بخش اول

مروری بر ASP.NET MVC

فصل اول

پیشگفتار

سال‌ها پیش، برای ساخت یک نرم‌افزار تلاش‌های بسیاری می‌شد تا نرم‌افزاری که دارای قابلیت‌های بالا باشد به همراه هزاران خط کد ایجاد گردد و به عوامل این کار برنامه‌نویس گفته می‌شد. با پیشرفت روزافزون کامپیوتر و نرم‌افزار کم‌کم محیط‌های ویژوال ایجاد گردید و برنامه‌نویسان تبدیل شدند به کدنویسان ماهر، هم اکنون می‌بینیم که دوران کدنویسی هم تمام شده و همه چیز به سمت زیرساخت[1] و ایجاد چارچوب‌های استاندارد و تبعیت از آنها در تولید بهتر نرم‌افزار در حرکت است. نرم‌افزار تقریباً در تمامی جنبه‌های زندگی ما مؤثر است و تبدیل به امری مهم در تجارت، صنعت، پزشکی، کشاورزی و تمامی فعالیت‌های روزانه ما شده است، نرم‌افزار اکنون بخشی از هر فعالیتی در سازمان‌ها است. هدف اصلی شرکت‌های بزرگ تولید نرم‌افزار (مانند مایکروسافت)، ارائهٔ سیستمی به کاربر است که علاوه بر فراهم آوردن قابلیت‌ها و کارایی موردنیاز، قابل نگهداری، قابل اطمینان، قابل پذیرش و مورد اعتماد باشد. این مبحث را در ذهن خود داشته باشید.

حال وارد دنیای وب می‌شویم:

وب چیست؟ ساختاری است برای دسترسی به سندهای پیوند شده در اینترنت. ظرف ده سال، وب از یک ابزار ارتباطی فیزیکدانان به چیزی تبدیل شده که بسیاری از مردم آن را همان اینترنت می‌دانند. وب در سال ۱۹۸۹ در مرکز اروپایی فیزیک هسته‌ای موسوم به CERN متولد شد. در این مرکز ده‌ها تیم تحقیقاتی از سراسر اروپا به کار روی فرضیه‌های فیزیک ذرات مشغول هستند. اغلب این آزمایشات چنان پیچیده است که ده‌ها دانشمند از چندین کشور مختلف سال‌ها روی آن کار می‌کنند. همین پراکندگی دانشمندان این مرکز در کشورهای مختلف و لزوم ارتباط پیوسته آن‌ها منجر به تولید وب شد. ایدهٔ سندهای پیوند شده را اولین بار یکی از فیزیکدانان CERN به نام تیم برنرزلی در مارس ۱۹۸۹ مطرح کرد و در دسامبر ۱۹۹۱ در کنفرانسی به نمایش در آمد. سپس یکی از محققان به نام مارک آندرسن مرورگر گرافیکی را در فوریه ۱۹۹۳ به بازار عرضه کرد، این مرورگر موزائیک نام داشت. بعد

1. Infrastructure

از آن شرکت‌های زیادی با ساخت مرورگر در رقابت با هم بودند، مایکروسافت هم مرورگر IE را ارائه کرد. در سال ۱۹۹۸ مایکروسافت صفحات پویای ASP را ارائه کرد، سپس مایکروسافت در جولای سال ۲۰۰۰ میلادی در کنفرانس پیاده‌کنندگان حرفه‌ای ابتکار جدید خود؛ یعنی دات نت را معرفی نمود و در کنار آن ASP.NET نسخه جدید ASP را ارائه کرد و حال بعد از چند سال تجربه در ASP.NET 1.0, 2.0, 3.0, 3.5، فریم ورک جدیدی با نام ASP.NET MVC را ارائه کرد. این فریم ورک اولین بار در دسامبر ۲۰۰۷ و با یک نسخه CTP توسط مایکروسافت معرفی شد. در مارس ۲۰۰۹ نسخه یک پایدارآن عرضه شد و یک ماه بعد؛ یعنی در آپریل ۲۰۰۹ شرکت مایکروسافت سورس کد فریم ورک ASP.NET MVC را تحت مجوز MS-PL[1] منتشر نمود و در اکتبر ۲۰۰۹ نگارش جدید MVC با عنوان ASPNET MVC 2 Preview1 منتشر شد و هم‌اکنون نسخه نهایی آن قابل دانلود است.

1. Microsoft Public License

فصل دوم

MVC چیست؟

مخفف سه کلمهٔ Model (مدل) و View (نمایشگر) و Controller (کنترلگر) است. فریم ورک ASP.NET MVC جایگزینی است بر الگوی فرم‌های وب ASP.NET که از آن برای ساختن برنامه‌های وبی با MVC استفاده می‌شود. برخی از برنامه‌نویسان، همچنان از ASP.NET که بر مبنای فرم‌های وب و Postback است، استفاده می‌کنند، برخی از ویژگی‌های MVC سود می‌برند و بعضی‌ها هم هر دو پلت فرم را ترکیب می‌کنند و این موضوع بیانگر این است که هیچکدام از پلت‌فرم‌ها ناقض یکدیگر نیستند. در واقع MVC برروی معماری‌های چند لایه‌ای جهت تفکیک بخش‌های مختلف برنامه (بخش‌های منطقی برنامه مانند داده‌ها، مجوزها، کنترل صحت داده‌ها و لایه‌های مرتبط با کاربر نهایی) قرار می‌گیرد. مفهوم تازه‌ای نیست، خیلی وقت است که در جاوا، PHP و بسیاری پلت‌فرم‌های دیگر از این الگو برای طراحی نرم‌افزار استفاده می‌شود. اما برای طراحان و توسعه‌دهندگان ASP.NET تازگی دارد.

اجزای تشکیل‌دهندهٔ MVC

۱– **Model (مدل)**: قسمتی از برنامهٔ کاربردی است که مسئول بازیابی داده از بانک اطلاعاتی، ذخیرهٔ آن، تبدیل آن به شئ یا آبجکت‌ها و پیاده‌سازی منطق برنامه برای داده‌های دامنهٔ مسئله است. در حقیقت بار اصلی معماری MVC برعهدهٔ این بخش است. مثلاً یک آبجکت Product ممکن است اطلاعات را از بانک اطلاعاتی بازیابی کرده، برروی آنها عملیاتی را انجام دهد و سرانجام نتیجه را در بانک اطلاعاتی و در جدول Products ذخیره کند.

۲– **View (نمایشگر)**: اجزایی از برنامه است که واسط کاربری برنامه (UI[1]) را می‌سازد. معمولاً این UI از داده‌های مدل ساخته می‌شود. در واقع نقطهٔ پایان برنامه کاربردی است و به کاربر نتایج عملیات و بازیابی و نمایش داده از طریق برقراری ارتباط با دو بخش دیگر؛ یعنی مدل و کنترلگر را نشان می‌دهد. برای مثال، هنگامی که کاربر در فرم ورود به سیستم رمز عبور خود را وارد می‌کند، اکثر برنامه‌نویسان

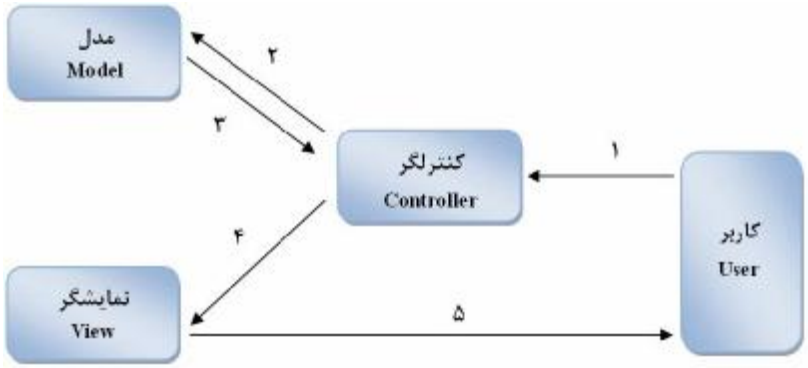
1. User Interface

در همان فرم اقدام به چک کردن رمز عبور می‌کنند که این عمل مغایر با قوانین MVC است. در MVC هنگامی که کاربر رمز عبور را وارد کرد، رمز عبور بدون هیچگونه اعمالی به بخش‌های دیگر فرستاده می‌شود و فقط یک نتیجهٔ ساده یا خبر از بخش‌های دیگر دریافت می‌کند که از طریق آن اجازهٔ ورود به برنامه داده می‌شود.

۳– Controller (کنترلگر): اجزایی از برنامه هستند که مدیریت تعامل با کاربر را بر عهده دارند. می‌توان گفت که واسط بین مدل و نمایشگر می‌باشد؛ یعنی با مدل کار می‌کند و در انتها نمایشگری را برای نشان دادن واسط کاربری انتخاب می‌کند. ورودی کاربر را مدیریت کرده و به آنها پاسخ می‌دهد و با کاربر تعامل می‌کند. برای مثال، کنترلگر عبارت‌های پرس‌وجوی بانک اطلاعاتی را مدیریت کرده و آن‌ها را به مدل ارسال می‌کند، وظیفهٔ اجرای پرس‌وجوها با مدل است.

درک الگوی MVC

برای درک بهتر MVC شکل زیر را معرفی می‌کنیم، نمودار شکل ۱ اساس الگوی MVC می‌باشد.



شکل ۱– نمودار الگوی MVC

تشریح شکل : یک درخواست در شکل بالا مسیرهای زیر را می‌پیماید:

۱. کاربر با مرورگر در ارتباط است. آدرس را در مرورگر وارد می‌کند و یا یک لینک یا کلید را در صفحهٔ وب کلیک می‌کند، این نقطهٔ شروع درخواست است (مرحلهٔ ۱ در شکل ۱).
۲. درخواست به کنترلگر فرستاده می‌شود، آن را اعتبارسنجی می‌کند و اجرای درخواست را در مدل الزامی می‌کند (مرحله ۲ در شکل ۱).
۳. مدل به بانک اطلاعاتی رجوع کرده و داده‌های بازیابی شده را به کنترگر می‌فرستد (مرحلهٔ ۳ در شکل ۱).
۴. کنترلگر داده‌ها را فرمت کرده و آنها را به نمایشگر می‌فرستد (مرحلهٔ ٤ در شکل ۱).
۵. در مرحلهٔ آخر نمایشگر داده‌های موردنیاز را دریافت کرده و جواب را به کاربر باز می‌گرداند (مرحلهٔ ٥ در شکل ۱).

چه زمانی برنامه‌های MVC را ایجاد کنیم؟

شما باید در هنگام انتخاب ASP.NET و ASP.NET MVC برای ساخت برنامه‌های وبی بسیار دقت کنید. MVC جایگزینی برای فرم‌های وب ASP.NET نیست و شما می‌توانید از هر کدام از آنها برای ساخت برنامه‌های وبی استفاده کنید. اگر شما یک برنامهٔ وبی بر مبنای فرم‌های وب دارید، می‌توانید همچنان آن را به همان شیوهٔ سابق ادامه دهید.

مزایای برنامه‌های وبی مبتنی بر MVC

- با تقسیم یک برنامه به سه قسمت مدل، نمایشگر و کنترلگر، مدیریت برنامه یا پروژه را ساده‌تر می‌کند.
- از ViewState و فرم‌های سروری استفاده نمی‌کند و از این نظر برای برنامه‌نویسانی که تسلط کامل بر رفتار برنامه را می‌خواهند عالی است.
- از الگوی کنترلگر جلو[1] استفاده می‌کند که درخواست‌های برنامه را توسط یک کنترلگر پردازش می‌کند. این مسئله باعث می‌شود تا بتوانیم برنامه‌هایی را طراحی کنیم که از زیر ساخت‌های غنی مسیریابی پشتیبانی می‌کند.
- پشتیبانی بهتری از طراحی و توسعهٔ آزمون محور[2] دارد.
- برای برنامه‌های پشتیبانی شده توسط تیم‌های بزرگ برنامه‌نویسان و طراحانی که کنترل بسیار بر رفتار برنامه را می‌خواهند، بهتر کار می‌کنند.

مزایای برنامه‌های وبی مبتنی بر فرم‌های وب

- از مدل رویداد استفاده می‌کند که وضعیت را روی HTTP حفظ می‌کند. این روش برای برنامه‌های وبی با منطق یک خطی مناسب است.
- از الگوی کنترلگر صفحه[3] استفاده می‌کند که به هر صفحه کارایی تابعی می‌دهد.
- از ViewState و فرم‌های سروری استفاده می‌کند که مدیریت اطلاعات وضعیت را ساده‌تر می‌کند.
- برای تیم‌های کوچک برنامه‌نویسی که می‌خواهند با استفاده از صدها کنترل موجود برای ساخت سریع برنامه وبی استفاده کنند، مناسب است.
- عموماً برای توسعهٔ وب ساده‌تر است، زیرا اجزای آن (کلاس Page، کنترل‌ها و...) متمرکز شده‌اند و معمولاً کد کمتری نسبت به MVC نیاز دارد.

---

1. Front Controller : http://msdn.microsoft.com/en-us/library/ms978723.aspx
2. Test Driven Development
3. Page Controller : http://msdn.microsoft.com/en-us/library/ms978764.aspx

ویژگی‌های ASP.NET MVC

- جداسازی وظایف برنامهٔ کاربردی (منطق ورودی، منطق کاری و منطق واسط کاربری) و امکان تست کردن برنامه و طراحی و توسعهٔ آزمون محور به‌صورت پیش‌فرض. تمام ارتباطات اصلی در MVCبر مبنای واسط است و می‌توان آن‌ها را با اشیای ساختگی Mock[1] تست کرد. می‌توانید کنترلگرها را بدون اینکه آنها را در ASP.NET اجرا کنید، تست کنید و این باعث افزایش سرعت و انعطاف تست برنامه می‌شود. می‌توانید از هر فریم ورک تست که با فریم ورک NET. منطبق است برای این کار استفاده کنید.
- فریم ورکی توسعه‌پذیر و قابل اتصال.[2] اجزای اصلی ASP.NET MVC به گونه‌ای طراحی شده‌اند که به راحتی جایگزین یا سفارشی شوند. می‌توانید به سادگی پیاده‌سازی‌های خودتان را جایگزین موتور نمایشگر، سیاست‌های مسیریابی آدرس، سریالی کردن پارامترهای متدهای عملیات یا سایر اجزا کنید. همچنین ASP.NET MVC از فریم ورک‌های تزریق وابستگی[3] و وارونگی کنترل پشتیبانی می‌کند. به شما امکان می‌دهد به جای این که اشیا را توسط کلاس‌ها بسازید، آنها را از جایی بیرون از کد مثل فایل‌های تنظیمات بگیرید و این مسئله تست برنامه را ساده‌تر می‌کند.
- نگاشت‌های قوی آدرس‌ها که امکان ساخت برنامه‌هایی با آدرس‌های با معنی و جستجو شدنی (قابل جستجو توسط موتورهای جستجوگر) را می‌دهد. نیازی نیست که آدرس‌ها پسوند فایل داشته باشند، آنها به گونه‌ای طراحی شده‌اند که از الگوهای نامگذاری قابل فهم برای موتورهای جستجو SEO[4] و آدرس‌های انتقال وضعیت قابل نمایش REST[5] پیروی کنند.
- پشتیبانی برای استفاده از نشانه‌گذاری[6] در صفحات ASP.NET (فایل‌های aspx.)، کنترل‌های کاربر (فایل‌های ascx.) و صفحات Master (با پسوند master.) به‌عنوان قالب‌های نمایشگر. شما می‌توانید از برخی ویژگی‌های موجود ASP.NET از قبیل صفحات master تو در تو، اسکریپت درونی (<%= %>)، کنترل‌های سرور اعلان شده، قالب‌ها، انقیاد داده‌ها[7]، محلی‌سازی و... در فریم ورک ASP.NET MVC استفاده کنید.
- پشتیبانی از ویژگی‌های موجود ASP.NET : ASP.NET MVC به ما این امکان را می‌دهد تا از ویژگی‌هایی مانند مجوز فرم[8] و تشخیص هویت[9] ویندوز، مجوز آدرس‌ها، عضویت و نقش‌ها، خروجی و کش کردن داده‌ها و Session و مدیریت وضعیت پروفایل، مانیتور کردن صحیح، سیستم پیکربندی و معماری عرضه‌کننده استفاده کنید.

---

۱. اشیایی که رفتار اشیای واقعی در برنامه را شبیه‌سازی می کنند.

2. Pluggable
3. Dependency Injection
4. Search Engine Optimization
5. Representational
6. Markup
7. DataBinding
8. Authorization
9. Authentication

فصل سوم

پیش‌نیازها و طریقهٔ دریافت و نصب ASP.NET MVC

پیش‌نیازها برای کار با ASP.NET MVC

۱- سیستم عامل موردنیاز:

Windows Server 2003; Windows Server 2008; Windows Vista; Windows XP

۲- ابزارهای موردنیاز:

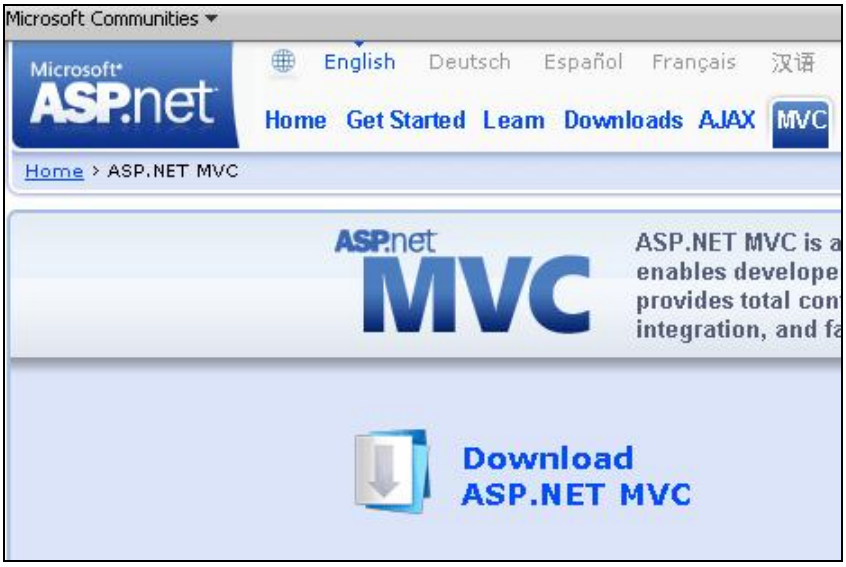
.NET 3.5 SP1; Visual Studio 2008; Visual Studio 2008 SP1 یا Visual Web Develper 2008 SP1

نکته: *نگارش MVC مورد بحث ما در این کتاب، نگارش یک است، اما ذکر این مطلب نیز ضروری است، Windows XP از MVC 2 پشتیبانی نمی‌کند (قابل توجه کاربران ویندوز XP)، اما خبر خوب برای کاربران Windows 7، خوشبختانه این سیستم عامل MVC 2 را پشتیبانی می‌کند.*

دریافت ASP.NET MVC:

۱. وارد آدرس زیر شوید: http://www.asp.net/mvc /

۲. سپس گزینه Download ASP.NET MVC را کلیک کنید (شکل ۱):



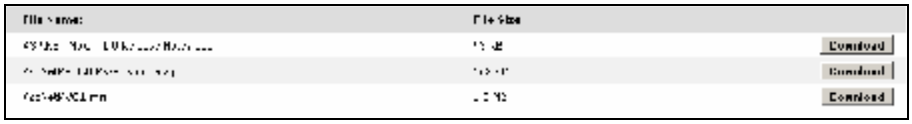
شکل ۱– صفحه معرفی MVC

۳. بعد از کلیک وارد آدرس http://www.asp.net/mvc/download/ می‌شوید.

٤. سپس صفحهٔ دانلود ASP.NET MVC (شکل ۲)

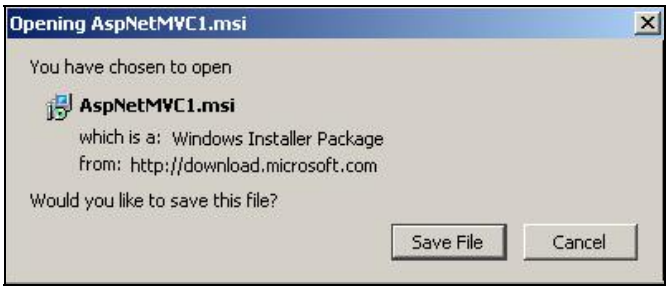
شکل ۲– صفحهٔ دانلود

۵. دانلود فایل‌های موردنظر (شکل ۳):



شکل ۳– دانلود فایل

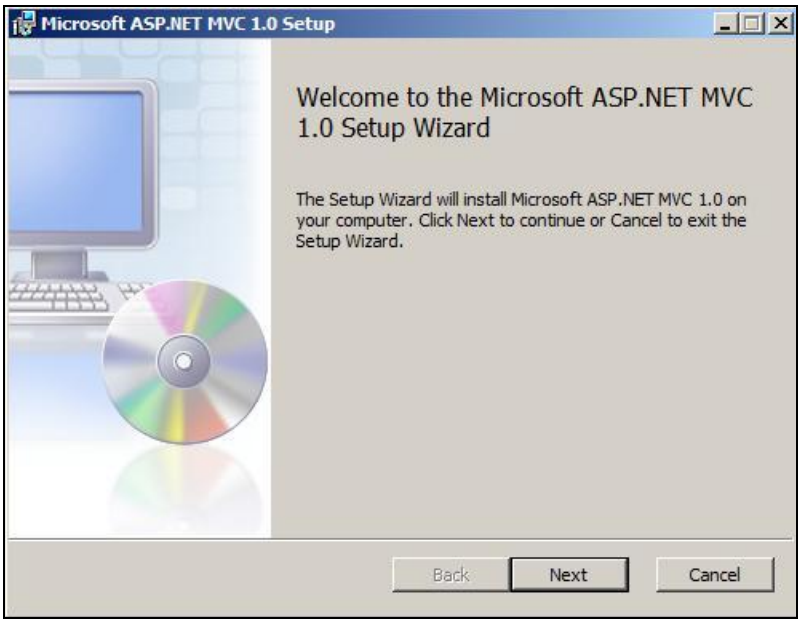
۶. ذخیره فایل در کامپیوتر (شکل ٤):



شکل ٤– ذخیره فایل در کامپیوتر

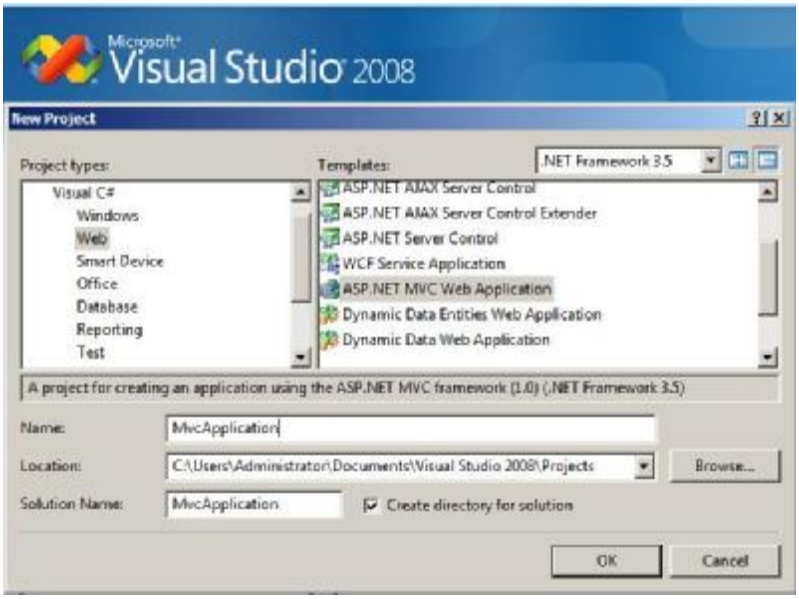
مراحل نصب ASP.NET MVC :

۱. نصب ASP.NET MVC (شکل ٥)



شکل ٥– صفحه اول نصب MVC

۲. کار با ASP.NET MVC در Visual Studio 2008 (شکل ٦)



شکل ٦– پنجره New Project در ویژوال استودیو ۲۰۰۸

فصل چهارم

درک فرآیند اجرای برنامه‌های ASP.NET MVC

درخواست‌های رسیده به برنامه وبی MVC، ابتدا به آبجکت UrlRoutingModule که یک ماژول HTTP است، منتقل می‌شود. این ماژول درخواست را تحلیل کرده و انتخاب مسیر را انجام می‌دهد. UrlRoutingModule، اولین شئ مسیر را که با درخواست رسیده منطبق باشد، انتخاب می‌کند. (یک شئ مسیر کلاسی است که RouteBase را پیاده‌سازی کرده و معمولاً شیئی از کلاس مسیر است.) اگر هیچ مسیری مطابق با درخواست پیدا نشد، UrlRoutingModule هیچ کاری انجام نمی‌دهد و اجازه می‌دهد که درخواست به‌صورت طبیعی توسط ASP.NET یا IIS پردازش شود.

UrlRoutingModule از آبجکت مسیر انتخاب شده، شئ IRoutHandler را که به مسیر تعلق دارد و در برنامه‌های MVC معمولاً MvcRouteHandler است، می‌گیرد. IRouteHandler، یک IHttpHandler می‌سازد و آن را به IHttpContext منتقل می‌کند. به‌صورت پیش‌فرض IRouteHandler در یک برنامه MVC، MvcHandler است. سپس MvcHandler کنترلگری را که باید درخواست را بررسی کند، انتخاب می‌کند.

نکته: وقتی که یک برنامه MVC در IIS 7.0 اجرا می‌شود، به پسوند فایل برای پروژه‌های MVC نیاز نیست. با این حال در IIS 6.0 نیاز است که پسوند mvc. را به ASP.NET ISAPI DLL اضافه کنیم. ماژول و اداره‌کننده‌ها اجزای ورود به فریم ورک ASP.NET MVC هستند. کار آنها:

- انتخاب کنترلگر مناسب در یک برنامه MVC
- به دست آوردن یک نمونه از کنترلگر
- فراخوانی متد Execute کنترلگر

است.

جدول زیر مراحل اجرای یک پروژه وب MVC را فهرست می کند:

مرحله	جزئیات
دریافت اولین درخواست برنامه	در فایل Global.asax اشیای Route به شئ RouteTable اضافه می‌شوند.
اجرای مسیریابی	ماژول UrlRoutingModule از اولین مسیر منطبق در RouteTable برای ساختن شئ RouteData استفاده می‌کند و بعد از آن برای ایجاد شئ RequestContext (IHttpContext) استفاده می‌کند.
ایجاد بررسی کنندهٔ درخواست MVC	شئ MvcRouteHandler یک نمونه از کلاس MvcHandler می‌سازد و نمونه RequestContext را به آن می‌فرستد.
ایجاد کنترلگر	شئ MvcHandler از RequestContext برای تعیین شئ IControllerFactory معمولاً یک نمونه از کلاس DefaultControllerFactory و برای ایجاد یک نمونه، از کنترلگر استفاده می‌کند.
اجرای کنترلگر	MvcHandler متد Execute را فراخوانی می‌کند.
فراخوانی عملیات (Action)	اغلب کنترلگرها از کلاس پایهٔ Controller مشتق شده‌اند. برای این کنترلگرها، شئ ControllerActionInvoker که متعلق به کنترلگر است تشخیص می‌دهد که کدام متد را فراخوانی کند و بعد آن متد را فراخوانی می‌کند.
اجرای نتیجه	یک متد عملیات معمولاً ورودی‌های کاربر را می‌گیرد، داده‌های پاسخ را آماده می‌کند و سپس نتیجه را با برگرداندن نوع نتیجه اجرا می‌کند. نوع نتیجه‌های پیش‌ساخته که می‌توان اجرا کرد شامل: ViewResult (که یک نمایشگر را تولید می‌کند و بیشترین کاربرد را دارد) و RedirectToRouteResult و RedirectResult و ContentResult و JsonResult و EmptyResult است.

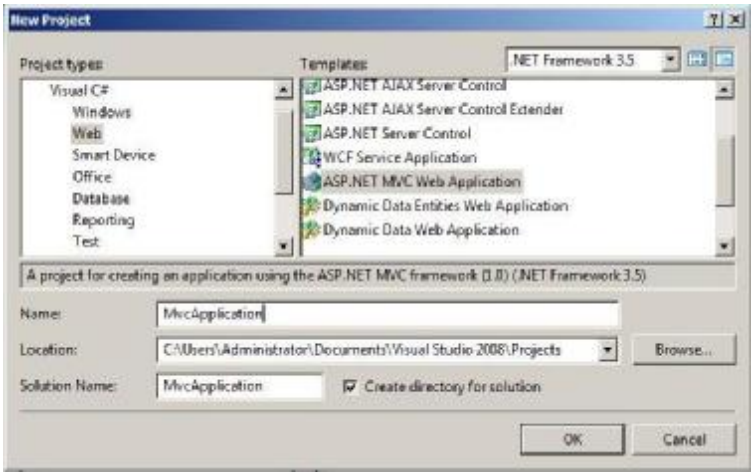
فصل پنجم

درک مدل‌ها، نمایشگرها و کنترلگرها در ASP.NET MVC

این آموزش 'C' و 'V' و 'M' را در ASP.NET MVC شرح می‌دهد. بعد از خواندن این آموزش، باید بدانید که چگونه قسمت‌های مختلف یک برنامهٔ ASP.NET MVC با هم کار می‌کنند. شما همچنین باید بدانید که یک برنامه ASP.NET MVC چه تفاوتی با یک برنامهٔ ASP.NET مبتنی بر فرم‌های وب دارد.

ساختن یک برنامه با تکنولوژی ASP.NET MVC

برای درک بهتر MVC با یک مثال ساده شروع می‌کنیم، از منو گزینهٔ File و New Project را انتخاب کنید (شکل ۱)، در پنجرهٔ New Project، زبان برنامه‌نویسی مورد علاقهٔ خود را انتخاب کنید و نام پروژه را تعیین کرده و دکمهٔ OK را کلیک کنید.



شکل ۱- ایجاد پروژهٔ جدید