密级： 版本： v1.0

文档作者：刘浪：liulang@mochasoft.com.cn 提交时间： 2010-08-24

JSON+JQuery 实现

页面动态加载与表单内容异步提交

1 概述

本文主要是描述使用 JSON+JQuery_AJAX 实现页面动态加载与页面表单数据的异步保存。首先页面通过调用 JQuery_AJAX 方法访问后台 Action，在后台 Action 中将获取到的 JavaBean、List、Map 等数据类型转换为 json-lib.jar 中的 `JSONObject` 和 `JSONArray` 类型，将格式化后的数据传给页面由 js 来处理。然后页面通过调用 jquery_ajaxSubmit 方法将页面表单中的数据传递给后台 Action 处理。

2 详述

2.1 依赖的 jar 包和 js 库文件（以下均为测试的环境，框架与 jar 包的版本均可以调整）

2.1.1. 该功能用于 J2EE 项目（web 项目）中，框架用的是 Spring+Struts+Ibatis。

2.1.2. JSON 依赖的 jar 包如下：

commons-httpclient-3.1.jar

commons-lang-2.4.jar

commons-logging-1.1.1.jar

json-lib-2.2.3-jdk13.jar

ezmorph-1.0.6.jar

commons-collections-3.2.1.jar

2.1.3. Jquery_AJAX 依赖的 js 如下：

jquery-1.3.1.js

jquery.form.js

2.2 JSON 格式化数据（主要是方便 js 操纵数据）

2.2.1. JSON 可以格式化的数据：

Java 中的数据类型	JSON 类型
普通的 JavaBean（有些特殊处理的不能被格式化，具体原因在研究 ing）	JSONObject
Map	JSONObject
List	JSONArray

常用的主要就以上三中类型

2.2.2. 普通的 JavaBean 格式化为 JSONObject：

1、建 2 个简单的 javaBean（TextBean.java，TextBean1.java）具体代码如下：

```
import java.util.Date;

public class TextBean {

    private String propertys1;

    private String propertys2;

    private Date propertys3;

    private TextBean1 tb1;

    public TextBean1 getTb1() {

        return tb1;

    }

    public void setTb1(TextBean1 tb1) {

        this.tb1 = tb1;

    }

    public Date getPropertys3() {

        return propertys3;

    }

    public void setPropertys3(Date propertys3) {

        this.propertys3 = propertys3;
```

```
    }
    public final String getPropertys1() {
        return propertys1;
    }
    public final void setPropertys1(String propertys1) {
        this.propertys1 = propertys1;
    }
    public String getPropertys2() {
        return propertys2;
    }
    public final void setPropertys2(String propertys2) {
        this.propertys2 = propertys2;
    }
}


public class TextBean1 {
    private String p1;
    private String p2;
    public String getP1() {
        return p1;
    }
    public void setP1(String p1) {
        this.p1 = p1;
    }
    public String getP2() {
        return p2;
    }
    public void setP2(String p2) {
        this.p2 = p2;
```

```
    }

}
```

2、在 Action 中格式化 TextBean，具体代码如下：

```
        TextBean tb = new TextBean();

        TextBean1 tb1 = new TextBean1();

        tb1.setP1("tb11111");

        tb1.setP2("tb2222222222");

        tb.setTb1(tb1);

        tb.setPropertys1("abcdef");

        tb.setPropertys2("123456");

        tb.setPropertys3(new Date());

        JSONObject jsonObj = JSONObject.fromObject(tb);
```

2.2.3. Map 格式化为 JSONObject：

```
        TextBean tb = new TextBean();

        TextBean1 tb1 = new TextBean1();

        tb1.setP1("tb11111");

        tb1.setP2("tb2222222222");

        tb.setTb1(tb1);

        tb.setPropertys1("abcdef");

        tb.setPropertys2("123456");

        tb.setPropertys3(new Date());

        Map map = new HashMap();

        map.put("textBean", tb);

        map.put("type", "map");

        JSONObject jsonObj = JSONObject.fromObject(map);
```

2.2.4. List 格式化为 JSONArray：

```
        TextBean tb = new TextBean();
        TextBean1 tb1 = new TextBean1();
        tb1.setP1("tb11111");
        tb1.setP2("tb2222222222");
        tb.setTb1(tb1);
        tb.setPropertys1("abcdef");
        tb.setPropertys2("123456");
        tb.setPropertys3(new Date());
        Map map = new HashMap();
        map.put("type", "map");
        List list = new ArrayList();
        list.add(tb);
        list.add(map);
        JSONArray jsonArray = JSONArray.fromObject(list);
```

2.2.5. 将 JSON 数据传递给页面，代码如下：

```
public void writeToResponse(HttpServletResponse response, JSONObject
jsonObj) {
        PrintWriter out = null;
        response.setContentType("text/html;charset=utf-8");
        try {
            out = response.getWriter();
            out.print(jsonObj);
            out.flush();
        } catch (IOException e) {
            e.printStackTrace();
```

```
        } finally {
            out.close();
        }
    }

    public void writeToResponse(HttpServletResponse response, JSONArray
jsonArr) {
        PrintWriter out = null;
        response.setContentType("text/html;charset=utf-8");
        try {
            out = response.getWriter();
            out.print(jsonArr);
            out.flush();
        } catch (IOException e) {
            e.printStackTrace();
        } finally {
            out.close();
        }
    }
```

2.3 页面通过 Jquery_AJAX 获取数据

2.3.1. Jquery_AJAX（使用前必须在页面引入 Jquery jquery-1.3.1.js）:

```
function getTextData() {
    $.ajax({
        url:
'/perform/jsp/perform/manager/editForm/pmpmy/PmpmyDataBGVAction.do?me
thod=getTextData',
```

```
        data: '',
        type: 'post',
        dataType: 'json',
        contentType: 'application/json;charset=utf-8',
        cache: false,
        success: function(data) {
            alert(data.propertys3.year);
        },
        error: function(xhr) {
            alert(xhr);
        }
    });
}
```

url：ajax 方法需要访问的后台 action 的处理方法，可以在后面跟参数。

data：传给后台处理 action 方法的参数。格式为{'key1':'value1', 'key2':'value2'}或"key1=value1&key2=value2"。（只有当 type 参数为 get 时 data 传的参数才起左右）

type：post/get

dataType：json（表明传递给后台的数据类型，以及后台传递来的数据类型）

success: function(data) { } 访问成功后执行的方面，其中参数data为后台action返回的数据。

2.3.2. 获取 JSONObject 格式的数据：

1) 获取javaBean数据（如TextBean：data.propertys2/data.tb1.p1等）
2) 获取Map数据（如上面Map： data.type/data.textBean.protertys2/ data.textBean.tb1.p1等）
3) 获取List数据（如上面List： data[0]. propertys2/data[0] .tb1.p1/data[1].type等）
4) 获取List的长度：data.length(data为js中的Array)

2.3.3. 当数据量比较大的时候要用 **post** 方法，此时如果要传递给 **action** 参数就必须要在 **url** 后面添加，如果要用如 **2** 中这样方便的操纵数据 **dataType** 就要指定为 **json**。

2.4 通过 jquery_ajaxSubmit 异步提交表单数据

2.4.1. jquery_ajaxSubmit（使用前必须在页面引入 Jquery jquery.form.js）：

```
function o_save(){  //用于保存 表单内容
    var succFalg=false;
    if(!checkForm()) {
        return false;
    }
    $("#form1").ajaxSubmit({
        async: false ,
        success: function(data) { succFalg=true;},
        url: '/perform/jsp/DRWPmpmyearDate.do?method=saveDate'
    });
   if(!succFalg){
        alert('存储数据出现异常！');
   }
    return succFalg;
}
```

```
$("#form1").ajaxSubmit({
  async: false ,
  success: function(data) {},
  url: ''
}) 用于异步提交表单 form1 中的数据到 url 指定的 action 方法。
```

3 分析

可以将这这些功能做成一个易用的框架，在下正在努力 working。

4 总结

5 附录

♥特别感谢我的老爸、老妈、老婆♥