabs">
<ul>
  <li><a href="#genreTab">Genre Information</a></li>
  <li><a href="#movieTab">Movie Information</a></li>
</ul> <!-- tab containers -->
<div id="genreTab">
  <!-- genreTab contents-->
</div>
<div id="movieTab">
  <!-- movieTab contents-->
</div>
</div>
```

다음은 tab이 적용된 화면의 모습이다.

Genre Information	Movie Information
Title *	Green Zone
Director *	Paul Greengrass
Genre *	Action
Actors *	Yigal Naor
Runtime	90 min.
Release Date	2010-03-25
Ticket Price	7000
Now Playing	Is this movie now playing ? <input checked="" type="checkbox"/>
Poster	
<input type="button" value="Update"/>	

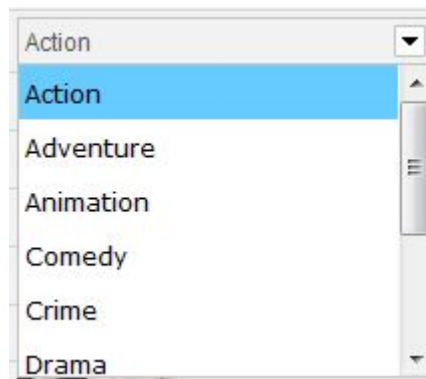
9.dropdown

jQuery의 msDropDown plugin을 사용하여 combobox를 구현할 수 있다. tab 컴포넌트와 마찬가지로 기존 HTML을 그대로 사용하면서 간단하게 구현할 수 있다.

```
<script type="text/javascript">
  $(function() {
    $('#genreId').msDropDown();
  })
</script>

<form:select id="genreId" path="genre.genreId" cssStyle="width:210px;">
  <form:options items="{genreList}" itemValue="genreId" itemLabel="name"/>
</form:select>
```

다음은 msDropDown plugin이 적용된 combobox 화면이다.



10.uploadify

Simpleweb plugin의 Sample Application에서는 파일 업로드를 위한 uploadify plugin의 적용을 위해 upload.jsp를 사용하고 있다. upload.jsp의 내용은 아래와 같다.

```
<%@ page language="java" autoFlush="true" contentType="text/html; charset=utf-8"
pageEncoding="utf-8"%>
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>

<script type="text/javascript">
$(document).ready(function() {
$("#uploadify").uploadify({
  uploader : "<c:url value='/simpleweb-jquery/jquery/uploadify/uploadify.swf' />",
  script : "<c:url value='/simpleFile/upload.do' />",
  cancelImg : "<c:url value='/simpleweb-jquery/images/cancel.png' />",
  queueID : "fileQueue",
  fileName : "fileData",
  auto : false,
  multi : false,
  width : 81,
  height : 24,
  sizeLimit : 10000000,
  buttonImg : "<c:url value='/simpleweb-jquery/images/uploadBrowse.png' />",
  onComplete : function(event, queueID, fileObj, response, data) {
    eval("var respJson="+response);

    $('#hiddenUploadedFiles').val($('#hiddenUploadedFiles').val()+','+respJson.uploadResult.fileName);
    $('#fileDir').val(respJson.uploadResult.fileDir);
    $('#uploadify').uploadifySettings('scriptData', {'fileDir' : $('#fileDir').val()});
  }
});
$('#uploadClearButton').click(function() {
  jquery('#uploadify').uploadifyClearQueue();
});
});
</script>

<table width="150" height="100" border="0" bordercolor="red">
<tr>
<td>
<div id="fileQueue" style="width:176px; height:60px;"></div>
<!-- hidden attributes -->
<input type="hidden" id="hiddenUploadedFiles" name="hiddenUploadedFiles" />
<input type="hidden" id="filePathsAttrName" name="filePathsAttrName" value="filePaths" />
<input type="hidden" id="fileDir" name="fileDir" />
<!-- browse/clear button -->
<div id="buttons" align="center" style="padding-top:5px">
<input type="file" id="uploadify" name="uploadify" width="80"/>

</div>
</td>
</tr>
</table>
```

script 속성으로 정의되어 있는 "/simpleFile/upload.do" 요청에 대한 Bean 정의는 아래와 같다.

```
<bean name="/simpleFile/upload.do"
class="org.anyframe.simpleweb.controller.FileUploadController">
```

```
<property name="idGenerationService" ref="UUIDGenService" />
</bean>
```

위의 JSP 코드에서 볼 수 있듯이 fileData와 fileDir을 **FileUploadController** [http://dev.anyframejava.org/docs/anyframe/plugin/optional/simpleweb-jquery/1.0.2/reference/htmlsingle/simpleweb-jquery.html#simpleweb_jquery_controller_fileuploadcontroller]에서 받아 처리하고 처리 결과에 대해 다시 Model 객체로 리턴해준다. 개발자는 uploadify plugin을 FileUploadController와 upload.jsp 파일을 이용하여 파일 업로드를 구현할 수 있으며 파일 업로드를 구현할 JSP 페이지에 upload.jsp 파일을 include 하고 submit tag의 setProperty 속성을 정의하고 upload에 대해 true로 설정해 준다.

```
<jsp:include page="/WEB-INF/jsp/simpleweb-map/common/upload.jsp" flush="true"/>
<anyframe:submit id="addlink" form="movieForm" action="/simplemap.do"
  service="simplewebMovieService.create(movie)" ...>
  <anyframe:setProperty name="upload" value="true" />
</anyframe:submit>
```

다음은 upload.jsp를 include하고 있는 form.jsp 페이지를 웹 브라우저를 통해 본 UI 모습이다. Browse 버튼을 클릭하여 파일을 첨부할 수 있다.



또한, uploadify()의 multi 속성을 true로 설정하면 여러 개의 파일을 업로드할 수 있다. (현재는 1개 파일 업로드 가능)

```
$("#uploadify").uploadify({
  ..
  multi : false,
  ...
})
```

업로드된 그림 파일을 JSP에서 확인하기 위해 c tag를 사용하여 아래와 같이 작성한다.

```
<c:forTokens var="token" items="${movie.filePaths}" delims=",">
  
  border="0" width="80" height="100" />
</c:forTokens>
```