**What is Web API and why to use it ?**

*Posted By : Shailendra Chauhan, 04 Apr 2013*

*Updated On : 07 Apr 2013*

*Keywords : when to use asp.net web api,advantage of asp.net web api,asp.net web api features,why asp.net web api*

Asp.Net Web API is a framework for building HTTP services that can be consume by a broad range of clients including browsers, mobiles, iphone and tablets. It is very similar to ASP.NET MVC since it contains the MVC features such as routing, controllers, action results, filter, model binders, IOC container or dependency injection. But it is not a part of the MVC Framework. It is a part of the core ASP.NET platform and can be used with MVC and other types of Web applications like Asp.Net WebForms. It can also be used as an stand-alone Web services application.

**Why Asp.Net Web API (Web API) ?**

Today, a web-based application is not enough to reach it's customers. People are very smart, they are using iphone, mobile, tablets etc. devices in its daily life. These devices also have a lot of apps for making the life easy. Actually, we are moving from the web towards apps world.

So, if you like to expose your service data to the browsers and as well as all these modern devices apps in fast and simple way, you should have an API which is compatible with browsers and all these devices.

For example twitter,facebook and Google API for the web application and phone apps.



Web API is the great framework for exposing your data and service to different-different devices. Moreover Web API is open source an ideal platform for building REST-ful services over the .NET Framework. Unlike WCF Rest service, it use the full featues of HTTP (like URIs, request/response headers, caching, versioning, various content formats) and you don't need to define any extra config settings for different devices unlike WCF Rest service.

**Web API Features**

1. It supports convention-based CRUD Actions since it works with HTTP verbs GET,POST,PUT and DELETE.
2. Responses have an Accept header and HTTP status code.
3. Responses are formatted by Web API’s MediaTypeFormatter into JSON, XML or whatever format you want to add as a MediaTypeFormatter.
4. It may accepts and generates the content which may not be object oriented like images, PDF files etc.
5. It has automatic support for OData. Hence by placing the new [Queryable] attribute on a controller method that returns IQueryable, clients can use the method for OData query composition.
6. It can be hosted with in the applicaion or on IIS.
7. It also supports the MVC features such as routing, controllers, action results, filter, model binders, IOC container or dependency injection that makes it more simple and robust.

**Why to choose Web API ?**

1. If we need a Web Service and don’t need SOAP, then ASP.Net Web API is best choice.
2. It is Used to build simple, non-SOAP-based HTTP Services on top of existing WCF message pipeline.
3. It doesn't have tedious and extensive configuration like WCF REST service.
4. Simple service creation with Web API. With WCF REST Services, service creation is difficult.
5. It is only based on HTTP and easy to define, expose and consume in a REST-ful way.
6. It is light weight architecture and good for devices which have limited bandwidth like smart phones.
7. It is open source.

**Difference between ASP.NET MVC and ASP.NET Web API**

*Posted By : Shailendra Chauhan, 05 Apr 2013*

*Keywords : ASP.NET MVC vs ASP.NET Web API,web api vs mvc,mvc4 vs web api,when to use Web API over MVC*

While developing your web application using MVC, many developers got confused when to use Web API, since MVC framework can also return JSON data by using JsonResult and can also handle simple AJAX requests. In previous article, I have explained the [Difference between WCF and Web API and WCF REST and Web Service](http://www.dotnet-tricks.com/Tutorial/webapi/JI2X050413-Difference-between-WCF-and-Web-API-and-WCF-REST-and-Web-Service.html) and when to use Web API over others services. In this article, you will learn when to use [Web API](http://www.dotnet-tricks.com/Tutorial/webapi/VG9K040413-What-is-Web-API-and-why-to-use-it-?.html) with MVC.



**Asp.Net Web API VS Asp.Net MVC**

1. Asp.Net MVC is used to create web applications that returns both views and data but Asp.Net Web API is used to create full blown HTTP services with easy and simple way that returns only data not view.
2. Web API helps to build REST-ful services over the .NET Framework and it also support content-negotiation(it's about deciding the best response format data that could be acceptable by the client. it could be JSON,XML,ATOM or other formatted data), self hosting which are not in MVC.
3. Web API also takes care of returning data in particular format like JSON,XML or any other based upon the Accept header in the request and you don't worry about that. MVC only return data in JSON format using JsonResult.
4. In Web API the request are mapped to the actions based on HTTP verbs but in MVC it is mapped to actions name.
5. Asp.Net Web API is new framework and part of the core ASP.NET framework. The model binding, filters, routing and others MVC features exist in Web API are different from MVC and exists in the newSystem.Web.Http assembly. In MVC, these featues exist with in System.Web.Mvc. Hence Web API can also be used with Asp.Net and as a stand alone service layer.
6. You can mix Web API and MVC controller in a single project to handle advanced AJAX requests which may return data in JSON, XML or any others format and building a full blown HTTP service. Typically, this will be called Web API self hosting.
7. When you have mixed MVC and Web API controller and you want to implement the authorization then you have to create two filters one for MVC and another for Web API since boths are different.
8. Moreover, Web API is light weight architecture and except the web application it can also be used with smart phone apps.

# How to pass javascript complex object to ASP.NET Web Api and MVC

*Posted By : Shailendra Chauhan, 12 Apr 2013*

*Updated On : 13 Apr 2013*

*Keywords : Passing Dynamic JSON Object to Web API or MVC method,Passing a Complex type in JSON*

ASP.NET Web API is one of the most powerful recent addition to ASP.NET framework. Sometimes, you have to post a form data using jQuery-JSON to Web API or MVC method, which have so many input fields. Passing each and every input field data as a separate parameter is not good practice, even when you have a strongly typed-view. The best practice is, pass a complex type object for all the input fields to the server side to remove complexity.

In this article, I am going to explain you how can you pass complex types object to the Web API and MVC method to remove complexity at server side and make it simple and useful.

### Model Classes

Suppose you have the following Product class and repository for product.

1. **public class Product**
2. **{**
3. **public int Id { get; set; }**
4. **public string Name { get; set; }**
5. **public string Category { get; set; }**
6. **public decimal Price { get; set; }**
7. **}**
8.
9. **interface IProductRepository**
10. **{**
11. **Product Add(Product item);**
12. ***//To Do : Some Stuff***
13. **}**
14.
15. **public class ProductRepository : IProductRepository**
16. **{**
17. **private List<Product> products = new List<Product>();**
18. **private int \_nextId = 1;**
19.
20. **public ProductRepository()**
21. **{**
22. ***// Add products for the Demonstration***
23. **Add(new Product { Name = "Computer", Category = "Electronics", Price = 23.54M });**
24. **Add(new Product { Name = "Laptop", Category = "Electronics", Price = 33.75M });**
25. **Add(new Product { Name = "iPhone4", Category = "Phone", Price = 16.99M });**
26. **}**
27.
28. **public Product Add(Product item)**
29. **{**
30. **if (item == null)**
31. **{**
32. **throw new ArgumentNullException("item");**
33. **}**
34.
35. ***// TO DO : Code to save record into database***
36. **item.Id = \_nextId++;**
37. **products.Add(item);**
38.
39. **return item;**
40. **}**
41. ***//To Do : Some Stuff***
42. **}**

### View (Product.cshtml)

1. **<script type="text/javascript">**
2. ***//Add New Item by Web API***
3. **$("#Save").click(function () {**
4.
5. ***//Making complex type object***
6. **var Product = {**
7. **Id: "0",**
8. **Name: $("#Name").val(),**
9. **Price: $("#Price").val(),**
10. **Category: $("#Category").val()**
11. **};**
12.
13. **if (Product.Name != "" && Product.Price != "" && Product.Category != "") {**
14. ***//Convert javascript object to JSON object***
15. **var DTO = JSON.stringify(Product);**
16. **$.ajax({**
17. **url: 'api/product', *//calling Web API controller product***
18. **cache: false,**
19. **type: 'POST',**
20. **contentType: 'application/json; charset=utf-8',**
21. **data: DTO,**
22. **dataType: "json",**
23. **success: function (data) {**
24. **alert('added');**
25. **}**
26. **}).fail(**
27. **function (xhr, textStatus, err) {**
28. **alert(err);**
29. **});**
30.
31. **}**
32. **else {**
33. **alert('Please Enter All the Values !!');**
34. **}**
35.
36. **});**
37.
38. **</script>**
39. **<div>**
40. **<div>**
41. **<h2>Add New Product</h2>**
42. **</div>**
43. **<div>**
44. **<label for="name">Name</label>**
45. **<input type="text" id="Name" title="Name" />**
46. **</div>**
47.
48. **<div>**
49. **<label for="category">Category</label>**
50. **<input type="text" id="Category" title="Category" />**
51. **</div>**
52.
53. **<div>**
54. **<label for="price">Price</label>**
55. **<input type="text" id="Price" title="Price" />**
56. **</div>**
57. **<br />**
58. **<div>**
59. **<button id="Save">Save</button>**
60. **<button id="Reset">Reset</button>**
61. **</div>**
62. **</div>**

### Web API Controller

1. **public class ProductController : ApiController**
2. **{**
3. **static readonly IProductRepository repository = new ProductRepository();**
4. **public Product PostProduct(Product item)**
5. **{**
6. **return repository.Add(item);**
7. **}**
8. **}**

### How it work ?





The same thing you have to done with MVC while calling MVC controller method using jQuery-JSON.