**ASP.NET Web API**

* [Guidance](https://docs.microsoft.com/en-us/aspnet/web-api/overview/)
  + [Getting Started](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/)
    - [Getting Started with Web API 2 (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api)
    - [Action Results in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/action-results)
    - [Using Web API with ASP.NET Web Forms](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/using-web-api-with-aspnet-web-forms)
    - [Creating Web API Help Pages](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/creating-api-help-pages)
    - [Hands On Lab: Build a Single Page Application (SPA) with ASP.NET Web API and Angular.js](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/build-a-single-page-application-spa-with-aspnet-web-api-and-angularjs)
  + [Routing](https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/)
    - [Routing in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/routing-in-aspnet-web-api)
    - [Routing and Action Selection in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/routing-and-action-selection)
    - [Attribute Routing in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/attribute-routing-in-web-api-2)
    - [Create a REST API with Attribute Routing in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/create-a-rest-api-with-attribute-routing)
  + [Working with Data](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/)
    - [Using Web API 2 with Entity Framework 6](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/)
      * [Create the Project](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-1)
      * [Add Models and Controllers](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-2)
      * [Seed the Database](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-3)
      * [Handling Entity Relations](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-4)
      * [Create Data Transfer Objects (DTOs)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-5)
      * [Create the JavaScript Client](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-6)
      * [Create the View (UI)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-7)
      * [Display Item Details](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-8)
      * [Add a New Item to the Database](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-9)
      * [Publish the App to Azure Websites](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-10)
  + [Mobile Clients](https://docs.microsoft.com/en-us/aspnet/web-api/overview/mobile-clients/)
    - [Calling Web API from a Windows Phone 8 Application (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/mobile-clients/calling-web-api-from-a-windows-phone-8-application)
  + [OData](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/)
    - [OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/)
      * [Create an OData v4 Endpoint](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/create-an-odata-v4-endpoint)
      * [Create an OData v4 Client App (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/create-an-odata-v4-client-app)
      * [Entity Relations in OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/entity-relations-in-odata-v4)
      * [Actions and Functions in OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/odata-actions-and-functions)
      * [Containment in OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/odata-containment-in-web-api-22)
      * [Create a Singleton in OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/using-a-singleton-in-an-odata-endpoint-in-web-api-22)
      * [Open Types in OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/use-open-types-in-odata-v4)
      * [Complex Type Inheritance in OData v4](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v4/complex-type-inheritance-in-odata-v4)
    - [OData v3](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v3/)
      * [Creating an OData v3 Endpoint](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v3/creating-an-odata-endpoint)
      * [Supporting Entity Relations in OData v3](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v3/working-with-entity-relations)
      * [OData Actions in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v3/odata-actions)
      * [Calling an OData Service From a .NET Client (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-v3/calling-an-odata-service-from-a-net-client)
    - [Supporting OData Query Options in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/supporting-odata-query-options)
    - [Using $select, $expand, and $value in Web API 2 OData](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/using-select-expand-and-value)
    - [Routing Conventions in Web API 2 OData](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-routing-conventions)
    - [Security Guidance for Web API 2 OData](https://docs.microsoft.com/en-us/aspnet/web-api/overview/odata-support-in-aspnet-web-api/odata-security-guidance)
  + [Serialization and Model Binding](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/)
    - [Media Formatters in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/media-formatters)
    - [JSON and XML Serialization in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/json-and-xml-serialization)
    - [BSON Support in Web API 2.1](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/bson-support-in-web-api-21)
    - [Content Negotiation in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/content-negotiation)
    - [Model Validation in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/model-validation-in-aspnet-web-api)
    - [Parameter Binding in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/formats-and-model-binding/parameter-binding-in-aspnet-web-api)
  + [Error Handling](https://docs.microsoft.com/en-us/aspnet/web-api/overview/error-handling/)
    - [Exception Handling in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/error-handling/exception-handling)
    - [Global Error Handling in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/error-handling/web-api-global-error-handling)
  + [Testing and Debugging](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/)
    - [Tracing in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/tracing-in-aspnet-web-api)
    - [Unit Testing Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/unit-testing-with-aspnet-web-api)
    - [Unit Testing Controllers in ASP.NET Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/unit-testing-controllers-in-web-api)
    - [Mocking Entity Framework when Unit Testing Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/mocking-entity-framework-when-unit-testing-aspnet-web-api-2)
    - [Troubleshooting HTTP 405 Errors after Publishing Web API 2 Applications](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/troubleshooting-http-405-errors-after-publishing-web-api-applications)
  + [Security, Authentication, and Authorization](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/)
    - [Authentication and Authorization in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/authentication-and-authorization-in-aspnet-web-api)
    - [Secure a Web API with Individual Accounts in Web API 2.2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/individual-accounts-in-web-api)
    - [External Authentication Services with Web API (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/external-authentication-services)
    - [Preventing Cross-Site Request Forgery (CSRF) Attacks in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/preventing-cross-site-request-forgery-csrf-attacks)
    - [Enabling Cross-Origin Requests in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/enabling-cross-origin-requests-in-web-api)
    - [Authentication Filters in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/authentication-filters)
    - [Basic Authentication in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/basic-authentication)
    - [Forms Authentication in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/forms-authentication)
    - [Integrated Windows Authentication](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/integrated-windows-authentication)
    - [Working with SSL](https://docs.microsoft.com/en-us/aspnet/web-api/overview/security/working-with-ssl-in-web-api)
  + [Deployment and Hosting](https://docs.microsoft.com/en-us/aspnet/web-api/overview/hosting-aspnet-web-api/)
    - [Use OWIN to Self-Host Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/hosting-aspnet-web-api/use-owin-to-self-host-web-api)
    - [Host Web API 2 in an Azure Worker Role](https://docs.microsoft.com/en-us/aspnet/web-api/overview/hosting-aspnet-web-api/host-aspnet-web-api-in-an-azure-worker-role)
  + [Advanced Topics](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/)
    - [Configuring Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/configuring-aspnet-web-api)
    - [Dependency Injection in Web API 2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/dependency-injection)
    - [Calling a Web API From a .NET Client (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/calling-a-web-api-from-a-net-client)
    - [HTTP Message Handlers in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/http-message-handlers)
    - [HttpClient Message Handlers in Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/httpclient-message-handlers)
    - [Sending HTML Form Data in Web API: Form-urlencoded Data](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/sending-html-form-data-part-1)
    - [Sending HTML Form Data in Web API: File Upload and Multipart MIME](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/sending-html-form-data-part-2)
    - [HTTP Cookies in ASP.NET Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/http-cookies)
  + [Releases](https://docs.microsoft.com/en-us/aspnet/web-api/overview/releases/)
    - [What's New in Web API OData 5.3](https://docs.microsoft.com/en-us/aspnet/web-api/overview/releases/whats-new-in-aspnet-web-api-odata-53)
    - [What's New in ASP.NET Web API 2.2](https://docs.microsoft.com/en-us/aspnet/web-api/overview/releases/whats-new-in-aspnet-web-api-22)
    - [What's New in ASP.NET Web API 2.1](https://docs.microsoft.com/en-us/aspnet/web-api/overview/releases/whats-new-in-aspnet-web-api-21)
  + [Older Versions](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/)
    - [Enabling CRUD Operations in Web API 1](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/creating-a-web-api-that-supports-crud-operations)
    - [Using ASP.NET Web API 1 with Entity Framework 5](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/)
      * [Overview and Creating the Project](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-1)
      * [Creating the Domain Models](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-2)
      * [Creating an Admin Controller](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-3)
      * [Adding an Admin View](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-4)
      * [Creating a Dynamic UI with Knockout.js](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-5)
      * [Creating Product and Order Controllers](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-6)
      * [Creating the Main Page](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/using-web-api-1-with-entity-framework-5/using-web-api-with-entity-framework-part-7)
    - [Self-Host Web API 1 (C#)](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/self-host-a-web-api)
    - [Build RESTful API's with ASP.NET Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/build-restful-apis-with-aspnet-web-api)
* [Videos](https://docs.microsoft.com/en-us/aspnet/web-api/videos/)
  + [Overview and Getting Started](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/)
    - [ASP.NET Web API](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/aspnet-web-api)
    - [ASP.NET Web API, Part 1: Your First Web API](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/your-first-web-api)
    - [ASP.NET Web API, Part 2: Getting Data](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/getting-data)
    - [ASP.NET Web API, Part 3: Delete and Update](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/delete-and-update)
    - [ASP.NET Web API, Part 4: Paging and Querying](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/paging-and-querying)
    - [ASP.NET Web API, Part 5: Custom Validation](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/custom-validation)
    - [ASP.NET Web API, Part 6: Authorization](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/authorization)
* [Web API Samples List](https://docs.microsoft.com/en-us/aspnet/web-api/samples-list)

# Get Started with ASP.NET Web API 2 (C#)

* 11/28/2017
* 8 minutes to read
  + [](https://github.com/MikeWasson)
  + [](https://github.com/ericmutta)
  + [](https://github.com/nschonni)
  + [](https://github.com/wadepickett)
  + [](https://github.com/jbennink)
  + +8

by [Mike Wasson](https://github.com/MikeWasson)

[Download Completed Project](https://code.msdn.microsoft.com/Sample-code-of-Getting-c56ccb28)

In this tutorial, you will use ASP.NET Web API to create a web API that returns a list of products.

HTTP is not just for serving up web pages. HTTP is also a powerful platform for building APIs that expose services and data. HTTP is simple, flexible, and ubiquitous. Almost any platform that you can think of has an HTTP library, so HTTP services can reach a broad range of clients, including browsers, mobile devices, and traditional desktop applications.

ASP.NET Web API is a framework for building web APIs on top of the .NET Framework.

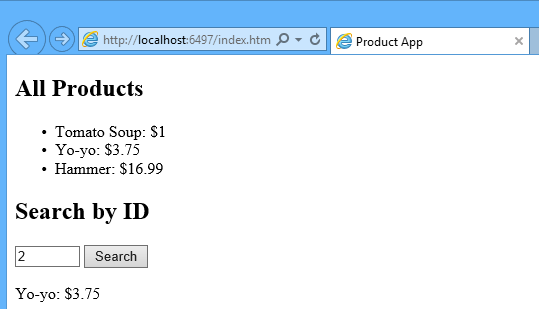
## Software versions used in the tutorial

* [Visual Studio 2017](https://visualstudio.microsoft.com/downloads/?utm_medium=microsoft&utm_source=docs.microsoft.com&utm_campaign=button+cta&utm_content=download+vs2017)
* Web API 2

See [Create a web API with ASP.NET Core and Visual Studio for Windows](https://docs.microsoft.com/en-us/aspnet/core/tutorials/first-web-api) for a newer version of this tutorial.

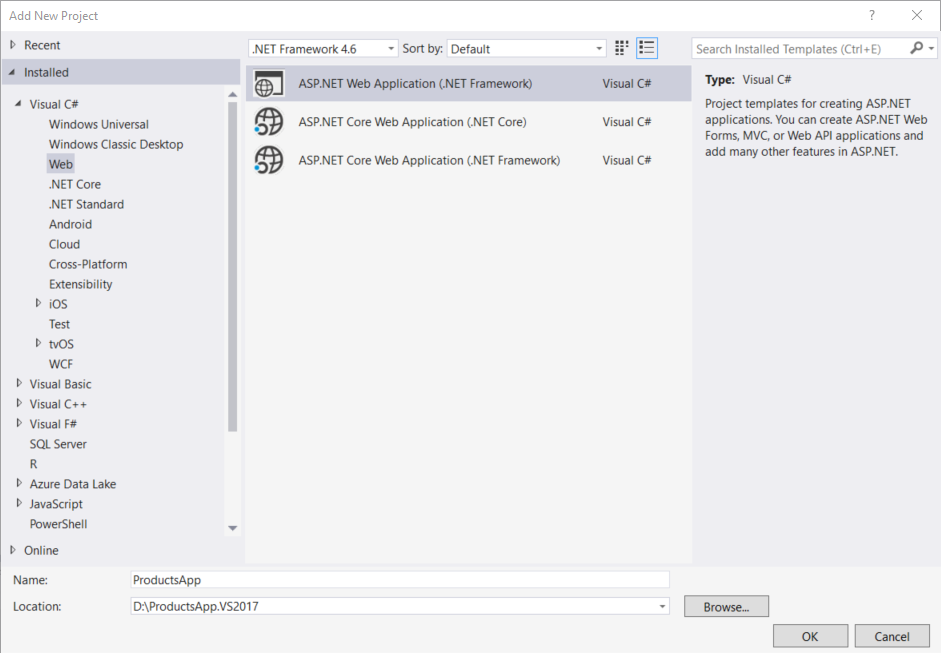
## Create a Web API Project

In this tutorial, you will use ASP.NET Web API to create a web API that returns a list of products. The front-end web page uses jQuery to display the results.

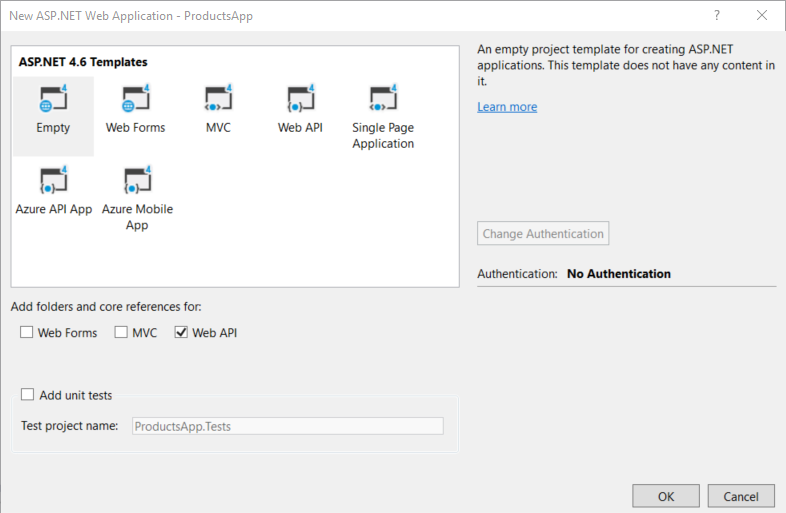


Start Visual Studio and select **New Project** from the **Start** page. Or, from the **File** menu, select **New** and then **Project**.

In the **Templates** pane, select **Installed Templates** and expand the **Visual C#** node. Under **Visual C#**, select **Web**. In the list of project templates, select **ASP.NET Web Application**. Name the project "ProductsApp" and click **OK**.



In the **New ASP.NET Project** dialog, select the **Empty** template. Under "Add folders and core references for", check **Web API**. Click **OK**.



**Note**

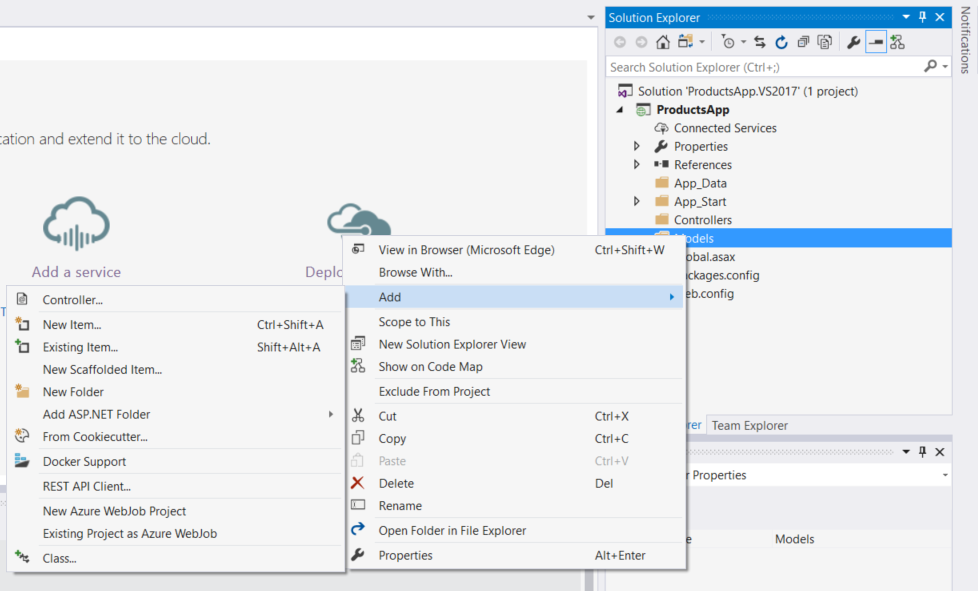
You can also create a Web API project using the "Web API" template. The Web API template uses ASP.NET MVC to provide API help pages. I'm using the Empty template for this tutorial because I want to show Web API without MVC. In general, you don't need to know ASP.NET MVC to use Web API.

## Adding a Model

A model is an object that represents the data in your application. ASP.NET Web API can automatically serialize your model to JSON, XML, or some other format, and then write the serialized data into the body of the HTTP response message. As long as a client can read the serialization format, it can deserialize the object. Most clients can parse either XML or JSON. Moreover, the client can indicate which format it wants by setting the Accept header in the HTTP request message.

Let's start by creating a simple model that represents a product.

If Solution Explorer is not already visible, click the **View** menu and select **Solution Explorer**. In Solution Explorer, right-click the Models folder. From the context menu, select **Add** then select **Class**.



Name the class "Product". Add the following properties to the Product class.

C#Copy

namespace ProductsApp.Models

{

public class Product

{

public int Id { get; set; }

public string Name { get; set; }

public string Category { get; set; }

public decimal Price { get; set; }

}

}

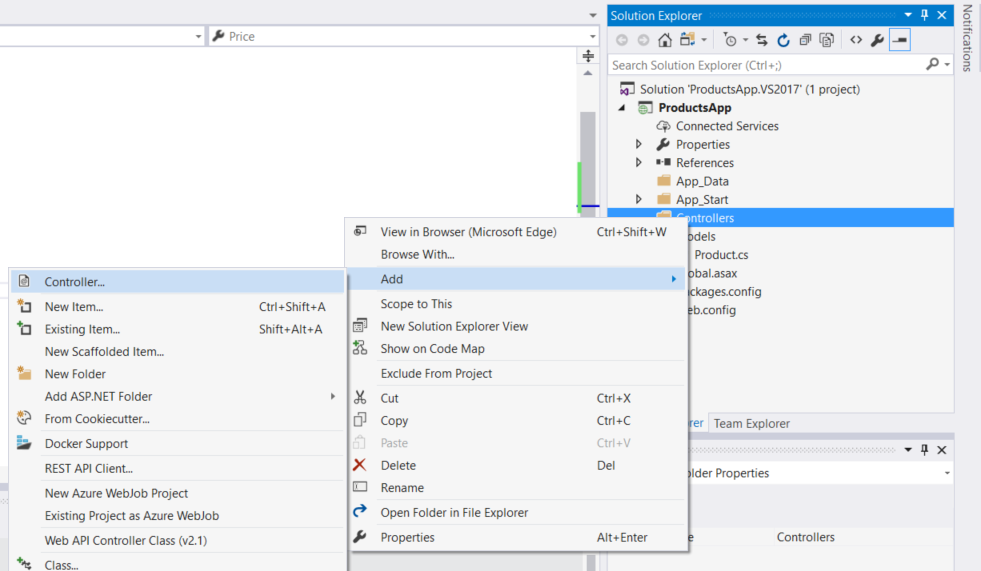
## Adding a Controller

In Web API, a controller is an object that handles HTTP requests. We'll add a controller that can return either a list of products or a single product specified by ID.

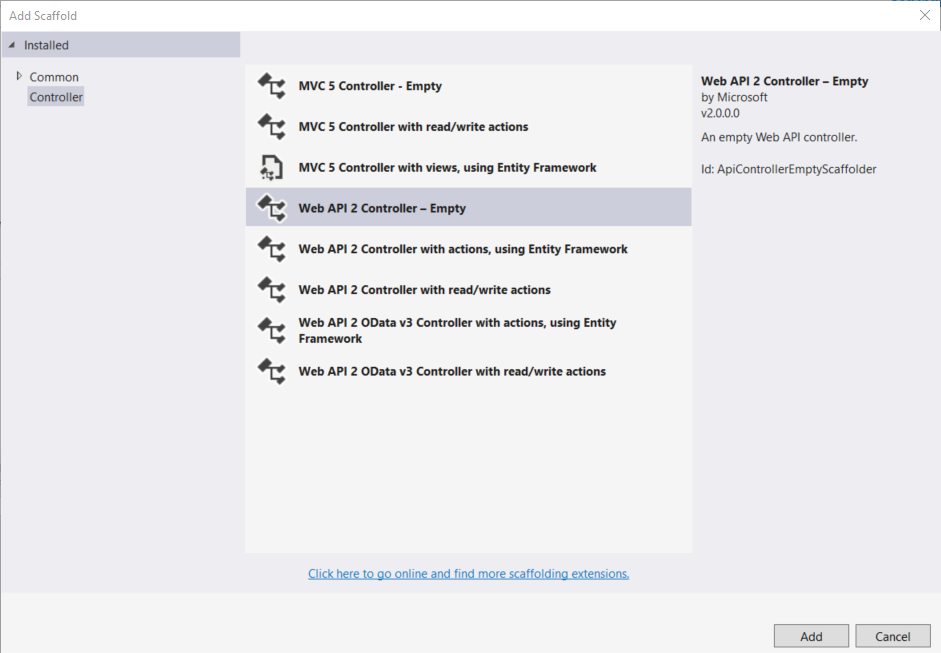
**Note**

If you have used ASP.NET MVC, you are already familiar with controllers. Web API controllers are similar to MVC controllers, but inherit the **ApiController** class instead of the **Controller** class.

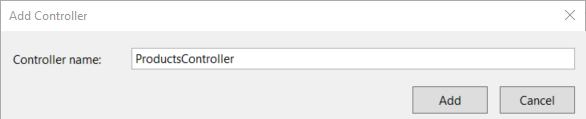
In **Solution Explorer**, right-click the Controllers folder. Select **Add** and then select **Controller**.



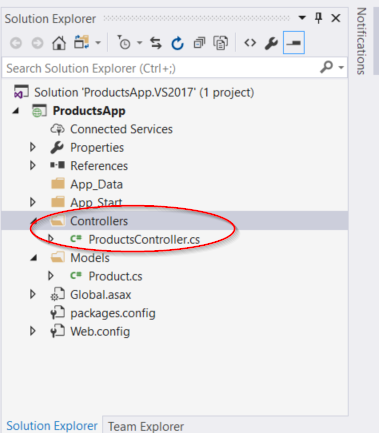
In the **Add Scaffold** dialog, select **Web API Controller - Empty**. Click **Add**.



In the **Add Controller** dialog, name the controller "ProductsController". Click **Add**.



The scaffolding creates a file named ProductsController.cs in the Controllers folder.



**Note**

You don't need to put your controllers into a folder named Controllers. The folder name is just a convenient way to organize your source files.

If this file is not open already, double-click the file to open it. Replace the code in this file with the following:

C#Copy

using ProductsApp.Models;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Net;

using System.Web.Http;

namespace ProductsApp.Controllers

{

public class ProductsController : ApiController

{

Product[] products = new Product[]

{

new Product { Id = 1, Name = "Tomato Soup", Category = "Groceries", Price = 1 },

new Product { Id = 2, Name = "Yo-yo", Category = "Toys", Price = 3.75M },

new Product { Id = 3, Name = "Hammer", Category = "Hardware", Price = 16.99M }

};

public IEnumerable<Product> GetAllProducts()

{

return products;

}

public IHttpActionResult GetProduct(int id)

{

var product = products.FirstOrDefault((p) => p.Id == id);

if (product == null)

{

return NotFound();

}

return Ok(product);

}

}

}

To keep the example simple, products are stored in a fixed array inside the controller class. Of course, in a real application, you would query a database or use some other external data source.

The controller defines two methods that return products:

* The GetAllProducts method returns the entire list of products as an **IEnumerable<Product>** type.
* The GetProduct method looks up a single product by its ID.

That's it! You have a working web API. Each method on the controller corresponds to one or more URIs:

| **TABLE 1** | |
| --- | --- |
| **Controller Method** | **URI** |
| GetAllProducts | /api/products |
| GetProduct | /api/products/id |

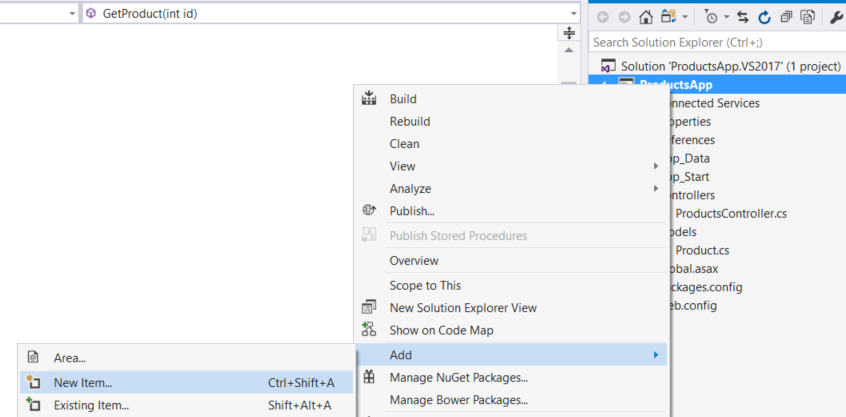
For the GetProduct method, the id in the URI is a placeholder. For example, to get the product with ID of 5, the URI is api/products/5.

For more information about how Web API routes HTTP requests to controller methods, see [Routing in ASP.NET Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/web-api-routing-and-actions/routing-in-aspnet-web-api).

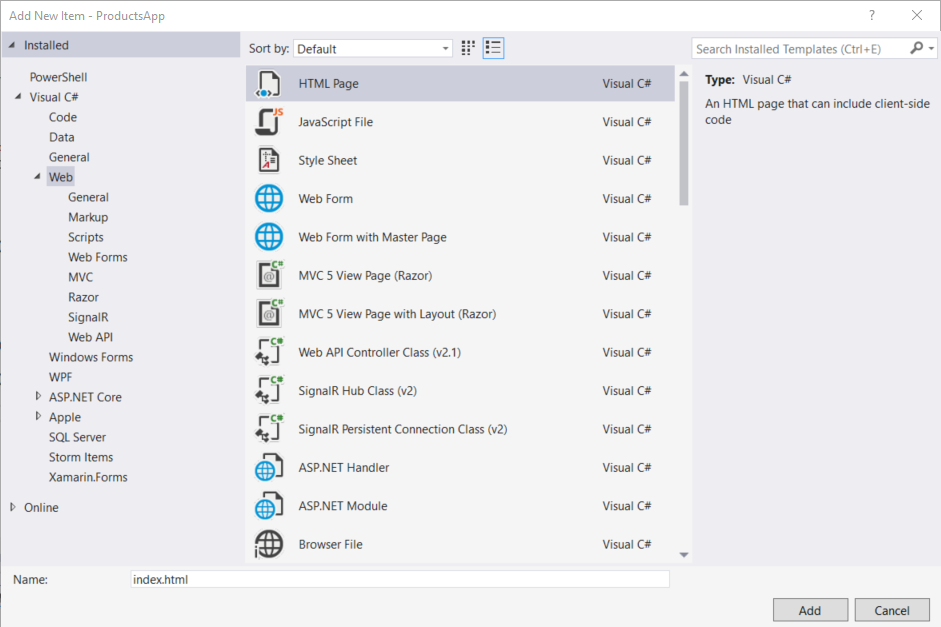
## Calling the Web API with Javascript and jQuery

In this section, we'll add an HTML page that uses AJAX to call the web API. We'll use jQuery to make the AJAX calls and also to update the page with the results.

In Solution Explorer, right-click the project and select **Add**, then select **New Item**.



In the **Add New Item** dialog, select the **Web** node under **Visual C#**, and then select the **HTML Page** item. Name the page "index.html".



Replace everything in this file with the following:

HTMLCopy

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">

<head>

<title>Product App</title>

</head>

<body>

<div>

<h2>All Products</h2>

<ul id="products" />

</div>

<div>

<h2>Search by ID</h2>

<input type="text" id="prodId" size="5" />

<input type="button" value="Search" onclick="find();" />

<p id="product" />

</div>

<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.0.3.min.js"></script>

<script>

var uri = 'api/products';

$(document).ready(function () {

// Send an AJAX request

$.getJSON(uri)

.done(function (data) {

// On success, 'data' contains a list of products.

$.each(data, function (key, item) {

// Add a list item for the product.

$('<li>', { text: formatItem(item) }).appendTo($('#products'));

});

});

});

function formatItem(item) {

return item.Name + ': $' + item.Price;

}

function find() {

var id = $('#prodId').val();

$.getJSON(uri + '/' + id)

.done(function (data) {

$('#product').text(formatItem(data));

})

.fail(function (jqXHR, textStatus, err) {

$('#product').text('Error: ' + err);

});

}

</script>

</body>

</html>

There are several ways to get jQuery. In this example, I used the [Microsoft Ajax CDN](https://docs.microsoft.com/en-us/aspnet/ajax/cdn/overview). You can also download it from <http://jquery.com/>, and the ASP.NET "Web API" project template includes jQuery as well.

### Getting a List of Products

To get a list of products, send an HTTP GET request to "/api/products".

The jQuery [getJSON](http://api.jquery.com/jQuery.getJSON/) function sends an AJAX request. The response contains array of JSON objects. The done function specifies a callback that is called if the request succeeds. In the callback, we update the DOM with the product information.

HTMLCopy

$(document).ready(function () {

// Send an AJAX request

$.getJSON(apiUrl)

.done(function (data) {

// On success, 'data' contains a list of products.

$.each(data, function (key, item) {

// Add a list item for the product.

$('<li>', { text: formatItem(item) }).appendTo($('#products'));

});

});

});

### Getting a Product By ID

To get a product by ID, send an HTTP GET request to "/api/products/id", where id is the product ID.

JavaScriptCopy

function find() {

var id = $('#prodId').val();

$.getJSON(apiUrl + '/' + id)

.done(function (data) {

$('#product').text(formatItem(data));

})

.fail(function (jqXHR, textStatus, err) {

$('#product').text('Error: ' + err);

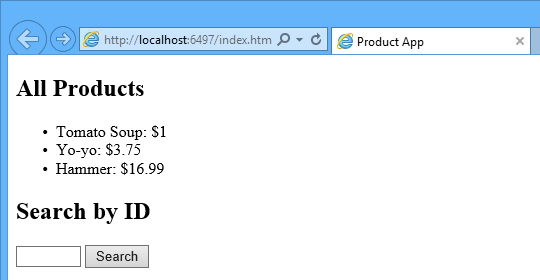
});

}

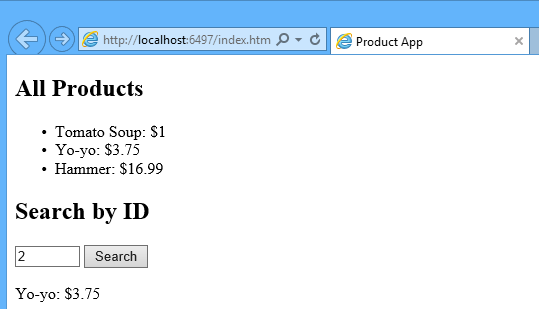
We still call getJSON to send the AJAX request, but this time we put the ID in the request URI. The response from this request is a JSON representation of a single product.

## Running the Application

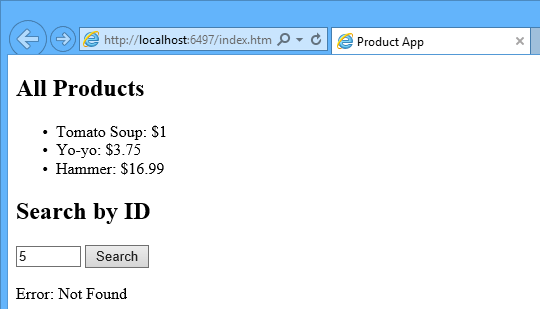
Press F5 to start debugging the application. The web page should look like the following:



To get a product by ID, enter the ID and click Search:

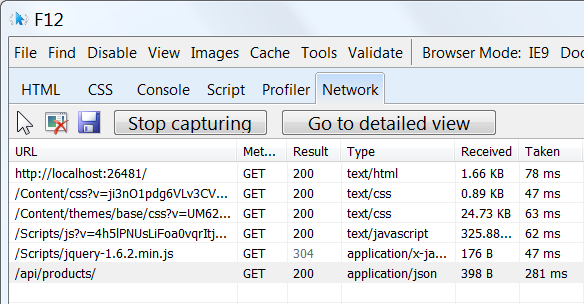


If you enter an invalid ID, the server returns an HTTP error:

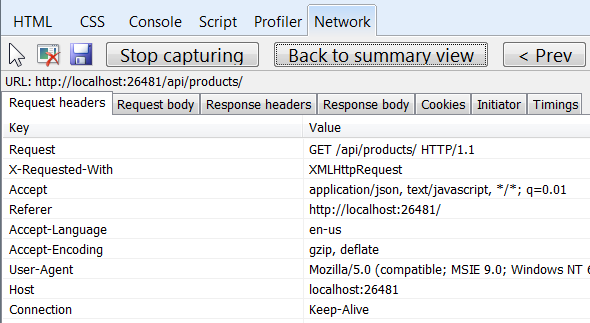


## Using F12 to View the HTTP Request and Response

When you are working with an HTTP service, it can be very useful to see the HTTP request and response messages. You can do this by using the F12 developer tools in Internet Explorer 9. From Internet Explorer 9, press **F12** to open the tools. Click the **Network** tab and press **Start Capturing**. Now go back to the web page and press **F5** to reload the web page. Internet Explorer will capture the HTTP traffic between the browser and the web server. The summary view shows all the network traffic for a page:



Locate the entry for the relative URI "api/products/". Select this entry and click **Go to detailed view**. In the detail view, there are tabs to view the request and response headers and bodies. For example, if you click the **Request headers** tab, you can see that the client requested "application/json" in the Accept header.



If you click the Response body tab, you can see how the product list was serialized to JSON. Other browsers have similar functionality. Another useful tool is [Fiddler](http://www.fiddler2.com/fiddler2/), a web debugging proxy. You can use Fiddler to view your HTTP traffic, and also to compose HTTP requests, which gives you full control over the HTTP headers in the request.

## See this App Running on Azure

Would you like to see the finished site running as a live web app? You can deploy a complete version of the app to your Azure account by simply clicking the following button.

[](https://deploy.azure.com/?WT.mc_id=deploy_azure_aspnet&repository=https://github.com/tfitzmac/WebAPI-ProductsApp#/form/setup)

You need an Azure account to deploy this solution to Azure. If you do not already have an account, you have the following options:

* [Open an Azure account for free](https://azure.microsoft.com/pricing/free-trial/?WT.mc_id=A443DD604) - You get credits you can use to try out paid Azure services, and even after they're used up you can keep the account and use free Azure services.
* [Activate MSDN subscriber benefits](https://azure.microsoft.com/pricing/member-offers/msdn-benefits-details/?WT.mc_id=A443DD604) - Your MSDN subscription gives you credits every month that you can use for paid Azure services.

## Next Steps

* For a more complete example of an HTTP service that supports POST, PUT, and DELETE actions and writes to a database, see [Using Web API 2 with Entity Framework 6](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-1).
* For more about creating fluid and responsive web applications on top of an HTTP service, see [ASP.NET Single Page Application](https://docs.microsoft.com/en-us/aspnet/single-page-application/).
* For information about how to deploy a Visual Studio web project to Azure App Service, see [Create an ASP.NET web app in Azure App Service](https://azure.microsoft.com/documentation/articles/web-sites-dotnet-get-started/).

## Recommended content

### [Creating Help Pages for ASP.NET Web API - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/creating-api-help-pages)

This tutorial with code shows how to create help pages for ASP.NET Web API in ASP.NET 4.x.

### [Using Web API with ASP.NET Web Forms - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/using-web-api-with-aspnet-web-forms)

Tutorial with code step by step to add Web API to an ASP.NET Forms application for ASP.NET 4.x

### [Testing and Debugging ASP.NET Web API](https://docs.microsoft.com/en-us/aspnet/web-api/overview/testing-and-debugging/)

Testing and debugging ASP.NET Web API

### [Enabling CRUD Operations in ASP.NET Web API 1 - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/older-versions/creating-a-web-api-that-supports-crud-operations)

Tutorial shows how to support CRUD operations in an HTTP service using ASP.NET Web API for ASP.NET 4.x.

Show more

**Recommended content**

**[Web API Guidance - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/)**

ASP.NET Web API guidance list for ASP.NET 4.x

**[Web API Samples List - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/samples-list)**

ASP.NET Web API samples list for ASP.NET 4.x

**[ASP.NET Web API Videos](https://docs.microsoft.com/en-us/aspnet/web-api/videos/)**

**[ASP.NET Web API](https://docs.microsoft.com/en-us/aspnet/web-api/videos/getting-started/aspnet-web-api)**

Debugging

**[Working with Data in ASP.NET Web API - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/data/)**

How to program CRUD (create, read, update, delete) operations in Web API for ASP.NET 4.x.

**[Advanced Topics for ASP.NET Web API - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/advanced/)**

More advanced topics for ASP.NET Web API development in ASP.NET 4.x, after you've learned the basics.

**[Get Started with ASP.NET Web API 2 (C#) - ASP.NET 4.x](https://docs.microsoft.com/en-us/aspnet/web-api/overview/getting-started-with-aspnet-web-api/tutorial-your-first-web-api)**

Tutorial with code. Use ASP.NET Web API to create a web API that returns a list of products.

Show less