

گرچه ASP.NET Web API به همراه ASP.NET MVC بسته بندی شده و استفاده می‌شود، اما اضافه کردن آن به اپلیکیشن‌های ASP.NET Web Forms کار ساده ای است. در این مقاله مراحل لازم را بررسی می‌کنیم.

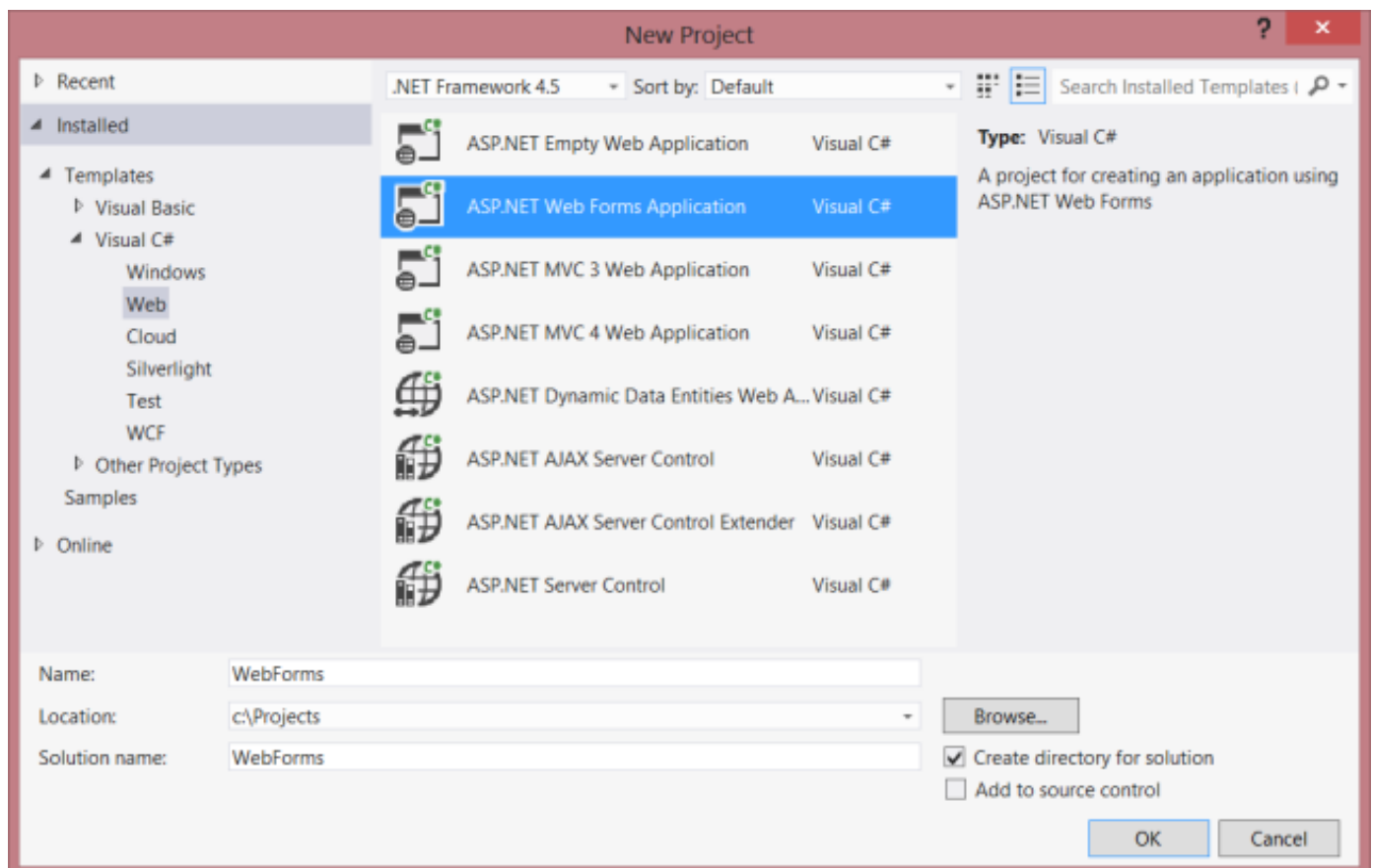
برای استفاده از Web API در یک اپلیکیشن ASP.NET Web Forms دو قدم اصلی باید برداشته شود:

اضافه کردن یک کنترلر Web API که از کلاس **ApiController** مشتق می‌شود.

اضافه کردن مسیرهای جدید به متد **Application_Start**.

یک پروژه Web Forms بسازید

ویژوال استودیو را اجرا کنید و پروژه جدیدی از نوع ASP.NET Web Forms Application ایجاد کنید.



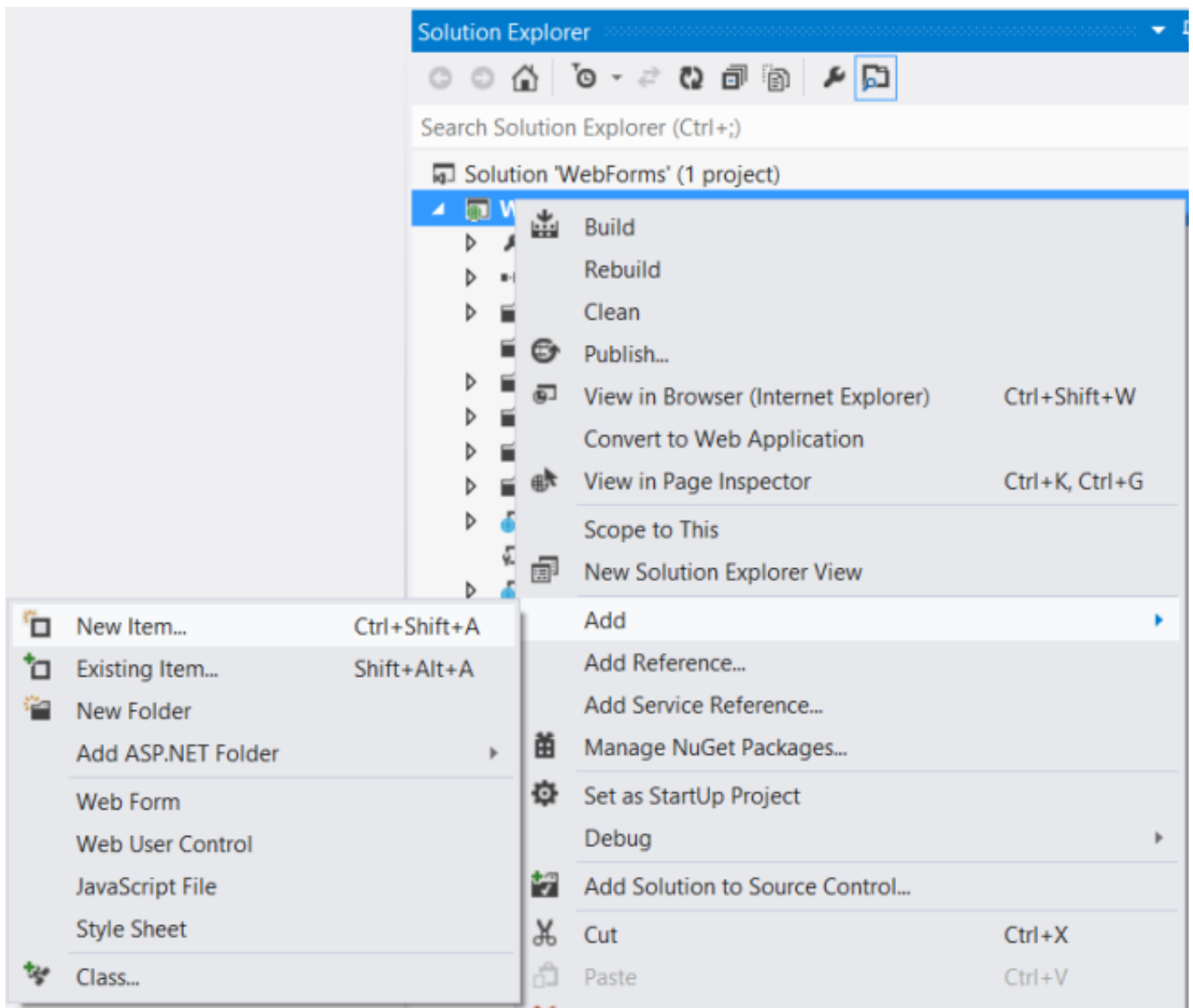
کنترلر و مدل اپلیکیشن را ایجاد کنید

کلاس جدیدی با نام Product بسازید و خواص زیر را به آن اضافه کنید.

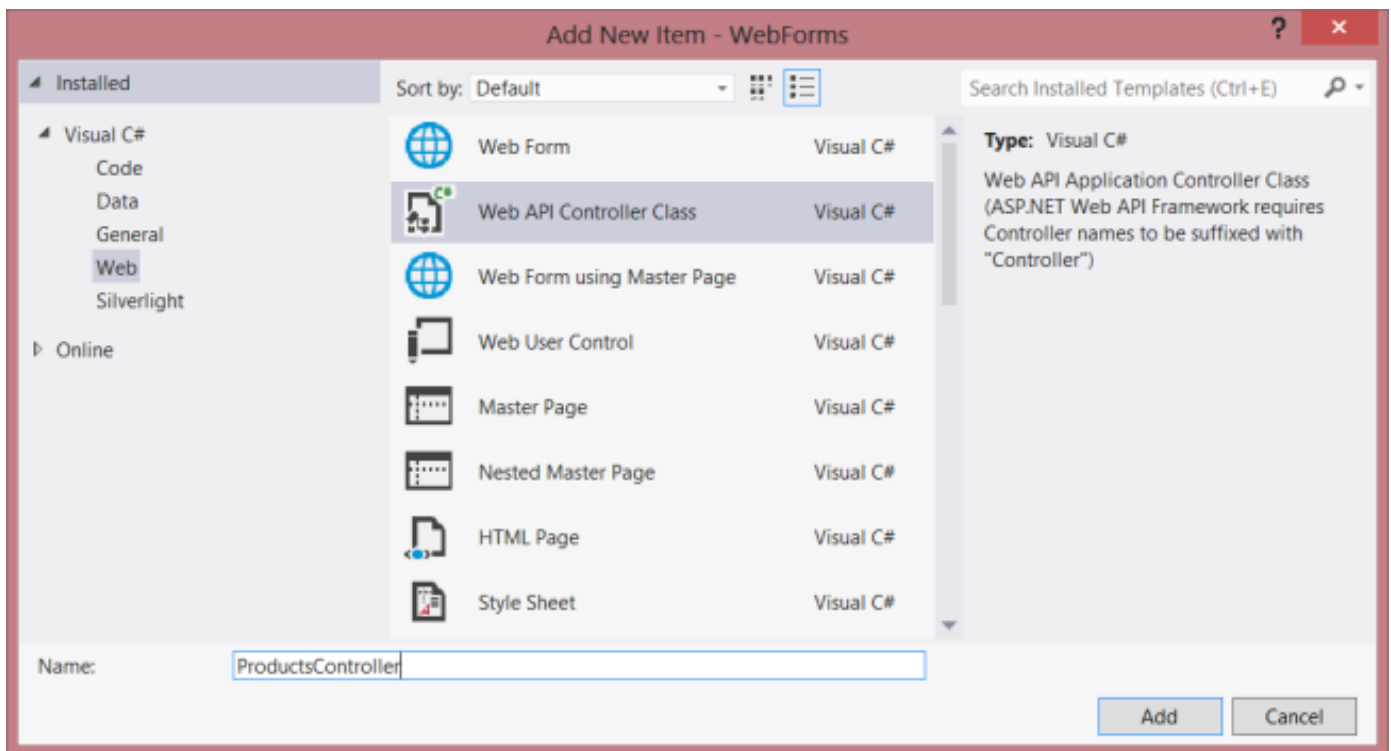
```
public class Product
{
    public int Id { get; set; }
    public string Name { get; set; }
    public decimal Price { get; set; }
    public string Category { get; set; }
}
```

}

همانطور که مشاهده می‌کنید مدل مثال جاری نمایانگر یک محصول است. حال یک کنترلر Web API به پروژه اضافه کنید. کنترلرهای Web API درخواست‌های HTTP را به اکشن متدها نگاشت می‌کنند. در پنجره Solution Explorer روی نام پروژه کلیک راست کنید و گزینه Add, New Item را انتخاب کنید.



در دیالوگ باز شده گزینه Web را از پانل سمت چپ کلیک کنید و نوع آیتم جدید را **Web API Controller Class** انتخاب نمایید. نام این کنترلر را به "ProductsController" تغییر دهید و OK کنید.



کنترلر ایجاد شده شامل یک سری متد است که بصورت خودکار برای شما اضافه شده اند، آنها را حذف کنید و کد زیر را به کنترلر خود اضافه کنید.

```
namespace WebForms
{
    using System;
    using System.Collections.Generic;
    using System.Linq;
    using System.Net;
    using System.Net.Http;
    using System.Web.Http;

    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 Product GetProductById(int id)
        {
            var product = products.FirstOrDefault((p) => p.Id == id);
            if (product == null)
            {
                throw new HttpResponseException(HttpStatusCode.NotFound);
            }
            return product;
        }

        public IEnumerable<Product> GetProductsByCategory(string category)
        {
            return products.Where(
                (p) => string.Equals(p.Category, category,
                    StringComparison.OrdinalIgnoreCase));
        }
    }
}
```

```
}
}
```

کنترلر جاری لیستی از محصولات را بصورت استاتیک در حافظه محلی نگهداری می‌کند. متدهایی هم برای دریافت لیست محصولات تعریف شده اند.

اطلاعات مسیریابی را اضافه کنید

مرحله بعدی اضافه کردن اطلاعات مسیریابی (routing) است. در مثال جاری می‌خواهیم آدرس‌هایی مانند "/api/products" به کنترلر Web API نگاشت شوند. فایل **Global.asax** را باز کنید و عبارت زیر را به بالای آن اضافه نمایید.

```
using System.Web.Http;
```

حال کد زیر را به متد Application_Start اضافه کنید.

```
RouteTable.Routes.MapHttpRoute(
    name: "DefaultApi",
    routeTemplate: "api/{controller}/{id}",
    defaults: new { id = System.Web.Http.RouteParameter.Optional }
);
```

برای اطلاعات بیشتر درباره مسیریابی در Web API به [این لینک](#) مراجعه کنید.

دریافت اطلاعات بصورت آژاکسی در کلاینت

تا اینجا شما یک API دارید که کلاینت‌ها می‌توانند به آن دسترسی داشته باشند. حال یک صفحه HTML خواهیم ساخت که با استفاده از jQuery سرویس را فراخوانی می‌کند. صفحه Default.aspx را باز کنید و کدی که بصورت خودکار در قسمت Content تولید شده است را حذف کرده و کد زیر را به این قسمت اضافه کنید:

```
<%@ Page Title="Home Page" Language="C#" MasterPageFile="~/Site.Master"
    AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="WebForms._Default" %>

<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
</asp:Content>

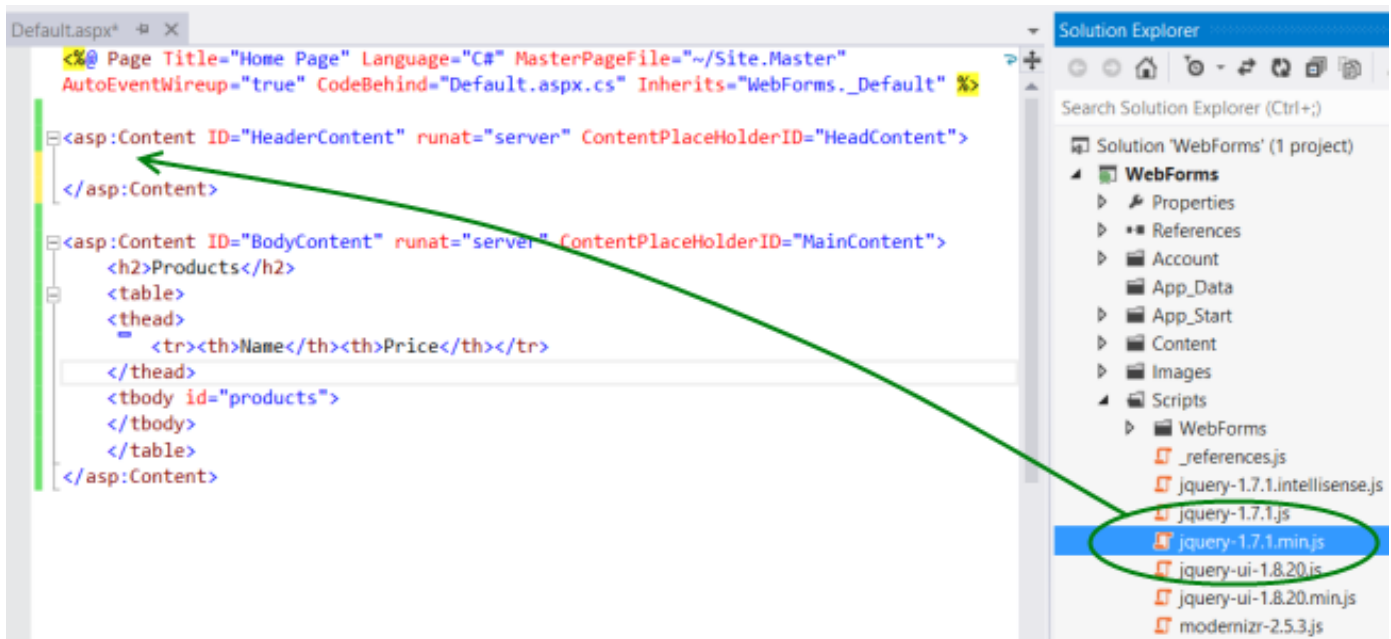
<asp:Content ID="BodyContent" runat="server" ContentPlaceHolderID="MainContent">
    <h2>Products</h2>
    <table>
    <thead>
        <tr><th>Name</th><th>Price</th></tr>
    </thead>
    <tbody id="products">
    </tbody>
    </table>
</asp:Content>
```

حال در قسمت HeaderContent کتابخانه jQuery را ارجاع دهید.

```
<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
    <script src="Scripts/jquery-1.7.1.min.js" type="text/javascript"></script>
</asp:Content>
```

همانطور که می‌بینید در مثال جاری از فایل محلی استفاده شده است اما در اپلیکیشن‌های واقعی بهتر است از CDN‌ها استفاده کنید.

نکته: برای ارجاع دادن اسکریپت‌ها می‌توانید بسادگی فایل مورد نظر را با drag & drop به کد خود اضافه کنید.



زیر تگ jQuery اسکریپت زیر را اضافه کنید.

```
<script type="text/javascript">
    function getProducts() {
        $.getJSON("api/products",
            function (data) {
                $('#products').empty(); // Clear the table body.

                // Loop through the list of products.
                $.each(data, function (key, val) {
                    // Add a table row for the product.
                    var row = '<td>' + val.Name + '</td><td>' + val.Price + '</td>';
                    $('<tr>', { text: row }) // Append the name.
                        .appendTo($('#products'));
                });
            });
    }

    $(document).ready(getProducts);
</script>
```

هنگامی که سند جاری (document) بارگذاری شد این اسکریپت یک درخواست آژاکسی به آدرس "/api/products" ارسال می‌کند. سرور ما لیستی از محصولات را با فرمت JSON بر می‌گرداند، سپس این اسکریپت لیست دریافت شده را به جدول HTML اضافه می‌کند.

اگر اپلیکیشن را اجرا کنید باید با نمایی مانند تصویر زیر مواجه شوید:



نظرات خوانندگان

نویسنده:

ارشیا

تاریخ:

۱۳:۲۷ ۱۳۹۲/۱۱/۰۵

اگر بخواهیم زمانی که برای فیلد price مقداری که وارد میکند حتما نوع عددی یا اعشاری که شما در نظر گرفتید باشد ، باید چه کدی را اضافه کنیم . تا زمانی که مقدار عددی و یا اعشاری وارد نکند اجازه اضافه کردن سطر دیگر را ندهد . امکان چنین کاری وجود دارد ؟

نویسنده:

آرمین ضیاء

تاریخ:

۱۹:۵۰ ۱۳۹۲/۱۱/۰۵

می تونید از جاوا اسکریپت و Remote Validation استفاده کنید.

نویسنده:

ارشیا

تاریخ:

۱۱:۲۲ ۱۳۹۲/۱۱/۰۶

امکانش هست لینک مثالی در این باره بفرمایید یا نمونه برنامه ای ؟

نویسنده:

محسن خان

تاریخ:

۱۲:۵ ۱۳۹۲/۱۱/۰۶

مفاهیم [اعتبارسنجی در MVC](#) با [Web Api](#) تقریبا یکی است.

نویسنده:

مهرداد

تاریخ:

۱۶:۴۵ ۱۳۹۳/۰۲/۱۵

- آیا برای عملیات CRUD می توان از آن استفاده کرد؟ اضافه ، حذف ، آپدیت؟ (مثال؟)
 - آیا استفاده از web api جهت عملیات CRUD بجای استفاده از MS AJAX بهتر است ؟
 - برای اینکه فقط یوزرهای سایت به این web api دسترسی داشته باشند ، کد خاصی باید اضافه شود ؟
 - در نهایت سوال آخر : اگر بخواهیم تمام عملیات CRUD سایت(ASP.NET Web forms) را با web api انجام دهیم کار درستی است ؟
 بسیار متشکرم

نویسنده:

وحید نصیری

تاریخ:

۱۷:۲۵ ۱۳۹۳/۰۲/۱۵

- بله. [گروه Web API](#) و EF را در سایت پیگیری کنید.
 - Web API یک بحث سمت سرور است. به آن به زبان ساده به چشم یک وب سرویس مدرن نگاه کنید. برای نمونه بجای وبمتهای استاتیک صفحات aspx یا فایل های ashx یا asmx و حتی سرویس های WCF از نوع REST و امثال آن، بهتر است از Web API استفاده کنید.
 - برای نمونه پایه مباحثی مانند Forms Authentication در اینجا هم کاربرد دارد (البته این یک نمونه است).
 - برای کار با Web API الزاما نیازی به ASP.NET ندارید (نه وب فرم ها و نه MVC)؛ به هیچکدام از نگارش های آن. سمت کاربر آن AngularJS و سمت سرور آن Web API باشد. کار می کند. (اهمیت این مساله در اینجا است که الان می شود یک فریم ورک جدید توسعه ی برنامه های وب را کاملا مستقل از وب فرم ها و MVC طراحی کرد)