# مرجع آموزشی
# ASP.NET 4.5

مهندس محمد اسماعیلی هدی

انتشارات پندار پارس

سرشناسه : اسماعیلی هدی، محمد، ۱۳۵۶ -
عنوان و نام پدیدآور : مرجع آموزشی ASP.NET 4.0/ تالیف و ترجمه محمد اسماعیلی‌هدی.
مشخصات نشر : تهران: پندار پارس، ۱۳۹۰.
مشخصات ظاهری : ۶۲۴ ص.:مصور،جدول، نمودار.
شابک : ۱۶۵۰۰۰ ریال: 978-964-2989-61-4
وضعیت فهرست نویسی : فیپا
موضوع : صفحه‌های سرور فعال
موضوع : ای‌اس.پی. (پروتکل شبکه کامپیوتری)
موضوع : وب— سایت‌ها— طراحی
رده بندی کنگره : TK۵۱۰۵/۸۸۸۵/ ص۷الف۵ ۱۳۹۰
رده بندی دیویی : ۰۰۵/۲۷۶
شماره کتابشناسی ملی : ۸۴۲۶۷۲۲

*انتشارات پندارپارس*

**دفتر فروش:** انقلاب، ابتدای کارگرجنوبی، کوی رشتچی، شماره ۱٤، واحد ۱٦ **www.pendarepars.com**

تلفن: ٦٦٥٧٢٣٣٥ - تلفکس: ٦٦٩٢٦٥٧٨همراه: ۰۹۱۲۲٤٥۲۳٤۸ **info@pendarepars.com**

**نام کتاب : مرجع آموزشی ASP.NET 4.0**

**ناشر : انتشارات پندار پارس ناشر همکار: مانلی**

**تالیف : محمد اسماعیلی هدی**

**چاپ اول : بهار ۹۰**

**شمارگان : ۱۰۰۰ نسخه**

**طرح جلد : محمد اسماعیلی هدی، زهرا مستغنی**

**لیتوگرافی : ترام‌سنج**

**چاپ، صحافی : صالحان، روشنک**

**قیمت : ۱٦۵۰۰ تومان به همراه DVD** **شابک : ٤-٦۱-۲۹۸۹-۹٦٤-۹۷۸**

*هرگونه کپی برداری، تکثیر و چاپ کاغذی یا الکترونیکی از این کتاب بدون اجازه ناشر تخلف بوده و پیگرد قانونی دارد *

# فهرست مطالب

# مقدمه

تکنولوژی ASP.NET[1]، یک تکنولوژی روز برای ساخت صفحات پویای وب[2] است که بعد از ASP، توسط شرکت مایکروسافت روانه بازار شده است. لازم است بدانید که این تکنولوژی، یک تکنولوژی کاملا جدید است و نسخه به روز شده‌ای از ASP نیست. تکنولوژی ASP.NET، جزو مجموعه Microsoft .NET محسوب می‌شود که به عنوان ابزاری قدرتمند، در دست برنامه‌نویسان وب قرار گرفته است. تکنولوژی ASP.NET می‌تواند با داشتن زیرساخت‌های مناسب، صفحات وب پویا با درجه امنیت بالاتر و استحکام بیشتر ایجاد نماید.

یکی از تفاوت‌های عمده بین تکنولوژی ASP.NET و سایر تکنولوژی‌ها، این است که در این تکنولوژی، کدها به جای تفسیر شدن، مانند سایر نرم‌افزارهای کاربردی .NET، کامپایل می‌شوند. در این حالت، فایل‌های dll. دریک فولدر موقت تشکیل شده و هنگام درخواست‌های بعدی از آنها استفاده خواهد شد.

از مزایای دیگر این زبان می‌توان به شیءگرا بودن آن اشاره کرد که اهمیت آن برای برنامه‌نویسان پوشیده نیست. پشتیبانی از گستره وسیعی از مرورگرها، راحتی انجام تنظیمات و امکان توسعه سریع سایت وب، از مزایای دیگر این زبان بسیار قدرتمند است.

در این کتاب، برنامه‌ها در ورژن Visual Studio 2010 و ASP.NET 4.0 تولید شده‌اند. بعد از مطالعه کامل این کتاب، خواهید توانست یک سایت ASP.NET با تمام امکانات را با استفاده از این نرم‌افزار تولید نمایید.

به فضل الهی، تالیف، ترجمه و جمع‌آوری مطالب این کتاب پس از ماه‌ها کار و تلاش به پایان رسید و امید است مورد توجه دانش‌پژوهان و علاقمندان به طراحی صفحات وب قرار گیرد. در تالیف این کتاب، همواره سعی بر این بوده تا مطالب به صورت ساده و شیوا ارائه گردد تا خوانندگان گرامی بتوانند بهترین استفاده را از آن بنمایند. مطالب این کتاب به صورت یک مرجع آموزشی است و دستورات آن به صورت ساده توضیح داده شده است تا همه افراد، در هر سطحی بتوانند از آن استفاده نمایند. بحث‌های جامعی که در این کتاب مطرح گردیده، باعث شده است تا برنامه‌نویسان وب نیز از آن به عنوان یک مرجع استفاده نمایند.

با تمام تلاشی که در تالیف و ترجمه کتاب انجام شده است، ممکن است معادل فارسی انتخاب شده، چندان گویای مطالب نباشد؛ بنابراین، در این کتاب زبان اصلی را نیز فراموش نکردیم. در حد امکان سعی بر آن بوده است تا

---

[1] Active Server Pages .NET

[2] استفاده از صفحات پویا، باعث ارتباط دوطرفه بین کاربران و سرور وب می‌شود.

اصطلاحات و ترجمه انگلیسی لغات به صورت پاورقی درج شود. همچنین، معنی تمام لغات تخصصی در آخر کتاب آورده شده است تا در صورت نیاز، مورد استفاده قرار گیرد.

لازم به ذکر است که این کتاب، حاصل ترجمه چند کتاب مرجع انگلیسی، به همراه تغییراتی است که مولف در آنها ایجاد کرده است. بخش‌های مختلفی از این کتاب، حاصل تجربیات نویسنده است که به صورت تالیف به کتاب اضافه شده است تا استفاده بهتری از کتاب حاصل شود.

از آنجایی که انجام تمرینات، نقش بسیار موثر و مهمی در یادگیری مطالب دارند؛ موکدا توصیه می‌شود خوانندگان گرامی، مثال‌ها و تمرینات ذکر شده در این کتاب را با دقت تمام انجام دهند.

در نهایت این که مطالب کتاب حاضر، حاصل سال‌ها تدریس و کار عملی است و هم‌اکنون پیش روی شما قرار گرفته است. امید است این کتاب مورد استفاده خوانندگان محترم و دانش‌پژوهان گرامی قرار گیرد. متخصصین محترم وب و خوانندگان گرامی می‌توانند نظرات، انتقادات، پیشنهادات و مطالب خود را از طریق انتشارات پندار پارس و یا پست الکترونیکی mohammadesmailihoda@gmail.com به اینجانب انتقال دهند. همچنین در صورت لزوم، پاسخ‌های مقتضی به آدرس الکترونیکی این افراد ارسال خواهد شد.

محمداسماعیل‌هدی –اردیبهشت‌ماه ۱۳۹۰

بخش اول

# مفاهیم اصلی ASP.NET

# فصل اول

# شروع کار با ASP.NET 4.0

از اوایل سال ۲۰۰۲ که شرکت مایکروسافت نسخه 1.0 .NET Framework را به بازار عرضه کرد، این شرکت، تلاش زیادی نمود تا ASP.NET را توسعه دهد. این تکنولوژی، به برنامه‌نویسان اجازه می‌دهد تا با جهت‌گیری جدیدی، اقدام به تولید صفحات پویای وب کنند. مایکروسافت، برای ایجاد این تکنولوژی از Java الهام گرفته و در آن، معایب و محدودیت‌های موجود در زبان‌های قبلی را حذف کرده است. قبل از این تکنولوژی، مایکروسافت ASP را برای ساخت این نوع صفحات عرضه کرده بود؛ اما با ورود ASP.NET 1.0 به بازار، برنامه‌نویسان وب به تکنولوژی جدید روی آوردند. یکی از نقاط قوت برنامه‌نویسی با ASP.NET این است که یک ابزار توسعه قدرتمند به نام Visual Studio .NET به برنامه‌نویسان امکان می‌دهد تا به صورت کاملا ویژوال، برنامه‌های‌وب با تکنولوژی طرف سروری تولید کنند.

در این فصل، به بررسی دلایل استفاده از تکنولوژی ASP.NET و مراحل پیشرفت آن، از زمان ایجاد تا نسخه ٤ آن خواهیم پرداخت.

## ۱,۱. دلایل استفاده از تکنولوژی ASP.NET

هم اکنون، تکنولوژی ASP.NET دارای علاقمندان زیادی در دنیا است که از آن برای ساخت صفحات وب پویا استفاده می‌کنند. دلیل این امر، مزایایی است که این تکنولوژی نسبت به محصولات دیگر دارد. این دلایل مهم عبارتند از:

### ۱,۱,۱. تکنولوژی بر پایه .NET Framework

قبل از هر چیز لازم است بدانید که ASP.NET بر پایه .NET Framework ساخته است و بنابراین می‌تواند از قابلیت های آن استفاده کند. در .NET Framework برای برنامه‌نویسی از ساختارهایی مانند class، interface و ... استفاده می‌شود. دسترسی کامل به .NET Framework باعث شده است تا توسعه‌دهندگان وب، از یک

روش مشترک و ساده برای کار با بانک‌های اطلاعاتی، فایل‌ها، پست الکترونیکی، ابزارهای شبکه و بسیاری دیگر از امکانات استفاده کنند.

شما می‌توانید از کتابخانه‌های عظیم موجود در .NET Framework استفاده نمایید. هزاران کلاس، در یک ساختار منطقی و به صورت سلسله مراتبی، در مفهومی به نام Namespace قرار گرفته‌اند. هر Namespace، یکسری از ویژگی‌ها و قابلیت‌ها را پشتیبانی می‌کند.

### ۱,۱,۲. خاصیت کامپایلری

کدهای نوشته شده در این تکنولوژی، برخلاف تکنولوژی‌های قبلی مانند ASP که تفسیر[1] می‌شدند، کامپایل[2] می‌شوند؛ بنابراین، نحوه برنامه‌نویسی با ASP.NET، بسیار شبیه به نوشتن برنامه‌های کاربردی[3] است. بنابراین، برنامه‌نویسان C# و Visual Basic به راحتی می‌توانند وارد صحنه تولید برنامه‌های وب شوند.

در واقع، برنامه‌های .NET باید در دو مرحله کامپایل شوند. در گام نخست، برنامه نوشته شده به زبان C# به یک کد میانی به نام MSIL[4] یا به اختصار IL ترجمه می‌شود. این همان زبانی است که .NET Framework. می‌تواند آن را درک کرده و اجرا نماید. نتیجه پردازش یک سایت ASP.NET، یک یا چند فایل با پسوند dll. است. این مرحله از کامپایل باعث می‌شود .NET. مستقل از زبان باشد. به عبارت دیگر می‌توانید به طور همزمان از زبان های مختلفی مانند C# و VB، برای تولید یک برنامه استفاده نمایید. مرحله اول کامپایل، می‌تواند در اولین درخواست صفحه توسط کاربر انجام شود یا آن که در توسط برنامه‌نویس پیش‌کامپایل[5] شود. کد IL تولید شده، یک کد اسمبلی خواهد بود.

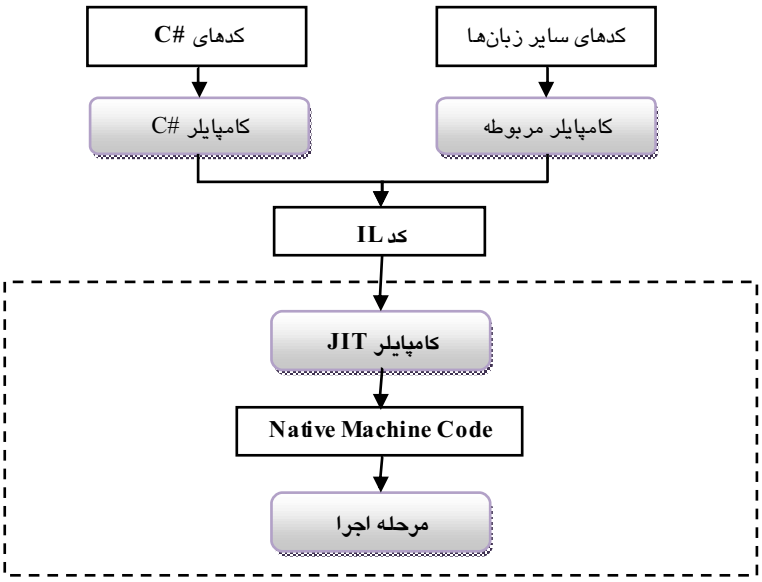
مرحله بعدی از کامپایل، درست قبل از اجرای صفحه اتفاق می‌افتد. در این مرحله، کد میانی IL به کد ماشین مورد نظر تبدیل می‌شود. این مرحله از کامپایل، *JIT*[6] نامیده می‌شود. شکل ۱-۱ نحوه کامپایل یک برنامه را نشان می‌دهد.

---

[1] Interpret
[2] Compile
[3] Application
[4] Microsoft Intermediate Language
[5] Precompiling
[6] Just-In-Time

کامپایل یک برنامه در دو مرحله، باعث راحتی در نوشتن برنامه‌ها شده و آن را قابل حمل و نقل می‌کند. کامپایلر قبل از آن که کد ماشین را تولید کند، لازم است نوع سیستم‌عامل و مشخصاتی مانند ۳۲ یا ٦٤ بیتی بودن سخت‌افزار را بداند. با توجه به دو مرحله‌ای بودن کامپایل می‌توانید کد IL برنامه خود را تولید کرده و آن را توزیع نمایید؛ سپس کاربران می‌توانند برنامه شما را دریافت کرده و مرحله دوم از کامپایل را انجام دهند.

شکل زیر نحوه کامپایل یک صفحه وب ASP.NET را نشان می‌دهد.




اگر لازم باشد تا برنامه‌ها در هر درخواست از طرف کاربران، مجددا کامپایل شوند کارایی برنامه‌ها کاهش‌خواهد یافت. خوشبختانه برنامه‌های ASP.NET نیازی به کامپایل مجدد در هنگام درخواست صفحه از طرف کاربر ندارند. کدهای IL فقط یکبار ایجاد می‌شوند؛ مگر آن که صفحه وب خود را ویرایش نمایید که در این صورت، لازم است این کدها مجددا ایجاد شوند. کدهای مربوط به یک ماشین محلی که اصطلاحا *Native Machine Code* نامیده می‌شود، در شاخه‌ای سیستمی مانند دایرکتوری c:\Windows\Microsoft.NET\Framework\v2.0.50\Temporary ASP.NET Files ذخیره[1] می‌شود.

[1] Cache

همچنین ASP.NET دارای ابزارهای پیش‌کامپایل است که به برنامه‌نویس اجازه می‌دهد تا کدهای خود را قبل از درخواست اولین کاربر، کامپایل نماید. این کار، باعث افزایش سرعت پاسخ‌دهی در اولین درخواست می‌شود و از بار سرور می‌کاهد.

### ۱,۱,۳. پشتیبانی از انواع زبان‌ها به طور همزمان

دو زبان شیءگرای Visual Basic .NET و C# که در حال حاضر محبوبیت زیادی دارند، توسط ASP.NET پشتیبانی می‌شوند. اگر چه شما احتمالا از یک زبان مانند C# برای برنامه‌نویسی استفاده خواهید کرد؛ اما می‌توانید از ترکیبی از زبان‌ها برای ساخت صفحات وب استفاده نمایید. دلیل این امر، استفاده از زبان میانی IL است که در کاپایل نهایی از آن استفاده می‌شود. برای مثال، برنامه زیر در زبان C# نوشته شده است:

```
using System;

namespace HelloWorld
{
    public class TestClass
    {
        static void Main(string[] args)
        {
            Console.WriteLine("Hello World");
        }
    }
}
```

این کد، ساده‌ترین کدی است که در زبان C# قابل نوشتن است. این کد، یک پیغام را در پنجره کنسول نشان می‌دهد. حال بیایید از یک زاویه دیگری به این کد نگاه کنیم. برنامه زیر، متد Main() را به صورت کدهای IL نشان می‌دهد:

```
.method private hidebysig static void Main(string[] args) cil managed
{
.entrypoint
// Code size 13 (0xd)
.maxstack 8
IL_0000: nop
IL_0001: ldstr "Hello World"
IL_0006: call void [mscorlib]System.Console::WriteLine(string)
IL_000b: nop
IL_000c: ret
} // end of method TestClass::Main
```

حال کد زیر را که در محیط Visual Basic نوشته شده است مشاهده نمایید:

```
Imports System

Namespace HelloWorld
    Public Class TestClass
        Shared Sub Main(args() As String)
            Console.WriteLine("Hello World")
        End Sub
    End Class
End Namespace
```

این برنامه نیز پس از کامپایل، همان کد IL را تولید خواهد کرد که برنامه نوشته شده با C# تولید می‌کند. اگر چه برخی مواقع ممکن است یک کامپایلر، روی کدهای IL عملیات بهینه‌سازی مخصوص به خود را انجام دهد؛ اما نتیجه کار بسیار شبیه به هم خواهد بود. به هر حال، برنامه IL فوق به یک کد نهایی تبدیل خواهد شد که به وسیله سیستم محلی قابل اجرا است.

### ۱,۱,۴. اجرا در محیط CLR

شاید یکی از جنبه‌های مهم ASP.NET این باشد که در داخل یک محیط Runtime به نام CLR[1] اجرا می‌شود. کل .NET Framework از جمله کلاس‌ها و فضای متغیرها، به یک کد مدیریت شده[2] اشاره دارد. برخی مزایای استفاده از CLR عبارتند از:

**مدیریت اتوماتیک حافظه**: هنگامی که یک عضو از کلاس را ایجاد می‌کنید، CLR حافظه لازم را به شما اختصاص خواهد داد. بعد از اتمام کار، لازم نیست این حافظه را به صورت دستی آزاد نمایید؛ بلکه هنگامی که از محدوده تعریف یک متغیر خارج می‌شوید، بخشی از CLR به نام Garbage Collector[3]، حافظه مربوطه را به صورت خودکار آزاد خواهد کرد. بنابراین کاربر با خیال آسوده می‌تواند به نوشتن برنامه‌های خود بپردازد.

**ایمنی**: هنگامی که یک برنامه کامپایل می‌شود، .NET به اسمبلی شما اطلاعاتی اضافه می‌کند که جزئیاتی مانند کلاس ها، اعضای آنها و نوع داده‌هایشان را مشخص می‌کند. در نتیجه، سایر برنامه‌های می‌توانند بدون نیاز به فایل‌های دیگر از آن استفاده نمایند. این لایه اضافی، همه خطاهای سطح پایین را از بین می‌برد.

---

[1] Common Language Runtime

[2] Managed Code

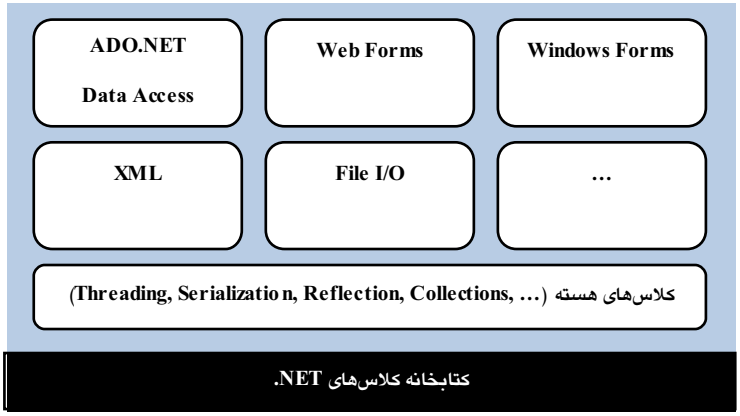
[3] این بخش از CLR به طور متناوب اجرا شده و حافظه‌های بدون استفاده را آزاد می‌کند. مدیریت این نوع از حافظه‌ها، نقش مهمی در برنامه‌های سروری دارند؛ زیرا اگر مدیریت درستی انجام نشود حافظه سرور به سرعت پر شده و قدرت سرویس دهی خود را از دست خواهد داد. در ضمن به این نکته نیز توجه کنید که ما نمی‌توانیم به راحتی، سرور را در هر زمانی خاموش کرده و یا مجددا راه‌اندازی نماییم.

**داده‌های اضافی قابل توسعه**: اطلاعاتی در مورد کلاس‌ها و اعضای آن، یکی از داده‌هایی است که .NET در فایل اسمبلی کامپایل شده قرار می‌دهد. داده‌های اضافی داخل یک فایل، برای توصیف کدها به کار می‌روند و به شما اجازه می‌دهند اطلاعات اضافی را برای زمان اجرا و سایر سرویس‌ها مهیا نمایید. برای مثال می‌توانید به دیباگ کننده[1] بگویید که چگونه کدهای شما را تعقیب نماید و یا آن که Visual Studio چگونه کنترل‌ها شخصی را در هنگام طراحی نشان دهد.

**بررسی ساخت‌یافته خطاها**: زبان‌های .NET امکانات گوناگونی برای بررسی خطاها در اختیار برنامه‌نویسان قرار می‌دهند تا بتوانید خطاها را سازمان‌دهی و مدیریت نمایید. بررسی خطا در بلوک‌های مختلف برنامه و همچنین بررسی اشکالات برنامه در لایه‌هایی با عمق مختلف از مزایای این زبان‌ها است.

اجرای همزمان[2]: این خاصیت باعث می‌شود تا بتوانید چند پروسه را به طور همزمان انجام دهید؛ بدون آن که نیاز به ایجاد Threadهای جدیدی داشته باشید. برای مثال می‌توانید چند متد را فراخوانی کرده و همزمان اقدام به خواندن از یک فایل نمایید.

شکل زیر، شمای کلی CLR و .NET Framework را نشان می‌دهد.




[1] Debugger
[2] Multithreading


### ۱,۱,۵. شیءگرایی ASP.NET

زبان سنتی ASP از شیءگرایی به طور نسبتا ضعیفی پشتیبانی می‌کرد و فقط دارای چند شیء محدود بود. در مقابل، ASP.NET یک زبان کاملا شیءگرا است. در ASP.NET نه تنها دسترسی کاملی به اشیاء موجود در .NET Framework دارید بلکه شما می‌توانید از تمام مزایای یک محیط برنامه‌نویسی شیءگرا[1] استفاده نمایید. برای مثال، شما می‌توانید کلاس‌هایی با قابلیت استفاده مجدد ایجاد کنید و یا آن که با استفاده از قوانین وراثت، آنها را گسترش دهید. همچنین می‌توانید توابع مفید خود را در یک مجموعه قابل توزیع و به صورت یک کامپوننت کامپایل شده قرار دهید.

یکی از بهترین مثال‌های تفکر شیءگرایی در ASP.NET را می‌توان در کنترل‌های سروری آن جستجو کرد. این کنترل‌ها، نشان دهنده کپسوله‌سازی در ASP.NET است. شما می‌توانید با برنامه‌نویسی، ظاهر آنها را تغییرداده و برای رویدادهایی که روی آنها اتفاق می‌افتد، واکنش‌های مناسبی تعیین نمایید. به جای مجبور کردن برنامه نویسان به نوشتن کدهای HTML، کنترل‌های سروری به طور اتوماتیک، کدهای لازم را درست قبل از ارسال صفحه به کاربر تولید خواهند کرد.

کد HTML استاندارد زیر، یک جعبه ویرایشی (Text Box) را در صفحه وب ایجاد می‌کند:

```
<input type="text" id="myText" runat="server" />
```

[1] Object-Oriented Programming

با خاصیت اضافی runat="server" این کد استاتیک HTML به یک کنترل سروری عملیاتی تبدیل می‌کند؛ بنابراین شما می‌توانید با استفاده از کدهای C# آن را کنترل نمایید. به عبارت دیگر این خاصیت باعث می‌شود بتوانید با یک منبع داده ارتباط پیدا کرده و یا برای رویدادهای این کنترل کدهای سروری بنویسید. برای مثال، با استفاده از کد C# زیر می‌توانید هنگام بارگذاری صفحه، مقدار جعبه ویرایشی را تعیین نمایید:

```
void Page_Load(object sender, EventArgs e)
{
    myText.Value = "Hello World!";
}
```

تابع فوق به طور اتوماتیک، هنگام بارگذاری صفحه اجرا شده و مقدار خاصیت Value مربوط به جعبه ویرایشی را تغییر می‌دهد. خروجی کار، یک متن است که در صفحه HTML دیده خواهد شد.

### ۱,۱,۶. پشتیبانی از تمامی مرورگرها

یکی از چالش‌های بزرگ توسعه دهندگان صفحات وب، تنوع مرورگرهایی است که هر کدام از آنها، دارای خصوصیات خاص خود هستند و از نسخه‌ها و دستورات متفاوت XHTML، CSS و JavaScript پشتیبانی می‌کنند. بنابراین، برنامه‌نویسان همواره باید در طراحی‌های خود به این موضوع توجه کنند که آیا صفحات طراحی شده آنها، در همه مرورگرها به درستی نشان داده خواهد شد یا خیر. در نهایت باید صفحه را طوری تغییر دهید که در همه مرورگرها قابل مشاهده باشد.

در تکنولوژی ASP.NET این مشکل به طور قابل ملاحظه‌ای کاهش یافته است. اگر چه در ASP.NET شما می توانید اطلاعاتی در مورد مرورگرها دریافت کرده و صفحاتی مطابق با آنها بسازید؛ اما ASP.NET کمک می‌کند تا از این ملاحظات صرف‌نظر نمایید و خود این وظیفه را به عهده می‌گیرد.

### ۱,۱,۷. راحتی توسعه و پیکربندی[1]

یکی از دردسرهای برنامه‌نویسان وب در هنگام توسعه برنامه‌ها، گسترش یک برنامه تکمیل شده به سرور است نه تنها فایل‌های متعلق به صفحات وب، فایل‌های بانک اطلاعاتی و کامپونت‌ها نیاز به انتقال دارند؛ بلکه این کامپونت ها نیاز به رجیستر شدن هم دارند و تنظیمات باید مجددا ایجاد شوند. تکنولوژی ASP.NET این پروسس را بسیار ساده کرده است.

[1] Configuration

هر نوع نصب برنامه .NET Framework شامل یکسری کلاس‌های هسته یکسان است. در نتیجه، گسترش برنامه های .NET نسبتا آسان است. شما فقط نیاز دارید تا همه فایل‌های لازم را در شاخه مجازی سرور، کپی نمایید. این کار می‌تواند از طریق یک برنامه FTP و یا حتی دستور XCOPY در خط دستور[1] انجام شود. تا وقتی که ماشین میزبان .NET Framework دارد، نیازی به انجام مراحل زمانبر رجیستر نمی‌باشد.

توزیع کامپوننت هایی که برنامه شما از آن استفاده می‌کند نیز کار ساده‌ای است. آن چیزی که شما باید انجام دهید این است که اسمبلی کامپوننت مورد نظر را به همراه فایل‌های سایت وب خود کپی نمایید؛ زیرا تمام اطلاعات کامپوننت شما مستقیما در داخل فایل‌های مربوطه قرار گرفته است و نیازی به اجرای برنامه‌های رجیستر کننده‌یا ویرایش رجیستری ویندوز نمی‌باشد. در صورتی که این کامپوننت‌ها را در محل مناسب (در شاخه Bin از سایت وب) کپی نمایید، موتور ASP.NET آن‌ها را به طور اتوماتیک شناسایی کرده و در اختیار کدهای صفحات وب قرار می‌دهد.

چالش دیگری که توسعه دهندگان نرم‌افزار با آن روبرو هستند، مساله پیکربندی است؛ به خصوص آن که شما نیاز به انتقال اطلاعات امنیتی، مانند حساب کاربران داشته باشید. تکنولوژی ASP.NET، این کار را با استفاده‌از کاهش وابستگی به برنامه IIS[2] انجام می‌دهد. به جای آن، ASP.NET تنظیمات را در یک فایل اختصاصی به نام web.config قرار داده است. این فایل در همان فولدری قرار می‌گیرد که سایت وب شما در آنجا قرار دارد. تنظیمات قرار گرفته در این فایل، به صورت سلسله مراتبی و با قالب XML می‌باشد و بنابراین، ویرایش آن می تواند در هر برنامه متنی مانند Notepad به راحتی انجام شود. نکته مهم دیگری که باید به آن توجه کنید این است که فایل web.config به وسیله سرور قفل نمی‌شود و بنابراین در هر زمانی قابل ویرایش و به روزرسانی است.

### ۱,۱,۸. تمرکز در کدنویسی

در این زبان، یک تمایز کامل بین HTML و کدهای ASP.NET وجود دارد. در زبان قدیمی ASP، کدهای طرف سروری در بین کدهای HTML نوشته می‌شدند و در بین آنها پراکنده بودند. این امر، ویرایش کدها را بسیار مشکل می‌کرد؛ اما تکنولوژی ASP.NET، به شما اجازه می‌دهد تا کدهای C# را جدای از کدهای HTML در یک فایل cs. قرار دهید. به این ترتیب، ویرایش کدها و نقل و انتقال آنها بسیار آسان‌تر شده است؛ به طوری که حتی ممکن است تصور کنید در حال نوشتن یک نرم‌افزار کاربردی با C# هستید نه طراحی یک سایت وب!

[1] Command Line
[2] Internet Information Services