Passing A List Of Objects Into An MVC Controller Method Using jQuery Ajax

Ask Question

97

39

I'm trying to pass an array of objects into an MVC controller method using jQuery's ajax() function. When I get into the PassThing() C# controller method, the argument "things" is null. I've tried this using a type of List for the argument, but that doesn't work either. What am I doing wrong?

```
<script type="text/javascript">
    $(document).ready(function () {
        var things = [
            { id: 1, color: 'yellow' },
            { id: 2, color: 'blue' },
            { id: 3, color: 'red' }
        ];

        $.ajax({
            contentType: 'application/json; charset=utf-8',
            dataType: 'json',
            type: 'POST',
            url: '/Xhr/ThingController/PassThing',
            data: JSON.stringify(things)
        });
    });
</script>

public class ThingController : Controller
{
    public void PassThing(Thing[] things)
    {
        // do stuff with things here...
    }

    public class Thing
    {
        public int id { get; set; }
        public string color { get; set; }
    }
}
```

c# asp.net-mvc jquery

edited Nov 6 '12 at 0:56
tereško
52.2k 20 77 135

asked Nov 6 '12 at 0:01
Halcyon
7,458 14 54 81

You can try

accepts an array. Change your method to accept a string, then deserialize it within the method. – Bob Horn Nov 6 '12 at 0:36

2 Your code is correct. I tested it and it worked using MVC 4. Please provide more data to figure it out. – Diego Nov 6 '12 at 1:47

This is great stuff but what if you need not just a list of strings to pass but need to include a separate id associated with the list of strings? So like, group id, list of groups under group id. – Nathan McKaskle Aug 16 '17 at 15:26

11 Answers

153

Using NickW's suggestion, I was able to get this working using `things = JSON.stringify({ 'things': things });` Here is the complete code.

```
$(document).ready(function () {
    var things = [
        { id: 1, color: 'yellow' },
        { id: 2, color: 'blue' },
        { id: 3, color: 'red' }
    ];

    things = JSON.stringify({ 'things

    $.ajax({
        contentType: 'application/jso
        dataType: 'json',
        type: 'POST',
        url: '/Home/PassThings',
        data: things,
        success: function () {
            $('#result').html('"PassT
        },
        failure: function (response)
            $('#result').html(respons
        }
    });
});
```

```
public void PassThings(List<Thing> th
{
    var t = things;
}

public class Thing
{
    public int Id { get; set; }
    public string Color { get; set; }
}
```

There are two things I learned from this:

1. The contentType and dataType settings are absolutely necessary in the ajax() function. It won't work if they are missing.

2. To pass in an array of objects to an MVC controller method, simply use the JSON.stringify({ 'things': things }) format.

I hope this helps someone else!

edited Feb 21 '18 at 10:57
Doomsknight
10.4k 15 63 130

answered Nov 6 '12 at 16:38
Halcyon
7,458 14 54 81

7 I was having the same problem and adding the contentType fixed it. Thanks! – chill182 Mar 19 '13 at 14:26

9 Two things to note: JSON.stringify and specifying 'contentType'. – dinesh ygv Nov 7 '14 at 5:40

Crud. Still not working for me. my request URL is `http://localhost:52459/Sales/completeSale?itemsInCart=[{"ItemId":1,"Quantity":"1","Price":3.5}]` and `Sales.completeSale` is `public ActionResult completeSale(ItemInCart[] itemsInCart)` , annotated as a `HttpGet` . – abalter Jun 29 '15 at 17:09

2 for whatever reason I had to to just use `data: JSON.stringify(things),` – Rob Scott Aug 5 '15 at 2:33

1 `dataType` is not necessary. If its omitted, the ajax function will work it out based on the return data – user3559349 Aug 4 '17 at 8:54

25

Couldn't you just do this?

```
var things = [
    { id: 1, color: 'yellow' },
    { id: 2, color: 'blue' },
    { id: 3, color: 'red' }
];
$.post('@Url.Action("PassThings")', {
   function () {
        $('#result').html('"PassThings
   });
```

and mark your action with

```
    // do stuff with things here...
}
```

answered Mar 26 '15 at 16:28

lanternmarsh
408 5 8

2 This should be the best answer. The JSON.stringify should not be used in this case – Dummy Apr 25 '16 at 9:52

This is not working for me..I am using [HttpPost] public int SaveResults(List<ShortDetail> model) {} and $.post("@Url.Action("SaveResults", "Maps")", {model: dataItems}, function (result) { }); – Samra Jul 7 '17 at 5:46

It worked for me. Absolutely the best answer. I don't know why the Halcyon implementation didn't work. The PassThings function was invoked but the 'things' input variable was empty even if it was filled in the javascript just before the call. – Leonardo Daga Sep 3 '17 at 22:46

Only this worked for me. Most simplest answer. – RP Nainwal Nov 30 '18 at 15:19

11

Formatting your data that may be the problem. Try either of these:

```
data: '{ "things":' + JSON.stringify(t
```

Or (from [How can I post an array of string to ASP.NET MVC Controller without a form?](https://stackoverflow.com/))

```
var postData = { things: things };
...
data = postData
```

edited May 23 '17 at 11:47

Community ♦
1 1

answered Nov 6 '12 at 0:51

nick_w
12.7k 2 41 60

Your code is close, but it doesn't work. I was able to get the code working thanks to your suggestion. See my answer above. – Halcyon Nov 6 '12 at 16:39

7

`JavaScriptSerializer.Deserialize<T>(string input)` , which is pretty strange if that's the default deserializer for MVC 4.

My model has nested lists of objects and the best I could get using JSON data is the uppermost list to have the correct number of items in it, but all the fields in the items were null.

This kind of thing should not be so hard.

```
    $.ajax({
        type: 'POST',
        url: '/Agri/Map/SaveSelfValuat
        data: { json: JSON.stringify(m
        dataType: 'text',
        success: function (data) {

    [HttpPost]
    public JsonResult DoSomething(stri
    {
        var model = new JavaScriptSeri
```

edited Jul 19 '17 at 5:14

Nathan Tuggy
2,188 9 25 33

answered Jul 19 '17 at 4:50

user1650892
133 2 3

To make this work, follow the format of the Ajax call closely. – Graham Laight Jul 24 '18 at 13:10

6

I have perfect answer for all this : I tried so many solution not able to get finally myself able to manage , please find detail answer below:

```
       $.ajax({
            traditional: true,
            url: "/Conroller/MethodTes
            type: "POST",
            contentType: "application/
            data:JSON.stringify(
               [
                { id: 1, color: 'yello
                { id: 2, color: 'blue'
                { id: 3, color: 'red'
                ]),
            success: function (data) {
                $scope.DisplayError(da
            }
        });
```

Controler

```
public class Thing
{
    public int id { get; set; }
```

```
   {
  //now  datav is having all your val
 }
```

edited Sep 7 '15 at 14:45

answered Jul 27 '15 at 14:42

veeresh i
554 5 12

4

This is working code for your query,you can use it.

Controler

```
    [HttpPost]
    public ActionResult save(List<List
    {
    //operation return
    Json(new { istObject }, JsonReques
    }
```

javascript

```
  $("#btnSubmit").click(function () {
    var myColumnDefs = [];
    $('input[type=checkbox]').each(fur
        if (this.checked) {
            myColumnDefs.push({ 'Statu
        } else {
            myColumnDefs.push({ 'Statu
        }
    });
   var data1 = { 'listObject': myColum
   var data = JSON.stringify(data1)
   $.ajax({
   type: 'post',
   url: '/Controller/action',
   data:data ,
   contentType: 'application/json; cha
   success: function (response) {
    //do your actions
   },
   error: function (response) {
    alert("error occured");
   }
   });
```

edited Mar 3 '17 at 4:00

answered Mar 2 '17 at 13:55

sach4all
41 4

4

I am using a .Net Core 2.1 Web Application and could not get a single answer here to work. I either got a blank parameter (if the method was combination of answers and finally got a working result.

In my case the solution was as follows:

Script - stringify the original array (without using a named property)

```
$.ajax({
    type: 'POST',
    contentType: 'application/jsor
    url: mycontrolleraction,
    data: JSON.stringify(things)
});
```

And in the controller method, use [FromBody]

```
[HttpPost]
public IActionResult NewBranch([Fr
{
    return Ok();
}
```

Failures include:

- Naming the content

  data: { content: nodes }, // Server error 500

- Not having the contentType = Server error 500

Notes

- `dataType` is not needed, despite what some answers say, as that is used for the *response* decoding (so not relevant to the *request* examples here).
- `List<Thing>` also works in the controller method

answered Jun 10 '18 at 19:45

Gone Coding
78.1k 18 150 170

1

```
    var List = @Html.Raw(Json.Encode(
$.ajax({
    type: 'post',
    url: '/Controller/action',
    data:JSON.stringify({ 'item': List
    contentType: 'application/json; ch
    success: function (response) {
        //do your actions
    },
    error: function (response) {
        alert("error occured");
```

answered Jan 4 '17 at 9:32

Athul Nalupurakkal
179 1 7

Try this code for passing List Of model objects using ajax. Model represents the IList<Model>. Use IList<Model> in controller to get the values. – Athul Nalupurakkal Jan 16 '17 at 8:00

0

If you are using ASP.NET Web API then you should just pass `data: JSON.stringify(things)` .

And your controller should look something like this:

```
public class PassThingsController : Ap
{
    public HttpResponseMessage Post(Li
    {
        // code
    }
}
```

answered Dec 25 '15 at 10:07

FleGMan
46 1 5

0

Modification from @veeresh i

```
 var data=[

                        { id: 1, color
                        { id: 2, color
                        { id: 3, color
                        ]; //parameter
        var para={};
        para.datav=data;   //datav fro


        $.ajax({
                    traditional: true,
                    url: "/Conroller/M
                    type: "POST",
                    contentType: "appl
                    data:para,
                    success: function
                        $scope.Display
                    }
                });

In MVC



public class Thing
    {
        public int id { get; set; }
        public string color { get; set
    }

    public JsonResult MethodTest(IE
```

By using our site, you acknowledge that you have read and understand our , , and our .

answered Apr 24 '17 at 7:56

Minhajul Islam
59 9

0

Wrapping your list of objects with another object containing a property that matches the name of the parameter which is expected by the MVC controller works. The important bit being the wrapper around the object list.

```
$(document).ready(function () {
    var employeeList = [
        { id: 1, name: 'Bob' },
        { id: 2, name: 'John' },
        { id: 3, name: 'Tom' }
    ];

    var Employees = {
      EmployeeList: employeeList
    }

    $.ajax({
        dataType: 'json',
        type: 'POST',
        url: '/Employees/Process',
        data: Employees,
        success: function () {
            $('#InfoPanel').html('It w
        },
        failure: function (response) {
            $('#InfoPanel').html(respo
        }
    });
});


public void Process(List<Employee> Emp
{
    var emps = EmployeeList;
}

public class Employee
{
    public int Id { get; set; }
    public string Name { get; set; }
}
```

answered Sep 21 '17 at 22:06

Hoodlum
398 4 14