Using Angular $http.put with ASP.NET MVC project

## Background

While working with AngularJS $http services, I was happy to see how we could send and receive data using $http.post and $http.get methods however when it comes to $http.get and $http.delete it was a nightmare. I also [posted question about this in Stackoverflow](http://stackoverflow.com/questions/31263554/angularjs-http-put-doesnt-work-with-asp-net-mvc-controller) but found not much help.   
   
***Error we may receive while using $http.put in ASP.NET MVC application is***

*The directory or file specified does not exist on the Web server. The URL contains a typographical error. A custom filter or module, such as URLScan, restricts access to the file. Things you can try: Create the content on the Web server. Review the browser URL. Check the failed request tracing log and see which module is calling SetStatus.*

Ultimately after a lot of struggle we got the solution and here it is.

## Submitting data to ASP.NET MVC using $http.get or $http.delete

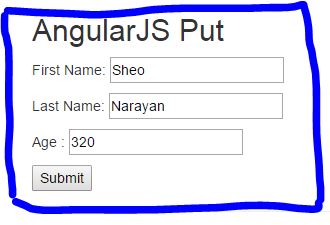
##### HTML Code

To demonstrate how to post data using AngularJS $http.post and $http.delete in ASP.NET MVC, let's create a simple form with three text boxes (FirstName, LastName and Age) and the form looks like this

<div ng-app="myApp" ng-controller="HttpPutController">  
    <form ng-submit="SendHttpPutData()">  
        <p>First Name: <input type="text" name="firstName" ng-model="firstName" required /></p>  
        <p>Last Name: <input type="text" name="lastName" ng-model="lastName" required /></p>  
        <p>Age : <input type="number" name="age" ng-model="age" required /></p>  
        <input type="submit" value="Submit" />  
        <hr />  
        {{ ServerResponse }}  
    </form>  
</div>

Above AngularJS form is pretty simple with all text boxes have a ng-model directive and ServerResponse is being written below the form.

When the form is submitted, ng-submit directive works and it calls "SendHttpPutData()" function declared in the AngularJS controller (in this case HttpPutController controller).



##### AngularJS JavaScript code

Now, let's see the AngularJS javascript code. In below code snippet, we have a controller defined for "myApp". In that SendHttpPutData() function is defined. Inside the function, we have formed a data variable with the help of $.param [jQuery](http://techfunda.com/Howto/jquery) function.

The $.param jQuery function receives JSON object as parameter and convert it to HTML Form data string so that it can be sent to the server.

Next, we have declared a config variable that is setting the headers to *x-www-form-urlencoded;charset=utf-8* that is passed to the $http.put method to set the type of content being sent to the server. Note that by default AngularJS sends JSON content type header to the server that is not understood by the server unless server accept JSON data.

<script>  
    var myApp = angular.module("myApp", []);  
    myApp.controller("HttpPutController", function ($scope, $http) {  
  
        $scope.SendHttpPutData = function () {  
  
            var data = $.param({  
                firstName: $scope.firstName,  
                lastName: $scope.lastName,  
                age : $scope.age  
            });  
  
            var config = {  
                headers: {  
                    'Content-Type': 'application/x-www-form-urlencoded;charset=utf-8;'  
                }  
            }  
  
            $http.put('/ServerRequest/PutDataResponse', data, config)  
            .success(function (data, status, headers, config) {  
                $scope.ServerResponse = data;  
            })  
            .error(function (data, status, header, config) {  
                $scope.ServerResponse = "Data: " + data +  
                    "\n\n\n\nstatus: " + status +  
                    "\n\n\n\nheaders: " + header +  
                    "\n\n\n\nconfig: " + config;  
            });  
        };  
  
    });  
</script>

In last part of above code snippet, we are calling $http.put method with following parameters

1. 1st parameter is the url to send put request to
2. 2nd parameter is the data to send along with request
3. 3rd parameter is the configuration to use while sending the request

Next, the success() and error() function is being called based on sever response. In case of success, $scope.ServerResponse property is being set to the data coming from the server.

##### Web.config code change (very important change)

Now, change the following settings in ASP.NET MVC root web.config file. Note the bold code of <add> tag. This tag instruct the ASP.NET MVC framework that our application is ready to accept any verb (by default it accepts only GET and POST) ie. now our application can also accept $http.put or $http.delete.

<system.webServer>  
    <modules>  
      <remove name="FormsAuthentication" />  
    </modules>  
    <handlers>  
      **<add name="ExtensionlessUrlHandler-Integrated-4.0" path="\*." verb="\*" type="System.Web.Handlers.TransferRequestHandler" preCondition="integratedMode,runtimeVersionv4.0" />**  
    </handlers>  
  </system.webServer>

This magic configuration help us to work with any other $http service method of AngularJS. Now when we run the application we should be able to send $http.put and $http.delete request to the server and we can receive these sent data using Request.Form.

##### ASP.NET MVC Controller code

In below code snippet, we are able to receive sent data using Request.Form["dataPropertyName"].

        [HttpPut]  
        public ContentResult PutDataResponse()  
        {  
            return Content("First name: " + Request.Form["firstName"] +  
                " | Last name: " + Request.Form["lastName"] +  
                " | Age: " + Request.Form["age"] +  
                " | Request Type: " + Request.RequestType.ToString());  
        }

|  |
| --- |
| * When should we use PUT and when should we use POST?   The HTTP methods POST and PUT aren't the HTTP equivalent of the CRUD's create and update. They both serve a different purpose. It's quite possible, valid and even preferred in some occasions, to use POST to create resources, or use PUT to update resources.  PUT 比較正確的定義是 Replace (Create or Update)，例如PUT /items/1的意思是替換/items/1，如果已經存在就替換，沒有就新增。PUT 必須包含items/1的所有屬性資料。 |