**jquery ajax向spring mvc controller中传值并接受及解析返回值**

2016年07月16日 23:31:44 [Believe勤能补拙](https://me.csdn.net/a355586533) 阅读数：49544 标签： [jquery ajax](http://so.csdn.net/so/search/s.do?q=jquery%20ajax&t=blog" \t "_blank)[controller 传值](http://so.csdn.net/so/search/s.do?q=controller%20%E4%BC%A0%E5%80%BC&t=blog) 更多

个人分类： [jquery](https://blog.csdn.net/a355586533/article/category/2546675" \t "_blank)[spring MVC](https://blog.csdn.net/a355586533/article/category/5870807)

版权声明：本文为博主原创文章，未经博主允许不得转载。 https://blog.csdn.net/a355586533/article/details/51927045

**第一种传值：**

controller中是几个单独的基本类型参数

Spring MVC-controller

1. @RequestMapping("update")
2. @ResponseBody*//此注解不能省略 否则ajax无法接受返回值*
3. public Map<String,Object> update(Long num, Long id, BigDecimal amount){
5. Map<String,Object> resultMap = new HashMap<String, Object>();
6. if(num == null || id == null || amount == null){
7. resultMap.put("result", "参数不合法！");
8. return resultMap;
9. }
10. *//xxx逻辑处理*
11. resultMap.put("result", "操作成功");
12. return resultMap;
13. }

jquery ajax

1. var params = {};
2. *//params.XX必须与Spring Mvc controller中的参数名称一致*
3. *//否则在controller中使用@RequestParam绑定*
4. params.num = num;
5. params.id = id;
6. params.amount = amount;
7. $.ajax({
8. async:false,
9. type: "POST",
10. url: "price/update",*//注意路径*
11. data:params,
12. dataType:"json",
13. success:function(data){
14. if(data.result=='SUCCESS'){
15. alert("修改成功");
16. }else{
17. alert("修改失败，失败原因【" + data + "】");
18. }
19. },
20. error:function(data){
21. alert(data.result);
22. }
23. });

**第二种传值：**

controller中是参数是实体bean，bean中属性都是基本数据类型

Spring MVC-controller

1. @RequestMapping("add")
2. @ResponseBody*//此处不能省略 否则ajax无法解析返回值*
3. public Map<String,Object> add(DataVo dataVo){
4. Map<String, Object> result = null;
5. if(dataVo.getNum() == null || StringUtils.isBlank(dataVo.geId())){
6. result = new HashMap<String, Object>();
7. result.put("msg", "参数不合法！");
8. return result;
9. }
10. *//xxx业务逻辑处理*
12. return result;
13. }

实体bean DataVo

1. public class DataVo {
2. */\*\**
3. *\* 编号*
4. *\*/*
5. private Long num;
6. */\*\**
7. *\* id*
8. *\*/*
9. private String id;
11. public Long getNum() {
12. return num;
13. }
14. public void setNum(Long num) {
15. this.num = num;
16. }
17. public String getId() {
18. return id;
19. }
20. public void setId(String id) {
21. this.id = id;
22. }
23. }

jquery ajax

1. var params = {};
2. params.num = $("#num").val();
3. params.id = $("#id").val();*//注意params.名称 名称与实体bean中名称一致*
4. $.ajax({
5. type: "POST",
6. url: "price/add",
7. data:params,
8. dataType:"json",
9. *// contentType: "application/json; charset=utf-8",//此处不能设置，否则后台无法接值*
10. success:function(data){
11. if(data.msg != ""){
12. alert( data.msg );
13. }
14. },
15. error:function(data){
16. alert("出现异常，异常原因【" + data + "】!");
17. }
18. });

**第三种传值：**

controller中是参数是实体bean，bean中属性有有数组或者自定义bean

Spring MVC-controller

1. @RequestMapping("add")
2. @ResponseBody*//此处不能省略 否则ajax无法解析返回值*
3. public Map<String,Object> add(@RequestBody DataVo dataVo){*//@RequestBody注解不能省略，否则无法接值*
4. Map<String,Object> resultMap = new HashMap<String, Object>();
5. *//业务逻辑处理*
6. return resultMap;
7. }

实体 DataVo

1. public class DataVo {

4. private BigDecimal[] nums;
5. private String id;
7. public Long getId() {
8. return id;
9. }
11. public void setId(Long id) {
12. this.id = id;
13. }
15. public BigDecimal[] getNums() {
16. return nums;
17. }
19. public void setNums(BigDecimal[] nums) {
20. this.nums = nums;
21. }
23. }

jquery ajax  需要jquery json的插件  进行json序列化，我这里使用了json.js

且配置

datatype:"json",

contentType: "application/json; charset=utf-8",

1. var params = {};
2. params.nums = [];
3. params.id = $("#id").val();*//parmas.参数名 注意与实体bean参数名称相同*
4. var prices = document.getElementsByName("prices");*//prices 是name="prices"一组input标签*
5. for (var i = 0; i < prices.length; i++) {
6. params.nums[i] = prices[i].value;
7. }
8. $.ajax({
9. type: "POST",
10. url: "price/add",
11. data:JSON.stringify(params),*//json序列化*
12. datatype:"json", *//此处不能省略*
13. contentType: "application/json; charset=utf-8",*//此处不能省略*
14. success:function(data){
15. alert(data);
16. },
17. error:function(data){
18. alert(data)
19. }
20. });