05-SpringMVC基于注解使用：JSON处理

json数据格式回顾：
java转换为json 的过程一般会称为 “序列化”
json转换为java 的过程一般会称为 “反序列化”
Json的属和字符串值 必须要用双引号“” 不能用单引号

java	json
String	"xxx"
Integer	123
javaBean\Map User 属性：id 、 name	{"id":1,"name","xushu"}
数组、集合： String[] \ List<String>	["a","b","c"]
List<User> List<Map>	[{"id":1,"name","xushu"}, {"id":1,"name","xushu"}, {"id":1,"name","xushu"}]
User 属性 id name Role role	{"id":1,"name","xushu","role":{"id":1,"name":"管理员"}}
User 属性 id name List<Role>	{"id":1,"name","xushu","role":[{"id":1,"name":"管理员"} {"id":2,"name":"普通员工"}]}

1、SpringMVC的返回JSON数据

到目前为止我们编写的所有Controller的方法的返回值都是String类型，但是大家应该都知道，我们有时候数据传递特别是在ajax中，我们返回的数据经常需要使用json，那么如何来保证返回的数据的是json格式呢？使用@ResponseBody注解

pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xs
    <modelVersion>4.0.0</modelVersion>

    <groupId>cn.tulingxueyuan</groupId>
    <artifactId>springmv_ajax</artifactId>
    <version>1.0-SNAPSHOT</version>
    <dependencies>
        <!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-webmvc</artifactId>
            <version>5.2.3.RELEASE</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-core -
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-core</artifactId>
            <version>2.10.3</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-databi
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-databind</artifactId>
            <version>2.10.3</version>
        </dependency>
        <!-- https://mvnrepository.com/artifact/com.fasterxml.jackson.core/jackson-annotat
        <dependency>
            <groupId>com.fasterxml.jackson.core</groupId>
            <artifactId>jackson-annotations</artifactId>
            <version>2.10.3</version>
        </dependency>
    </dependencies>
</project>
```

springmvc.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<beans xmlns="http://www.springframework.org/schema/beans"
       xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
       xmlns:context="http://www.springframework.org/schema/context"
       xmlns:mvc="http://www.springframework.org/schema/mvc"

       xsi:schemaLocation="http://www.springframework.org/schema/beans http://www.springfr

    <context:component-scan base-package="cn.tulingxueyuan"></context:component-scan>

    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
        <property name="prefix" value="/WEB-INF/page/"></property>
        <property name="suffix" value=".jsp"></property>
    </bean>
    <mvc:default-servlet-handler></mvc:default-servlet-handler>
    <mvc:annotation-driven></mvc:annotation-driven>
</beans>
```

JsonController.java

```java
package cn.tulingxueyuan.controller;

import cn.tulingxueyuan.bean.User;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.ResponseBody;

import java.util.ArrayList;
import java.util.Date;
import java.util.List;

@Controller
public class JsonController {

    @ResponseBody
    @RequestMapping("/json")
    public List<User> json(){
        List<User> list = new ArrayList<User>();
        list.add(new User(1,"zhangsan",12,"男",new Date(),"1234@qq.com"));
        list.add(new User(2,"zhangsan2",12,"男",new Date(),"1234@qq.com"));
```

```
        list.add(new User(3,"zhangsan3",12,"男",new Date(),"1234@qq.com"));
        return list;
    }
}
```

User.java

```
package cn.tulingxueyuan.bean;

import com.fasterxml.jackson.annotation.JsonFormat;
import com.fasterxml.jackson.annotation.JsonIgnore;

import java.util.Date;

public class User {

    private Integer id;
    private String name;
    private Integer age;
    private String gender;
    @JsonFormat( pattern = "yyyy-MM-dd")
    private Date birth;
    @JsonIgnore
    private String email;

    public User() {
    }

    public User(Integer id, String name, Integer age, String gender, Date birth, String em
        this.id = id;
        this.name = name;
        this.age = age;
        this.gender = gender;
        this.birth = birth;
        this.email = email;
    }

    public Integer getId() {
        return id;
    }
```

```
    }
```

```
34
35      public void setId(Integer id) {
36          this.id = id;
37      }
38
39      public String getName() {
40          return name;
41      }
42
43      public void setName(String name) {
44          this.name = name;
45      }
46
47      public Integer getAge() {
48          return age;
49      }
50
51      public void setAge(Integer age) {
52          this.age = age;
53      }
54
55      public String getGender() {
56          return gender;
57      }
58
59      public void setGender(String gender) {
60          this.gender = gender;
61      }
62
63      public Date getBirth() {
64          return birth;
65      }
66
67      public void setBirth(Date birth) {
68          this.birth = birth;
69      }
70
71      public String getEmail() {
72          return email;
73      }
```

```
74
75      public void setEmail(String email) {
76          this.email = email;
77      }
78
79      @Override
80      public String toString() {
81          return "User{" +
82                  "id=" + id +
83                  ", name='" + name + '\'' +
84                  ", age=" + age +
85                  ", gender='" + gender + '\'' +
86                  ", birth=" + birth +
87                  ", email='" + email + '\'' +
88                  '}';
89      }
90  }
```

同时@ResponseBody可以直接将返回的字符串数据作为响应内容

```
1  package cn.tulingxueyuan.controller;
2
3  import cn.tulingxueyuan.bean.User;
4  import org.springframework.http.HttpEntity;
5  import org.springframework.stereotype.Controller;
6  import org.springframework.web.bind.annotation.RequestBody;
7  import org.springframework.web.bind.annotation.RequestMapping;
8  import org.springframework.web.bind.annotation.ResponseBody;
9
10 @Controller
11 public class OtherController {
12     @ResponseBody
13     @RequestMapping("/testResponseBody")
14     public String testResponseBody(){
15         return "<h1>success</h1>";
16     }
17 }
```

2、SpringMVC的获取JSON数据

ajax我们经常用到，传的数据是json数据，json数据又有对象，数组。以下给大家总结了4种常用的获取json方式：

ajax.jsp

前端

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
  <head>
    <title>$Title$</title>
    <script src="https://cdn.staticfile.org/jquery/3.5.1/jquery.min.js"></script>
    <script type="text/javascript">
      // 也没加载事件简写方式
      $(function(){

        $("#btnJson1").click(function(){
          $.ajax({
            url:"${pageContext.request.contextPath}/json/request01",
            method:"post",
            data:"张三",
            contentType:'application/json',
            dataType:"json",
            success:function(user){
              alert(user.name);
            }
          });
        });



        $("#btnJson2").click(function(){

          var user={'id':'1','name':'张三'};   // 定义js对象
          var jsonValue=JSON.stringify(user); // 对象转换为json字符串
          console.log(jsonValue)
          $.ajax({
            url:"${pageContext.request.contextPath}/json/request02",

            method:"post",
```

```
            data:'{"id":"1","name":"张三","birthady":"2019-01-01"}',
            contentType:'application/json',
            dataType:"json",
            success:function(user){
              alert(user.name);
            }
          });
        });


        $("#btnJson3").click(function(){

          $.ajax({
            url:"${pageContext.request.contextPath}/json/request03",
            method:"post",
            data:'{"idxx":"1","namexx":"张三","birthadyxx":"2019-01-01"}',
            contentType:'application/json',
            dataType:"json",
            success:function(user){
              alert(user.name);
            }
          });
        });


        $("#btnJson4").click(function(){
          var listUser=new Array();
          var user1={"id":"1","name":"张三","birthady":"2019-01-01"};
          var user2={"id":"2","name":"李四","birthady":"2019-01-01"};
          listUser.push(user1)
          listUser.push(user2)

          $.ajax({
            url:"${pageContext.request.contextPath}/json/request04",
            method:"post",
            //data:'[{"id":"1","name":"张三","birthady":"2019-01-01"},{"id":"2","name":"
            data:JSON.stringify(listUser),
            contentType:'application/json',

            dataType:"json",
```

```
            success:function(user){
              alert(user.name);
            }
          });
        });

      })

    </script>
  </head>
  <body>
<input type="button" value="发送单个参数的json数据" id="btnJson1"/><br/>
<input type="button" value="发送对象的json数据用javaBean接收" id="btnJson2"/><br/>
<input type="button" value="发送对象的json数据用Map接收" id="btnJson3"/><br/>
<input type="button" value="发送数组对象的json数据用List<User>接收" id="btnJson4"/><br/>
  </body>
</html>
```

1、以@RequestBody接收

前端传来的是json数据不多时：

```
@PostMapping("/json/request01")
@ResponseBody
public  User responseJson(@RequestBody String name){
    User user = new User(1, "徐庶","12346",new Date());
    System.out.println(name);
    return user;

}
```

2、以实体类方式接收

前端传来的是一个json对象时：{ id:1,name:xx},可以用实体类直接进行自动绑定

```
@PostMapping(value="/json/request02",consumes = "application/json")
@ResponseBody
public  User requestJson02(@RequestBody User user){
    User user2 = new User(1, "徐庶","12346",new Date());
    System.out.println(user);
    return user2;
```

```
7
8 }
```

前端

3、以Map接收

前端传来的是一个json对象时：{ id:1,name:xx},可以用Map来获取

```
1
2 @R
3 @PostMapping(value="/json/request03",consumes = "application/json")
4 @ResponseBody
5 public  User requestJson03(@RequestBody Map<String,String> map){
6     User user2 = new User(1, "徐庶","12346",new Date());
7     System.out.println(map);
8     return user2;
9
10 }
```

4、以List接收

当前端传来这样一个json数组：[{ id:1,name:xx},{ id:1,name:xx},{ id:1,name:xx},...]时，用List<E>接收

```
1
2
3 @PostMapping(value="/json/request04",consumes = "application/json")
4 @ResponseBody
5 public  User requestJson04(@RequestBody List<User> list){
6     User user2 = new User(1, "徐庶","12346",new Date());
7     System.out.println(list);
8     return user2;
9
10 }
```