Programming

مرجع کامل :

ASP.NET

مرجعی کامل برای تمامی برنامه نویسان ASP.NET

محمد بشیری

m.bashiry@gmail.com

تقدیم به:

تمام ایرانی ها

برنامه نویسان

اساتید

دانشجویان و

مشتاقان علم

توجه: مطالبی که در پیش رو دارید تماماً از سایت زیر بدون هیچگونه تغییری در محتوای اصلی آنان آورده شده است.

www.iranasp.net

گرد آورنده: محمد بشیری

تماس با من:

m.bashiry@gmail.com

M_bashiry@walla.com

Mohamad_bashiry@yahoo.com

استان همدان – شهر سامن

آدرس مقالات و کتب دیگر:

http://Bashiry.persiangig.com/Ebook

۱۳۸۴/۳/۱۷

فهرست مطالب

کلیات

ASP.NET

مقدمه ای بر Microsoft.NET[1]

معرفی فناوری .NET و بررسی قسمتهای اصلی آن

NET عضوی از بدنه .NET و متعلق به میکروسافت است. شرکت میکروسافت موفقیت خود را با سرنوشت .NETگره زده است. بنابراین شاید جالب باشد که بدانیم .NET چیست؟ بدنه .NET دارای دو قسمت اصلی است: قسمت اول یک کتابخانه عظیم از کلاس های آماده است و قسمت دوم یک محیط زمان اجرا می باشد.

کتابخانه کلاس ها در .NET

کتابخانه .NET دارای بیش از ۳۴۰۰ کلاس آماده جهت استفاده در برنامه ها است. بعضی از این کلاس ها همان کلاس های پایه مانند آرایه و رشته هستند. اما عمده این کلاس ها به پیاده سازی اعمال تخصصی مانند کار با فایل و یا تولید مستندات XML پرداخته اند.

فضانام ها در .NET

انبوه کلاس های موجود در .NET به حال خود رها نشده اند بلکه این کلاس ها در یک دسته بندی سلسله مراتبی به نام namespace یا فضانام چیده شده اند. بعنوان مثال کلیه کلاس های مربوط به کار با فایل ها در فضای نام System.IO دسته بندی شده اند. بنابراین با استفاده از نام کلاس و فضانام مربوطه هر کلاس بصورت منحصربفردی مشخص می شود. بعنوان مثال جهت دسترسی به کلاس File در فضانام System.IO می توان از عبارت System.IO.File استفاده کرد.یک دسته از فضانام ها بطور خاص برای برنامه نویسی در ASP.NET در نظر گرفته شده اند. این فضانام ها با System.Web شروع می شوند. بعنوان مثال کلیه ابزارهای HTML در فضانام

[1] مدیریت سایت iranasp.net

System.Web.UI.HTMLControlsو ابزارهای Web در فضانام System.Web.UI.WebControls قرار دارند.

مجموعه ها در.NET

کلاس های موجود در .NET. در قالب مجموعه ها (Assemblies) جاسازی شده اند. نکته مهم این است که در اینجا یک مجموعه با یک فضانام اشتباه نشود. یک مجموعه عبارت است از یک یا چند فایل که کد برنامه مربوط به کلاس ها در آن قرار دارد. برای مثال، کلیه کلاس های موجود در فضانام System.IO در یک مجموعه بنام Mscorlib.dllقرار دارد. مجموعه Mscorlib.dll یک فایل معمولی است که در دیسک سخت یک کامپیوتر حاوی .NETدر کنار سایر فایل ها قرار می گیرد. برای یک فضانام مهم نیست که کلاس های آن بروی دیسک چگونه ذخیره شده است. کلاس های یک فضانام ممکن است در قالب چند مجموعه ذخیره شده باشند و از طرف دیگر یک مجموعه ممکن است حاوی کلاس های چند فضانام باشد. یک فضانام عبارت است از دسته بندی منطقی کلاس ها در محیط برنامه نویسی در حالیکه یک مجموعه، دسته بندی فیزیکی کلاس ها بروی دیسک سخت است

زبان مشترک زمان اجرا در.NET

قسمت دوم بدنه .NET. یک محیط یا زبان برای زمان اجرا است که زبان مشترک زمان اجرا .NET. The) (Common Language Runtime - CLRنام دارد. در اولین فراخوانی یک صفحه ASP.NET آن صفحه ابتدا کامپایل شده و سپس اجرا می گردد و کد حاصله جهت مراجعات بعدی و جلوگیری از کامپایل مجدد بروی سرور نگهداری می شود. این کد کامپایل شده یک کد میانی است که زبان آن زبان جدیدی است بنام زبان میانی میکروسافت (Microsoft Intermediate Language) یا

MSILیا به اختصار IL نام دارد MSIL .مشابه کد زبان اسمبلی است. با این تفاوت که دارای ویژگیهای شئ گرا است و مستقل از پردازنده کامپیوتر طراحی شده است. صفحات ASP.NET با هر زبانی که نوشته شوند در نهایت به کد MSIL تبدیل می شوند .سپس کد MSIL در زمان اجرا توسط کامپایلری بنام JIT به کد زبان ماشین مربوطه تبدیل می شود. در واقع اینکه صفحات ASP.NET قبل از اجرا به کد میانی MSIL تبدیل می شوند دلایل مهمی دارد. دلیل اول اینکه این مساله دست برنامه نویس را در انتخاب زبان برنامه نویسی باز می گذارد بدون آنکه تفاوتی در نتیجه چه از جهت سرعت و چه از جهت کارآئی برنامه احساس شود. بعنوان مثال شما می توانید یک صفحه ASP.NETرا با زبان ویژوال بیسیک بنویسید در حالیکه سرعت اجرای آن همان سرعت برنامه ای است که با C# یا C++نوشته شده است. و این به آن دلیل است که هر زبانی که برنامه نویس انتخاب کند در نهایت برنامه تولید شده به کد MSIL تبدیل خواهد شد.

دلیل دوم اینکه چون همه زبان های NET. به یک زبان میانی تبدیل می شوند، لذا این زبان ها در کنار یکدیگر بخوبی کار می کنند. بعنوان مثال شما می توانید از کلاسی که به زبان ویژوال بیسیک نوشته شده است در زبان C# استفاده نمائید.

ASP.NET چیست؟[1]

آشنائی با ASP.NET و بررسی ویژگیهای آن در مقایسه با ASP کلاسیک

نسل بعدی Active Server Pages یا ASP است که توسط شرکت میکروسافت ارائه شده است .این محصول توسط میکروسافت بعنوان شاخص اصلی فناوری در ساخت سایتهای وب در نظر گرفته شده است. با استفاده از ASP.NETمی توان هم اینترانت کوچک یک شرکت را ساخت و هم یک سایت وب تجاری خیلی بزرگ را طراحی و پیاده سازی نمود. مهمترین نکاتی که در طراحی این محصول در نظر گرفته شده است راحتی استفاده و بالا بودن کارائی و قابلیت آن می باشد. در زیر برخی ویژگیهای ASP.NET را بررسی می کنیم.

- صفحات ASP.NET کامپایل می شوند.

هنگامی که یک صفحه ASP.NET برای اولین بارتوسط یک مراجعه کننده به سایت فراخوانی می شود، آن صفحه ابتدا کامپایل شده و بر روی سرور نگهداشته می شود و در فراخوانی های بعدی از آن استفاده می شود. این بدین معنی است که صفحات ASP.NET خیلی سریع اجرا می شوند.

- صفحات ASP.NET با ابزارهای روی سرور ساخته می شوند.

با ابزارهای موجود در ASP.NET می توان صفحات پیچیده وب را براحتی طراحی نمود. بعنوان مثال با استفاده از ابزار DataGrid می توان به آسانی داده های موجود در یک بانک اطلاعاتی را تحت وب نمایش داد.

- مجموعه ASP.NETعضوی از بدنه .NET است.

[1] مدیریت سایت iranasp.net

بدنه .NET دارای بیش از ۴۵۰۰ کلاس آماده جهت استفاده در ASP.NET است. این کلاس ها تقریبا هر نیازی را در برنامه نویسی برآورده می کنند. بعنوان مثال از این کلاس ها می توان جهت تولید تصاویر بر حسب تقاضا، به رمز درآوردن یک فایل و یا ارسال یک نامه استفاده کرد.

مقایسه ASP.NET و ASP کلاسیک

ASP.NETنسل بعدی ASP یا ASP کلاسیک است. اما این یک پیشرفت تکاملی است بطوریکه این دو فناوری تقریبا از یکدیگر متفاوتند. صفحات ASP با زبان های دستورالعمل نویسی مانند VBScript یا JScript ایجاد می شوند اما در ASP.NET ما یک فرایند کامل برنامه نویسی با زبانهای Visual Basic یا) C# سی-شارپ تلفظ شود) داریم. همچنین در ASP کلاسیک تنها پنج کلاس استاندارد (Request, Response, Application Session, Server)وجود دارد حال آنکه در ASP.NET می توان از بیش از ۴۵۰۰ کلاس استاندارد موجود در بدنه .NETبهره جست.همچنین علیرغم قدرت و امکانات زیاد و متعددASP.NET ، استفاده از آن در مقایسه با ASP کلاسیک بسیار آسانتر است. بعنوان مثال با استفاده از چند ابزار در یک صفحه ASP.NET می توان یک صفحه بسیار پیچیده HTML بدست آورد که ساخت آن در ASP کلاسیک ممکن است نیاز به چند روز کار داشته باشد.

زبانهای برنامه نویسی درASP.NET

شما در ASP.NET می توانید از هر زبان برنامه نویسی که با بدنه .NET سازگار باشد استفاده کنید. این زبانها عبارتند از Visual Basic.NET و C# و . JScript.NET این بدین معنی است که شما جهت نوشتن برنامه در ASP.NET نیاز به فراگیری زبان جدیدی ندارید و اگر یکی از زبانهای ویژوال بیسیک یا C++ یا جاوا را می دانید هم اکنون می توانید در ASP.NET برنامه بنویسید.از طرف دیگر تعدادی زبانهای دیگر توسط بعضی از شرکتهای فعال در این زمینه به مجموعه زبانهای استاندارد ASP.NET افزوده شده است. بعنوان مثال اگر مایل باشید حتی می توانید از PERL و COBOL هم در ASP.NET استفاده کنید.

ابزارهای ASP.NET

سالهاست که برنامه نویسان ویژوال بیسیک جهت ساخت فرم های خود از ابزارهای ویژوال بیسیک مانند TextBox و ListBox استفاده کرده اند. در ASP.NET هم شما می توانید از ابزارهای فراوان موجود در آن برای ساخت فرم ها و صفحات خود استفاده نمائید. در ASP.NET چهار دسته عمده از ابزارها موجود است:

- ابزارهای اصلی مانند TextBox ، RadioButton، ListBox و Button.
- ابزارهای اعتباری برای حصول اطمینان از ورود و تائید صحت اطلاعات ورودی فرم ها.
- ابزارهای داده ای برای ارتباط با بانک اطلاعاتی و دستکاری داده.
- ابزارهای پیشرفته جهت نمایش عناصر پیچیده در واسط کاربر مانند تقویم و آگهی های تبلیغاتی.

با استفاده از Visual Studio.NET شما براحتی می توانید با چیدن تصویری این ابزارها بر روی فرم مورد نظر، صفحه دلخواه خود را بسازید. در صورت تمایل حتی می توانید در یک ویرایشگر ساده متن مانند Notepad برنامه مورد نظر را نوشته و از این ابزارها استفاده کنید.

دریافت ASP.NET

جهت شروع برنامه نویسی در ASP.NET تنها کافی است که مجموعه ASP.NET را بهمراه بدنه .NET از سایت میکروسافت دریافت کنید .

دریافت .NET Framework

msdn.microsoft.com/downloads/default.asp@url=_2fcode_2fsample.asp@url=_2fmsdn-files_2f027_2f000_2f976_2fmsdncompositedoc.xml

ASP.NETبا سیستم عامل های Windows 2000(نسخه Server و Professional) و Windows XPکاملا سازگار است.

یک صفحه ASP.NET ساده[1]

اولین برنامه در ASP.NET و شرح قسمتهای آن

یک صفحه ASP.NET بطور عمده دارای دو بخش است: قسمت تعریف کد و قسمت اجرا. قسمت تعریف کد شامل تعریف کلیه موارد و زیربرنامه هایی است که در قسمت اجرای کد استفاده می شوند. قسمت اجرای کد بخشی از صفحه است که در هنگام فراخوانی صفحه اجرا می شود و حاصل آن کد HTML است که به مرورگر ارسال می گردد.

بعنوان مثال در زیر یک صفحه ASP.NET ساده آمده است که حاصل اجرای آن نمایش زمان جاری است. این صفحه به ویژوال بیسیک نوشته شده است.

```
<Script Runat="Server">
Sub Page_Load
myLabel.Text = DateTime.Now()
End Sub
</Script>
<html>
<head><title>Simple.aspx</title></head>
<body>
<asp:Label
ID="myLabel"
Runat="Server" />
</body>
</html>
```

[1] مدیریت سایت iranasp.net

قسمت تعریف کد

قسمت تعریف کد در لیست ۱ آن قسمت از صفحه است که با برچسب <Script Runat="Server"> شروع شده و با برچسب </Script> تمام می شود. در لیست بالا یک زیربرنامه بنام Page-Load تعریف شده است که در هنگام فراخوانی صفحه، بطور خودکار اجرا می شود. این زیربرنامه صفت Text مربوط به ابزار Label را با زمان و تاریخ جاری مقداردهی می کند.

قسمت اجرای کد

قسمت اجرای کد در این صفحه عبارت است از مابقی صفحه در زیر قسمت تعریف کد. شما حتما تاکنون متوجه شده اید که عمده قسمت اجرای کد می تواند همان HTML معمولی باشد. در این قسمت از ابزار Label از مجموعه ابزارهای ASP.NET برای نمایش یک متن یا برچسب بروی صفحه HTML استفاده شده است. نحوه تعریف ابزار Labelدر زیر آمده است:

```
<asp:Label ID="myLabel" Runat="Server"/>
```

هنگامی که این صفحه اجرا می شود این ابزار هم به کد متناظر HTML تبدیل می شود.

رویدادهای یک صفحه ASP.NET[1]

بررسی مجموعه رویدادهایی که هنگام فراخوانی یک صفحه ASP.NET روی می دهند مانند Init و Load...

هنگامی که یک صفحه ASP.NET فراخوانی شود مجموعه رویدادهای زیر به ترتیب رخ می دهند:

Init: اولین رویدادی که هنگام فراخوانی صفحه انجام می گردد.

Load: این رویداد قبل از همه رویدادهای مربوط به کنترل های موجود درون صفحه رخ می دهد.

PreRender :این رویداد بعد از همه رویدادهای مربوط به کنترل های موجود درون صفحه رخ می دهد.

Unload :هنگامی که صفحه از حافظه تخلیه می گردد این رویداد رخ می دهد.

Disposed :این رویداد بعد از آزاد سازی حافظه از صفحه روی می دهد.

جهت رسیدگی به این رویدادها می توان برای هر یک، روال مجزائی نوشت .بعنوان مثال روال رسیدگی کننده به رویداد Load می تواند مانند زیر باشد:

```
<Script Runat="Server">

Sub Page_Load
  ' Place any code that you want to execute here
End Sub
</Script>
```

[1] مدیریت سایت iranasp.net

توجه داشته باشید که شما با ایجاد روالی بنام Page_Load می توانید به رویداد Load مربوط به یک صفحه رسیدگی کنید. بعبارت دیگر جهت رسیدگی به رویدادهای Disposed ، Unload، PreRender، Load، Initباید از روال هایی با نامهای Page_Unload ، Page_PreRender، Page_Load، Page_Initو Page_Disposed استفاده نمود.

معمولا مفیدترین رویداد یک صفحه رویداد Load می باشد. بعبارت دیگر از این رویداد جهت مقداردهی به کنترل های موجود درون صفحه استفاده می گردد. برای مثال، مقداردهی یک برچسب یا مقداردهی یک کنترل DataGridبا داده های یک بانک اطلاعاتی در این رویداد انجام می شود.

مهمترین تفاوت میان رویدادهای Load و PreRender این است که رویداد Load قبل از همه رویدادهای مربوط به کنترل های درون صفحه انجام می شود. جهت روشن شدن مطلب به مثال زیر توجه فرمائید. این صفحه دارای سه رویداد Button_Click ، Page_Loadو Page_PreRender می باشد. اگر بر روی دکمه موجود کلیک شود ابتدا روالPage_Load ، بعد روال Button_Click و در نهایت روال Page_PreRender اجرا می گردد.

```
<Script Runat="Server">

Sub Page_Load
  Response.Write( "<li> Page_Load" )
End Sub

Sub Button_Click( s As Object, e As EventArgs )
  Response.Write( "<li> Button_Click" )
End Sub
```

```
Sub Page_PreRender
  Response.Write( "<li> Page_PreRender" )
End Sub

</Script>

<html>
<head><title>EventOrder.aspx</title></head>
<body>

<form runat="Server">

<asp:Button
  Text="Click Here!"
  OnClick="Button_Click"
  Runat="Server" />

</form>

</body>
</html>
```

نشان دادن قابلیتهای مرورگر در ASP.NET[1]

این مقاله نحوه نشان دادن قابلیتهای مرورگر با استفاده از ASP.NET را نشان می دهد .

لینک دریافت کد : www.iranasp.net/download/shahoo01.zip

اگرچه در حال حاضر جنگ مرورگرها تقریبا تمام شده است اما این موضوع دلیلی بر شناخته نشدن قابلیتهای مرورگرها نیست. در اینجا توانایی ASP.NET در نشان دادن قابلیتهای مرورگرها بحث شده است. بعنوان نمونه، مثال ۱ نوع مرورگر را به ما نشان می دهد.

```
<html><body>
You are using <% =Request.Browser.Type %>
</body></html>
```

برای نمونه اگر شما از IE 5 استفاده می کنید نتیجه خروجی چنین باید باشد :

You are using IE5

در مثال ۱ Request.Browser.Type یک رشته را که همان نام ونسخه‌ی مرورگر است را بر می گرداند. اما این موضوع چگونه صورت میگیرد ؟

شئ HTTPBrowserCapabilities

در حقیقت خاصیت Browser در شئ Request کلاسی از HTTPBrowserCapabilities است که در فضانام System.Web قراردارد .وقتی که این کلاس روی یک صفحه ASP.NET نمونه سازی می شود خواص صفحه سرویس گیرنده ای را نشان می دهد که از آن برای اجرا شدن کد استفاده شده است. شئ Request در

[1] شاهو طوفانی

برگیرنده این خاصیت مرورگر است که این کلاس را میتوان معادل کلاس MSWC. BrowserCapabilities در ASPکلاسیک در نظر گرفت.

در لیست زیر اکثر خاصیتهای شی HTTPBrowserCapabilities تشریح شده است:

ActiveXControls:نشان می دهد که مرورگر اکتیویکس را ساپورت می کند یا نه.

AOL:چک می کند که مرورگر از نوع AOL است یا نه.

Cookies:نشان می دهد که مرورگر کوکی ها را ساپورت می کند یا نه باید توجه داشت که این خاصیت وضعیت فعال بودن یا غیر فعال بودن کوکی ها را نشان نمی دهد.

Crawler:نشان میدهد که مرورگر سرویس گیرنده از موتورهای جستجو تاثیر می پذیرد یا نه.

Browser:نوع مرورگر را نشان می دهد.

Frames:نشان می دهد که مرورگر از قابلیت Frame برخورداراست یا نه.

MajorVersion:نسخه اصلی مرورگر را نشان می دهد بعنوان مثال در IE5 عدد ۵ نشانگر نسخه اصلی است.

MinorVersion:نسخه جزیی (کوچکتر) مرورگر را نشان می دهد بعنوان مثال در IE5.1 عدد ۱. نشانگر نسخه جزیی است.

Type:نوع ونسخه مرورگر را بصورت یک رشته بازمیگرداند..

VBScript:نشان می دهد که مرورگر VBScript را ساپورت می کند یا نه.

Version:نسخه اصلی و جزیی مرورگر را بعنوان یک رشته برمی گرداند.

در زیر نمونه کامل یک مثال آورده شده است.

```
<%@ page language="VB" %>
<%@ Import Namespace="System.Web" %>
<html>
<body>
<head><title>HTTPBrowserCapabilities Demo</title></head>

<%
Dim browserObj As HTTPBrowserCapabilities
browserObj = Request.Browser
%>
<font face="verdana, arial" size=2>

<p>Your browser supports ActiveX controls: <%=browserObj.ActiveXControls %>
</p>
<p>Your browser type: <%=browserObj.Type %> </p>
<p>Your browser version: <%=browserObj.Version%> </p>

... Add any other property that you would like to display

</font>
</body>
</html>
```

چرا به دات نت احتیاج داریم؟[1]

آشنائی با دات نت و بررسی مشکلاتی که حل آنها سبب تولد دات نت گردید .

به طور معمول نسل های جدید زبان های برنامه نویسی به این دلیل متولد می شوند که زبان های قدیمی تر دارای امکانات محدود بودند و یا قدرت استفاده از تکنولوژی های فعلی را به صورت مطلوب و ساده ندارند.

مهمترین نیازی که به عنوان آخرین تکنولوژی وجود دارد، برنامه نویسی در محیط اینترنت است. اینترنت در مدت تقریبا ۸ سال جای خود را به عنوان یکی از مهمترین وسایل ارتباطی برای کارهای روزمره و تجارت باز کرده است. سیستم های برنامه نویسی قدیمی تر امکان برنامه نویسی برای اینترنت را فراهم کرده بودند اما هر کدام دارای اشکالات بزرگی هستند، برای مثال تکنولوژی COM اولین بار در ویندوز به کار گرفته شد. در سال ۱۹۷۰ نیز سیستم هایی برای Unix نوشته شده بودند، جاوا نیز در اصل برای ابزارهای الکترونیکی بود و نه برای اینترنت.

سپس برای اولین بار یک سیستم جامع برای برنامه نویسی تحت اینترنت ایجاد شد. این سیستم .NET- از مراحل سطح پایین که به زبان ماشین می باشد تا بالاترین سطح که برنامه نویسی ویژوال آن می باشد برای استفاده در اینترنت طراحی شده است. البته. NETفقط برای اینترنت نیست و با استفاده از آن می توان برنامه های کامل تحت Clientنیز ایجاد کرد، اما بزرگترین مزیت آن دربرابر سیستم های دیگر امکانات اینترنت آن است.

برای اینکه مزایای استفاده از. NETرا بهتر متوجه بشویم بهتر است در ابتدا معایب سیستم های پیشین را ذکر کنیم. شرکت مایکروسافت تا قبل از سال ۱۹۹۵ به برنامه نویسی در محیط های Client و Server می پرداخت، اما از آن سال به بعد توجه بیشتری به مساله برنامه نویسی در اینترنت کرد. مایکروسافت COM و COM+ را ایجاد کرد و آنها را در ویژوال استودیوی ۶ به کار گرفت. در سال ۱۹۹۹ حدود ۵۰ در صد از بزرگترین سایتهای تجارت الکترونیکی

[1] حامد بنایی

از محصولات مایکروسافت استفاده می کردند. اما هنوز هم مشکلات بزرگی در سیستم های مایکروسافت وجود داشت که یکی از آنها دشواری نوشتن برنامه در اینترنت با محصولات مایکروسافت بود. شرکت مایکروسافت برای راحتی کار برنامه نویس ها ASP یا Active Server Page را ایجاد کرد. با اینکه این یک قدم بزرگ بود و کارها را بسیار ساده کرد ولی هنوز از برنامه نویسی شی گرا پشتیبانی نمی کرد. همچنین در ویژوال استودیوی ۶ قسمتی برای Internet Applicationایجاد شده بود و در آنها امکان ساختن Web Class وجود داشت ولی هیچ وقت به عنوان یک ابزار کار آمد برای برنامه نویسی وب درنظر گرفته نشد.

مدل برنامه نویسیDNA

مایکروسافت یک مدل برنامه نویسی به نام Distributed interNet Application دارد که بر پایه برنامه نویسی n-tier و COM بنا نهاده شده است. مدل DNA از سه بخش اساسی تشکیل شده است.

بخش اول به نام Presentation tire معروف است. در این بخش رابط تصویری کاربر وجود دارد و خود نیز به دو نوع Internet Browser و Win 32 GUI تقسیم می شود که هر کدام مشکلات خاص خود را دارند. در مدلی که از Win32 GUI یا همان نرم افزارهای معمولی استفاده می شود دو مشکل بزرگ وجود دارد ؛ دشواری بروز رسانی نرم افزار و دیگری DLL Hell که در ادامه توضیح داده خواهد شد. در نوع دوم مشکلاتی از قبیل نبود امکانات برنامه نویسی کافی در محیط مرورگر، نبود رابط قوی با کاربر، نبودن مرورگر های یکسان و... وجود دارد. همچنین همیشه یک اتصال به اینترنت یا اینترانت لازم است .در این نوع از برنامه نویسی می توان از Java Applet ها یا ActiveX استفاده کرد ولی مرورگر باید امکان استفاده از آن را داشته باشد، مخصوصاً هنگام استفاده از ActiveX که باید فقط از IE استفاده کرد.

بخش دوم که Middle tier نام دارد، مکانی است که اطلاعات و قوانین تجاری در آن وجود دارد. منظور از قوانین ، متد ها و اجزائی هستند که اعمال کاربران را کنترل می کنند. مهمترین و آسان ترین زبان برای نوشتن این

اجزا از DNA ویژوال بیسیک است. برنامه نویسی که بخواهد در این رده برنامه بنویسد باید آشنایی کاملی با COM و پروتکل های رایج داشته، همچنین باید مهارت کافی در استفاده از ADO و ADSI داشته باشد. مشخص است که یک اشتباه در این لایه باعث بروز خطا و نقص در کل سیستم می شود.

بخش سوم یا Data tier مکانی است که اطلاعات سازمان در آن ذخیره می شود. معمولاً در این قسمت از بانکهای پیشرفته رابطه ای مانند SQL Server و Oracle استفاده می کنند.

محدودیت های COM

همانطور که دیدید مهمترین قسمت در DNA همان COM است که در جای جای آن استفاده می شود. در اینجا برخی معایب COM ذکر می شود: (ر ابتدای متن ذکر شد که برای درک نیاز به .NETباید ابتدا معایب سیستم های قدیمی را بشناسیم)

DLL Hell: گرچه کوچکترین تغییری در یک COM ایجاد شود، دیگر برنامه هایی که از ورژن قبلی استفاده می کردند قادر به فعال ساختن نسخه جدید نیستند. هنگامی که در ویندوز، یک COM نصب شود برایش در رجیستری یک GUID ثبت می شود که اطلاعات آن COM را در خود ذخیره می کند. اگر یک برنامه از نسخه اول یک COM استفاده کند و بعد از مدتی شما تغییراتی در نسخه اول بدهید و بخواهید آن را دوباره در سیستم نصب کنید ویندوز به شما پیغام خطا می دهد چون ورژن آن تکراری است، اگر هم آن را به ورژن دوم ارتقا دهید نرم افزار قبلی هنوز به دنبال نسخه اول می گردد. این امر باعث می شود که شما مجبور شوید یکبار دیگر کل برنامه را کامپایل کرده و در کامپیوترتان نصب کنید.

کمبود در وراثت :در نسخه های COM که در حال حاضرهستند چیزی به نام وراثتی که در C++ وجود دارد نمی باشد، بلکه وراثت تنها در واسط یک COM می باشد، استفاده از آن هم چندان کمکی به برنامه نویسی نمی کند.

برخی محدودیت های برنامه نویسی اینترنتی در مدلDNA

۱ -وجود دو محیط برنامه نویسی برای اینترنت وClient

نقصان در نوشتن برنامه هایی با رابط گرافیکی خوب که در اینترنت کار می کردند کاملاً مشهود است، نمونه بارز آن اختلاف در برنامه نویسی در ویژوال بیسیک و ASP است .ویژوال بیسیک با رابط گرافیکی کاملاً سطح بالا و ASP تقریباً رابط گرافیکی ندارد .همین امر باعث می شد که یک برنامه نویس مجبور باشد طیف وسیعی از تکنیک ها و زبان ها را فرابگیرد تا بتواند برنامه ساده ای در اینترنت بنویسد.

۲ -نبودن حالت های ذخیره اطلاعات رابط گرافیکی در صفحه های اینترنتی

نمونه این حالت زمانی است که در یک textbox متنی وجود داشته باشد .در برنامه های Win32 GUI متن داخل textbox تا زمانی که کاربر یا برنامه آن را تغییر نداده بر جای خود وجود دارد. اما در محیط اینترنت و نوع ASPبا هر بار refresh کردن صفحه کل اطلاعات ازبین می رود. البته این مشکل با استفاده از شئ های Request و Responseتقریباً قابل حل است ولی احتیاج به برنامه نویسی برای هر تکه از صفحه ASP دارد.

۳ -نداشتن Event Handler در محیط برنامه نویسی اینترنت

یکی از مهمترین ابزاری که در برنامه نویسی Win32 GUI وجود دارد استفاده از Event ها است. با تکنولوژی که در حال حاضر وجود دارد تنها راه رسیدن به این مهم استفاده از ActiveX است که به علت مسایل امنیتی در بیش از ۹۵ در صد مواقع توسط کاربر استفاده از آن رد می شود.

معایب استفاده ازAPI

API ها توابعی هستند که از ویندوز نسخه ۱ تا امروز در برنامه نویسی کاربرد داشته و دارند. مهمترین کاری که این توابع انجام می دهند انجام کارهای سخت و سطح پایین سیستمی است که احتیاج به برنامه نویسی زیادی دارند و یا حتی امکان ایجاد آن با زبان هایی مثل ویژوال بیسیک نیست. اما هر API از هر نسخه ویندوز تا نسخه دیگر آن می تواند دچار تغییرات بشود. برای مثال برنامه ای که در ویندوز ۹۸ نوشته شده باشد می تواند در ویندوز ۹۵ اجرا نشود. همچنین هم اکنون ابزارهای جدیدی به بازار آمده است که برای آنها نیز می توان برنامه نویسی کرد، مانند تلفن های سیار، کیوسک تلفن، دستگاه های کامپیوتری جیبی و غیره. در این نوع دستگاه ها دیگر ویندوز به مفهومی که در حال حاضر وجود دارد قابل اجرا نیست و در نتیجه API هم وجود ندارد. لازم به ذکر است که ویندوز CE برای دستگاه های مذکور می باشد ولی قابلیت های آن با ویندوزهای دیگر تفاوت زیادی دارد.

خلاصه از کتاب Professional VB.NET از Wrox

ویژگیهای دات نت[1]

آشنائی با مایکروسافت دات نت و شناخت قابلیتها و بررسی ویژگیهای آن

با اینکه مایکروسافت می دانست با ابزارهای قبلی شرکت می توان برنامه های اینترنتی نوشت ولی برای قبضه کردن بازار احتیاج به تکنولوژی جدیدی داشت . مایکروسافت از سال ۱۹۹۸ که ویژوال استودیو ۶ را به بازار وارد کرد در پی حل این مشکلات بود تا در سال ۲۰۰۰ NET. را در کنفرانس برنامه نویسان حرفه ای PDC به جهان معرفی کرد . از آن روز تا به حال مایکروسافت حدود ۸۰ ٪ از توانش را برای تکمیل NET. مصرف کرده است . در همین راه پروتکلهای جدیدی مانند SOAP یا Simple Object Access Protocol را ایجاد کرد . همچنین نسل جدیدی از برنامه نویسی به عنوان Web Service را تهیه کرده است.

مهم ترین اهداف NET.

طراحی برنامه های اینترنتی بر سبک برنامه های Win32 GUI : همانطور که اشاره شد برنامه نویسی برای Win32 GUI از قدرت خوبی برخوردار است ، در NET. برنامه های اینترنتی نیز از همین قدرت برخوردارند.

داشتن رابط گرافیکی خوب اینترنتی : به علت تغییرات اساسی که در برنامه در این سیستم داده شده برنامه اینترنتی قابلیت گرافیکی در حد برنامه های Win32 GUI دارند.

انتقال ساده به سیستم های دیگر : در NET. براحتی می توان برنامه ها را با یک کپی ساده به کامپیوتر های دیگر انتقال داد.

[1] حامد بنایی

پشتیبانی از زبانهای مختلف : در .NET به زبانهای برنامه نویسی مایکروسافت مثل ویژوال بیسیک ، سی شارپ و یا C++ محدود نیستیم . به طوری که در حال حاضر نسخه های Cobol.NET و Pascal.NET در حال ساخته شدن است . اما شرکت مایکروسافت زبان ویژوال بیسیک را به عنوان زبان اصلی برگزیده است . این مساله ریشه در تاریخ مایکروسافت دارد!

Platformهای آینده :هم اکنون .NET برای ویندوز نوشته شده است ولی در آینده نزدیک نسخه های Unixو Linux و همچنین برای Mobile و PDA نیز ارائه خواهد شد . این امر این امکان را می دهد که برنامه ای که برای ویندوز در .NET نوشته اید در تمامی سیستم عامل ها و دستگاه های بالا قابل اجرا باشد. در ادامه توضیحات کامل برای این مبحث ارائه خواهد شد.

ساختار.NET

مهمترین ویژگی در .NET Framework. این است که تمام لایه های برنامه نویسی را در بالای سیستم عامل دربرمی گیرد . که این شامل تمامی تکنولوژی های موجود که از طرف مایکروسافت یا شرکتهای دیگر ارائه شده است، می شود . در .NET. تمام اعمال تخصیص حافظه و سازماندهی فایل برعهده .NET Framework است و همین اصل باعث می شود که بتوان برنامه هایی نوشت که به سیستم عامل متکی نباشد. در آدرس اینترنتی زیر می توانید تصویر ساختار .NETرا مشاهده کنید.

www.geocities.com/hamedbanaei/netframe.htm

CLR زیر ساختار.NET

قلب .NET Framework همان CLR یا Common Language Runtime می باشد . CLR مسئول اجرای فایل ها ، فراخوانی آنها به حافظه و کامپایل کردن آنها به زبان MSIL یا Microsoft Intermediate

Languageاست . بعداً کدهای IL در هنگام اجرا، بوسیله برنامه کامپایلر just-in-time به زبان ماشین تبدیل می شود . این بدین معناست که در .NET. دو مرحله برای کامپایل شدن وجود دارد . اولین مرحله وقتی است که برنامه به هر زبان .NET. که باشد به IL کامپایل می شود که این کد کامپایل شده به IL قابلیت پخش در تمام .NET Frameworkرا دارد و بستگی به سیستم عامل ندارد . مرحله دوم زمان اجرا است که کامپایلر just-in-time کد ILرا به زبان آن ماشینی که برنامه در آن می خواهد اجرا شود کامپایل می کند .عهده دار برنامه نویسی شی گرا در سطح زبان های .NET. است ، برای مثال شما می توانید یک object در سی شارپ داشته باشید و آن را در ویژوال بیسیک فرا بخوانید و همچنین بوسیله وراثت تغییراتی در آن object بدهید . همچنین CLR بر Garbage Collectionها نیز نظارت می کند . بحث کامل درباره CLR در ادامه خواهد آمد.

کلاس های پایه در.NET Framework

این لایه حاوی تمامی کلاس ها و آبجکت هایی است که معمولاً مورد نیاز برنامه نویسان می باشد ، از جمله ADO.NETکه نسل جدید ADO است ، XMLکه قسمت زیادی از .NET. از این تکنولوژی استفاده می کند ، Threadingیا آبجکت هایی برای برنامه نویسیِ هر thread .

ASP.NET و Windows Forms

در مرحله بعدی دو روش کلی برنامه نویسی تحت اینترنت و تحت client قرار دارد که هر کدام خواص و آبجکت های مخصوص آن روش برنامه نویسی را دارا هستند.

خلاصه از کتاب Professional VB.NET ازWrox

مروری بر ساختار برنامه های.NET[1]

هر برنامه ای که بر مبنای دات نت تعریف می شود از سه قسمت مهم و اصلی تشکیل شده است: اسمبلی، ماژول و تایپ .

هر برنامه ای که بر مبنای .NET تعریف می شود از سه قسمت مهم و اصلی تشکیل شده است: Assemblies ، Modules و Types .اسمبلی ها اصلی ترین جز برای انتقال برنامه های .NET هستند .(Deployment) ماژول ها فایلهایی هستند که اسمبلی از روی آنها ساخته می شود و تایپ ها، واحد های پایه برای تعریف داده ها، property ها و توابع هستند.

اسمبلی ها

اسمبلی تشکیل شده از manifest و یک یا چند فایل ماژول، HTML.یاXML

Manifestنیز دارای اجزای زیر است:

•اطلاعاتی درباره خود اسمبلی که به صورت text ذخیره شده است. نمونه این اطلاعات، نام، ورژن، عمومی یا غیر عمومی بودن اسمبلی و ... است.

•نوع حفاظتی اسمبلی را توضیح می دهد. هر اسمبلی می تواند برای اجرا شدن نوع خاصی از لایه امنیتی داشته باشد که بر سه نوع است Required :، Optionalو.Denied

•اطلاعاتی درباره اسمبلی های دیگر که یک اسمبلی به آنها وابسته است از قبیل نام و نسخه آنها.

[1] حامد بنایی

•اطلاعاتی از قبیل زبان محلی اسمبلی، تاریخ، واحد پول و غیره.

ماژول ها

ماژول ها یا فایلهای DLL هستند یا فایلهای EXE Windows PE (Portable Executable) که حاوی IL، Meta Dataو به صورت اختیاری دارای manifest می باشد. هر اسمبلی فقط یک manifest می تواند داشته باشد، بنابراین اگر ماژولی حاوی manifest نیز بود فقط همان ماژول است که manifest دارد CLR .دو روش برای کامپایل هر فایل IL دارد، یکی install-time است که در زمان نصب برنامه فعال می شود و دیگری JIT یا کامپایلر just-in-timeکه به صورت method by method برنامه را کامپایل می کند. یعنی هنگامی که برنامه هر متد را صدا می زند کامپایل هم می شود . به صورت عادی برنامه ها به روش JIT کامپایل می شوند Meta Data .حاوی اطلاعات بیشتری درباره تعریف تایپ ها می باشد و به صورت IL است.

تایپ ها

تایپ ها دو نوع هستند، Valueو . Reference هر تایپ دارای property ، methodو field است. در ادامه درباره تایپ ها توضیحات بیشتری داده خواهد شد.

خلاصه از کتاب Professional VB.NET ازWrox

بررسی CLR[1]

بررسی و آشنائی با محیط زمان اجرای دات نت یعنی CLR و ویژگیهای آن از جهت انتقال برنامه ها، بحث نسخه و مدیریت حافظه

CLRیا Common Language Runtime در مرکز .NET platform واقع شده است .در حقیقت CLRعملی را که runtime ها در زبان های پیشین انجام می داند انجام می دهد .برای مثال ویژوال بیسیک ۶ دارای یک Runtime مخصوص به خود بود که کارهای مدیریت حافظه، فراخوانی فایلها و از این دست اعمال را برای برنامه های بیسیک انجام می داد، همچنین ویژوال سی نیز دارای چنین ابزاری بود .CLR یک runtime مشترک است برای تمامی زبانهایی که بر مبنای .NET طراحی شده اند.

نسخه

در ویژوال بیسیک ۶ و کلاً سیستم های قبلی که بر COM استوارند نسخه یک COM مساله بزرگی برای انتشار آن است. در حالی که شما می توانستید نسخه های مختلفی از یک COM داشته باشید ولی در استفاده از ProgIDها دارای محدودیت بودید. برای مثال در ویژوال بیسیک هنگام استفاده از Word.Application نمی توانستید نسخه آن را انتخاب کنید، یعنی برای مثال .Word.Application.9 در CLR تمام اجزا در GAC یا Global Assemblies Cacheبار می شوند. بعبارت دیگر اطلاعات اسمبلی ها در GAC قرار می گیرند. در CLR دو ویژگی برای اسمبلی هایی که در GAC قرار می گیرند وجود دارد:

Side-by-side versioning: بدین معنی که نسخه های مختلف یک اسمبلی در کنار هم بدون تداخل وجود دارند.

[1] حامد بنایی

Automatic QFE:اگر نسخه جدیدی از یک اسمبلی که با نسخه قبلی سازگاری دارد در GAC قرار گیرد، CLRاین مساله را تشخیص می دهد و از آن لحظه به بعد از نسخه جدید استفاده می کند.

نسخه ها در CLR از چهار قسمت Major.Minor.Revision.Build تشکیل شده اند. اگر در Major و Minorتغییری داده شود، این بدین معناست که این نسخه از اسمبلی دیگر با نسخه های قبلی سازگاری ندارد.

انتقال

برنامه هایی که به زبان ویژوال بیسیک قدیمی نوشته می شوند در هنگام انتقال به کامپیوترهای دیگر دچار مشکلات فراوان می شوند. تمامی اجزا باید در رجیستری ثبت شوند. اگر نسخه جدیدی ایجاد شود امکان دارد برنامه هایی که از نسخه قدیمی استفاده می کردند را از کار بیندازد. ولی در .NET هم مساله نسخه ها تقریباً حل شده است و هم نحوه انتقال. کافی است در کامپیوتر مقصد .NET Framework نصب شده باشد. در این صورت به راحتی با استفاده از دستور xcopy داس می توانید برنامه خود را انتقال بدهید!

شایان ذکر است که این نوشته صرفاً توضیح مختصری درباره هر کدام از قسمت های .NET می دهد و برای هر کدام از این اجزا می توان یک مقاله مفصل نوشت.

مدیریت حافظه

درباره مدیریت حافظه CLR بیشتر در زمینه VB بحث می کنیم.

مدیریت بهتر بر Garbage Collection :

Runtimeدر ویژوال بیسیک دارای سیستم خود کار برای Garbage Collection ها است. در ویژوال بیسیک ۶، runtime به شکل خودکار منابع را از شئ هایی که دیگر توسط برنامه استفاده نمی شوند می گیرد. برای

مثال فرض کنید می خواهیم یک فایل LOG ایجاد کنیم، برای این کار می توان از دو شئ Scripting.Stream و Scripting.FileSystemObjectاستفاده کرد. اگر این دو شئ را در تابعی به کار ببریم، بعد از اتمام عملیات، runtimeمنابع اختصاص داده شده به این دو شئ را می گیرد. اما مواردی وجود دارد که runtime به آسانی امکان تشخیص شئ و زمان بلااستفاده بودن آن را ندارد.

:Cyclical References

یکی از مهمترین زمانهایی که runtime مربوط به VB نمی تواند تشخیص بدهد که آیا یک شئ به منابعی احتیاج دارد یا اینکه کارش به پایان رسیده زمانی است که در متن برنامه حالتی به نام Cyclical Reference بوجود بیاید. به عنوان مثال اگر شئ A به صورت reference از شئ B استفاده کند و همچنین شئ B از A.

هر شئ COM مسئول نگهداری تعداد دفعاتی است که فعال شده است. بدین صورت که هر بار از روی آن ساخته شود با AddRef و هربار که یک برنامه آن را رها کند با Release از IUnKnown interface شمارنده ای را یکی افزایش یا کاهش می دهد. اگر آن شمارنده به صفر برسد شئ تمامی منابع خود را آزاد می کند، اما اگر هنگامی که یک شئ COM غیر فعال شود ولی از Release استفاده نشود یک شئ بلا استفاده در حافظه می ماند.

runtime مربوط به VB 6 این کار را به صورت خودکار انجام می دهد، اما هنگامی که دو شئ به هم اشاره می کنند بحث فرق می کند و runtime به شکل حالت ساده نمی تواند یک شئ را از حافظه بردارد. برای مثال به قطعه برنامه زیر توجه کنید.

```
'Class : CCyclicalRef

Dim m_objRef as Object
```

```
Public Sub Initialize(objRef as Object)
   Set m_objRef = objRef
End Sub

Private Sub Class_Terminate()
   Call Msgbox("Terminating.")
   Set m_objRef = Nothing
End Sub
```

در کلاس CCyclicalRef متدی با نام Initialize تعریف شده است که در پارامتر های خود یک شئ را می پذیرد و از روی آن یک شئ دیگر می سازد. حال از کلاس تعریف شده در کد زیر استفاده می کنیم.

```
Dim objA as New CCyclicalRef
Dim objB as New CCyclicalRef

Call objA.Initialize(objB)
Call objB.Initialize(objA)

Set objA = Nothing
Set objB = Nothing
```

ابتدا دو نمونه از روی کلاس CCyclicalRef با نامهای objA و objB ساختیم، حال هر دوی این شئ ها یک بار ساخته شده اند و شمارنده آنها یک است. با استفاده از متد Initialize هر دوی شئ ها، آدرس حافظه یکدیگر را مبادله می کنند و به این شکل از روی هر دوی آنها یک بار دیگر ساخته می شود و شمارنده هر دو، ۲ می شود . شماره یک مربوط به خود برنامه است و شماره دوم مربوط به شئ هایی که به هم اشاره می کنند. سپس هر دوی آنها را

برابر nothing قرار می دهیم تا آنها را از حافظه حذف کنیم. در اینجا شمارنده هر دو یکی کم و برابر یک می شود. بنابراین برنامه terminate شده است ولی هنوز شئ در حافظه وجود دارد.

Garbage Collector در CLR :

روش Garbage Collector در CLR با VB 6 Runtime تفاوت دارد. در روش جدید که خودِ CLR و GCمسئول اجرای آن هستند، آخرین باری که از nothing استفاده می شود ولی هنوز شئ در حافظه مانده باشد، GCاین وضعیت را درک می کند و آن شئ را بعد از مدتی از حافظه خارج می کند. شئ هایی هستند که مانند COMمسئول غیر فعال کردن خود نیستند، GCآنها را نیز تشخیص داده و اگر برنامه ای دیگر از آنها استفاده نکند، از حافظه حذف می شوند.

Finalize:

GC متد Object.Finilize را قبل از اینکه هر شئ را از حافظه حذف کند صدا می زند. این متد می تواند در هر زمانی بعد از اینکه برنامه دیگر از شئ استفاده نکرد صدا زده شود، لذا در هنگام استفاده از این متد باید نکته را در نظر گرفت. در .NET بهتر است قبل از اینکه یک شئ بوسیله nothing از بین برود از متد Dispose یا Close استفاده شود تا سریعاً از حافظه حذف شود.

روش سریعتر تخصیص حافظه به شئ ها

هر زمان که یک برنامه یک شئ بسازد حافظه ای به آن اختصاص می یابد، آن حافظه در virtual memory است که برای هر برنامه اختصاص می یابد و به آن heap گفته می شود .CLR مفهومی به نام Managed Heap دارد بدین معنی که شئ ها را مدیریت می کند و آنها را در یکِ Heap مدیریت شده قرار می دهد و CLR مسئول حفاظت آنهاست.

از مزایای این روش این است که راندمان سرعت اختصاص دادن حافظه بالا می باشد. به طور کلی وقتی کُد های مدیریت نشده در Heap های مدیریت نشده قرار می گیرند، به دنبال جایی در حافظه می گردند که بتوانند خود را در آن قرار دهند. در CLR شیءای که ساخته می شود همیشه در بالای آخرین شیءای که ساخته شده در heap قرار می گیرد. تصویر مربوطه را در آدرس زیر مشاهده کنید.

www.geocities.com/hamedbanaei/memorialoc.htm

ابتدا CLR حافظه ای برای شئ هایA ، Bو C بر روی Heap در نظر می گیرد. سپس شئ B از حافظه حذف می شود، در ادامه نیز برنامه یک شئ دیگری به نام D می سازد و آن را بر روی آخرین شیءای که ساخته شده بود یعنی C قرار می دهد. این روش برای اختصاص حافظه سریع است. اما اگر CLR به انتهای heap برسد چه عملی باید انجام دهد؟ همان طور که مشاهده کردید این روش به شکل افزایشی شئ ها را در حافظه قرار می دهد. وقتی CLRدیگر نتواند از حافظه استفاده کند GC را فرا می خواند .GC هم فضاهایی مانند B را کاملاً آزاد می کند و هم شئ ها را فشرده می کند و در پایین heap قرار می دهد تا بالای heap آزاد باشد.

خلاصه از کتاب Professional VB.NET از Wrox

پردازش های مبتنی بر Client Server[1]

بررسی معماری Client/Server در مدلهای تک لایه، دو لایه، سه لایه و چند لایه و اصول طراح معماری برنامه های تحت وب

در اواسط دهه ۸۰ میلادی و زمانیکه اولین بار تولیدکنندگان تجهیزات شبکه، محصولات خود را به بازار عرضه کردند، واژه Client/Server وارد عرصه کامپیوتر گردید. در آن زمان واژه فوق صرفا در رابطه با تجهیزات سخت افزاری (کامپیوتر) استفاده می شد و کامپیوتری که از آن بعنوان مرکز ثقل ارائه خدمات در یک شبکه یاد می شد، را با نام Server و کامپیوتری که از این امکانات استفاده می کرد را بعنوان Client می شناختند (سایه نرم افزار بر این واژه حضور سنگینی نداشت.)

امروزه واژه فوق دارای یک معنی خاص است که چندان مرتبط با سخت افرار نمی گردد. اغلب مردم هنوز واژه Clientرا به یک کامپیوتر فیزیکی نسبت داده و واژه Server را به کامپیوتر فیزیکی دیگری که به آن متصل و سرویس هائی را ارائه می نماید، اطلاق می نمایند. مطلب فوق با اینکه درست است ولی صرفا یک بخش اندک از تمامی واقعیت های موجود در این زمینه است. واژه فوق امروزه در مقیاس وسیعتری به خدمت گرفته می شود. بمنظور آشنائی بیشتر با این واژه لازم است در ابتدا با ساختار و یا معماری عمومی یک نرم افزار آشنا شویم.

اغلب برنامه های کاربردی دارای سه لایه اصلی می باشند:

لایه Presentation :(بالاترین لایه) این لایه مسئول ایجاد ارتباط متقابل بین انسان و کامپیوتر است (رابط کاربر). لایه فوق مسئولیت گرفتن اطلاعات ورودی از صفحه کلید، ماوس و سایر دستگاههای ورودی و نمایش اطلاعات ذیربط بر روی دستگاههای خروجی نظیر صفحه نمایشگر است .

[1] محمد جواد سخائی

لایه Application یا Business Logic :لایه فوق مسئول اعمال و پیاده سازی سیاست های مورد نظر در یک نرم افزار است، در حقیقت با عملکرد لایه فوق است که می توان تفاوت بین یک نرم افزار از نرم افزار دیگر را مشاهده و بعنوان مثال تفاوت بین یک نرم افزار ثبت سفارش و یا انبارداری را حس کرد.

لایه Service: این لایه مسئول ارائه سرویس های خاص و مورد نیاز برای سایر لایه ها نظیر سرویس های مربوط به فایل، چاپ، ارتباطی و از همه مهمتر دسترسی به بانک های اطلا عاتی است. در ادامه بحث خود را بر روی مجموعه ای از نرم افزارها ئی متمرکز خواهیم کرد که نیازمند سرویس های بانک اطلاعاتی باشند.

تعداد طبقات(Tires) ، در یک نرم افزار Client Server به نحوه ارتباط (متراکم، معمولی) هر یک از سه لایه گفته شده بستگی خواهد داشت. در ادامه به بررسی مدل های رایج در این زمینه خواهیم پرداخت.

مدل One-Tire

در این نوع نرم افزارها سه لایه گفته شده بصورت متراکم و فشرده در کنار یکدیگر قرار می گیرند. در مدل فوق لایه Presentation دارای آگاهی خاص و جزئی از ساختار بانک اطلاعاتی است. لایه Application اغلب بصورت موجی با لایه های Presentation و Service مرتبط خواهد بود. تمام سه لایه گفته شده بهمراه بانک اطلاعاتی، اغلب بر روی یکدستگاه کامپیوتر قرار گرفته و اجرا خواهند شد .نرم افزارهائی با این خصوصیت بسادگی طراحی شده و بکمک ابزارهای برنامه نویسی امروزی بسرعت نوشته خواهند شد.

در صورتیکه بخواهیم یک نرم افزار One-tire با چندین کاربر را طراحی نمائیم، می توان نرم افزار را بر روی چندین کامپیوتر اجرا و با به اشتراک گذاشتن بانک اطلاعاتی زمینه استفاده از داده های موجود در بانک را برای سایر کاربران نیز فراهم نمود. بانک اطلاعاتی را می توان بر روی یکدستگاه کامپیوتر معمولی در یک شبکه نظیر به نظیر (Peer to Peer) و یا بر روی یک سرویس دهنده فایل (File Server) نصب نمود. در این حالت هر یک از

کامپیوترهائی که برنامه بر روی آنها اجرا می گردد می بایست دارای یک نسخه از Database Engine بوده تا قادر به استفاده از داده های موجود در بانک اطلاعاتی باشند. در این مدل صرفا داده ها به اشتراک گذاشته شده و منطق بانک اطلاعاتی به اشتراک گذاشته نشده است. این نوع از نرم افزارها (چند کاربره (One Tire تا زمانیکه تعداد کاربران کم باشد موفق عمل می نمایند ولی با افزایش تعداد کاربران، با مشکل مواجه می شوند.

علت عمده بروز مشکل پایبند بودن این نوع از نرم افزارها به انجام عملیات مربوط به بانک های اطلاعاتی بر روی هر یک از سرویس گیرندگان است. مثلا اگر برنامه ای از این نوع نیاز داشته باشد که لیست تمامی کاربرانی را که نام آنها Reza است، را نمایش دهد، می بایست تمامی اطلاعات (رکوردهای داده و ایندکس های مربوطه) بمنظور پاسخگوئی به درخواست واصل شده، بر روی شبکه فرستاده شود. در برخی حالا ت خاص و با توجه به پیچیدگی درخواست های صادر شده برای اطلاعاتی خاص، ممکن است تمامی بانک اطلاعاتی برای سرویس گیرنده ارسال گردد.

اگر از یک سطح فنی به مسئله فوق نگاه کنیم، مدیریت Database Engine های مستقل بر روی سرویس گیرندگان بمنظور ممانعت از بروز تعارض (Conflict) بین دو سرویس گیرنده جهت تلاش برای دستیابی و یا بهنگام سازی برخی رکوردها مشکل است (مسئله Record Locking).

مدل Two Tire

بمنظور حل مشکل مطرح شده در مدل One-tire از بعد کارائی و مسائل فنی مربوطه، مدل فوق معرفی گردید. نرم افزارهائی که با اتکا بر مدل فوق طراحی و پیاده سازی می گردند در اغلب موارد دارای عملکردی مشابه مدل One Tireبوده با این تفاوت مهم که Database Engine بر روی سرویس گیرنده ها اجرا نخواهد شد.

در مدل فوق بانک اطلاعاتی بر روی سرویس دهنده اجرا می گردد. از روش های متعددی برای ارتباط بین لایه Application(Logic) و Database Service استفاده می گردد. SQL بان ساختیافته پرس و جو) از

متداولترین روش های موجود در این زمینه است. دستورات SQL به سرویس دهنده بانک اطلاعاتی ارسال شده و در آنجا عملیات مربوطه بصورت محلی انجام و نتیجه (اطلاعات مربوط به درخواست ارسال شده) برای سرویس گیرنده ها ارسال خواهد شد. در مدل فوق صرفا سرویس دهنده بانک اطلاعاتی از برنامه مجزا شده و لایه های Presentation و Busines Logicهمچنان در هم تنیده هستند. دو لایه فوق همچنان دارای آگاهی اساسی (محرمانه) از بانک اطلاعاتی خواهند بود.

نوشتن برنامه هائی از این قبیل تا اندازه ای پیچیده تر از مدل قبل است. امروزه ابزارهای برنامه نویسی نیز مجهز به پتانسیل هائی شده اند که طراحی و نوشتن این نوع از برنامه ها را سرعت می بخشد. اغلب ابزارهای برنامه نویسی دارای امکاناتی جهت استفاده از DataBase Engines بوده که می توان از آنها در طراحی برنامه های One-Tire استفاده کرد (نظیر Jet Engine که توسط اکسس و ویژوال بیسیک استفاده می گردد) اما نرم افزارهای Two Tire نیازمند محصولات مجزای بانک اطلاعاتی نظیر Sybase , IBM DB2 , Oracle و SQL Sever می باشند.

مدلThree Tire

این مدل همانگونه که احتمالا حدس زده اید تمامی سه لایه گفته شده را در بخش های مستقل قرار می دهد. در مدل فوق Business Logic یک سرویس است و می تواند بر روی کامپیوتر اختصاصی خود فعال و اجرا گردد . زمانیکه Business بصورت یک سرویس دهنده در نظر گرفته می شود با نام Application Server نامیده می شود. یک Application Server اغلب ممکن است بر روی همان کامپیوتری که DataBase Engine قرار دارد، نصب گردد. شاید یکی از دلایل مهم جهت انجام این کار افزایش کارآئی سیستم باشد.

یکی از مزایای مهم و کلیدی، داشتن یک Application Server این است که بتوان آن را در محلی قرار داد که به بهترین نوع ممکن خدمات خود را ارائه نماید. در این مدل مسئله حائز اهمیت در این است که تمامی Application Serverها بتوانند و می بایست سرویس بانک اطلاعاتی خود را از یک کامپیوتر مرکزی دریافت دارند. (

ممکن است در برخی حالات تعدادی از کاربران نرم افزار از یک Application Server که بر روی یک کامپیوتر مجزا قرار گرفته است استفاده نمایند و یک کاربر از راه دور ممکن است Application Server را بر روی یکدستگاه کامپیوتر اختصاصی اجرا نماید.) بهرحال محل Application Server و Database Server ارتباطی با کاربر نداشته و تمامی آنها با یک روش یکسان از نرم افزار و توانائی آن استفاده می نمایند.

در مدل فوق لایه Presentation دارای آگاهی خصوصی از بانک اطلاعاتی نبوده و لایه فوق از طریق لایه Application Server و بکمک یک استراتژی خاص با بانک اطلاعاتی مرتبط خواهد بود. مرورگرها در حالت خاص دارای هیچگونه شناختی از ساختار بانک اطلاعاتی در سایت Amazon.com نمی باشند ولی با این حال قادر به ارتباط با بانک اطلاعاتی و خرید یک کتاب هستند. در مدل فوق با نگرش وب، سرویس گیرنده از طریق یک پروتکل خاص با یک Application Server مرتبط می گردد. برنامه هائی از این نوع (مدل Three Tire) پیچیده تر از مدل های قبلی بوده و هنوز ابزارهای برنامه نویسی خاصی در این زمینه وجود ندارد و برنامه نویسان مجبور به نوشتن حجم بالائی از کدها خواهند بود.

مدل N Tire

این مدل امروزه بسرعت رایج و مطرح شده است. در حقیقت مدل Three Tire در حالت خاص به سمت-N Tireمیل خواهد کرد. در این حالت یک Application Server می تواند درخواست خود را از چندین سرویس دیگر داشته باشد. هر یک از سرویس های صدا زده شده نیز خود می توانند سرویس های دیگری را جهت پاسخگوئی به درخواست واصل شده، فعال نمایند. واژه MiddleWare اغلب جهت تشریح ارتباط یک برنامه یا Business Logic بر روی یک Application Server استفاده می گردد.

چه میزان از Bussines Logic می بایست بر روی Application Server قرار گیرد؟

بدون شک یکی از بخش های مهم هر نرم افزار که دائما می تواند دستخوش تغییرات گردد، مجموعه قوانینی است که با اعمال آنها سیاست عملکردی یک نرم افزار تعیین می گردد. مثلا در یک سیستم بازرگانی می توان قانونی را داشته باشیم که برای خریدهای بالای یکصد هزار تومان مجوز مدیر مربوطه فرض است. در این حالت می توان قانون فوق را بصورت یک روتین (سرویس) و بصورت جامع طراحی و در لایه Application قرار داد، سرویس فوق می تواند توسط سایر سرویس های موجود در این لایه و یا سایر لایه ها مورد استفاده قرار گیرد. بدیهی است در صورتیکه این سیاست به نوعی تغییر نماید و قرار شود از این پس خریدهای بالای یکصد و پنجاه هزار تومان مکلف به تایید مدیریت مربوطه باشند، بسادگی با اعمال تغییر در روتین فوق و تزریق سیاست جدید، زمینه استفاده اتوماتیک از آن برای سایر سرویس های استفاده کننده فراهم می گردد.

نحوه و زمان تغییر سیاست فوق از دیدگاه استفاده کننده و لایه Presentation مهم نبوده و تغییرات بصورت خودکار در تمامی سرویس های موجود در سایر لایه ها حس خواهد شد. بنابراین مجموعه قوانین و سیاست هائی که در روند عملیاتی یک نرم افزار نقش تعیین کننده ای را دارند، می بایست در لایه Application قرار گرفته تا بدینوسیله امکان درج تغییرات و اعمال سیاست های جدید مرکزیت یافته و مسائل مربوط به پشتیبانی و ارتقا یک نرم افزار با اطمینان خاطر و صرف کمترین زمان و هزینه صورت پذیرد.

در برخی از موارد می توان این سیاست ها را در قالب مجموعه ای از سرویس ها در لایه Presentation قرار داد. بررسی صحت داده های ورودی یک نمونه مناسب در این زمینه است. در این مورد اغلب قوانین جهت بررسی اعتبار و صحت داده های ورودی بر روی لایه Presentation قرار خواهد گرفت. بدیهی است در چنین حالتی بجای ارسال اطلاعات بررسی نشده به لایه Application و بکارگیری یک روتین جهت بررسی صحت داده ها، می توان این عملیات را در لایه Presentation قرار داد تا بدینوسیله از یکطرف ترافیک محیط انتقال داده ها افزایش نیابد و از طرف دیگر کاربران رودرو با لایه Presentation بازخوردهای سریعی را از سیستم داشته باشند. بهرحال در چنین حالاتی بخشی از منطق عملکرد یک نرم افزار را در لایه Presentation قرار داده ایم. در صورتیکه حجم Logic

اضافه شده در لایه Presentation کم و ناچیز باشد، در اینصورت لایه فوق بصورت انحصاری مسئولیت های پیش فرض خود را دنبال خواهد کرد. در چنین وضعیتی سرویس گیرنده را Thin Client می گویند. در حالتیکه بر روی سرویس گیرنده، Logic بالائی قرار گرفته باشد، به آن Fat Client می گویند.بهترین نمونه از یکThin Client ، مرورگرهای وب بوده که قادر به ارتباط با انواع نرم افزهائی است که بر روی وب سایت قرار دارند.

جمع بندی

واژه Client Server دارای معانی بمراتب بیشتری نسبت به جداسازی یک کامپیوتر سرویس گیرنده و سرویس دهنده از یکدیگر است واژه فوق بسرعت در دنیای نرم افزار نیز مطرح و دارای جایگاه ویژه ای در این زمینه شده است . از دیدگاه فوق یک روتین (سرویس) می تواند ارائه دهنده خدمات خاصی به سایر سرویس ها باشد. در چنین وضعیتی سرویس ارائه دهنده خدمات را Server و سرویس استفاده کننده از یک خدمات را Client می گویند. با تعمیم سیاست های طراحی نرم افزار از مدل های One Tire به Two-Tire و Three Tire و نهایتا N-Tire و تاکید بر نگرش ساختیافته و اصولی به عملکرد هر یک از لایه ها، مفهوم روتین های سرویس دهنده (Server) و روتین های سرویس گیرنده (Client) جایگاه ممتازی را پیدا نمودند.

یک سرویس می تواند در عین خدمات دهی به سایر سرویس های متقاضی، خود نیز از خدمات سایر سرویس ها استفاده نماید. بنابراین یک سرویس دهنده در چنین حالتی بصورت اختصاصی صرفا رسالت سرویس دهی و یا سرویس گیری را انجام نخواهد داد. اگر از دیگاه هر لایه به عملکرد سرویس ها نظری داشته باشیم، قطعا تمامی آنها مسئولیت ارائه یک سرویس خاص را در لایه مربوطه برعهده خواهند داشته و قدرمطلق تمامی آنها ارائه خدمات است. مهمترین مزیت نگرش فوق حرکت ب iسمت تولید سرویس هائی خواهد بود که اولا امکان استفاده از آنان در چندین نرم افزار فراهم شده و ثانیا زمینه تحقق اصل بسیار مهم استفاده مجدد از کدهای نوشته شده (Reusable Code) نیز فراهم

می گردد. امروزه با توجه به نیاز روزافزون به طراحی و پیاده سازی نرم افزارهای متکی بر وب، مدل های Three Tire و N-Tire بشدت مورد توجه طراحان و پیاده کنندگان نرم افزارهای متکی بر بستر وب قرار گرفته است.

فرستادن Email از طریق ASP.NET[1]

در این مقاله نحوه فرستادن Email از یک صفحه ASP.NET نشان داده شده است .

لینک دریافت کد: www.iranasp.net/download/shahoo04.zip

برای فرستادن Email از یک صفحه ASP.NET بایستی از کلاس SmtpMail که در فضانام System.Web.Mail قرار دارد استفاده کرد که شامل متد استاتیک Send است .بهرحال ساده ترین راه برای فرستادن Email فراخوانی یک نمونه متد Send از کلاس MailMessage است. کلاس MailMessage در فضانام System.Web.Email قرار دارد که پیغام Email را نشان می دهد.

کلاس MailMessage در برگیرنده خاصیتهای مشابه شئ CDONTS است (CDONTSرایجترین شئ برای فرستادن Email در ASP کلاسیک است).

بعنوان مثال خاصیتهایی چون:

Body ،Priority ،Subject ،BodyFormat ،Bcc ،Cc ،From ،To.

بهرحال برای فرستادن Email بایستی نمونه ای از کلاس MailMessage ساخته شود و خواص آن مشخص شود:

۱ -ابتدا نمونه ای از کلاس MailMessage ساخته می شود :

```
Dim ObjMM As New MailMessage
```

۲ -سپس آدرس email دریافت کننده :

[1] شاهو طوفانی

```
ObjMM.To="someone@someaddress.com"
```

۳ -آدرس فرستنده :email

```
ObjMM.From="someoneelse@someotheraddress.com"
```

۴ -همچنین می توان فیلدهای Cc و Bcc را داشت :

```
ObjMM.Cc="someone2@someaddress.com"
ObjMM.Bcc="someone3@someaddress.com"
```

۵ -بوسیله خاصیت BodyFormat نحوه فرستادن email بصورت text و یا html مشخص می شود :

```
Obj.MM.BodyFormat=MailFormat.Text
```

۶ -خاصیت Priority میزان Security میل و فرستادن آنرا مشخص می کند که شامل سه انتخاب High ، Normalو Low است :

```
ObjMM.Priority=MailPriority.Normal
```

۷ -خاصیت Subject عنوان میل را مشخص می کند :

```
ObjMM.Subject="Hello there"
```

۸ -بوسیله Body بدنه Email مشخص می شود (پیغام):

```
ObjMM.Body = "This is body!"
```

توجه داشته باشید که فضانام System.Web.Mail در صفحه ASP.NET فراخوانی شود :

```
<% @import NameSpace="System.Web.Mail" %>
```

بعد از تعیین خواص MailMessage ، متد Send از کلاس SmtpMail برای کلاس شبیه سازی شده MailMessage بصورت زیر فراخوانی می شود :

```
SmtpMail.Send(ObjMM)
```

کلاس SmtpMail برای فرستادن Email از سرویسSmtp ی استفاده می کند که در ساختار IIS وجود دارد. به تعبیر دیگر با بکار بردن متدSend ، از سرور داخلی SMTP برای فرستادن Email استفاده می شود. برای مشخص کردن سرور SMTP از نوع دیگر (غیر از پیش فرض) باید خاصیت SmtpServer آن بصورت زیر تعیین شود :

```
SmtpMail.SmtpServer= emailservername
```

در غیر اینصورت SmtpServer پیش فرض (Default) بصورت زیر است :

```
SmtpMail.SmtpServer=""
```

یکی از مهمترین کاربردهای فرستادن Email از طریق صفحات وب دریافت نظرات، اطلاعات و ... از طریق یک فرم html و ارسال آن به مدیر سایت است.(Feedback)

کد نمونه برنامه در آدرس www.iranasp.net/download/shahoo04.zip بطور کامل این فرآیند را نشان می دهد.

اصول پیاده سازی نرم افزارهای مبتنی بر وب[1]

در این مقاله تاریخچه و اصول پیاده سازی نرم افزار بخصوص بر روی بستر وب تشریح شده است .

بمنظور بررسی مقوله پیاده سازی نرم افزار بر روی بستر وب بحث خود را بر روی دو موضوع عمده متمرکز می کنیم: شناخت مدل های رایج جهت پیاده سازی نرم افزار از ابتدا تا کنون و شناخت وب بعنوان بستر مربوطه بهمراه تکنولوژی هائی که در این زمینه مورد استفاده قرار می گیرند.

هدف ما رسیدن به نقطه ای است که مشخص نمائیم، برای طراحی و پیاده سازی نرم افزار بر روی بستر وب، اولا از چه نوع مدلی برای پیاده سازی استفاده می گردد و ثانیا روند توسعه و تکامل وب را با تاکید بر نیازهای نرم افراری از بعد ابزارشناسی دنبال کرده و از این رهگذر جایگاه هر ابزار (تکنولوژی) تبین شده تا بدین وسیله بتوانیم از هر چیز در جایگاه خود و در زمان مربوطه استفاده کنیم. بهرحال وب یک واقعیت انکار ناپذیر بوده و سایه حضور آن را در همه جا می توان احساس کرد. نرم افزار نیز مجری خواسته های انسانی در سرزمین سخت افزار است، این بار با یک چالش جدی مواجه شده است : پاسخگوئی به خیل نیازهای جدید مطرح شده متکی بر بستر وب.

در بخش اول این مقاله موضوع اول یعنی شناخت مدل های پیاده سازی نرم افزار تشریح خواهد شد. به این امید که از این رهگذر به نقطه ای برسیم که یک مدل مناسب جهت پیاده سازی برنامه های تحت وب را معرفی و آن را بعنوان پایه و اسا س کار خود قرار دهیم. در ابتدا لازم است به این اصل بدیهی اشاره شود که یک برنامه کامپیوتری حاصل ترکیب داده ها و منطق است. منطق یک برنامه از طریق کدهای مربوطه که به یکی از زبانهای برنامه نویسی نوشته خواهند شد، مسئول تحقق خواسته های تعریف شده برای یک نرم افزار از طریق انجام عملیات مورد نیاز بر روی داده ها است. داده ها خود می توانند به اشکال و روش های متنوعی سازماندهی و در اختیار یک نرم افزار قرار گیرند .

Program = Logic(Code) + Data

[1] محمد جواد سخائی

مدل های پیاده سازی یک نرم افزار از ابتدا تاکنون متاثر از عوامل متفاوتی بوده است که جایگاه سخت افزارها بعنوان پتانسیل هائی که می بایست توان خود را در جهت بالفعل نمودن نرم افزارها قرار دهند، نقش برجسته ای دارد. مدل های پیاده سازی نرم افزار را می توان در پنج گروه عمده بشرح زیر تقسیم نمود:

MainFrame Architecture

در این مدل دو عنصر فیزیکی مورد اهتمام جدی بودند: کامپیوتر اصلی که با نام Host شناخته می شد و سخت افزارهای استفاده کننده از کامپیوتر اصلی که با نام ترمینال شناخته می شدند. تمامی منطق یک برنامه (Logic) بهمراه داده های مربوطه (Data) بر روی Host نصب می شد و کاربران با استفاده از ترمینال ها که بمنزله پایانه هائی جهت ورود و خروج (نمایش) اطلاعات بودند، قادر به ارتباط با سیستم و اجرای یک برنامه بودند. تمرکز منطق برنامه در یک محل (Host) از مهمترین ویژگی های این مدل است.

File Server Architecture

از این مرحله دو واژه معروف Server و Client پا به عرصه وجود گذاشتند. حیات و معنی این واژه ها محدود به سخت افرار بود و به مرزهای نرم افزار نرسیده بود. در این راستا کامپیوتری که برای دیگران سرویس هائی را ارائه می کرد با نام Server یا در این حالت خاص (File Server) و کامپیوترهائی که از این خدمات بهره مند می شدند را Clientمی گفتند. مدل فوق پاسخی اولیه به نیازهای کاربران یک شبکه کامپیوتری بود. در مدل فوق منطق یک برنامه بر روی یک Client نصب و داده ها بر روی Server قرار می گرفتند. دراین مدل داده ها در یک فایل (با یک ساختار خاص) قرار گرفته و یک بانک اطلاعاتی را بوجود می آوردند و سرویس دهنده مسئول ارائه تسهیلاتی برای جابجائی و ارسال اطلاعات موجود در فایل ها بود. تمرکز منطق برنامه در یک محل (Client) از مهمترین ویژگی های این مدل است.

Client Server Architecture

مدل فوق در پاسخ به اشکالات بوجود آمده در مدل قبل ارائه گردید. در مدل فوق کامپیوتر ارائه کننده خدمات را همچنان Server و کامپیوترهای استفاده کننده را Client می نامیدند. داده های یک برنامه (بانک های اطلاعاتی) همچنان بر روی سرویس دهنده قرار داشت ولی در رابطه با منطق برنامه اصل توزیع پردازش مورد توجه جدی قرار گرفت. بنابر اصل فوق بخشی از منطق یک برنامه را در حد امکان بر روی سرویس گیرنده اجرا و بخش دیگر از منطق برنامه بر روی سرویس دهنده اجرا می گردید. در مدل فوق برای اجرای یک برنامه دو پردازش جداگانه یکی بر روی سرویس دهنده و دیگری بر روی سرویس گیرنده فعال و هر یک نقشی در اجرای یک برنامه را برعهده می گرفت. مهمترین ویژگی مدل فوق مطرح کردن اصل پردازش توزیع شده است .

Two Tire Architecture

در مدل فوق اصل تقسیم وظیفه بصورت یک واقعیت انکار ناپذیر مورد توجه جدی قرار گرفت در این مدل همچنان کامپیوترهای سرویس دهنده و سرویس گیرنده جایگاه قبلی خود را داشتند با این تفاوت بسیار مهم که حوزه انجام هر عملیات (منطق) تا اندازه ای شفاف تر گردید. مثلا جهت دستیابی به بانک های اطلاعاتی تمامی DataBase Engineبر روی سرویس گیرنده قرار می گرفت و سرویس گیرندگان جهت استفاده از داده های موجود در بانک اطلاعاتی نیازمند نصب امکانات نرم افزاری و آگاهی از ساختار بانک اطلاعاتی نبودند. از این مرحله واژه های سرویس گیرنده و سرویس دهنده پا به عرصه نرم افزار نیز گذاشتند و مفاهیمی نظیر سرویس دهنده بانک اطلاعاتی و رایج شد. مهمترین ویژگی مدل فوق تاکید بر اصل تقسیم فعالیت در چهارچوب ارائه طبقات (Tires) بود.

Three Tire Architecture

در مدل فوق اصل تفکیک مجموعه قوانین (سیاست های) مربوط به عملکرد یک نرم افزار مورد توجه جدی قرار گرفت. بدیهی است با حجیم شدن یک نرم افزار از یکطرف و افزایش تعداد کاربران از طرف دیگر و تغییرات متوالی در

سیاست های راهبردی و عملیاتی یک نرم افزار در یک سازمان، مسائل مربوط به پشتیبانی و ارتقاء یک نرم افزار از مسائل بسیار مهم و حیاتی در موفقیت افزایش طول عمر یک نرم افزار محسوب می گردد.

در مدل فوق همچنان واژه های سرویس دهنده و سرویس گیرنده حضور مستمر خود را ادامه دادند با این تفاوت بسیار مهم که حوزه عملکرد این واژه ها در رابطه با نرم افزار بسیار برجسته گردید. در این مدل از سه لایه استفاده می گردد: لایه اول مسئول تماس و ارتباط با کاربر و ارائه دهنده محیط رابط کاربر، لایه دوم (میانی (مسئول نگهداری و اجرای سیاست ها و قوانین کلیدی و راهبردی حاکم بر نرم افزار و لایه سوم مسئولیت نگهداری بانک اطلاعاتی و ارائه سرویس و خدمات به لایه متقاضی (لایه دوم) است. عملکرد لایه دوم (میانی) بسیار گسترده بوده و می توان با همگرا نمودن این عملکردها به چند بخش، لایه های دیگری را نیز در این بخش داشته باشیم، در چنین حالتی این مدل اصطلاحا N-Tire نامیده می شود.

مدل فوق بهترین انتخاب برای پیاده سازی نرم افزار بر روی بستر وب است. کلید طلائی طراحی این نوع نرم افزارها توانائی نوشتن عناصر (اجزا) بگونه ای است که از یکطرف امکان بکارگیری آنها بسادگی در لایه ها و حتی چندین برنامه فراهم شده و از طرف دیگر امکان ارتباط این عناصر با یکدیگر صرفنظر از زبان برنامه نویسی استفاده شده و سایر موارد مرتبط فراهم گردد. ما می بایست جعبه های سیاهی را طراحی کنیم که صرفنظر از ماهیت درون هریک، قادر به استفاده از توان آنها در بخش یا بخش هائی از یک و یا چندین نرم افزار باشیم. مطلب فوق شاید مهمترین دلیل رویکرد شرکت های عظیم نرم افزاری جهت ارائه یک ساختار استاندارد برای تولید این عناصر باشد .تکنولوژی Component Object Model یا COM پاسخ شرکت مایکروسافت به این نیاز حیاتی بود.

تکنولوژی COM

مهمترین ویژگی تکنولوژی فوق قابلیت استفاده مجدد و ارتباط متقابل برای عناصر(اشیاء) توزیع شده است. بدین ترتیب پیاده کنندگان نرم افزار این امکان را پیدا خواهند کرد تا با در کنار هم قرار دادن این عناصر و استفاده چندباره

از آنان (حتی اگر تولیدکنندگان آنها متفاوت باشند) قادر به خلق آثار ماندگار خود در سریعترین زمان ممکن و متکی بر اصول مهندسی نرم افزار باشند.

ریشه COM

تکنولوژی COM بصورت ناگهانی مطرح نگردید و ریشه در تلاش هائی دارد که از مدت ها قبل بعنوان یک نیاز مطرح شده بود. معرفی تکنولوژی Object Linking & Embedding یا OLE در سال ۱۹۹۱ اولین تلاش در این زمینه بود که توسط شرکت مایکروسافت برای ارتباط و پیوستگی بین مستندات در چهارچوب مجموعه برنامه های آفیس مطرح گردید. حوزه عملکرد تکنولوژی فوق بر روی مستندات (Documents) متمرکز بود. در ادامه شرکت مایکروسافت به این نکته پی برد که تکنولوژی فوق نباید صرفا متمرکز بر روی مستندات باشد و می تواند عملکردی جامع تر را تحت پوشش خود قرار دهد .بدین منظور نسخه شماره ۲، OLEدر سال ۱۹۹۵ مطرح گردید و این نسخه در ادامه تمامی عناصر و اجزای موجود در محیط ویندوز را شامل گردید و بدین ترتیب COM مطرح شد.

در اوایل، تکنولوژی فوق در رابطه با عناصر و اجزای توزیع شده امکانات قابل توجه ای ارائه نکرده بود. شاید یکی از مهمترین دلایل آن عدم عرضه یک سیستم عامل شبکه ای از طرف مایکروسافت تا آن زمان بود. همزمان با عرضه ویندوز ۹۵ و ویندوز NT در سال ۱۹۹۶ و مطرح شدن امکانات شبکه ای و ضرورت توزیع، اجرا و ارتباط بین عناصر توزیع شده تکنولوژی Distributed COM یا DCOM مطرح گردید. سرانجام در سال ۱۹۹۷ نسخه توسعه یافته این تکنولوژی با نام COM+ توسط شرکت مایکروسافت ارائه گردید.

همزمان با گرایش بسمت طراحی و پیاده سازی نرم افزارهای متکی بر مدل Three Tire از یکطرف و نیاز شدید به پیاده سازی نرم افزار های متکی بر وب، ضرورت توجه و بازنگری در نحوه طراحی و پیاده سازی عناصر توزیع شده مورد اهتمام جدی شرکت های عظیم نرم افزاری قرار گرفت. شرکت مایکروسافت در این زمینه منادی تکنولوژی

های+COM/DCOM/COM ، Internet Explorerو ActiveX گردید. در مقابل شرکت های نرم افزاری دیگر، NetScape، Java/J2EE (شرکت سان) و CORBA را مطرح کردند.

اولین نسخه CORBA در سال ۱۹۹۲ توسط Object Management Group یا OMG که بالغ بر ششصد عضو دارد ارائه گردید. آخرین نسخه آن (نسخه شماره ۲) در سال ۱۹۹۶ عرضه شده است. عملکرد کلی تکنولوژی فوق نظیر COM است. بهرحال هدف اکثر تکنولوژی های فوق در این است که امکانات و استانداردهائی را برای تولید عناصر بگونه ای ارائه نمایند که با پیاده سازی آنها، قادر به اخذ سرویس و خدمات بصورت محلی و یا از را دور باشیم.

در این راستا شاید مناسب باشد که به عملکرد هر Tire در نرم افزارها از بعد سرویس دهی متمرکز شده و هر Tireرا بعنوان مجموعه ای از سرویس ها در نظر بگیریم که مسئول ارائه سرویس به عناصر موجود در Tire خود و یا سایر Tire های مرتبط باشد. با این نگرش می توان گفت تمامی نرم افزارها خدمات و سرویس های خود را در سه بخش ارائه می نمایند :

- User Sevices
- Business Services
- Data Services

در مدل Three Tire مسئولیت ارائه هر یک از سرویس های فوق به یک Tire واگذار می گردد. عناصر موجود و مسئول ارائه سرویس و خدمات در هر Tire قادر به ارتباط و درخواست سرویس از عناصر موجود در Tire خود و سایر Tire های موجود در بالا و یا پایین خود خواهند بود. نکته بسیار مهم در رابطه با وضعیت فوق این است که یک درخواست جهت اخذ سرویس نمی تواند یک Tire را حذف و خود مستقیما با Tire ثانویه (بعدی) مرتبط و اصطلاحا یک Tire را دور بزند. مثلا عناصر موجود در لایه User Services نمی توانند مستقیما درخواست خود را برای لایه Data Servicesارسال دارند .البته لایه فوق نیز چنین امکانی را نخواهد داشت. هر یک از سه بخش فوق مسئولیت

های خاصی را برعهده گرفته و در زمانیکه یک بخش به خدمات یک بخش دیگر نیاز داشته باشد، درخواست خود را برای اخذ سرویس در اختیار بخش مورد نظر قرارداده و بخش مربوطه سرویس درخواستی را در قالب اجرای یک یا چندین عنصر انجام و ماحصل را در اختیار بخش مربوطه قرار خواهد داد.

مدل فوق که بر اساس همگرائی نوع سرویس ها و خدمات در یک نرم افزار ارائه شده است، صرفا یک مدل منطقی است و نشاندهنده یک مدل فیزیکی نیست. دراین راستا چهار مدل فیزیکی برای پیاده سازی نرم افزارهای Three Tireارائه شده است:

- Single Server
- Business Server
- Transaction Server
- Web Server

Single Server

در این مدل محل استقرار تمامی عناصر بین سرویس گیرنده و سرویس دهنده شبکه تقسیم می گردد. در مدل فوق تمامی عناصر مربوط به بانک های اطلاعاتی (Data Services) بر روی سرویس دهنده قرار می گیرد. عناصر مربوط به User Service در صورتیکه بگونه ای طراحی شده اند که ممکن است مورد استفاده چندین نرم افزار دیگر قرار بگیرند، می بایست آنها را بر روی سرویس دهنده شبکه نصب نمود .عناصر مربوط به Business Services که مسئولیت پیاده سازی سیاست ها و قوانین در یک نرم افزار را برعهده دارند، عمدتا بر روی سرویس دهنده شبکه نصب می گردنند مگر اینکه در رابطه با یک نرم افزار، اعمال یک سیاست بخصوص را می بایست در سطح لایه User Servicesپیاده سازی نمود (بررسی صحت داده های ورودی، انجام برخی محاسبات خودکار با توجه به رفتار داده ها و). در این حالت عنصر مجری سیاست فوق می بایست در لایه User Services و بصورت محلی و مختص به آن نصب و فعال گردد.

Bussines Server (Application)

در مدل فوق یک سرویس دهنده اضافی با نامApplication Server ، استفاده می گردد. سرویس دهنده فوق مسئولیت استقرار تمامی عناصری را که می بایست به اشتراک گذاشته شوند، بر عهده خواهد گرفت. در این راستا در صورتیکه برخی از عناصر مربوط به لایه User Service باشند ولی بصورت مشترک مورد استفاده چندین نرم افزار قرار می گیرند نیز از این قاعده مستثنی نبوده و بهترین محل برای استقرار آنان، سرویس دهنده Application است. در مدل فوق تمامی عناصر مربوط به Data service بر روی سرویس دهنده Data قرار خواهند گرفت. ارتباط تمامی سرویس گیرندگان در ابتدا با Application Server آغاز خواهد گشت. سرویس گیرندگان خواسته خود را به لایه Applicationارسال و لایه فوق مسئولیت ارتباط با لایه Data را بر عهده خواهد گرفت.

Transaction Server

Transaction واحد انجام یک فعالیت بوده که خود می تواند شامل چندین عملیات دیگر باشد. سلسله عملیات فوق می بایست تماما با موفقیت اجرا گردند. در مدل فوق سرویس دهنده Transaction مسئولیت مدیریت و ذخیره سازی عناصر لازم برای یک فعالیت Transaction را برعهده خواهد گرفت. در این مدل می توان از چندین سرویس دهنده دیگر بمنظور استقرار عناصر مربوطه استفاده کرد. استقرار عناصر بر روی سرویس دهنده ها می بایست پویا بوده و در صورت افزایش ترافیک، امکان جابجائی آنها بر روی سایر سرویس دهنده ها وجود داشته باشد. سرویس دهنده Transaction مسئولیت های نگهداری عناصرActiveX ، ارسال درخواست یک برنامه به یکی از سرویس دهنده ها، اتمام اجرای یک برنامه، بررسی صحت عملکرد یک عنصر را بر عهده خواهد گرفت.

Web Server

در مدل فوق یک سرویس دهنده در شبکه اضافه و مسئولیت سرویس های وب را بر عهده خواهد گرفت. سرویس گیرنده ها مجهز به نرم افزارهای ارتباطی نظیر مرورگرها بوده تا بدین طریق قادر به درخواست صفحات ایستا و

پویا از سرویس دهنده وب باشند. برنامه های مبتنی بر وب تمامی تاکید خود را بر استاندارد نمودن نرم افزارهای مرورگری معطوف می دارند. چراکه با استاندارد شدن این نوع از نرم افزارها تمامی سرویس گیرنده ها با یک ابزار واحد استاندارد شده از سرویس دهنده های وب خواسته های خود را مطرح خواهند نمود. بدیهی است در چنین حالتی پاسخگوئی به این درخواست ها از طرف سرویس دهنده های وب بمراتب ساده تر و با اطمینان خاطر بیشتری صورت می پذیرد.

در سه مدل گفته شده قبلی، بر این نکته تاکید وجود داشت که تمامی سیاست های راهبردی نرم افزار متمرکز شده تا بدین طریق اعمال تغییرات بسادگی صورت پذیرد. در مدل فوق چون سرویس دهنده وب این پتانسیل را دارا است که بصورت اتوماتیک عناصری را بر روی کامپیوتر سرویس گیرنده مستقر نماید، ضرورت تمرکز سیاست های راهبردی نرم افزار بر روی سرویس دهنده چندان مهم بنظر نمی آید. در این مدل بیشتر سرعت اجرای این عناصر مورد توجه است. در چنین حالتی اگر یک برنامه مبتنی بر وب بر روی بستر اینترنت اجرا می گردد، می توان برخی از عناصر را برای سرویس گیرنده ارسال تا بصورت محلی بر روی کامپیوتر وی اجرا شوند. در چنین حالتی سعی می شود که زمان ارتباط و درخواست از سرویس دهنده به حداقل زمان ممکن کاهش پیدا کند چراکه پهنای باند و ارتباط با سرویس گیرنده ها را نمی توان همواره بدون نگرانی تضمین نمود. در صورتیکه برنامه مبتنی بر وب بر روی بستر اینترانت اجرا می گردد، می توان بر اساس توان کامپیوترهای سرویس گیرنده و سرویس دهنده و پهنای باند موجود، برخی از عناصر را بر روی سرویس دهنده و برخی دیگر از عناصر را بر روی سرویس گیرنده اجرا نمود.

بهرحال مدل فوق یک ایده جدید برای اجرای یک برنامه را مطرح کرده است .سرویس دهنده وب بسادگی و بصورت اتوماتیک قادر به نصب اجزای مورد نیاز یک سرویس گیرنده خواهد بود. بدین ترتیب از یکطرف ضرورت استقرار تمامی عناصر بر روی سرویس گیرنده از بین رفته و از طرف دیگر به سرویس گیرنده ها استقلال لازم داده شده و در صورت ضرورت می توان آنها را نیز در اجرای برخی از عناصر سهیم کرد.

جایگاه ASP.NET در مقایسه صفحات ایستا و پویا[1]

بررسی مفهومی صفحات وب ایستا و پویا و بررسی روش های تزریق این پویائی از جایگاه سرویس دهنده و سرویس گیرنده بهمراه بررسی فناوریهای متداول در این خصوص با اشاره به ASP.NET

ASP.NET، یک تکنولوژی جدید و قوی برای نوشتن صفحات وب پویا است .تکنولوژی فوق با دو تکنولوژی اصلی دیگر مایکروسافت یعنی ASP و دات نت همگرا است ASP .دارای عمری شش ساله بوده که تاکنون توانسته است در زمینه ساخت صفحات وب پویا موفق عمل نماید. دات نت یک فناوری جدید ارائه شده توسط مایکروسافت با هدف ایجاد تغییرات بنیادین در رابطه با طراحی و پیاده سازی برنامه های کامپیوتری است.

برای شروع کار و استفاده از ASP.NET نیازی به آگاهی و دانش لازم در مورد ASP نبوده و با آشنائی مختصر با HTML می توان از تکنولوژی فوق برای ایجاد صفحات وب پویا استفاده کرد. این فناوری بمراتب قدرتمندتر از هم نام قدیم خود یعنی ASP است ASP.NET .دارای مجموعه ای از کنترل های تعریف شده است که با استفاده از آنها، هم در زمان صرفه جوئی خواهد شد و هم بهره وری افزایش خواهد یافت.

در این مقاله سعی می شود که با مفاهیم صفحات وب ایستا(Static) ، صفحات وب پویا(Dynamic) ، تکنولوژیهای رایج جهت ایجاد صفحات وب پویا وASP.NET ، تشریح خواهند شد.

صفحات وب ایستا

ما امروز بر روی اینترنت و وب سایت های متعدد، با موارد بیشماری از صفحات وب ایستا برخورد می کنیم. واژه "ایستا" در رابطه با یک صفحه وب چگونه تعریف می گردد؟ این نوع صفحات، صفحاتی هستند که شامل کدهای Htmlبوده و در یک محیط ادیتور تایپ و فایل مربوطه با انشعاب Htm و یا Html ذخیره می گردد. مولف صفحه

1 محمد جواد سخایی

وب قبل از اینکه هر نوع درخواستی برای آن وجود داشته باشد، بطور کامل محتوی صفحه را مشخص کرده است. محتویات این نوع از صفحات (متن، تصویر، لینک ها و) و شکل ظاهری آنها همواره یکسان خواهد بود، صرفنظر از اینکه چه کسی ، در چه زمانی و یا چگونه صفحه را مشاهده خواهد کرد. بنابراین می توان گفت محتویات این قبیل از صفحات قبل از اینکه درخواستی ایجاد گردد، توسط مدیریت سایت ایجاد و مشخص شده اند.

مراحل آماده سازی صفحات وب ایستا

۱ -یک مولف صفحات وب، یک صفحه که شامل کدهای Html است را ایجاد و آن را با انشعاب Htm و یا Html بر روی سرویس دهنده وب و در مسیر مربوطه ذخیره می کند.

۲ -کاربری از طریق برنامه مرورگر خود، در خواست استفاده از یک صفحه را می نماید. درخواست فوق از مرورگر برای سرویس دهنده ارسال می گردد.

۳ -سرویس دهنده وب فایل درخواستی با انشعاب Htm را پیدا خواهد کرد.

۴ -سرویس دهنده وب کدهای Html فایل مزبور را از طریق شبکه برای مرورگر ارسال میدارد.

۵ -مرورگر کدهای Html را پردازش و صفحه فوق را نمایش خواهد داد.

محدودیت های صفحات وب ایستا

فرض کنید می خواهیم یک صفحه وب را بگونه ای طراحی کنیم، که بمحض ورود هر کاربر زمان جاری سیستم بهراه یک پیام مناسب نمایش داده شود. در این زمینه با چندین محدودیتی مواجه خواهیم بود که صرفا و به تنهائی بکمک کدهای Html قادر به برطرف کردن محدودیت های فوق نخواهیم بود (اگر با این عقیده موافق نیستید، دست بکار شوید و یک صفحه وب را که صرفا شامل کدهای Html است، تهیه نمائید که زمان را به وقت محلی بر روی سرویس دهنده نمایش دهد!).

ما می دانیم که یک کاربر در یک زمان خاص به ملاقات صفحه خواهد آمد ولی قطعا زمان آن را نمی دانیم. اگر بخواهیم زمان را بصورت کد در صفحه Html خود داشته باشیم، نتیجه همواره یکسان بوده و همیشه یک زمان ثابت و یکسان برای تمامی ملاقات کنندگان صفحه نمایش داده خواهد شد. کدهای Html به تنهائی هیچگونه امکاناتی برای شخصی سازی صفحات وب در اختیار قرار نمی دهند. نمایش هر صفحه برای هر کاربر یکسان خواهد بود (نظیر رستورانی که همواره و صرفنظر از ذائقه مشتریان خود، یک غذای ثابت و از قبل آماده شده را برای همه آماده و در اختیار قرار می دهد!).

Html دارای هیچگونه امنیتی نیز نبوده و کدهای آن را همه می توانند مشاهده و حتی تکثیر نمایند. شاید تنها مزیت این نوع از صفحات طراحی آسان، سرعت تکثیر و توزیع آنها در یک شبکه باشد. بدیهی است این نوع صفحات دارای هیچگونه امکانات لازم جهت آفرینش صفحات پویا نیستند. چون ما نمی توانیم کدهای Html اضافه ای را بعد از درخواست یک صفحه به آن اضافه نمائیم. پس می بایست بدنبال روشی بود که بتوان کدهای Html را بعد از درخواست یک صفحه وب، ایجاد نمود. بمنظور نیل به هدف فوق از دو روش عمده استفاده می گردد:

- **صفحات پویای نزد سرویس گیرنده: بهره گیری از تکنولوژیهائی که پویائی یک صفحه را از خاستگاه سرویس گیرنده تحقق خواهند داد.(Client Side Dynamic Web Page)**

- **صفحات پویای نزد سرویس دهنده: بهره گیری از تکنولوژیهائی که پویائی یک صفحه را از جایگاه سرویس دهنده تحقق نمایند(Server Side Dynamic Web Page) .**

قبل از پرداختن به هر یک از موارد فوق لازم است در ابتدا با مفهوم و جایگاه یک سرویس دهنده وب بیشتر آشنا شویم. یک سرویس دهنده وب، نرم افزاری است که مدیریت صفحات وب را برعهده گرفته و آنها را برای سرویس گیرندگان مجهز به مرورگرها قابل دستیابی و استفاده می نماید. تاکنون سرویس دهنده های وب متعددی طراحی و به بازار عرضه شده استApache ،. IIS و نمونه هائی از این نوع نرم افزارها هستند. یکی از سرویس دهندگان وبی که

ASP.NETبر روی آن اجرا می گردد، IISمتعلق به شرکت مایکروسافت است IIS .را می توان در زمان نصب ویندوز ۲۰۰۰ و یا XP نیز بر روی سیستم نصب نمود IIS 5.0 .بهمراه ویندوز ۲۰۰۰ و IIS 5.1 بهمراه ویندوز XP آ نصب خواهند شد .بهرحال جایگاه یک سرویس دهنده وب در ارائه امکانات و زیرساخت های مناسب برای طراحی صفحات وب پویا و بالطبع سایت های پویا یک امر برجسته است.

همانگونه که گفته شد برای خلق صفحات وب پویا از دو رویکرد متفاوت استفاده می گردد. استفاده همزمان از دو روش فوق هیچگونه تعارضی با هم نداشته بلکه بالعکس توانائی یک صفحه وب پویا را افزایش خواهد داد. در ادامه به بررسی دو رویکرد فوق خواهیم پرداخت.

صفحات پویای نزد سرویس دهنده

در مدل فوق ماژول هائی (Plugin) که به مرورگر ملحق شده اند تمامی عملیات لازم جهت ایجاد صفحات پویا را انجام خواهند داد. کدهای Html از طریق یک فایل مجزا که شامل مجموعه ای از دستورالعمل ها است برای مرورگر ارسال خواهد شد.مرورگرها دستورات فوق را جهت تولید کدهای Html و در زمان درخواست یک صفحه توسط کاربر، استفاده خواهد کرد. بنابراین محتویات یک صفحه بر اساس درخواست کاربران و بصورت پویا ایجاد خواهد شد.

مراحل آماده شدن یک صفحه وب پویا با تاکید بر روش های: Client-Side

۱ -یک مولف صفحه وب مجموعه ای از دستورالعمل را برای ایجاد کدهای Html نوشته و آنها را در فایلی با انشعاب Htm ذخیره می نماید.

۲ -کاربران درخواست یک صفحه را از طریق مرورگر خود برای سرویس دهنده وب ارسال خواهند کرد.

۳ -سرویس دهنده فایل درخواستی (در صورت نیاز فایل دیگری که شامل دستورالعمل ها باشد) را

پیدا خواهد کرد.

۴ -سرویس دهنده وب فایل حاوی کدهای Html و در صورت وجود دستورالعمل های مربوطه را برای متقاضی ارسال خواهد کرد.

۵ -یک ماژول همراه مرورگر، دستورالعمل ها را پردازش و کدهای Html را در همان صفحه Html برمی گرداند .

۶ -در نهایت کدهای Html توسط مرورگر نمایش داده می شوند.

تاکنون تکنولوژیهای متعددی بر اساس رویکرد فوق به موازات هم بوجود آمده و در اختیار طراحان و مؤلفان صفحات وب پویا قرار گرفته شده است. جاوا اسکریپت،VBScript ، کنترل های ActiveX و اپلت های جاوا نمونه هائی از این نوع تکنولوژیها بوده که برای شناخت خوانندگان در این بخش بصورت خیلی مختصر در رابطه با هر یک توضیحاتی ارائه خواهد شد.

جاوا اسکریپت(JavaScript)

اولین زبان اسکریپت در رابطه با مرورگرها است. زبانهای اسکریپت بعنوان حد میانه بین کدهای Html و زبانهای معمولی برنامه نویسی قرار دارند و بصورت مفسر عمل می نمایند. جاوااسکریپت را نباید با زبان برنامه نویسی جاوا اشتباه گرفت. شرکت نت اسکیپ در ابتدا زبان اسکریپتی با نام LiveScript پیاده سازی نمود. و بهمراه مرورگر NetScape 2.0در اختیار علاقه مندان قرار گرفت. زمانیکه شرکت نت اسکیپ با شرکت Sun متحد گردید، نام آن را جاوااسکریپت گذاشتند.

بخشی از گرامر زبان فوق نظیر ساختار اولیه، از جاوا گرفته شده است(خود جاوا نیز اغلب ساختار خود را از زبان C گرفته است). جاوااسکریپت دارای امکانات متعدد و قدرتمندی جهت کنترل و مدیریت رفتار و محتویات یک مرورگر

است. اما زبان فوق توانائی انجام عملیاتی نظیر: عملیات روی فایل ها را دارا نمی باشد (شاید یکی از دلایل آن مسائل امنیتی باشد). فراگیری جاوااسکریپت نسبت به جاوا بمراتب راحت تر است. جاوا اسکریپت بگونه ای طراحی شده است که قادر به خلق برنامه های کوچک و در عین حال موثر جهت انجام عملیات متعددی نظیر برخورد با رویدادهای بوجود آمده در سطح کاربر نظیر: کلیک نمودن بر روی یک آیتم، بستن یک پنجره، فعال شده یک صفحه، خارج شدن از یک صفحه، حرکت موس روی یک آیتم و است. مایکروسافت نسخه اختصاصی خود از جاوااسکریپت را با نام Jscript و همزمان با معرفی مرورگر IE 3.0 در اختیار علاقه مندان قرار داد.

VBScript

شرکت مایکروسافت همزمان با عرضه مرورگر IE 3.0 زبان اسکریپت اختصاصی خود یعنی VBScript را مطرح نمود. زبان اسکریپت فوق بر اساس زبان برنامه نویسی ویژوال بیسیک و با هدف رقابت با جاوااسکریپت در اختیار علاقه مندان قرار گرفت. شاید از معدود امتیازات این زبان نسبت به جاواسکریپت بتوان به عدم حساسیت آن در رابطه با حروف بزرگ و کوچک (Case Sensitive) نام برد. کدهای نوشته شده توسط زبان فوق صرفا از طریق مرورگر شرکت مایکروسافت (IE) قابل تفسیر و اجرا بوده و نت اسکیپ این زبان را حمایت نمی کند. گرچه با افزودن برخی Plug-In امکان استفاده از این زبان در مرورگر نت اسکیپ نیز فراهم خواهد شد. مطمئا استفاده از زبان جاوااسکریپت بمراتب نسبت به زبان VBScript رایج تر است.

اگر قصد انتخاب یک زبان اسکریپت برای پردازش های متکی بر سرویس گیرنده را داشته باشیم، بدون شک جاوااسکریپت یک گزینه مناسب خواهد بود. جاواسکریپت و VBScript هر دو بعنوان یک ماژول با نام Script Engineکه بهمراه مرورگرها ارائه شده است، مسئولیت تفسیر و اجرای دستورالعمل های مربوطه را بر عهده خواهند گرفت. در پروژه دات نت، شرکت مایکروسافت VB.NET را جایگزین VBScript نموده است.

کنترل هایActiveX

یک کنترل اکتیوایکس عنصری است که توسط یکی از زبانهای برنامه نویسی نظیر C++ و یا جاوا پیاده سازی می گردند. در زمانیکه این نوع اکتیوایکس ها را با صفحات خود استفاده نمائیم، امکان انجام بخشی از عملیات متکی بر سرویس گیرنده نظیر ایجاد یک Timer، Bar Chart، تایید کاربر و یا دستیابی به بانک اطلاعاتی فراهم می گردد. کنترل های اکتیوایکس از طریق تگ <Object> به صفحات وب اضافه می شوند.

منادی تکنولوژی فوق شرکت مایکروسافت بوده و تا نسخه شش مرورگر نت اسکیپ امکان استفاده از آنها توسط مرورگر فوق وجود ندارد. البته با نصب برخی Plug-in زمینه استفاده از کنترل های اکتیوایکس در مرورگر نت اسکیپ بگونه ای فراهم شده است. نکته جالب توجه در رابطه با تکنولوژی فوق این است که امکان انجام عملیات متفاوت بر روی کامپیوترهای کاربران نظیر کار با فایل ها و ریجستری ویندوز بوجود می آید و این خود می تواند از لحاظ امنیتی مشکل و گاها با توجه به وجود فایروال ها تحقق ناپذیر باشد. بهرحال نمی توان بر روی تکنولوژی فوق بعنوان یک راه حل جامع و فراگیر برای خلق صفحات وب پویا استفاده کرد، مگر اینکه مخاطبان سایت خود را صرفا از بین کسانی انتخاب نمائیم که ویندوز را بعنوان سیستم عامل و مرورگر IE را بعنوان مرورگر خود برگزیده اند.

Java Applet

جاوا یک زبان برنامه نویسی مستقل از Platform است. جاوا نسبت به زبانهای اسکریپت دارای قابلیت های بمراتب بیشتری است. هدف، استفاده از کدهای جاوا به شکل اپلت است. عناصر فوق بسادگی و توسط تگ <Applet> به صفحات وب ملحق خواهند شد. خوشبختانه مرورگرهای مایکروسافت و نت اسکیپ هر دو از طریق ایجاد یک ماشین مجازی جاوا (JVM) از اپلت های جاوا حمایت می کنند. بمنظور استفاده از اپلت های جاوا در یک صفحه وب از چندین روش می توان استفاده کرد: استفاده از تگ استاندارد <Object> یا استفاده از تگ غیراستاندارد .<Applet> تگ های فوق به مرورگر خواهند گفت که یک فایل جاوا را از طریق سرویس دهنده وب DownLoad و سپس بکمک ماشین مجازی جاوا(JVM) ، موجود در مرورگر، آن را اجرا نماید.

همانطور که حدس زده اید یکی از مسائل موجود در رابطه با استفاده از اپلت ها جاوا، زمان اضافه ای است که صرف Download کردن می گردد. بنابراین در زمان استفاده از اپلت های جاوا سعی در نوشتن اپلت ها با کد کم باشیم. از رایج ترین موارد کاربرد اپلت های جاوا می توان به ایجاد Drop-Down Menu و انیمیشن های متفاوت اشاره کرد.

صفحات پویای نزد سرویس دهنده

در این مدل کدهای Html بهمراه مجموعه ای از دستورالعمل ها برای سرویس دهنده ارسال و مجددا از دستورالعمل های فوق برای تولید کدهای Html برای صفحه ای که کاربر درخواست کرده، استفاده شده و در نهایت صفحه بصورت پویا بر اساس درخواست کاربر ایجاد خواهد شد.

مراحل آماده شدن یک صفحه وب پویا با تاکید بر روش های Server-Side :

۱ -یک مولف صفحه وب، مجموعه ای از دستورالعمل ها را برای ایجاد کدهای Html نوشته و دستورالعمل ها را در یک فایل ذخیره می کند.

۲ -کاربران از طریق مرورگر خود، درخواست یک صفحه وب را نموده و این درخواست برای سرویس دهنده وب ارسال خواهد شد.

۳ -سرویس دهنده وب محل فایل حاوی دستورالعمل را پیدا خواهد کرد.

۴ -سرویس دهنده وب دستورات موجود در فایل را بمنظور تولید کد Html اجرا خواهد کرد.

۵ -سرویس دهنده وب کدهای تولید شده جدید را از طریق شبکه برای مرورگر ارسال می نماید .

۶ -مرورگر کدهای Html را پردازش و در نهایت صفحه وب نمایش داده خواهد شد.

نکته مهم در سناریوی فوق، اجرای تمامی پردازش ها بر روی سرویس دهنده قبل از ارسال صفحه برای مرورگر است. یکی از مزایای عمده مدل فوق نسبت به مدلClient-Side ، این مورد است که : در یک صفحه وب صرفا شاهد کدهای Html خواهیم بود .این بدان معنی است که منطق صفحات وب در نزد سرویس دهنده وب مخفی نگهداری خواهد شد و می توان این اطمینان را داشت که اکثر مرورگرها قادر به نمایش نتایج پردازش های اجرا شده بر روی سرویس دهنده باشند ASP .و ASP.NET از مدل فوق تبعیت می کنند.

یکی دیگر از نکات مهم در رابطه با مدل فوق در این است که یک صفحه تا زمانیکه درخواستی برای آن دریافت نشده باشد، محتویات آن بوجود نخواهد آمد. در ادامه به بررسی برخی تکنولوژیهای موجود دراین مدل خواهیم پرداخت.

Common Gateway InterfaceیاCGI

مکانیزمی برای ایجاد اسکریپت بر روی سرویس دهنده بوده تا بدین طریق امکان ایجاد برنامه های متکی بر وب فراهم گردد CGI .ماژولی است که می بایست به سرویس دهنده وب اضافه گردد. قدمت استفاده از تکنولوژی فوق بمراتب بیشتر از ASP است و تا کنون تعداد بیشماری از صفحات وب پویا با استفاده از تکنولوژی فوق و بکمک یک زبان اسکریپت ایجاد شده اند CGI .این امکان را فراهم خواهد کرد که کاربر، یک برنامه دیگر (نظیر یک اسکریپت Perl) را بر روی سرویس دهنده برای ایجاد صفحات وب پویا استفاده نماید. استفاده از زبانهائی نظیرC ، Perl و C++ بهمراه تکنولوژی فوق بسیار رایج است. بهرحال تکنولوژی فوق امروزه در خیلی از سایت های بزرگ خصوصا سایت های متکی بر یونیکیس رایج بوده و قابلیت اجرا بر روی چندین Platform را دارا می باشد.

Active Server PageیاASP

تکنولوژی فوق که پس از عرضه ASP.NET با نام ASP کلاسیک نامیده می شود یکی از متداولترین روش های موجود و استفاده شده برای ایجاد صفحات وب پویا است. تکنولوژی فوق با بهره گیری از توان زبانهای اسکریپت نظیر جاوااسکریپت و VBScript توانسته است پاسخی شایسته به طراحی صفحات وب پویا را ارائه نماید .ASP یک ماژول جداگانه است که در کنار سرویس دهنده وب قرار می گیرد .(ASP.dll)تکنولوژی فوق نسبت به برخی از تکنولوژیهای همگروه از کارآئی پایین تری برخوردار بوده و در زمینه استفاده از زبانهای اسکریپت در کنار خود، نیز دارای محدودیت جدی است. بهرحال تکنولوژی فوق با سابقه شش ساله تاکنون توانسته است به خیل عظیم درخواست ها برای ایجاد صفحات پویا درست پاسخ دهد. ولی با ظهور خواسته ها و انتظارات جدید به چالش جدی کشیده شده است و شاید ظهور و تولد ASP.NET دلیلی و پاسخی به برخی از انتقادات مطروحه در این زمینه باشد.

JSP یا Java Server Page

تکنولوژی فوق امکان ترکیب Html و یا Xml را با کدهای جاوا فراهم می نماید. این فناوری برخلاف ASP که صرفا توسط سرویس دهنده وب مایکروسافت (IIS) حمایت می گردد، توسط سرویس دهندگان متعددی حمایت شده است JSP .در مقایسه با ASP بمراتب دارای قدرت و سرعت بیشتری بوده و برنامه نویسان جاوا بخوبی با قابلیت های متعدد آن آشنائی دارند JSP .این امکان را فراهم می کند که برنامه های جاوا از ویژگی محیط های متکی بر Java2 نظیر JavaBeans و Java2 Libraries بخوبی استفاده نمایند.

ColdFusion

با استفاده از تکنولوژی فوق، امکان ساخت صفحات وب پویا فراهم می گردد. این تکنولوژی بصورت یک ماژول جداگانه است که می بایست بر روی سرویس دهنده وب نصب گردد. صفحاتی که توسط تکنولوژی فوق بوجود می آیند توسط هر نوع مرورگری قابل خواندن و نمایش خواهند بود. تکنولوژی فوق از مجموعه زیادی تگ که توسط نرم افزار ColdFusion ارائه شده است، استفاده می کند. نرم افزار فوق بر روی سرویس دهندگان متعددی حتی IIS نصب و

قابل استفاده است. مهمترین مسئله در رابطه با تکنولوژی فوق در این است که از تگ های Html-Like استفاده می گردد (در ASP.NET از زبانهای برنامه نویسی و اشیا استفاده می گردد). یکی دیگر از نکات مهم در رابطه با تکنولوژی فوق در این است که تهیه آن رایگان نبوده و می بایست بیش از هزار دلار برای تهیه آن هزینه نمود!

PHP

تکنولوژی فوق که در ابتدا Personal Home Page نامیده می شد و اخیرا PHP Hypertext Preprocessorنامیده می شود، یکی دیگر از تکنولوژیهای رایج برای ایجاد صفحات وب پویا است. تکنولوژی فوق بر خلاف ASP.NET بصورت Cross-Platform بوده و بر روی اغلب سیستم ها نظیر ویندوز NT و اغلب نسخه های یونیکس قابل استفاده است. گرامر زبان فوق نظیر C و Perl است. تکنولوژی فوق دارای برخی از ویژگی های برنامه نویسی شئ گرا بوده که امکان سازماندهی و کپسوله نمودن کدها را فراهم می آورد.

ASP.NET

پس از معرفی تکنولوژیهای رایج در این گروه، زمینه مناسب برای آشنائی با تکنولوژی ASP.NET بوجود آمده است ASP.NET .نیز بعنوان یک ماژول بر روی سرویس دهنده قرار می گیرد .(aspnetIsapi.dll) در کنار تکنولوژی فوق مجموعه عظیم دات نت قرار دارد که در صورت لزوم می توان برخی از عملیات و خواسته های خود را برای پردازش به دات نت ارسال کرد. در ابتدای این مقاله، یک تعریف از ASP.NET ارائه شد (یک تکنولوژی جدید و قوی برای ساخت صفحات وب پویا) در این مرحله می توان تعریف فوق را بصورت زیر اصلاح نمود:

ASP.NETیک تکنولوژی جدید و قوی از مجموعه تکنولوژیهای Server Side برای ایجاد صفحات وب پویا است.

ASPکلاسیک دررابطه با استفاده از زبانهای اسکریپت محدود بوده و صرفا به جاوااسکریپت و VBScript ختم می گردد (VBScript هم صرفا توسط سیستم های متکی بر ویندوز قابل استفا ده خواهد بود). ASP.NET .امکان استفاده از مجموعه وسیعی از زبانهای برنامه نویسی را فراهم می کند. زبانهائی نظیر VB.NET ، C#، Jscript.NET ، Perl، Pythonو نمونه هائی از زبانهائی می باشند که می توان از آنها بهمراه ASP.NET استفاده کرد.

Caching در ASP.NET[1]

آشنائی با مفهوم Caching و نحوه استفاده و بکارگیری آن در صفحات ASP.NET

۱- Caching چیست؟ چگونگی و ضعف آن در ASPکلاسیک

عمل Caching در صفحات ASP به منظور کاهش بار و میزبان و افزایش سرعت دریافت صفحات وب می باشد. بدین ترتیب که با توجه به دستورالعمل گفته شده، یک نسخه از صفحه درخواستی در میزبان ایجاد می شود و با توجه به شرایط Caching پس از آن اگر درخواستی برای آن صفحه صورت گرفت، به جای اجرا کردن فایل ASP مربوطه یک کپی از نسخه ایجاد شده در میزبان به مشتری (Client) فرستاده می شود (همانند عملیات ارسال فایل های HTML به طرف مشتری).

اما این عمل در ASP کلاسیک اگرچه در مواردی مفید بود اما هیچگاه به عنوان یک ابزار کاملا سودمند به آن پرداخته نشد، زیرا فقط با تعیین یک خصوصیت که همان زمان مورد نظر برای Caching بود، دستورالعمل پایان می یافت و هیچگاه نمی توانستیم شرایطی را اعمال کنیم که فقط در آن صورت Caching رخ دهد. همچنین قادر نبودیم که عملیات Caching را برای عناصر خاصی از صفحه تعریف کنیم. بدین ترتیب کارایی این روش در ASP کلاسیک چندان بالا نبود. اما در ASP.NET ما شاهد یک تحول شگرف در عملیات Caching هستیم بطوریکه ضعف هایی را که در ASP کلاسیک شاهد آن بودیم در ASP.NET آنها را برطرف شده می یابیم.

۲ -ساختار Caching در ASP.NET چگونه است؟

روش مورد استفاده در ASP.NET با ASP کلاسیک تفاوت دارد. شکل کلی آن به صورت زیر است :

```
<%@ OutputCache Duration = "..." VaryByParam = "..." %>
```

[1] محمود مروج

همانطور که می دانید در ASP.NET ما قادر خواهیم بود چندین هدایت کننده ای (<%@ ... %>) را در یک صفحه قرار دهیم .خصوصیات این هدایت کننده همانطور که در بالا مشخص شده است Duration و VaryByParamمی باشد که در ادامه مقاله به شرح آن خواهیم پرداخت.

الف- خصوصیت Duration :

این خصیصه مدت زمان مورد نیاز به Caching را به ثانیه تعیین می کند. به مثال زیر توجه کنید. فرض کنید هدایت کننده Caching مورد نظر به صورت زیر باشد :

```
<%@ OutputCache Duration = "15" VaryByParam = "none" %>
```

و صفحه ASP.NET ما شامل هدایت کننده فوق و یک Label و Button باشد .به صورتیکه با فشردن دکمه مورد نظر زمان جاری میزبان در Label مذبور به نمایش درآید. پس از اینکه صفحه برای اولین بار نمایش داده شد (فرض کنید زمان ۲۱:۳۰:۴۵ را نمایش دهد) پس از آن اقدام به فشردن دکمه موردنظر نمایید. خواهید دید که زمان نمایش داده شده تغییر نخواهد کرد. این عمل را تا ۱۵ ثانیه ادامه دهید. پس از آن خواهید دید که زمان نمایش داده شده ۲۱:۳۱:۰۰ خواهد بود. البته مهم این است که پس از Download شدن صفحه ۱۵ ثانیه بگذرد و پس از آن هر درخواستی که انجام گیرد به صورت معمول اجرا خواهد شد و بلافاصله پس از اجرا یک دوره جدید Caching به مدت ۱۵ ثانیه ایجاد خواهد شد. مثلا فرض کنید پس از ۳۰ ثانیه که از نمایش ۲۱:۳۰:۴۵ گذشت شما دکمه را فشار دهید. در آن صورت زمان ۲۱:۳۱:۱۵ به نمایش در خواهد آمد و پس از آن یک Caching جدید به مدت ۱۵ ثانیه برای صفحه ایجاد خواهد شد.

ب- خصوصیت VaryByParam :

این خصیصه که از ویژگی های جالب Caching در ASP.NET است، معین کننده این است که استفاده از Caching در چه صورت انجام نشود یا به عبارت دیگر اگر پارامترهایی که به صفحه ASP.NET مورد نظر ارسال می شوند همنام با موارد معین شده در خصوصیت VaryByParam بود، دیگر از Caching قبلی استفاده نکند و صفحه جدید را با توجه به مقادیر جدید دوباره اجرا و کد HTML ایجاد شده را به مشتری مورد نظر بفرستد. بعنوان مثال فرض کنید هدایت کننده Caching به صورت زیر باشد :

```
<%@ OutputCache Duration = "120" VaryByParam = "name" %>
```

همچنین در نظر بگیرید که فرم وب مورد نظر دارای دو Label باشد. یکی نشاندهنده زمان جاری میزبان و دیگری نشاندهنده مقدار پارامتر name که به این صفحه ارسال می شود. فرض کنید مقدار Label زمان در هنگامی که صفحه برای اولین بار درخواست می شود دارای مقدار ۲۱:۲۲:۱۰ باشد. حال فرض کنید مدت زمان Caching به پایان نرسیده باشد و ما مجددا تقاضای صفحه مذبور را داشته باشیم. اگر همانند قسمت اول بود، یعنی مقدار خصیصه VaryByParamبا none تنظیم شده بود، علی الاصول باید از صفحه Cache شده استفاده کند. ولی در اینجا شرط دیگری داریم که در واقع می گوید اگر صفحه با مقدار جدیدی از پارامتر name درخواست شد، باید صفحه را اجرا و از Caching صرف نظر کند .مثلا فرض کنید پس از ۳۰ ثانیه صفحه مورد نظر به صورت زیر درخواست شود :

```
http://www.iranasp.net/myexample.aspx?name=mahmoud
```

چون این متغیر ارسالی قبلا مقداری نداشته است بنابراین صفحه ASP.NET مورد نظر دوباره اجرا شده و پارامتر ورودی name را نیز مد نظر می گیرد. بنابراین Label ای که زمان را نشان می دهد به ۲۱:۲۲:۴۰ و Label ای که مقدار name را نشان می دهد دارای مقدار mahmoud می شود. حال اگر صفحه موردنظر را با همین مقادیر پس از ۱۵ ثانیه دوباره درخواست کنیم خواهیم دید که همان اطلاعات قبلی را نشان می دهد. توجه کنید در این حالت

زمان همان مقدار قبلی ۲۱:۲۲:۴۰ را خواهد داشت. بنابراین تغییر مقدار پارامتر مورد نظر در Caching موثر است. حال فرض کنیم پس از ۵ ثانیه که زمان سیستم میزبان ۲۱:۲۳:۰۰ خواهد بود صفحه را به صورت زیر درخواست کنیم :

http://www.iranasp.net/myexample.aspx?name=mahmoud&student_no=80622121

خواهید دید که هیچگونه تغییری مشاهده نخواهد شد. بنابراین متغیرهایی که در VaryByParam مشخص نشده باشند در اجرای دوباره صفحه ASP.NET تاثیری ندارد.

توجه کنید دو خصوصیت ذکر شده با هم OR می شوند. یعنی هر کدام که شرایط را برقرار کنند باعث انجام عمل مربوطه خواهند شد و برقراری یا عدم برقراری خصوصیت دیگر در اجرای عمل تاثیری ندارد. پس اگر بعد از گذشت ۱۲۰ ثانیه هرگونه تقاضایی شود بدون توجه به پارامترهای ارسالی، صفحه درخواستی مجددا اجرا شده و نتیجه اجرای آن متعاقبا ارسال می گردد. و بالعکس آن یعنی زمانی که مدت زمان نگهداری Caching به پایان نرسیده باشد و موارد ذکر شده در خصوصیت VaryByParam شرط را برقرار کنند از Caching صرف نظر می کند (مانند مثال قبلی).

دقت کنید که شما می توانید صفحه ASP.NET خود را به تمام پارامترها حساس کنید. یعنی به جای ذکر تک تک موارد با قرار دادن * برای خصیصهVaryByParam ، باعث شوید که در صورت تغییر هر کدام از پارامترهای ارسالی، صفحه دوباره اجرا گردد .با این حال توجه کنید که اگر در این حالت هیچ یک از پارامترهای ارسالی تغییر نکند، شرط مورد نظر برقرار نمی شود. حالت مقابل آن none است که در مثال اول مورد استفاده آن نشان داده شده است.

۳ Caching -بخشی از صفحه

یکی دیگر از موارد جالب در Caching صفحات وب در ASP.NET این است که شما قادر خواهید بود قسمت خاصی از صفحه را تحت تاثیر Caching قرار دهید. در واقع تکنیک این کار استفاده از UserControl ها می باشد (فایل های که پسوند ascx دارند). بدین ترتیب که شما یک هدایت کننده Caching را در صفحه UserContol خود قرار می دهید. مثلا فرض کنید که صفحه ASP.NET عادی شما فاقد عمل Caching باشد ولی UserControlقرار داده شده در آن شامل Caching به صورت زیر باشد :

```
<%@ OutputCache Duration = "30" VaryByParam = "none" %>
```

همچنین UserControl ما شامل یک Label که زمان جاری سیستم میزبان را نشان میدهد باشد. همچنین صفحه اصلی ما نیز شامل یک Label زمان باشد. همچنین در هر دو روال Page_Load مربوط به صفحه اصلی و UserControlنسبت دهی زمان سیستم به این Label ها قرار داده شود. مشاهده خواهیم برای بار اول درخواست صفحه، زمان نمایش داده شده در هر دو Label یکسان است. اما برای بار دوم درخواست در صورتی که فاصله زمانی دو درخواست کمتر از ۳۰ ثانیه باشد مقدار زمان موجود در Label صفحه اصلی تغییر می کند ولی زمان Label موجود در UserControl تغییر نمی کند. توجه داشته باشید که برای درخواست های با فاصله زمانی بیش از ۳۰ ثانیه زمان هر دو یکسان خواهد بود. به هر حال زمان موجود در Labelصفحه اصلی در هر درخواست تغییر می کند.

بنابراین مشاهده شد که توسط UserControl ها می توان Caching را فقط در قسمت خاصی از صفحه قرار داد که این هم سرعت صفحات را بالا می برد و هم از دوباره کاری جلوگیری می کند. توجه کنید Caching در UserControlها می تواند توسط خصوصیت VaryByParam نیز تحت تاثیر قرار گیرد که این باعث می شود شرایط Caching را گسترده و گسترده تر کرد. بنابراین برعکس ASP کلاسیک که Caching در آن چندان مورد استفاده قرار نمی گرفت (اگرچه لزوم آن احساس می شد). لزوم استفاده از Caching در بالا بردن سرعت صفحات در ASP.NETبه وفور دیده می شود.

نکاتی جهت بهینه سازی برنامه های ASP.NET[1]

در این مقاله با نکاتی آشنا می شوید که جهت بهینه سازی و افزایش کارآیی و سرعت برنامه های ASP.NETلازم می باشند .

اگرچه ASP.NET مدل جدیدی از برنامه نویسی تحت وب می باشد و آمده است که نواقص و کمبودهای قبلی را برطرف نموده و به کارآیی برنامه های تحت وب بیافزاید، اما شما همچنان نیاز دارید که جهت کارآیی و بهینه بودن برنامه هایتان نکاتی را رعایت کنید. این نکات جهت افزایش کارآیی، سرعت و صرفه جویی در مصرف منابع بر روی سرور است. جهت دستیابی به لیست کامل و مفصل این نکات به آدرس زیر در سایت MSDN مراجعه نمائید.

msdn.microsoft.com/library/default.asp@url=_2flibrary_2fen-us_2fcpguide_2fhtml_2fcpcondevelopinghigh-performanceaspnetapplications.asp

چکیده برخی از این نکات بشرح زیر است:

Session Stateرا در صورت عدم نیاز حتما غیرفعال نمائید.

بصورت پیش فرض وضعیت Session State برای هر صفحه فعال است. اگر مطمئن هستید که در آن صفحه از متغیرهای Session استفاده نمی کنید، حتما آن را در دایرکتیو Page بالای صفحه تان با قراردادن آن برابر False غیرفعال کنید تا بدین ترتیب منابع اضافی در سرور مصرف نشود. اگر هم مطمئن هستید که در کل برنامه یا سایت تان نمی خواهید از متغیرهای Session استفاده کنید می توانید آن را در web.config بطور کلی غیرفعال کنید.

از رفت و برگشت های زیاد از حد پرهیز کنید.

[1] مدیریت سایت iranasp.net

فرم های وب جهت پردازش باید به سرور ارسال شده و نتیجه پردازش برگشت داده شود. به این عمل یک رفت و برگشت یا round trip می گویند. از آنجا که این رفت و برگشت ها برای سرور ایجاد بار پردازشی و ترافیک می کنند، سعی کنید بگونه ای از کنترلهای وب استفاده کنید که حداقل رفت و برگشت را داشته باشید و تا جائیکه می توانید از پردازش های سمت کاربر برای کارهای ساده استفاده کنید.

از کنترل های سرور با احتیاط استفاده کنید.

اگرچه کنترلهای سرور دارای قابلیتهای بسیار زیاد و جالبی می باشند اما این قابلیتها ناشی از اجرای آنها بر روی سرور است. از آنجا که هر اجرائی بر روی سرور ایجاد بار اضافی در سرور می کند لذا در استفاده از این نوع کنترلها خست بخرج دهید و اگر جائی می توانید از کنترلهای معمولی HTML استفاده نمائید حتما این کار را بکنید.

از Page.IsPostback استفاده کنید.

استفاده از Page.IsPostback سبب کاهش رفت و برگشت های (round trip) اضافی می گردد. همچنین به شما کمک می کند که تشخیص دهید چه موقع نیاز به تولید داده جدید برای یک صفحه دارید.

تا آنجائیکه ممکن است از View State استفاده نکنید.

View Stateهم مانند Session State برای نگهداری اطلاعات و سابقه یک فرم از منابع سرور استفاده کرده و به حجم صفحات اضافه می کند. از طرفی این حالت بصورت پیش فرض فعال می باشد. لذا در صورتیکه به آن نیاز ندارید از آن استفاده نکنید و حالت آن را غیرفعال نمائید.

از "Strict="true در صفحات تان استفاده کنید.

جهت استفاده از مزایای early binding از عبارت فوق در دایرکتیو Page استفاده کنید. این مساله سبب می شود تا شما مجبور شوید تایپ هر متغیر را در لحظه تعریف آن متغیر بصورت صریح مشخص کنید. همانطور که می دانید در VB.NETو JScript.NET جهت پوشش برنامه های ASP کلاسیک که بدون تایپ بود، می توانید تایپ یک متغیر را در لحظه تعریف آن مشخص نکرده بلکه بعدا هنگام انتساب مقادیر به آن، این کار (تعیین تایپ) بصورت خودکار انجام شود. این مساله سبب صرف منابع اضافی در سرور می گردد.

از Stored Procedure استفاده کنید.

اگر از بانک اطلاعاتی در برنامه تان استفاده می کنید، سعی کنید که حتما پرس و جوهای (query) خود را به روال های ذخیره شده (stored procedure) بسپارید. از آنجائیکه این روال ها برای مراجعه اول کامپایل شده و از آن پس جهت اجرا در اختیار مدیر بانک اطلاعاتی (DBMS) قرار می گیرد، نسبت به پرس و جوهای معمولی درون برنامه ای بسیار کاراتر و سریعتر هستند. از طرفی با بهینه سازی خاصی که برای گرداننده SQLServer (driver) در مجموعه دات نت شده است، سرعت اجرای روال های ذخیره شده نسبت به روش های معمولی OLEDB یا ODBC بیش از دو الی سه برابر می باشد.

از Data Reader استفاده کنید.

اگر حجم بالائی از داده ندارید یا تنها می خواهید از بانک اطلاعاتی عمل خواندن را انجام دهید حتما از Data Readerاستفاده کنید. استفاده از Data Set سبب کندی کار و مصرف بالای منابع می گردد.

از Caching استفاده کنید.

Cachingشاهکار ASP.NET است. تا آنجائیکه ممکن است از caching استفاده کنید. استفاده از cachingسبب می شود تا از تولید مجدد کد HTML لازم برای یک صفحه تکراری جلوگیری شده و کاربر هم صفحات را خیلی سریعتر دریافت کند.

حالت Debug را غیرفعال کنید.

هنگامیکه کار برنامه نویسی شما تمام شده است و زمان استفاده واقعی از برنامه فرا رسیده است، فراموش نکنید که حالت Debug در کامپایل نهائی برنامه تان را غیرفعال نمائید.

ویژگی های امنیتی ASP.NET[1]

بررسی مختصر مساله امنیت در برنامه های ASP.NET در نحوه ارتباط آن با کاربر

مقدمه

امنیت یکی از مسائل اصلی برای توسعه دهندگان و معماران برنامه های کاربردی است. همانطور که تعداد بیشماری از سایتهای وب با انواع امنیت ها مورد نیاز است، توسعه دهندگان نیز باید بدانند چگونه با مقوله امنیت کار کنند و چه مدل امنیتی مناسبی برای برنامه های کاربردیشان انتخاب کنند.

بعضی از سایتهای وب اطلاعات خاصی را از کاربر معمولی دریافت نمی کنند، اما اطلاعات موجود در خود را منتشر می کنند. همانند موتورهای جستجو در حالیکه سایتهای دیگری وجود دارند که نیازمند جمع آوری اطلاعات حساس از کاربرانشان هستند) برای مثال شماره کارت های اعتباری و دیگر اطلاعات شخصی). این سایتهای وب به پیاده سازی امنیت مستحکم تری برای جلوگیری از حمله احتمالی موجودیت های خارجی نیازمندند.

تفاوت جریان امنیتی ASP وASP.NET

جریان امنیتی صفحات ASP.NET از جریان امنیتی ASP کلاسیک متفاوت است. درASP ، بصورت پیش فرض IIS خود را به عنوان یک کاربر معتبر معرفی می نماید در حالیکه در ASP.NET توسعه دهنده کنترل بیشتری بر روی تنظیم امنیت در سطوح مختلف را دارا می باشد.

عملیات اساسی امنیتیASP.NET

[1] مهدی میرزائی

Authentication: عبارت است از روند اعتباردهی هویت یک کاربر به پذیرفتن یا رد کردن یک درخواست، یعنی دریافت گواهی نامه ها (برای مثال نام کاربر و کلمه عبور) از کاربران و اعتبار دهی آن. بعد از اینکه هویت بررسی شد و معتبر تشخیص داده شد، کاربر بصورت قانونی مطرح می شود و درخواست های دسترسی به منابع انجام می شود. بصورت ایده آل درخواست های آتی همان کاربر تا هنگام خروج از سیستم مرتبط با روند Authentication نیست.

Authorization: عبارت است از روند تضمین نمودن اینکه کاربران با هویت های معتبر مجاز به دسترسی به منابع مشخصی هستند.

Impersonation: این روند یک برنامه کاربردی را قادر می سازد تا به نوبت هویت کاربر و درخواستهای بعدی او برای سایر منابع را تضمین کند. دسترسی به منابع متناسب با هویت کاربری که جایگزین شده است (Impersonated)تایید یا رد می گردد. به عبارت دیگر Impersonation یک پروسه سرویس دهنده (Server Process)را قادر می سازد تا با استفاده از امنیت گواهی نامه سرویس گیرنده ها (Client) اجرا شود.

ارسال نامه در ASP.NET[1]

در این مقاله یاد خواهیم گرفت که چگونه می توان از طریق یک صفحه ASP.NET یک Email فرستاد .

لینک دریافت کد: www.iranasp.net/download/pourshahid02.zip

یکی از پر کاربردترین سرویسها بر روی اینترنت ارسال email می باشد که میخواهیم به آن بپردازیم. در دات نت برای این عمل باید از نامکده System.Web.Mail استفاده نمایم و به منظور وارد کردن آن از کد زیر استفاده مینمایم .

```
<%@ Import Namespace="System.Web.Mail" %>
```

همانطور که می دانید در زبانهای شئ گرا برای استفاده از اشیا باید ابتدا از آنها یک کپی یا در اصطلاح Instanceبرداشت به این منظور از کد زیر استفاده می نماییم .

```
Dim mail As New MailMessage
```

حال نوبت به استفاده از خواص و متدهای شئ کپی برداری شده می باشد، که اجزا مهم آن به شرح زیر می باشد:

mail.From:آدرس فرستندهemail

mail.To:آدرس گیرندهemail

mail.Subject: موضوع

mail.Body: متن email که می تواند به صورت HTML باشد

[1] علیرضا پورشاهید

mail.BodyFormat:نوع ساختار متن را مشخص می کند

SmtpMail.SmtpServer:سرویس دهنده محلی (Local) شما می باشد

SmtpMail.Send(mail):متد برای ارسال در انتهای کار

در اینجا یک مثال مشاهده می کنید که با استفاده از فرم و دکمه Submit یک آدرس پست الکترونیکی را دریافت کرده و پیام پیش فرض را به آن ارسال می دارد .توجه داشته باشید که در اینجا از یک صفحه برای هر دو عملیات دریافت آدرس و ارسال استفاده شده و با استفاده از یک کنترل Label که از کنترل های سرور در دات نت می باشد نشان میدهیم که در حال دریافت آدرس هستیم یا ارسال.

```
<%@ Page Language="VB" EnableSessionState="False"
EnableViewState="False" Trace="False" Debug="False" Strict="True" %>
<%@ Import Namespace="System.Web.Mail" %>
<script language="VB" runat=server>
Sub Page_load(Sender as Object, E as EventArgs)

   If request.form("EmailAddress") = "" Then
      dim strResponse as string = "<h2>Send Email using ASP.NET formatted in
HTML</h2>"
      lblMessage.Text = strResponse
   Else
      dim strResponse as string = "You just sent an email message formatted in HTML
to:<br><h2>" &request("EmailAddress") & "</h2>"
      lblMessage.Text = strResponse
   End If
```

```
End Sub

Sub btn_Click(sender as Object, e as System.EventArgs)
   If request.form("EmailAddress") <> ""
      Dim mail As New MailMessage
      mail.From = youraddress@domain.com
      mail.To = request.form("EmailAddress")
      mail.Subject = "Message sent using ASP.NET and CDONTS"
      mail.Body = "HTML Message sent from Iranasp.net using ASP.NET and
Cdonts<br>Wonder how this is done?<br><br>Wonder How to setup CDONTS?"
      mail.BodyFormat = MailFormat.Html
      SmtpMail.SmtpServer = "LocalServerName"
      SmtpMail.Send(mail)
   End If
End Sub
</script>
<html>
<head>
</head>
<body>
<h1 align="center">Sending Email via ASP.NET and CDONTS..</h1>
<b>How do I setup my server to use CDONTS?</b>
<br/>
<br/>
<br/>
<br/>
```

```
<asp:Label id="lblMessage" Font-Name="Verdana" Width="400px"
BorderStyle="solid" BorderColor="#cccccc" runat="server"/>
<form method="post" name="form1" runat="server" runat="server">
Email Address:<input type="text" name="EmailAddress" size="30" value=""><br><br>
<input type="Submit" id="btnSubmit" OnServerClick="btn_Click" value="Sending
Email with ASP.NET" name="b1" runat="server" />
</form>
</body>
</html>
```

Smart Navigation چیست؟[1]

این مقاله به توضیح مفهوم smart navigation (هدایت هوشمندانه!) و چگونگی استفاده از آن می پردازد .

مفهوم Smart Navigation و فواید آن

Smart Navigationیکی از بهترین ابزارهای جدیدی است که ASP.NET آنرا عرضه کرده است. این ابزار جدید باعث شده ظاهر برنامه های وب و احساسی که نسبت به آن وجود دارد شباهت بیشتری با برنامه های عادی و نوشته شده برای ویندوز پیدا کند.

یکی از موانع بزرگ برنامه های تحت وب به معماری و ساختار HTTP برمی گردد. جاییکه مجبوریم اطلاعات جمع آوری شده در سمت مشتری را به سرور بازگردانیم. به همین دلیل مجبور به رسم مجدد و کامل صفحه ای که قبلا دیده ایم می باشیم، که این نه تنها باعث می شود یک حالت فلش مانند در این رفت و برگشت و رسم مجدد رخ دهد، بلکه برای صفحه های بلند که برای دیدن تمام صفحه نیازمند به scrollingهستیم، باعث می شود که دیدمان را به اول صفحه انتقال دهد، چیزی که هم شاید دلخواه ما نباشد و هم اینکه ممکن است باعث سردرگمی کاربر گردد. همچنین این فرآیند باعث تغییر فوکوس کنترل ها و بسیاری از اتفاقات دیگرنیز می شود.

در برنامه های عادی ویندوز ما به طور معمول فقط قسمت هایی از صفحه را به روز می کنیم که تغییری در آن ایجاد شده باشد یا تحت تاثیر چیزی قرار گیرند و این بدون نیاز به تغییر در کل برنامه می باشد (مثلا فقط یک عضو به listbox ما اضافه می شود. بدون تغییر و رسم مجدد فرم برنامه).

[1] محمود مروج

Smart Navigationیا به عبارتی هدایت هوشمندانه این توانایی موجود در برنامه های ویندوز را برای برنامه های تحت وب فراهم می کند! اما قبل از هر چیز باید بدانید که این ابزار فقط برای IE می باشد و آن هم نسخه های ۵ به بالاتر آن. با این وجود شما می توانید Smart Navigation را فعال یا غیرفعال سازید، بدون آنکه تاثیری در برنامه شما بگذارد. حتی اگر شما در پروژه تان مرورگرهای محتلفی را مدنظر قرار داده باشید، می توانید Smart Navigationرا فعال سازید. در این صورت ASP.NET نوع مرورگر را تشخیص داده و Smart Navigation را فقط برای مرورگرهای پشتیبانی شده فعال می سازد.

چهار مورد برجسته ای که Smart Navigation فراهم می کند عبارتند از:

- صفحه در میان درخواست ها یک نمایش ممتد را داراست و به عبارتی حالت فلش زدن به خود نمی گیرد.
- موقعیت Scroll را حفظ می کند.
- فوکوس عضو دارنده فوکوس را نگه می دارد.
- آخرین صفحه درون تاریخچه (History) نگهداری می شود.

این ابزار در حالت واقع گرایانه برای برنامه هایی که ارسال به عقب Postback (!) فراوانی دارند طراحی شده است ولی با توجه به این نکته که محتوای صفحه نباید زیاد تغییر نکند. احتمالا بنا به دلایل کارایی و نه اینکه در تغییرات زیاد ایرادی بهم بزند - مترجم. شاید یک چیز شگفت آور در مورد این ابزار این باشد که شما در حقیقت نیاز به نوشتن هیچ کد و برنامه ای ندارید.

نحوه استفاده

Smart Navigationدرون هدایت کننده صفحه(<%@ %> : Page directive) ، برای تنظیم یک صفحه و درون web.config برای تنظیم کل برنامه استفاده می شود. برای تنظیم در Page Directive به صورت زیر عمل کنید:

```
<%@ Page SmartNavigation=true %>
```

و برای تنظیم در web.config از ساختار زیر استفاده نمایید :

```
<Configuration>
<System.web>
<Pages SmartNavigation=true />
</System.web>
</Configuration>
```

روش کار اینگونه است که کل صفحه بدرون یک فریم دورنی مخفی (hidden IFrame) بارگذاری (load) می شود و سپس فقط قسمت های تغییر کرده دوباره رندر (render)می شوند.

برگرفته از کتاب Professional ASP.NET 1.0 انتشارات Wrox

معماری فایل ASP.NET[1]

آشنایی با انواع فایل و پسوندهای آنها در برنامه های کاربردی ASP. NET .

در عمل و ساختار برنامه های کاربردی ASP.NET از فایل های زیادی استفاده می شود. در ادامه فایل های تشکیل دهنده برنامه کاربردی ASP.NET را فهرست و نقش آنها در برنامه را خواهیم دید.

.asax

این پسوند برای فایل خاصی بنام Global.asax استفاده می گردد. این نوع فایل محتوی ترکیب نحوی رویداد برای نوشتن رویدادهای سطح برنامه کاربردی ASP.NET است. آن را می توان در دایرکتوری ریشه یک برنامه کاربردی ASP.NET پیدا کرد.

.ascx

این نوع فایل نمایانگر یک کنترل تعریف شده توسط برنامه نویس ASP.NET است .(User Control) صفحات ASP.NET عموما از کنترلهای سرویس دهنده (کادرهای متنی، کادرهای فهرست، دکمه ها و غیره) تشکیل شده اند که عناصر اولیه صفحه وب را می سازند .ASP.NET درست مانند برنامه سازی مرسوم ویژوال بیسیک امکان ایجاد کنترلهای تعریف شده توسط برنامه نویس را فراهم می کند. این کنترل ها معمولا از ترکیب کنترل های سرویس دهنده و برنامه سازی برای انجام یک وظیفه یا مجموعه ای از وظایف بخصوص تشکیل می شوند.

.asmx

[1] مدیریت سایت

این پسوند برای سرویس های XML وب استفاده می گردد .این نوع فایل از سوی آن دسته از سرویس های وب میزبان مورد استفاده قرار می گیرد که از راه دور یا بطور محلی در معرض برنامه های کاربردی .NET هستند. سرویس وب موجودیتی قابل برنامه ریزی است که برای برنامه کاربردی یک عنصر خاص کارکرد معینی را تامین می کند.

.aspx

از این پسوند که اصلی ترین پسوند ASP.NET است برای فرم های وب و صفحات معمولی ASP.NET استفاده می گردد.

.axd

این نوع فایل مربوط به tracing برنامه کاربردی ASP.NET است و به ASP.NET اجازه می دهد تا به جمع آوری اطلاعات درباره درخواستهای HTTP برای یک برنامه کاربردی بپردازد.

.vsdisco

این نوع فایلXML ، لینک ها را در معرض منابع دیگری قرار می دهد که سرویس وب را توصیف می کنند. از فایل VSDISCO برای کشف سرویس های وب استفاده می شود که بطور عمومی موجود هستند.

.htm

فایل های معمولی از نوع HTML را در برمی گیرند.

.xml

این نوع سند XML مخصوص استفاده در برنامه های کاربردی ASP.NET است. این فایل XML برای مقاصد مختلفی استفاده می شود از جمله نگهداری اطلاعات برنامه کاربردی و نیز مجموعه داده های بازگشته از بانک اطلاعاتی.

.vb

این فایل شامل کد ویژوال بیسیک است که آن نیز به نوبه خود به یک فایل ASPX یا ASCX به ارث می رسد و کلیه عملیات برنامه ای مربوط به صفحات ASP.NET در آن قرار می گیرد. این نوع فایل اصطلاحا code-behindنیز نامیده می شود.

.cs

همانند پسوند .vb است اما بجای کد ویژوال بیسیک حاوی کد زبان C# است.

.config

پسوند فایل web.config است.این نوع فایل بر یک فایل پیکربندی دلالت دارد یعنی فایلی که از آن برای تعیین صفات مشخصه مختلف برنامه کاربردی استفاده می شود. این صفات مشخصه شامل تنظِمات اشکال زدایی(debuging)، تائید امنیت(authentication)، کارکرد پیگیری(tracing) ، نگهداری جلسه کاری (state management)و عمومی سازی (globalization & localization) می شوند.

(برگرفته از کتاب آموزشی ASP.NET از انتشارات کانون نشر علوم)

www.nashreoloom.com/showbooks.aspx@id=110

codebehind چیست؟[1]

بررسی مفهوم codebehind و مزایا و نحوه بکارگیری و پیاده سازی آن به سه روش مختلف

codebehind اصطلاحی است که به مجموعه کدهای برنامه ای که یک فرم وب(یا سرویس وب یا کنترل کاربری یا دیگر اشیاء ASP.NET) استفاده می کند می گویند. این کدها درون فایل ASPX یا (ASMX یا ASCX) نبوده بلکه درون فایل دیگری با پسوند-aspx.cs برای C# (برای VB.NET بصورت aspx.vb) قرار می گیرند. جداسازی کد برنامه از عبارات HTML در یک فرم وب دارای محاسن و مزیتهای بسیار زیادی است. برخی از این مزیتها عبارتند از:

- امکان جداسازی تیمهای تولید یک سایت وب بدین صورت که طراحان صفحات می توانند بدون آنکه کاری به کدهای برنامه صفحه داشته باشند یا آسیبی به آن برسانند با عبارات HTML مربوطه کار کنند.
- **حفاظت و امنیت برنامه**
- **پشتیبانی بسیار خوب محیط VS.NET (ویژوال استودیو دات نت) از این مفهوم**
- **سرعت تولید سایتهای وب**
- **نگهداری آسان سایتها و برنامه های ایجاد شده**

VS.NETتقریبا هیچگونه امکاناتی را جهت استفاده از بلوکهای <Script runar="server"> که مفهوم مقابل codebehind است را ندارد.

[1] مدیریت سایت

حال که با مزایای codebehind آشنا شدیم، سوال بعدی این است که این مفهوم چگونه کار می کند؟ در ASP.NETهمه چیز شئ یا object می باشد. در ASP.NET حتی یک صفحه ASPX بدون هرگونه کد و برنامه هم یک شئ می باشد که از کلاس System.Web.UI.Page به ارث گرفته شده است. هنگامی که صفحه مذکور برای اولین بار فراخوانی (اجرا) می شود محتویات آن صفحه توسط هسته ASP.NET کامپایل شده و نسخه کامپایل شده آن تا زمانی که تغییری در صفحه ایجاد نشده است، جهت فراخوانی های بعدی در سیستم cache می شود.

از آنجا که یک صفحه ASPX بصورت شئ عمل می کند لذا ما می توانیم بخوبی از مفهوم وراثت در برنامه نویسی شئ گرا بهره ببریم. بدین صورت که می توانیم کلاس پدر آن را بطور دلخواه تعریف و پیاده سازی کنیم بشرطی که این کلاس پدر خود بنوعی از کلاس System.Web.UI.Page به ارث گرفته شده باشد.

کلاس codebehind

بجای اینکه صفحه ASPX شما مستقیما از کلاس System.Web.UI.Page به ارث گرفته شود، صفحه شما می تواند از یک کلاس دلخواه دیگر بصورت زیر به ارث گرفته شود :

```
<%@ Page Inherits="YourNamespace.NewPage" %>
```

بمنظور عمل نمودن عبارت فوق، کلاس NewPage باید بصورت مستقیم یا غیر مستقیم از کلاس System.Web.UI.Pageبه ارث گرفته شده باشد. هنگام فراخوانی صفحه فوق، هسته ASP.NET در شاخه \bin یا اسمبلی سراسری cache شده (Global Assembly Cache) بدنبال کلاس NewPage می گردد و در صورتیکه آن را پیدا نکند خطائی را اعلام می کند. با استفاده از این روش می توان یک کلاس پایه بدین منظور در سراسر یک برنامه یا سایت وب داشت بطوریکه همه صفحات از آن کلاس به ارث گرفته شوند. بعنوان مثال مجله الکترونیکی angryCoder ← (www.angrycoder.com/default.htm) از همین روش استفاده کرده است. بهرحال بیشتر برنامه ها از کلاس های متفاوتی جهت کلاس codebehind صفحات خود استفاده می کنند.

شما می توانید فایلی را که محتوی codebehind شماست را به هسته ASP.NET معرفی کنید. این معرفی سبب می شود که ASP.NET خود بقیه کارهای لازم) کامپایل کردن فایل codebehind و قرار دادن حاصل آن که در اصل یک فایل dll است در شاخه bin\ برنامه) را انجام دهد. این کار را با استفاده از مشخصه Src بصورت زیر می توانید انجام دهید :

```
<%@ Page Src="NewPage.aspx.cs" Inherits="YourNamespace.NewPage" %>
```

اگر از VS.NET استفاده می کنید بصورت پیش فرض مقدمات استفاده از codebehind برای شما بصورت زیر فراهم شده است :

```
<%@ Page Codebehind="NewPage.aspx.cs" Inherits="YourNamespace.NewPage"
%>
```

VS.NETکلیه فایلهای codebehind را در قالب یک فایل dll یا اسمبلی کامپایل کرده و حاصل را در شاخه bin\ موجود در شاخه اصلی برنامه یا سایت قرار می دهد. دقت داشته باشید که الویت مشخصه Src از مشخصه Codebehindبالاتر است و در صورت استفاده همزمان از آنها در دایرکتیوPage ، فایل معرفی شده در عبارت Src بعنوان فایل codebehind در نظر گرفته می شود.

خلاصه

با توجه به مطالب فوق، سه روش جهت معرفی codebehind وجود دارد. روش اول بدین صورت است که با فرض موجود بودن dll مربوطه در شاخهbin\ ، تنها به ذکر نام کلاس codebehind در عبارت Inherits اکتفا نمائیم. روش دوم استفاده از مشخصه Src جهت معرفی نام فایل codebehind است. این روش سبب می شود تا ASP.NETبا استفاده از کامپایلر JIT محتوای فایل codebehind را قابل استفاده نماید. روش سوم که روش

VS.NETاست از مشخصه Codebehind استفاده می کند. این روش همانند روش اول است با این تفاوت که خود VS.NETکلیه کارهای لازم را بصورت خودکار برای ما انجام می دهد.

Namespace چیست؟[1]

ارائه تعریف و مفهوم Namespace با اشاره بهnamespace های موجود در چارچوب دات نت و نحوه بکارگیری و تعریف آنها در برنامه .

یک نکته مهم که در زمان استفاده از .NET Framework باید به آن توجه داشت آن است که فضانام (namespaceیا نامکده) ها در ساختمان برنامه کاربردی قرار دارند. فضانام یک طرح نامگذاری منطقی برای گروه بندی کلاس های مرتبط است. این طرح مانع از آن می شود تا کلاس هایی که برای متدها و خصوصیات از یک شناسه یکسان استفاده می کنند تداخل داشته باشند.

مثلا .NET Framework برای گروه بندی تایپ ها به مقوله های منطقی عملکرد، از قبیل چارچوب برنامه کاربردیASP.NET ، از یک طرح نامگذاری سلسله مراتبی استفاده می کند. ابزارهای طراحی از فضانام ها با هدف تسهیل مرور و ارجاع تایپ ها در برنامه بهره برداری می کنند. مثلا فرض کنید در حال نوشتن کد زیر هستید:

```
Public Class NewClass
  [Procedures and Functions]
End Class

Public Class NewClass
  [Procedures and Functions]
End Class
```

[1] مدیریت سایت

این کد به خطا منجر می شود چون کامپایلر راهی برای تشخیص کلاس ها از یکدیگر ندارد. برای غلبه بر این مشکل می توان از یک فضانام استفاده کرد که اجازه می دهد دو کلاس همنام در صفحه با هم وجود داشته باشند. قطعه برنامه زیر تعریف این دو کلاس در فضانام های منحصر بفرد را نشان می دهد:

```
Namespace One
  Public Class NewClass
    [Procedures and Functions]
  End Class
End Namespace

Namespace Two
  Public Class NewClass
    [Procedures and Functions]
  End Class
End Namespace
```

در این کد برخوردی بین دو کلاس با نام NewClass وجود ندارد چون هر کدام در یک فضانام جداگانه قرار داده شده است. کلاس اول را می توان با استفاده از ترکیب One.NewClass صدا زد، حال آنکه کلاس دوم را می توان با استفاده از ترکیب Two.NewClass صدا زد.

شما می توانید در فضانام های خود از یک ساختار سلسله مراتبی استفاده کنید. قرار دادن اشیاء مشابه تحت زیرعنوانها در یک فضانام مشترک تشخیص هدف فضانام را آسانتر می کند، و در عین حال باعث می شود برنامه به مراتب شئ گراتر شود.

برای توضیح فضانام می توان ساختار فایل و دایرکتوری (کشو، فولدر) در یک کامپیوتر را در نظر گرفت. در این مثال کلاس ها به مثابه فایل ها و فضانام ها مانند دایرکتوری ها هستند. بدیهی است همانگونه که می توانیم دایرکتوریهای تو در تو داشته باشیم، فضانام ها هم در حالیکه کلاس ها را در خود جای داده اند می توانند بصورت تو در تو باشند.

فضانام ها در ساخت برنامه های کاربردی ASP.NET نقش مهمی ایفا می کنند. خوشبختانه لازم نیست برای همه اشیایی که می توانند به وسیله صفحات ASP.NET مورد استفاده قرار بگیرند سیستم طبقه بندی فضانام ایجاد کنید. مایکروسافت این مساله را برای شما حل کرده است. دو فضانام ریشه، و فضانامهای فرزند آنها را می توان وارد صفحات ASP.NET خود کرد. اولی System نامیده می شود، و دومی Microsoft نام دارد. این فضانامها با جزئیات بیشتر در ادامه مورد بحث قرار گرفته اند.

فضانام System

فضانام System فضانام اصلی برای ساخت ASP.NET و همه برنامه های کاربردی دیگر مبتنی بر NET .Frameworkاست. هر چیزی که در برنامه کاربردی شما قابل انجام باشد از طریق فضانام System کنترل می شود. به عنوان مثال کنترل آرایه، عملیات ریاضی، و تبدیل نوع داده ها از طریق فضانام System و فضانامهای فرزند آن اداره می شوند. ۹ فضانام پیش فرض (فضانام System و ۸ فرزند آن (وجود دارند که به صورت خودکار به صفحات ASP.NETاضافه می شوند:

- System
- System.ComponentModel.Design
- System.Data
- System.Drawing
- System.Web.SessionState

- System.Web
- System.Web.UI
- System.Web.UI.WebControls
- System.Web.UI.HTMLControls

هشت فضانام (بجز فضانام System) در زمان ساخت Visual Studio.NET یا VS.NET بطور خودکار به صفحات ASP.NET وارد می شوند. این فضانام ها در زیر به اختصار شرح داده شده اند.

- **System.ComponentModel.Design:** دربرگیرنده کلاس هایی است که می توان از آنها برای طراحی پشتیبانی سفارشی اجزا و زمان طراحی و دسترسی به سرویس های تامین شده توسط معماری .NET Framework استفاده کرد.

- **System.Data:** امکان دسترسی به کلاس ها و رابطهایی را فراهم می کند که معماری ADO.NETرا برای دسترسی به داده های عمومی تشکیل می دهند.

- **System.Drawing :** دربرگیرنده کلاس ها و رابطهایی است که عملکرد گرافیکی اولیه را تامین می کنند. فضانام System.Drawing نیز از طریق فضانام System.Drawing.Drawing2Dو System.Drawing.Imaging عملکرد پیشرفته تری فراهم می کند.

- **System.Web :** کلاس ها و رابطهایی تامین می کند که ارتباط مرورگر/سرویس دهنده را امکان پذیر می کنند. این فضانام دربرگیرنده کلاس HTTPRequest (فراهم کننده اطلاعات وسیعی درباره درخواست HTTP جاری)، کلاس HTTPResponse (فراهم آورنده امکان دسترسی به فرآیندها و یوتیلیتی های سمت سرویس دهنده) است.

- **System.Web.SessionState:** فراهم کننده کلاس ها و متدهایی برای مدیریت وضعیت جلسات کاری می باشد.

- **System.Web.UI:** فراهم کننده کلاس ها و رابطهایی برای رابط واسط کاربر برنامه کاربردی ASP.NET است که موجب می شوند برنامه کاربردی با سطوح مختلف صفحه، ارتباط برقرار کند. کلاس اصلی این فضانام، کلاس Page می باشد که دربرگیرنده همه خصوصیتها، متدها، و سازنده های صفحه است. اشیاء اصلی Active Server Page زیر خصوصیتهایی در کلاس Page هستند : Application، Response، Request، Server و Session.

- **System.Web.UI.HTMLControls:** کلاس هایی برای عناصر HTML استاندارد، شامل فرم ها، کنترل های ورودی، آنکور، جداول، قسمتهای متنی، و غیره فراهم می کند. این کنترلها همانند تگهای عادی HTML هستند با این تفاوت که داری دو صفت runat="server" و id="*controlname*" می باشند.

- **System.Web.UI.HTMLControls:** برای کنترلهای سرویس دهنده ای که شبیه کنترلهای HTML هستند ولی انعطاف پذیری بیشتر و عملکرد پیچیده تری دارند کلاس هایی را تامین می کند.

برخی فضانام های مهم و پرکاربرد دیگر به شرح زیر می باشند.

- **System.IO:** دربرگیرنده رابطها و کلاس هایی است که امکان خواندن و نوشتن همگام و غیرهمگام فایل ها و جریانهای داده را فراهم می کنند.

- **System.Data.OleDb:** امکان دسترسی به کلاس ها و رابطهای مخصوص دسترسی به یک منبع داده از طریق ADO را فراهم می کند.

- **System.Data.SqlClient:** امکان دسترسی به کلاس ها و رابطهای مخصوص دسترسی به داده های خاص Microsoft SQL Server از طریق ADO را فراهم می کند.

- **System.Web.Security:** امکان دسترسی به کلاسها و رابطهای مخصوص امنیت برنامه کاربردی ASP.NET را فراهم می کند. دستیابی به رمزنگاری، مجوزها، و تنظیمات خط مشی برنامه کاربردی در این فضانام قرار می گیرند.

- **System.XML:** امکان دسترسی به کلاسها و رابطهای مخصوص پردازش اسناد XML را فراهم می کند.

فضانام Microsoft

علاوه بر فضانام System که در چارچوب .NET یافت می شود، مایکروسافت چند فضانام اضافه کرده است که برای زبان برنامه سازی ای که می خواهید از آن در برنامه کاربردی خود استفاده کنید عملکرد لازم را تامین می کنند. ممکن است شما بصورت مستقیم با این فضانام کاری نداشته باشید.

Microsoft.VisualBasic: این فضانام محتوی CLR یا زمان اجرای Visual Basic.NET است. از این زمان اجرا با زبان Visual Basic.NET استفاده می شود. این فضانام همچنین دربرگیرنده کلاس هایی است که از کامپایل و تولید کد با استفاده از زبان ویژوال بیسیک پشتیبانی می کنند.

Microsoft.CSharp: این فضای نام دربرگیرنده کلاس هایی است که از کامپایل و تولید کد با استفاده از زبان C# پشتیبانی می کنند.

Microsoft.JScript :این فضای نام دربرگیرنده کلاس هایی است که از کامپایل و تولید کد با استفاده از زبان JScript پشتیبانی می کنند.

Microsoft.Win32 :کلاسها و رابطهای مورد نیاز برای کار با کلیدها وhive های رجیستری را تامین می کند.

با وجود آنکه فضانام ها از قبل تامین می شوند، می توانید برای استفاده از برنامه کاربردی ASP.NET فضانام های خود را ایجاد کنید. برای هر کلاس ایجاد شده توسط سازنده یک فضانام تولید می شود.

استفاده از فضانام ها در صفحاتASP.NET

دو راه برای افزودن فضانام به برنامه کاربردی ASP.NET وجود دارد. از شبه دستور (Directive) صفحه Import@برای صفحات ASPX و از کلمه کلیدی Imports برای افزودن فضانام به برنامه codebehind مربوطه در ویژوال بیسیک استفاده می شود و برای زبان C# از دستور using استفاده می گردد. قطعه برنامه زیر ترکیب نحوی برای افزودن فضانام System.Web.UI.WebControls به صفحه ASP.NET شما است.

```
<%@ Import namespace = "System.Web.UI.WebControls" %>
```

همین فضانام را در قسمت codebehind بصورت زیر به برنامه اضافه می کنیم.

```
Imports System.Web.UI.WebControls (vb.net)
using System.Web.UI.WebControls; (C#)
```

(به تفاوت Import و Imports دقت کنید)

در صورتیکه می خواهید چند فضانام را به صفحه ASP.NET خود و یا صفحه codebehind اضافه کنید باید هرکدام را جداگانه اضافه کنید. بعنوان مثال، برای افزودن فضانام System.Web.UI.HTMLControls به صفحات با فضانام های موجود، درست بعد از آخرین عبارت مهم به خط بعد بروید و Imports System.Web.UI.HTMLControlsرا اضافه کنید. به محض آنکه Imports System. را تایپ کنید، VS.NETفهرستی از فضانام ها را ظاهر می کند، و می توانید به سادگی فضانام مورد نظر را با ماوس برگزینید. امتیاز این فهرست آن است که مجبور نیستید همه فضانام های .NET را از حفظ بدانید، بلکه می توانید به آسانی از فهرست انتخاب کنید. این ویژگی با عنوان Intellisense شناخته می شود. برای صفحات ASP.NET از این ترکیب استفاده کنید:

```
    <%@ Import namespace = "System.Web.UI.WebControls" %>
<%@ Import namespace = "System.Web.UI.HTMLControls" %>
<%@ Import namespace = "namespace name" %>
    ...
```

برای صفحات codebehind ویژوال بیسیک از این ترکیب استفاده کنید:

```
Imports System.Web.UI.WebControls
Imports System.Web.UI.HTMLControls
Imports namespace
    ...
```

(برگرفته از کتاب آموزشی ASP.NET از انتشارات کانون نشر علوم)

www.nashreoloom.com/showbooks.aspx@id=110

upload کردن فایل به سرور در ASP.NET[1]

این مقاله به چگونگی upload کردن فایل به سرور در ASP.NET می پردازد .

کدهای مربوط به Upload کردن فایل به سرور را باید به دو قسمت تقسیم نمود. قسمت اول شامل کدهایی است که نحوه نمایش در مرورگر را شامل می شوند و اساس آن بر HTML است. و قسمت دوم مربوط به طرف سرور است که شامل کدهایی است که با ASP.NET نوشته می شوند. برنامه زیر حاوی کدهای مربوطه برای انجام این کار به صورات خیلی ساده و ابتدایی می باشد. همانطور که ملاحظه می کنید برخلاف ASP کلاسیک، نیاز به وجود componentخاصی نمی باشد و این کار در ASP.NET بصورت خیلی ساده انجام پذیر است.

```
Upload.aspx:
<HTML>
<HEAD>
<TITLE>Uploading File...</TITLE>
<Script language="vb" runat="server">
sub page_load(s as object,e as EventArgs)
   If Not (MyInputFile.PostedFile Is Nothing) Then
      MyInputFile.PostedFile.SaveAs("c:\filename.ext")
      Response.write("Your File was saved on the server...")
   End If
end sub
</script>
</HEAD>
<BODY>
```

[1] محمود مروج

```
  <FORM method="post" encType="multipart/form-data">
    <input id="MyInputFile" type="file" name="MyInputFile" runat="server">
    <input type="submit" value="Submit">
  </FORM>
</BODY>
</HTML>
```

در زیر به شرح و توضیح برنامه بالا می پردازیم:

کدهای مربوط به طرفClient

به دلیل آنکه در این قسمت، ارسال اطلاعات به سرور را داریم، باید کدهای مربوطه درون یک فرم قرار گیرند. اما لازم است که ویژگی encType آن فرم را به صورت زیر مقدار دهی کنیم، زیرا در غیر اینصورت فایل انتخاب شده توسط کاربر به سرور ارسال نخواهد شد:

```
encType="multipart/form-data"
```

در HTML کنترلی وجود دارد که وظیفه دریافت نام فایل و ارسال آن به سرور را عهده دار می باشد:

```
<input id="MyInputFile" type="file" name="MyInputFile" runat="server">
```

توجه کنید که این کنترل را به صورت یک سرور کنترل در می آوریم تا بتوانیم در ASP.NET از توانایی های آن استفاده نماییم و به همین دلیل ویژگی runat="server" را به کنترل مربوطه اضافه کرده ایم. بنابراین کد مربوط به قسمت Client به صورت زیر خواهد بود:

```
<body>
<form method="post" encType="multipart/form-data">
```

```
<INPUT id="MyInputFile" type="file" name="MyInputFile" runat="server">
<input type="submit" value="Submit">
</form>
</body>
```

کدهای مربوط به طرف Server

این قسمت کدهایی را شامل می شود که برای ذخیره کردن فایل بر روی سرویس دهنده مورد استفاده قرار می گیرند.همانطور که گفته شد شئ مربوط به دریافت نام فایل را به صورت سرور کنترل قرار دادیم تا بتوانیم از خصوصیات آن در سرور استفاده کنیم (در اینجا نام آنرا MyInputFile درنظر گرفته ایم). یکی از خصوصیات این شئ، شئ PostedFileمی باشد که به فایل ارسالی توسط کاربر اشاره می کند. این شئ نیز شامل متدی است که اقدام به ذخیره فایل می کند. بنابراین قبل از ذخیره کردن باید مطمئن شد که آیا شئ PostedFile به چیزی اشاره می کند و یا اینکه تهی است. برای این منظور کد مربوط به ذخیره سازی را در شرط زیر قرار می دهیم:

```
If Not (MyInputFile.PostedFile Is Nothing) Then
```

و درصورت صحیح بودن شرط، اقدام به ذخیره سازی می نماییم. متد SaveAs شئ PostedFile دارای پارامتری است که محل و نام فایلی که قرار است بر روی سرور ذخیره شود را مشخص می کند. در انتها نیز پیغامی مبنی بر اینکه فایل با موفقیت ذخیره شده است را به کاربر می دهیم.

```
Response.write("Your File was saved on the server successfuly...")
```

بنابراین کد طرف سرور به صورت زیر خواهد بود:

```
<script language="vb" runat="server">
  sub page_load(s as object,e as EventArgs)
```

```
    If Not (MyInputFile.PostedFile Is Nothing) Then
      MyInputFile.PostedFile.SaveAs("c:\filename.ext")
      Response.write("Your File was saved on the server...")
    End If
  end sub
</script>
```

چند نکته

دیگر خصوصیات شئ PostedFile به صورت زیر است :

- **ContentLength:** به اندازه فایل ارسالی اشاره میکند.
- **ContentType: MIME :** فایل ارسالی را مشخص می کند که در تشخیص نوع فایل دریافتی کمک می کند.

(بعضی از انواع رایج MIME عبارت اند از:

text/plain ، image/gif ،image/jpeg ،audio/mpeg ،application/msword ،

application/octed-stream ،video/mpeg ،audio/wav و...)

- **FileName:** نام فایل و مسیر آن در کامپیوتر شخص کاربر را معین می کند.(مانند C:\images\personal\myface.jpg)
- **InputStream:** یک شئ stream که به فایل upload شده اشاره می کند و آنرا برای خواندن محتوایش آماده می سازد .

یک نکته مهم

توجه کنید که به طور پیش فرض طول داده های ارسالی در یک request حداکثر ۴۰۹۶ بایت (یا همان 4 kb می باشد). بدین معنی که یک فایل ارسالی حداکثر می تواند 4 kb باشد. برای اینکه این مقدار را به مقدار لازم و مورد نیاز افزایش دهید باید در فایل web.config ویا machine.config در قسمت <system.web> در صورتیکه قسمت زیر موجود نیست آنرا اضافه کنید

```
<httpRuntime maxRequestLength="1000000"/>
```

و در صورت وجود مقدار maxRequestLength را به مقدار دلخواه تنظیم کنید. این مقدار بر حسب بایت می باشد. در مثال بالا مقدار ماکزیمم فایل ارسال یک میلیون بایت حدود (1 MGB) می باشد. به عنوان نمونه یک فایل web.configکه فقط نیاز فوق را برآورده سازد به صورت زیر خواهد بود:

```
Configuration file : web.config
<?xml version="1.0" encoding="utf-8" ?>
<configuration >
  <system.web>
    <httpRuntime maxRequestLength="1000000" />
  </system.web>
</configuration>
```

Caching در ASP.NET[1]

بررسی مساله Caching در ASP.NET از طریق معرفی انواع آن که عبارتند از Fragment Caching، Page Output Caching و Data Caching .

۱ -مقدمه ای بر Caching در ASP.NET

- Caching:عبارت است از قرار دادن حاصل اجرای یک صفحه در یک حافظه سریع جهت دسترسی سریع و استفاده مجدد از آن در ارجاعات بعدی.
- Caching: مهمترین فاکتور در ایجاد یک برنامه تحت وب با توانائی و کارائی بالا می باشد.
- محل های Caching عبارتند از Web server, proxy server و client browsers.
- انواع Caching عبارتند :

Output caching

Fragment caching

Data caching

۲-Output Caching

Output Cachingچیست؟

[1] مدیریت سایت iranasp.net

- صفحاتی که از Output Caching استفاده می کنند برای بار اول اجرا می شوند و سپس حاصل آن cache می گردد. جهت پاسخگوئی به درخواستهای بعدی برای همین صفحه، از نسخه cacheشده استفاده می گردد.
- فواید Output Caching در کاهش محسوس زمان پاسخ دهی سرور و کاهش بار اضافی بر روی CPU در سرور می باشد.
- استفاده صحیح از Output Caching سرعت و کارائی سایت را بطور محسوسی افزایش می دهد.
- از Output Caching می توان در فایل های با پسوند .asmx ، .aspxو .ascx استفاده نمود.
- با استفاده از دایرکتیو @OutputCache در بالای فایلهای فوق بصورت زیر می توان Output Cachingرا فعال کرد:

```
<%@ OutputCache Duration="600" Location="Any" VaryByParm="none" %>
```

- یا از درون برنامه بصورت زیر هم می توان این کار را انجام داد:

```
[C#]
Response.Cache.SetExpires(DateTime.Now.AddSeconds(600));
Response.Cache.SetCacheability(HttpCacheability.Public);
```

صفاتOutputCaching

- Duration:مدت زمان معتبر بودن cache را مشخص می کند. مقداردهی این صفت الزامی است و مقدار آن بر اساس ثانیه است.

- Location: حل قرار گرفتن cache را مشخص می کند.

در حالت Server صفحه cache شده در حافظه سرور نگهداری می شود.

در حالت Downstream صفحه cache شده بر روی proxy server نگهداری می گردد.

در حالت Client صفحه cache شده بر روی مرورگر کاربر قرار می گیرد.

در حالت Any صفحه cache شده بر روی هر یک از موارد فوق می تواند قرار گیرد.

در حالت None صفحه مربوطه عملا در هیچ کجا cache نخواهد شد.

- VaryByParam: نسخه های cache متفاوتی از صفحه مورد نظر براساس پارامترهای موجود در QueryString و Form یا ترکیبی از آنها ایجاد می گردد.

```
<%@ OutputCache Duration="10" VaryByParam="location;count" %>
```

- VaryByHeader:نسخه های cache متفاوتی براساس مقادیر مختلف پارامتر تعیین شده در HTTP header ایجاد می گردد.

```
<%@ OutputCache Duration="60" VaryByHeader="Accept-Language" %>
```

- VaryByCustom: اگر مقدار این صفت کلمه خاص "Browser" باشد، cache مورد نظر براساس نوع و نسخه اصلی مرورگرایجاد خواهد شد. اگر مقدار آن یک رشته دلخواه باشد، آنگاه لازم است که شما متد HttpApplication.GetVaryByCustomString را در فایل Global.asaxرا بگونه دلخواه بازنویسی کنید.

Fragment Caching-۳

- علاوه بر اینکه شما می توانید تمام یک صفحه را cache کنید، شما حتی می توانید بخشی از یک صفحه را cache کنید. به این عمل Fragment Caching گویند.
- بدین منظور لازم است که شما بخش های مورد نظر را بصورت User Controlیا کنترل کاربری ایجاد کرده باشید.
- هر کنترل کاربری دایرکتیو @OutputCache مخصوص به خود را دارا می باشد.
- صفات مورد استفاده در اینجا عبارتند از VaryByParam و VaryByControl.
- امکان تعیین محل cache با استفاده از صفت Location مقدور نمی باشد و محل cache همواره بر روی سرور در نظر گرفته می شود.

- VaryByControl:ششمین صفت موجود در دایرکتیو @Outputcache می باشد. تنها می توان در کنترلهای کاربری از آن استفاده کرد. استفاده از آن سبب می شود تا cacheهای متعددی براساس خواص (properties) کنترل کاربری ایجاد گردد.

```
[*.ascx]
<%@ Language="C#" %>
<%@ OutputCache Duration="10" VaryByControl="State;Country"
VaryByParam="*"%>
<script runat=server>
```

```
public String State {
get { return state.Value; }
set { state.Value = State; } }

public String Country {
get { return country.Value; }
set { country.Value = Country; } }
</script>
```

- می توان بصورت تودرتو از کنترلهای کاربری با قابلیت cache استفاده نمود. در این صورت یک cache سلسله مراتبی بسیار قدرتمند خواهیم داشت. این مساله اگر چه نیاز به هیچ نوع برنامه نویسی خاصی ندارد اما می تواند سبب مصرف حافظه زیادی گردد.
- سعی نکنید بصورت برنامه ای به یک کنترل کاربری موجود در cache دسترسی داشته باشید. در غیر این صورت شما با یک exception برخورد خواهید کرد. زیرا این نوع کنترلها در درخت کنترلهای موجود قرار نمی گیرند.

Data Caching-۴

•با استفاده از data cache می توان داده های برنامه مانند رشته ها، DataSetها و سایر اقلام داده و آبجکت را بصورت زیر cache کرد:

```
Cache ("counter") = mycount.text
```

•اگرچه این مساله مانند استفاده از متغیرهای از نوع Application است، اما بسیار قوی تر و کارآمدتر می باشد.

•هر قسمت از برنامه که از داده های موجود در cache استفاده می کند، باید قادر باشد تا در صورت غیرمعتبر بودن cache، بتواند آن را دوباره بسازد.

```
Public Function GetProductData() As DataSet
  If (IsNothing(Cache("ProductData")) Then
    Cache("ProductData") = LoadDataSet()
  End If
  Return Cache("ProductData")
End Function
```

•توصیه می شود جهت استفاده از cache همواره از مدل فوق استفاده نمائید.

•جهت داشتن کنترل بیشتر بر cache یا استفاده از امکانات پیشرفته آن از متدهای Cache.Insert و Cache.Addاستفاده نمائید.

•با استفاده از متد Cache.Remove می توانید داده مورد نظر را از cache حذف نمائید.

انتقال مقادیر بین صفحات یک برنامه[1]

در این مقاله با یک روش جدید و مختص ASP.NET بنام Context جهت تبادل مقادیر میان صفحات یک برنامه آشنا می شوید .

ASP.NETدارای امکانات خوبی جهت انتقال مقادیر و متغیرها میان صفحات می باشد. خوشبختانه علاوه بر امکان استفاده از روش های قدیمی مانند متغیرهای Application ، SessionوQueryString ، قابلیت جدیدی نیز به این مجموعه اضافه شده است که بسیار کارآمد هم می باشد. این امکان جدید شئ Context می باشد و از این قابلیت می توان در کنار Server.Transfer بخوبی استفاده نمود. استفاده از Context همانند استفاده از Session است:

```
Context.Items.Add("NameOfVariable","ValueOfVariable");
Server.Transfer("WebForm2.aspx");
```

در صفحه دوم جهت بازیابی مقدار ذخیره شده بصورت زیر عمل می کنیم:

```
Label1.Text=Context.Items["NameOfVariable"].ToString();
```

بنابراین، در صفحه دوم می توانیم مقدار ارسالی را در قالب یک شئ از تایپ object دریافت کنیم که لازم است قبل از استفاده، آن را به تایپ مورد نظر تبدیل یا Cast کنیم. این روش شاید برای تعداد محدودی از مقادیر یا متغیرها کارآمد باشد، اما اگر تعداد زیادی از این متغیرها را بخواهیم به ازای هر کاربر یا Session نگهداری کنیم در این صورت ممکن است دچار مشکل شویم یا به علت اشکالات تایپی در نام این نوع متغیرها به خطای منطقی در برنامه برخورد نمائیم. جهت حل این معضل از روش زیر استفاده می کنیم.

[1] مدسریت سایت

```
Imports System.Collections
Public Class StaticContainer
   Private Shared thing As Hashtable
   Public Shared Sub InitThing()
      thing = New Hashtable()
   End Sub
   Public Shared Sub KillThing()
      thing.Clear()
      thing = Nothing
   End Sub
   Public Shared Sub AddContainer(ByVal key As Object)
      thing.Add(key, New ArrayList())
   End Sub
   Public Shared Sub RemoveContainer(ByVal key As Object)
      thing.Remove(key)
   End Sub
   Public Shared Sub AddToContainer(ByVal newname As String, ByVal index As
Integer,
         ByVal stuff As String, ByVal key As Object)
      Dim temp As ArrayList = thing.Item(key)
      thing.Remove(key)
      Dim t As New triple()
      t.name = newname
      t.ID = index
      t.data = stuff
      temp.Add(t)
```

```
        thing.Add(key, temp)
    End Sub
    Public Shared Function GetEntry(ByVal x As Integer, ByVal key As Object) As triple
        Dim temp As ArrayList = thing.Item(key)
        Dim result As triple
        If temp.Count > x And x >= 0 Then
            result = CType(temp(x), triple)
        End If
        Return result
    End Function
End Class
Public Structure triple
    Public name As String
    Public ID As Integer
    Public data As String
End Structure
```

جهت استفاده از کلاس فوق لازم است در فایل Global.asax تغییرات زیر را اعمال کنیم:

```
Sub Application_Start(ByVal sender As Object, ByVal e As EventArgs)
    ' Fires when the application is started
    StaticContainer.InitThing()
End Sub
Sub Session_Start(ByVal sender As Object, ByVal e As EventArgs)
    ' Fires when the session is started
    Session.Add("key", DateTime.Now.Ticks.ToString("x"))
    StaticContainer.AddContainer(Session.Item("key"))
```

```
End Sub
Sub Session_End(ByVal sender As Object, ByVal e As EventArgs)
   ' Fires when the session ends
   StaticContainer.RemoveContainer(Session.Item("key"))
End Sub
Sub Application_End(ByVal sender As Object, ByVal e As EventArgs)
   ' Fires when the application ends
   StaticContainer.KillThing()
End Sub
```

برای ذخیره شناسه یا ID مربوط Container و کاربر از Session استفاده نمودیم. هنگامی که برنامه برای اولین بار اجرا می شود Hashtable مورد نظر ایجاد شده و هنگامی که Session شروع می شود فضای مربوط به نگهداری مقادیر به ازای هر کاربر نیز ایجاد می گردد. نحوه استفاده از این برنامه فوق بصورت زیر است:

```
StaticContainer.AddToContainer(TextBox1.Text, Integer.Parse(TextBox2.Text),
TextBox3.Text, Session.Item("key"))
Dim t As triple = StaticContainer.GetEntry(Integer.Parse(TextBox4.Text),
Session.Item("key"))
ListBox1.Items.Add(t.name)
ListBox1.Items.Add(t.ID.ToString())
ListBox1.Items.Add(t.data)
```

همانگونه که ملاحظه می فرمائید دیگر نیازی به تبدیل به تایپ های مختلف نداشته و در عوض بجای ساختار tripleشما هر نوع ساختاری را با هر نوع تعداد متغیر را می توانید تعریف و استفاده نمائید.

منبع: http://www.dotnetjunkies.com/tutorials.aspx?tutorialid=600

ارسال نامه در ASP.NET از نگاهی دیگر[1]

نحوه ارسال نامه در ASP.NET با امکانات و تنظیمات بیشتر با استفاده از CDOSYS.

دقیقا نمی دانم که این موضوع چقدر برای شما می تواند مهم باشد اما برای من و سایت IranASP.NET که باعث شده بود تا پل ارتباطی با مراجعه کنندگان سایت یعنی همان E-mail و صفحه ارتباط با ما، ثبت نام در راهنمای ASP.NETایران، ارسال خبرنامه و بالاخره پذیرش اشتراک خبرنامه را از دست بدهیم خیلی مهم بود. در همه قسمتهای فوق نامه یا نامه هایی توسط سایت بصورت خودکار به آدرسی ارسال می شود که حتما شما نحوه ارسال یک نامه را در ASP.NET بخوبی می دانید. اگر هم نمی دانید کافی است به کد ساده زیر برای ارسال یک نامه نمونه و ساده دقت فرمائید:

```
[C#]
MailMessage objMM = new MailMessage();
objMM.From = "me@myhost.com";
objMM.To = "you@yourhost.com";
objMM.Subject = "Welcome!";
objMM.Body = "Welcome to IranASP.NET";
SmtpMail.Send(objMM);
```

از آنجا که نامه های ارسالی به سایت بصورت روزانه دریافت و خوانده می شوند، یک روز متوجه شدم که به طرز مشکوکی هیچ نامه ای برای سایت ارسال نشده است. به صفحه ارتباط با ما رفته و سعی کردم که نامه ای تستی برای سایت ارسال کنم که با Exception یا خطای زیر مواجه شدم:

```
The "SendUsing" configuration value is invalid.
```

[1] مدیریت سایت

پس از بررسی های متعدد متوجه شدم که با یک مساله امنیتی و تغییری در تنظیمات سرور مواجه هستم. دلیلی که برای این مشکل پیدا کردم این بود: احتمالا نصب Service Pack 3 برای Microsoft Exchange Server در سرور مربوطه! اطلاعات دقیق و روشنی هم از host مربوطه نتوانستم بگیرم. آیا این دلیل درست بود یا نه مهم نبود . آنچه که مهم بود این بود که دیگر نامه های ارسالی از طریق سایت با کد برنامه فوق کار نمی کردند و ارسال نمی شدند. بعبارت دیگر برنامه سایت احتمالا بدلیل مسائل امنیتی و محدودیتهای جدید آن دیگر نمی توانست بصورت مستقیما از SMTP Server پیش فرض و موجود در IIS استفاده کرده و از طریق آن نامه ها را ارسال کند. مساله مهمی بود و باید هرچه زودتر حل می شد.

جستجوها در اینترنت شروع شد و نتایج جالبی هم بدست آمد. حاصل آن بود که باید از روش دیگری نامه را ارسال کرد. آن روش استفاده از CDOSYS بود بجای CDONTS. ظاهرا این کامپوننت فقط هم برای ویندوز ۲۰۰۰ می باشد و برای NT چیزی پیدا نکردم. البته کامپوننت CDOEX هم برای استفاده از Exchange هم موجود است.

استفاده از CDOSYS دست شما را برای ارسال یک نامه از طریق برنامه کاملا باز می گذارد و محدودیتهای CDONTSرا دیگر ندارید. موارد زیر را می توانید برای ارسال یک نامه و قبل از ارسال آن تعیین نمائید:

- آدرس SMTP Server بعنوان سرور فرستنده: این سرور می تواند هر سرور خارجی یا داخلی باشد.
- پورت مورد استفاده جهت ارسال نامه: دیگر مجبور نیستید همواره از پورت ۲۵ استفاده کنید.
- username و password برای حالتی که سرور SMTP شما نیاز به Authentication دارد.

•محل یا دایرکتوری ذخیره نامه های ارسالی پیش از ارسال: تعیین این محل در اختیار شماست. سرور فرستنده شما هر وقت فرصت پیدا کند سری به این محل زده و در صورت وجود نامه ای در آن، نامه یا نامه های موجود و منتظر را ارسال می کند.

نکته :همین مساله سبب می شود تا شما بصورت مستقیم با سرور فرستنده در ارتباط نباشید. همین مساله احتمالا مشکل سایت IranASP.NET را حل کرد!

•و بسیاری موارد دیگر که می توانید در راهنمای مربوطه مطالعه نمائید .

ماحصل همه گفتار فوق کد زیر است که در سایت IranASP.NET هم اکنون استفاده می شود و بخوبی هم جواب داده است:

```
[C#]
int cdoSendUsingPickup = 1;
string cdoSendUsingMethod = "http://schemas.microsoft.com/
cdo/configuration/sendusing";
string cdoSMTPServerPickUpDirecory = "http://schemas.microsoft.com/
cdo/configuration/smtpserverpickupdirectory";
// You can uncomment each line when you need it!
//string cdoSMTPServer = "http://schemas.microsoft.com/
cdo/configuration/smtpserver";
//string cdoSMTPServerPort = "http://schemas.microsoft.com/
cdo/configuration/smtpserverport";
//string cdoSMTPConnectionTimeout = "http://schemas.microsoft.com/
cdo/configuration/smtpconnectiontimeout";
//string cdoSMTPAuthenticate = "http://schemas.microsoft.com/
```

```
cdo/configuration/smtpauthenticate";
//string cdoSendUserName = "http://schemas.microsoft.com/
cdo/configuration/sendusername";
//string cdoSendPassword = "http://schemas.microsoft.com/
cdo/configuration/sendpassword";

//For CDOSYS, the pickup directory is located at
c:\inetpub\mailroot\pickup
string strPickup = "c:\\inetpub\\mailroot\\pickup";

CDO.Message objMM = new CDO.Message ();
ADODB.Fields Flds = objMM.Configuration.Fields;

Flds[ cdoSendUsingMethod ].Value = cdoSendUsingPickup;
Flds[ cdoSMTPServerPickUpDirecory ].Value = strPickup;

Flds.Update ();
objMM.HTMLBodyPart.Charset = "utf-8";
objMM.MimeFormatted = true;

objMM.From = "me@myhost.com";
objMM.To = "you@yourhost.com";

objMM.Subject = "Welcome!";
objMM.HTMLBody = "<HTML><HEAD></HEAD><BODY><b>Welcome to
IranASP.NET!</b></BODY></HTML>";
```

```
objMM.Send ();
```

دقت کنید چون خواسته ایم که نامه ما احتمالا فارسی باشد لذا از utf-8 و HTMLBody استفاده کرده ایم.

همچنین برای اجرای این برنامه به دو فایل DLL زیر هم در شاخه bin سایتتان یا در GAC یا هرجای دیگر هم نیاز دارید (من در bin کپی کردم):

- adodb.dll
- Interop.CDO.dll

نگران نباشید اگر ویندوز ۲۰۰۰ را دارید این فایلها در کامپیوتر شما موجود است. این دو فایل باید در قسمت References در Visual Studio .NET به پروژه سایت شما الصاق شده تا برنامه شما کامپایل شود.

جهت اطلاعات بیشتر به موارد زیر رجوع نمائید:

1. CDO for Windows 2000

msdn.microsoft.com/library/default.asp@url=_2flibrary_2fen-us_2fcdosys_2fhtml_2f_cdosys_about_cdo_for_windows_2000.asp

2. CDO for Win2000: Messaging Configuration

msdn.microsoft.com/library/default.asp@url=_2flibrary_2fen-us_2fcdosys_2fhtml_2f_cdosys_messaging_configuration.asp

3. CDO for Win2000: Configuring the Message Object

msdn.microsoft.com/library/default.asp@url=_2flibrary_2fen-us_2fcdosys_2fhtml_2f_cdosys_configuring_the_message_object.asp

4. CDO for Win2000: Creating and Sending a Message

msdn.microsoft.com/library/default.asp@url=_2flibrary_2fen-us_2fcdosys_2fhtml_2f_cdosys_messaging_examples_creating_and_sending_a_message.asp

5. PRB: Microsoft Exchange 2000 Server Service Pack 3 Security Modification and CDOEX/CDOSYS

support.microsoft.com/@kbid=324037

6. Sending Email Via an External SMTP Server Using CDO

www.asp101.com/articles/john/cdosmtprelay/default.asp

ایجاد قالب برای صفحات وب در ASP.NET[1]

در این مقاله به بررسی روش های ایجاد قالب (template) برای صفحات وب پرداخته می شود .

منظور از قالب سازی برای صفحات چیست؟ (مقایسه با روش های دیگر ...)

قالب سازی بدین معنی است که با ایجاد یکسری نقاط مشترک برای تمامی صفحات بصورت جداگانه و قرار دادن نوعی ارجاع از آن نقاط در تمامی صفحات وب، یک نوع مدیریت جامع را به ساختار وب سایت تحمیل کنیم. تعریف ذکر شده کمی گنگ و پیچیده به نظر می آید. بنابراین با ذکر مثالی سعی در باز کردن مساله فوق می نمایم.

فرض کنید که قرار است سایتی را با بیش از ده صفحه ایجاد نمایید .مسلما برای ایجاد این سایت طرحی را درنظر می گیرید. بطور مثال اینچنین فرض می کنید که آرم و نام سایت را در قسمت بالا قرار میدهید، منوهای اصلی سایت را در سمت راست، در پایین موارد مربوط به کپی رایت، در چپ تبلیغات سایت و نهایتا فضای باقیمانده را به قسمت اصلی سایت (که از این پس آنرا Main Part می نامیم) یعنی مکانی برای نمایش اطلاعات و عملکرد سایت خود اختصاص می دهید. مسلما تمامی قسمت های ذکر شده در تمامی صفحات ثابت هستند (یا تقریبا ثابت هستند) و فقط فضای باقیمانده (Main Part) که معمولا بیشترین فضای صفحه را به خود اختصاص می دهد پویا می باشد، که در این مکان با توجه به درخواست و عملکرد کاربر اطلاعاتی نمایش داده می شود. حالت معمول طراحی بدین گونه هست که یکبار شمای ذکر شده طراحی و سپس در تمامی صفحات قرار می گیرد. پس از آن در هر صفحه با توجه به نیاز آن، مواردی به Main Part آن اضافه می شود (مواردی چون datagrid، buttons،forms، انواع کنترلهای وب، متن های مورد نیاز صفحه و...).

[1] محمود مروج

حال فرض کنید که قرار شود یک منوی جدید به لیست منوها اضافه گردد .بنابراین مجبور هستیم که این منو جدید را در تمامی ده صفحه فوق اضافه کنیم. شاید در نگاه اول این مساله مهمی نباشد ولی اگر صفحات فوق به صد صفحه و یا بیشتر افزایش یابد، دیگر این مساله کوچک نیست.

استفاده از فریم(Frame)ها به عنوان اولین راه حل

شاید خیلی ها به فریم های موجود در HTML متوسل شوند، که این به نوبه خود اگرچه مفید هست ولی سختی ها و کاستی های خود را دارد. در استفاده از فریم ها نیاز به دانشHTML ، آشنایی با مدل شی سند (DOM)و همچنین تا حدودی نیاز به JavaScript می باشد. در ضمن در حالت های پیشرفته تر نیاز بیشتری به موارد فوق است و این در حالی است که یک ساختار اجباری را به سایت نیز تحمیل می کند. بطور مثال اسکرولهای هر فریم جداگانه می باشد و اغلب در بسیاری از حالتها نحوه نمایش صفحه را از یک حالت پیوسته به صورتی گسسته درمی آورد. به عنوان نمونه در نظر بگیرید منوهای سمت راست صفحه در یک فریم راست و صفحه اصلی (Main Partدر فریم سمت چپ تعریف شود. حال در صورتی که منوها و متن صفحه اصلی از اندازه صفحه خارج شوند، نیاز به اسکرول پیدا می کنند و در نتیجه در صفحه دو اسکرول یافت می شود که این ساختار معمولا کاربر پسند نیست. همچنین در Address bar مربوط به اکسپلورر فقط آدرس فریم دربرگیرنده ظاهر می شود، که در صورت فشرده شدن دکمه refresh اکسپلورر، دوباره به صفحه اول باز می گردد که این به نوبه خود یک فاجعه در navigating می باشد!

پس راه حل چیست؟

خوب مسلما اولین راه حلی که از دید آشنایان با ASP.NET بنظر می رسد، استفاده از کنترلهای کاربر یا همان User controls می باشد. مسلما این یک راه حل بسیار خوب و جامع است ولی معایبی دارد که آنرا نیز می توان با اعمال روشهایی خاص برطرف ساخت. از آنجایی که راه حل اصلی مورد نظر در این مقاله بر پایه همین مدل

است، ابتدا به بررسی این روش پرداخته و سپس آنرا کامل می کنیم. بنابراین برای درک مرحله بعد نیاز است که این مرحله بخوبی درک شده باشد. در ضمن در کدهای زیر از روش code behind استفاده شده است تا مطلب واضح تر باشد.

قالب سازی به کمک User Control ها

نحوه عمل

در این روش بدین صورت عمل می شود که قسمتهای یکسان موجود در تمام صفحات در قالب user controlها تعریف می شوند و سپس این کنترلها در تمامی صفحات قرار داده می شوند. برای بررسی این روش به یک مثال نیاز داریم. در این مثال بدین گونه فرض می شود که صفحه وب ما شامل سه قسمت می باشد:

(1)مکان لوگو	
(2)منوها	(3)قسمت اصلی(Main Part)

و با انتخاب هر منو متن قسمت ۳ یا همان Main Part تغییر می کند.

طراحی بدین صورت شکل می گیرد که به جز قسمت ۳ که همیشه غیر ثابت و متغیر هست، قسمت های ۱ و ۲ بصورت User Control (که از اینجا به بعد به منظور خلاصه سازی با نام اختصاری UC نامبرده می شود) تعریف و استفاده می شوند. این قسمت ها را پنل (Panel)نیز می گویند. پنل ۱ را به صورت زیر تعریف می کنیم: (فایل با نام Panel_logo.ascx)

```
---Panel_logo.ascx file---
```

```
<%@ Control Language="vb" AutoEventWireup="false" Inherits="site.Panel_logo"
TargetSchema= "http://schemas.microsoft.com/intellisense/ie5" %>

<IMG height="74" src="images/logo.gif" width="370" vspace="2" border="0">
```

این تعریف کنترل panel_logo می باشد. در ضمن در این قسمت از هیچگونه برنامه نویسی استفاده نشده است.

نکته مهم :باید توجه کنید که به هیچ وجه درUCها از فرمهای با ویژگی runat=server استفاده نشود. اگرچه این کار مجاز است ولی عملا کاربرد کنترل شما را بسیار محدود می کند. توجه کنید، کنترلهایی که بدین صورت ایجاد می گردند در هنگام ترجمه کدهای HTML آن بصورت مستقیم در مکان فراخوانی کنترل قرار خواهند گرفت.

حال کنترل دوم را ایجاد می کنیم. این کنترل را بصورتی ایجاد می کنیم که با انتخاب هر منو، پس از رفتن به صفحه مربوطه، منوی مربوط به آن صفحه غیر فعال شود:

```
---Panel_right.ascx file---

<%@ Control Language="vb" AutoEventWireup="false"
Codebehind="Panel_right.ascx.vb" Inherits="site.Panel_right" TargetSchema=
"http://schemas.microsoft.com/intellisense/ie5" %>

<table >
  <tr>
    <td id="td1" runat="server"><A href="index.aspx"> Home Page </A></td>
    <td id="td2" runat="server"><A href="search.aspx"> Searching </A></td>
    <td id="td3" runat="server"><A href="AboutUs.aspx"> About Us </A></td>
```

```
  </tr>
</table>
```

همانطور که دیده می شود در این پنل سه لینک به سه صفحه متفاوت موجود است. همچنین این سه لینک هر کدام در یک تگ <td> با ویژگی runat=server قرار داده شده اند، تا بتوان از آنها در Codebehind مربوطه استفاده نمود. توجه کنید که قرار است با رفتن به هر صفحه، به این کنترل بفهمانیم که در کدام صفحه است و کدام لینک را باید غیرفعال کند. پس نیاز است که یک property برای این کنترل معرفی شود که این کار را برای ما انجام دهد. بنابراین در Codebehind مربوط به پنل راست داریم:

```
---Panel_right.ascx.vb file---

Protected WithEvents td1 As System.Web.UI.HtmlControls.HtmlTableCell
Protected WithEvents td2 As System.Web.UI.HtmlControls.HtmlTableCell
Protected WithEvents td3 As System.Web.UI.HtmlControls.HtmlTableCell
```

که باعث معرفی td1,td2,td3 در کد می شود. سپس property مربوطه به صورت زیر تعریف می شود:

```
Public WriteOnly Property current_page() As Integer
  Set(ByVal Value as Integer)
    Select Case Value
      Case 1
        td1.InnerHtml = "Home Page"
      Case 2
        td2.InnerHtml = "Searching"
      Case 3
        td3.InnerHtml = "About Us"
```

```
    End Select
  End Set
End Property
```

بدین ترتیب مثلا اگر خصیصه current_page به مقدار ۲ تنظیم شود، به جای لینکsearching ، فقط متن آن قرار خواهد گرفت که نشانگر غیرفعال بودن آن لینک و یا به عبارتی نشانه قرار داشتن در صفحه searching می باشد.

هم اکنون جزئیات قالب ما معین است و فقط کافی است که شکل صفحه قالب را معین کنیم. صفحه قالب را با نام template.aspx ایجاد و از آن استفاده می نماییم:

```
---template.aspx---
<%@ Register TagPrefix="Mysite" TagName="Panel_right" Src="Panel_right.ascx"
%>
<%@ Register TagPrefix="Mysite" TagName="Panel_logo" Src="Panel_logo.ascx" %>
<%@ Page Language="vb" AutoEventWireup="false" Codebehind="template.aspx.vb"
Inherits="site.template"%>
<HTML>
<head> My site </head>
<body>
<table>
<tr>
<td colspan="2">
<mysite:panel_logo id="panel_logo1" runat="server"></mysite:panel_logo>
</td>
</tr>
```

```
<td>متن و کنترلهایی که مخصوص هر صفحه هست و جداگانه تعریف می شوند در اینجا قرار] [می گیرند (Main
Part)
</td>
<td>
<mysite:panel_right id="panel_right1" runat="server"></mysite:panel_right>
</td>
<tr>
</tr>
</table>
</body>
</HTML>
```

بدین ترتیب قالب ما ساخته می شود و برای ایجاد هر صفحه جدید کافی است کدهای فوق (البته به غیر از directiveمربوط به page که باید مخصوص همان صفحه باشد)را در آنجا قرار داد. نمونه ساده ای از استفاده این قالب به صورت زیر است: (بطور مثال برای صفحه(aboutus.aspx)

```
---aboutus.aspx---
<%@ Register TagPrefix="Mysite" TagName="Panel_right" Src="Panel_right.ascx"
%>
<%@ Register TagPrefix="Mysite" TagName="Panel_logo" Src="Panel_logo.ascx" %>
<%@ Page Language="vb" AutoEventWireup="false" Codebehind="aboutus.aspx.vb"
Inherits="site.aboutus"%>
<HTML>
<head> My site </head>
<body>
<table>
```

```
<tr>
<td colspan="2">
<mysite:panel_logo id="panel_logo1" runat="server"></mysite:panel_logo>
</td>
</tr>
: 1-...2-...درباره ما... سابقه کاری شرکت ما به صورت زیر می باشد<td>
</td>
<td>
<mysite:panel_right id="panel_right1" current_page="3"
runat="server"></mysite:panel_right>
</td>
<tr>
</tr>
</table>
</body>
</HTML>
```

همانطور که در این کد ملاحظه می شود، directiveمربوط به Page تغییر کرده است، متن قسمت "صفحه اصلی (Main Part) "نیز با متن مورد نیاز در این صفحه تنظیم شده و نهایتا ویژگیcurrent_page ، از کنترل panel_right، با مقدار ۳ تنظیم شده که باعث می شود لینک سوم در هنگام بازدید از این صفحه غیرفعال گردد. توجه کنید که تغییرات فوق فقط مختص این صفحه است و به هیچکدام از صفحات دیگر در ارتباط نیست .بنابراین اصل قالب بندی به هیچ وجه زیر سوال نمی رود. حال برای درک بیشتر این قضیه فرض کنید که ۲۰۰ صفحه با قالب فوق ایجاد کردید و حال تصمیم می گیرید که در پنل مربوط به لوگو مکانی را برای قرار دادن یک عکس تبلیغاتی اختصاص دهید. فقط کافی است که بدرون کنترل panel_logo یک خط بصورت زیر اضافه شود:

```
<IMG height="74" src="images/adv.gif" width="570" vspace="2" border="0">
```

همین!... با تغییر فوق این تغییر را تمامی صفحات مشاهده خواهید کرد و دیگر لازم نیست که این خط را به تمامی صفحات اضافه و از درست عمل کردن آن مطمئن شوید. و یا اینکه تصمیم می گیرید یک منوی جدید به لیست منوها اضافه کنید. همانند مورد فوق خط مشابهی را در صفحه panel_right.ascx اضافه کرده و یک case دیگر نیز به دستور select در code behind مربوطه اضافه می کنید و دیگر تمام! حتی لازم نیست current_page مربوط به صفحات دیگر را تغییر دهید. همچنین شما می توانید به کنترلهای خود ویژگی visible را هم اضافه نمایید تا در صفحات مورد نیاز از نمایش بعضی پنل ها(مانند پنل تبلیغات) جلوگیری به عمل آورید. چیزی که اجرای آن با فریم ها بسیار مشکل و دردسر ساز می باشد. همانطور که دیده می شود، اصول شئ گرایی تماما در این روش رعایت شده است.

در این روش شما می توانید با استفاده از caching کنترلها، سرعت و کارایی برنامه را تا حد قابل توجهی بالا ببرید .

همچنین در موارد پیشرفته تر و تعداد صفحات بالا مشاهده خواهید کرد که:

- از حجم برنامه شما بسیار کاسته می شود.
- خوانایی کدهای HTML شما افزایش می یابد و درنتیجه ایراد زدایی از کدهای HTML نیز بسیار آسانتر خواهد شد.
- سرعت کار و تمرکز شما نیز بطور قابل توجهی افزایش می یابد.

روش ذکر شده بسیار خوب و کامل است. ولی این روش ایراداتی دارد که پس از اشاره به آنها، به نحوه رفع آن با روش دوم خواهیم پرداخت:

فرض کنید پس از مدتی تصمیم می گیریم که ظاهر سایت را تغییر دهیم .مثلا یک پنل تبلیغات به سمت چپ اضافه کنیم. اگر دقت کنید، با مشکلی همانند اولین مشکل ذکر شده در مقاله برخورد خواهیم کرد. در اینجا دیگر پنلی به این نام در صفحات ما وجود ندارد و ما مجبور خواهیم بود که این پنل را ایجاد و در تمامی صفحات قرار دهیم. و این در حالی است که ساختار HTML موجود در صفحات بهم نخورد. اغلب ساختار HTML صفحات بسیار پیچیده تر از مثال فوق است و بنابراین نیاز است که در کدهای HTML تمامی صفحات نیز بازنگری شود و این نیز بر مشکل اول افزوده می شود. مشکل اساسی آنجا آغاز می شود که قصد کنیم ظاهر سایت را کاملا تغییر داده و کل قالب را عوض کنیم. اگر چه شاید این مشکل مانند مشکلات مدل ایجاد صفحات بدون قالب نباشد، ولی حداقل نیاز است که این تغییرات برای تمامی صفحات موجود اجرا شود و قالب جدیدی نیز ساخته شود .پس از آن است که دیگر نیاز به تغییرات نیست، ولی اگر دوباره نیاز به اینچنین تغییری پیدا شد چه؟... مسلما این راه حل کاملا پاسخگو نیست .

در راه حل زیر مشکل فوق بررسی و برطرف می شود:

قالب سازی پویا بوسیله سگمنت کردن قالب اولیه

اگر در روش قبلی دقت کرده باشید، مشکل از وجود کدهای HTML موجود درقالب بود و اگرچه در تمامی صفحات ثابت بودند ولی با اندکی تغییر در ساختار صفحه نیاز به تغییر آن و در نتیجه تغییر قالب و نهایتا قالب گیری مجدد صفحات ساخته شده است. اگر بتوان بصورتی عمل کرد که قالب اصلی ما همیشه شکل ثابتی داشته باشد و بهیچ وجه تغییر نکند، مشکل برطرف می شد. به عبارتی با قالب سازی برای قالب اولیه و یا به عبارتی پویا سازی قالب، دیگر و در هیچ شرایطی نیاز به تغییر قالب مورد استفاده صفحات را نداشتیم و با تغییر شکل قالب، خودبخود در تمامی صفحات نیز تغییرات صورت می گرفت. قبل از بررسی روش مورد نظر توجه شما را به نکته ای جلب می کنم:

همانطور که احتمالا همه شما تا کنون متوجه شده اید، هر سایت مجموعه ای از پنل های فرعی و تقریبا ثابت، و یک قسمت اصلی (Main Part) که حاوی مطالب صفحه مورد نظر است می باشد. برای درک بیشتر این موضوع

همین الان به چند سایت مختلف نگاه کنید...، بطور مثال همین سایت IranASP.NET شامل دو پنل چپ و راست و یک پنل بالا و پایین است. قسمت وسط آن نیز، قسمت اصلی صفحه (Main Part) را تشکیل می دهد. در سایت یاهو نیز در صفحه مربوط به میل آن یک پنل در چپ، یک پنل در بالا و یک پنل تبلیغات در بالاترین نقطه وجود دارد. در قسمت راست صفحه هم اصلی ترین قسمت (Main Part)وجود دارد. می توان گفت که این قسمت یا همان Main Partبیشترین فضای صفحه را در تمامی سایتها اشغال می کند.

پس هم اکنون به این امر واقف هستید که "قسمت اصلی" جزو لاینفک هر سایت هست و کلیه تغییرات در سایت حول این قسمت می باشد.

برای توضیح این روش از مثال ذکر شده در روش قبل استفاده می کنیم و آنرا تکمیل می نماییم. اکنون همانند مثال ذکر شده در روش قبل یک قالب ایجاد کنید .مکانی که باید کدهای قسمت اصلی در آنجا قرار بگیرند را در نظر بگیرید. از همین مکان کدها را به دو قسمت بالایی و پایینی تفکیک کنید. یعنی کدهای HTML ای که بالای قسمت Main Partقرار دارند به صورت زیر خواهد بود:

```
<%@ Register TagPrefix="Mysite" TagName="Panel_right" Src="Panel_right.ascx"
%>
<%@ Register TagPrefix="Mysite" TagName="Panel_logo" Src="Panel_logo.ascx" %>
Inherits="site.template"%>
<HTML>
<head> My site </head>
<body>
<table>
<tr>
  <td colspan="2">
```

```
    <mysite:panel_logo id="panel_logo1" runat="server"></mysite:panel_logo>
  </td>
</tr>
  <td>
```

و بلافاصله پس از خط آخر کدهای قسمت اصلی قرار دارند:

```
[Codes that are for main part of the page, must put here.]
```

و بلافاصله پس از این کدها، کدهای HTML مربوط به ادامه قالب قرار دارند:

```
      </td>
  <td>
    <mysite:panel_right id="panel_right1" runat="server"></mysite:panel_right>
  </td>
<tr>
</tr>
</table>
</body>
</HTML>
```

هم اکنون این دو قسمت بالایی و پایینی را در دو کنترل جداگانه تعریف می کنیم. اولین کنترل را سگمنت بالایی می نامیم(segment_top.ascx) :

```
    ---segment_top.ascx---
<%@ Register TagPrefix="Mysite" TagName="Panel_logo" Src="Panel_logo.ascx" %>
<%@ Control Language="vb" AutoEventWireup="false"
```

```
Codebehind="segment_top.ascx.vb" Inherits="site.segment_top" TargetSchema=
"http://schemas.microsoft.com/intellisense/ie5" %>
<HTML>
<head> My site </head>
<body>
<table>
<tr>
  <td colspan="2">
    <mysite:panel_logo id="panel_logo1" runat="server"></mysite:panel_logo>
  </td>
</tr>
  <td>
```

از آنجایی که کد HTML فوق شامل پنل راست نیست، directiveمربوطه را نیز از آن حذف می کنیم.

همچنین این قمست دارای code behind خاصی نیست و از آن صرف نظر می کنیم.

کنترل دوم را سگمنت پایینی می نامیم و شامل کدهای قسمت پایینی قالب است(segment_down.ascx) :

```
---segment_down.ascx---
<%@ Register TagPrefix="Mysite" TagName="Panel_right" Src="Panel_right.ascx"
%>
<%@ Control Language="vb" AutoEventWireup="false"
Codebehind="segment_down.ascx.vb" Inherits="site.segment_down" TargetSchema=
"http://schemas.microsoft.com/intellisense/ie5" %>
  </td>
  <td>
    <mysite:panel_right id="panel_right1" runat="server"></mysite:panel_right>
```

```
  </td>
<tr>
</tr>
</table>
</body>
</HTML>
```

همانطور که دیده می شود، directiveحذف شده در این سگمنت آورده شده است.همچنین کدهای HTML نیز در هیچکدام از دو کنترل فوق بصورت کامل نمی باشند (به عبارتی well formed نیست) و در واقع بصورت تنها، اشتباه و بدون کاربرد می باشد .بنابراین توجه داشته باشید که این دو سگمنت با هم بکار برده می شوند. همچنین VS.NETدر بعضی شرایط خود اقدام به تکمیل این کدها می کند که در صورت مشاهده این عمل، خود اقدام به حذف کدهای اضافی نمایید.

انجام یک کار در هنگام ایجاد سگمنت ها مورد اجباری است و آن اینکه از آنجا که پس از استفاده از صفحه قالب ایجاد شده، امکان دسترسی به خصوصیات کنترلهای موجود در این سگمنت ها نیست، باید این خصوصیات را به این سگمنت ها انتقال داد. بطور مثال در پنل راست خصیصه ای بود که با آن صفحه جاری را معین می کردیم. حال این خصیصه را به صورت زیر به عنوان یک خصیصه در سگمنت پایینی (چون این کنترل در این سگمنت قرار دارد) تعریف می کنیم. ابتدا باید کنترل panel_right را بصورت زیر در درون codebehind تعریف کنیم، زیرا بصورت خودکار این تعریف ایجاد نمی شود.

```
---segment_down.ascx.vb---

Protected WithEvents Panel_right1 As Panel_right
```

سپس این خصیصه را به کنترل سگمنت پایین اضافه می کنیم:

```
WriteOnly Property current_page() As Integer
   Set(ByVal Value As Integer)
      Panel_right1.current_page = Value
   End Set
End Property
```

بدین ترتیب این خصیصه را به کنترل پایینی اضافه کرده ایم. حال صفحه قالب ما در تمامی حالات فقط از شکل زیر برخوردار است:

```
---template.aspx---

<%@ Page Language="vb" AutoEventWireup="false" Codebehind="template.aspx.vb"
%>
<%@ Register TagPrefix="mysite" TagName="segment_down"
Src="segment_down.ascx" %>
<%@ Register TagPrefix="mysite" TagName="segment_top" Src="segment_top.ascx"
%>
<mysite:segment_top id="Segment_top1" runat="server"></ mysite:segment_top>
<mysite:segment_down id="Segment_down1" runat="server"></
mysite:segment_down>
```

فقط اینکه در صفحات مورد استفاده از قالب ها، page directive باید مربوط به خود آن صفحه باشد.

حال به عنوان نمونه صفحه aboutus.aspx را با این قالب جدید پیاده سازی می کنیم:

```
---aboutus.aspx---
```

```
<%@ Page Language="vb" AutoEventWireup="false" Codebehind="aboutus.aspx.vb"
%>
<%@ Register TagPrefix="mysite" TagName="segment_down"
Src="segment_down.ascx" %>
<%@ Register TagPrefix="mysite" TagName="segment_top" Src="segment_top.ascx"
%>
< mysite:segment_top id="Segment_top1" runat="server"></ mysite:segment_top>
درباره ما... سابقه کاری شرکت ما به صورت زیر می باشد: ۱-...۲...-
< mysite:segment_down id="Segment_down1" runat="server" current_page=" 3" ></
mysite:segment_down>
```

همانطور که ملاحظه می شود، متن قسمت اصلی در میان دو سگمنت قرار می گیرد. همچنین مقدار خصیصه current_pageنیز در سگمت پایینی معین شده است .

حال فرض کنید می خواهیم یک پنل تبلیغات در سمت چپ قرار دهیم. فقط کافی است در سگمنت بالایی کدهای زیر را در انتهای آن اضافه کنیم: (فرض کنید کنترل تبلیغات به اسم panel_adv.ascx طراحی شده است)

```
<%@ Register TagPrefix="Mysite" TagName="Panel_adv" Src="Panel_adv.ascx"
%>
<mysite:panel_adv id="panel_adv1" runat="server"></myite:panel_adv>
```

و فقط همین! دیگر نیازی به تغییر در هیچکدام از فایلها نیست و صفحه" درباره ما" با شکلی جدید به نمایش خواهد آمد. در این شرایط اگر ظاهر سایت نیز به کلی تغییر کند طوری که تمام کدهای HTMLعوض گردند، حداکثر تغییر در کنترل های سگمنت بالایی و پایینی است. یعنی حداکثر کدهای HTML این دو تغییر خواهند کرد و تمامی صفحات از قالب جدید پیروی خواهند کرد .

در دو طراحی آخر فقط به این مورد می توان اشاره کرد که در صورت استفاده از قالب بندی به یکی از این دو روش، در هر صورت تمامی فایلهای سایت شما باید تبدیل به aspx شوند. شاید این از نظر برخی ناخوشایند باشد. بطور مثال تمامی صفحات با محتوای ثابت که می توانند به صورت HTML نیز باشند، باید به صورت aspx درآیند و ازآنجا که هیچ برنامه نویسی سمت سرور در این فایلها نیست، در واقع کار بی موردی انجام داده ایم و به سرور بار اضافی تحمیل کرده ایم. خوب این بستگی به طراح سایت دارد که کدام مدل برای وی بیشتر به صرفه باشد. چیزی که اینجا مشخص است، برای سایت های با حجم و ترافیک محدود شاید چند صفحه ثابت ارائه شده بصورت aspx آنچنان بر عملکرد سایت موثر نباشد. در ضمن قابل ذکر است که این یک مدل و شیوه است و اجباری بر اجرای آن بر روی تمامی صفحات نیست.

در آخر به عنوان نمونه ای از کارهای انجام شده با این روش می توان به سایت www.hudhud-dubai.com و www.irantbc.com (بزودی) اشاره کرد.

تازه های NET Framework 1.1.[1]

دات نت نسخه1.1 ، توانسته است نسخه قبلی را با ارائه قابلیتهای جدید، بهبود موارد موجود و افزایش حجم و کیفیت مستندات، توسعه دهد .

دات نت نسخه ۱,۱، توانسته است نسخه قبلی را با ارائه قابلیتهای جدید، بهبود موارد موجود و افزایش حجم و کیفیت مستندات، توسعه دهد.

پشتیبانی محلی از برنامه‌های دستگاههای موبایل

پیش از این در دات نت جهت ایجاد برنامه های قابل اجرا برروی دستگاههای موبایل مانندPDA ، گوشی های تلفن موبایل و غیره شما مجبور بودید که علاوه بر موتور اصلی دات نت یعنی.NET Framework 1.0 ، بسته نرم افزاری دیگری بنام Microsoft Mobile Internet Toolkit را برروی کامپیوتر خود نصب نمائید تا قابلیت ایجاد چنین برنامه هایی و استفاده از کنترلهای مخصوص این نوع برنامه ها برای شما میسر گردد. هم اکنون با ظهور دات نت نسخه ۱,۱ شما دیگر نیازی به نصب این بسته نرم افزاری نداشته بلکه قابلیت نوشتن برنامه های تحت موبایل از ابتدا در این نسخه جدید و همچنین نسخه جدید Visual Studio .NET 2003 دیده شده است.

مدیریت نسخه‌های متفاوت: اجرای پهلو به پهلو

پشتیبانی از اجرای پهلو به پهلو یا Side-by-Side در دات نت نسخه ۱,۱ مدیران سیستم را قادر می سازد تا نسخه های متفاوتی از برنامه ها و کامپوننت(Component)های مختلف را همزمان بر روی یک کامپیوتر نگهداری و اجرا نمایند .(Version Control)بعبارت دیگر شما می توانید علاوه بر داشتن نسخه های متفاوت دات نت، نسخه های متفاوت برنامه هایی که از هر یک از نسخه های گوناگون دات نت استفاده می کنند را در هر لحظه برروی یک

[1] مدیریت سایت

کامپیوتر داشته باشید. دقت داشته باشید که این بدین معنی نیست که هر نسخه از برنامه های موجود می تواند برروی هر نسخه از دات نت اجرا شود، بلکه هر برنامه یا هر نسخه تنها می تواند با نسخه ای از دات نت کار کند که از قبل برای آن تنظیم شده است. این مساله می تواند توسط مدیر سیستم از طریق فایلهای ساختاربندی تنظیم گردد . جهت اطلاع بیشتر لینک زیر را را ببینید.

msdn.microsoft.com/netframework/productinfo/versioncomparison/default.asp

توانایی اجرای اسمبلی‌های Windows Forms با مبدا اینترنت

در دات نت ۱,۱ می توان اسمبلی Assembly)یا همان DLL سابق)های مربوط به Windows Forms را که دارای مبداء اینترنتی هستند را اجرا نمود. بعنوان مثال کنترلهای Microsoft Windows Forms که در درون یک برنامه تحت وب (Web based) یا اسمبلی های Windows Forms که برروی یک میزبان وب قرار دارند را می توان از راه دور استفاده و اجرا کرد.

توانایی تعیین امنیت دسترسی کد برای برنامه هایASP.NET

مدیران سیستم از این پس می توانند با استفاده از قابلیت "امنیت دسترسی کد" یا Code Access Securityموارد امنیتی بیشتری را برای برنامه های ASP.NET و سرویس های وب (Web Services) تعیین نمایند. اگرچه کلیه موارد امنیتی برای برنامه ها از طریق کاربر اختصاصی سیستم عامل که برنامه ها تحت لوای آن اجرا می شوند، تعیین می گردد، اما قابلیت "امنیت دسترسی کد" مربوط بهCLR ، مدیران سیستم را قادر می سازد تا موارد خاص امنیتی بیشتری را بطور جداگانه برای برنامه های مختلف اعمال نمایند. از این قابلیت می توان بخصوص در سرورهای مشترک مانند آنچه که درISP ها و میزبانهای وب موجود است استفاده شود.

پشتیبانی محلی جهت ارتباط با پایگاه داده‌های ODBC وORACLE

تا قبل از دات نت نسخه ۱,۱ برنامه نویسان جهت استفاده از راه انداز(Data Provider - Driver)های ارتباط با پایگاه داده های ODBC و ORACLE می بایست ابتدا بسته های نرم افزاری مربوطه را بصورت جداگانه از طریق وب دریافت و سپس نصب می کردند. اما همینک این دو راه انداز بهمراه بسته نرم افزاری دات نت ۱,۱ ارائه می شود. استفاده کنندگان از ODBC باید راه انداز خود را در نامکده(Namespace)ای بنام System.Data.Odbc یافته و استفاده کنندگان ORACLE از نامکده System.Data.OracleClient بهره می برند.

مدل برنامه نویسی یکپارچه برای برنامه‌هایSmart Client

Microsoft .NET Compact Framework همانگونه که از نامش پیداست نسخه فشرده ای از دات نت است که برای برنامه نویسی دستگاههایی که اصطلاحا Smart Client نامیده می شوند، ارائه شده است. این دستگاهها عبارتند از Pockect PC 2002 Phone Edition ، Pocket PC 2000/2002و سایر دستگاههایی که براساس استفاده از سیستم عامل Windows CE .NET 4.1 طراحی شده اند (نسخه های قدیمی تر Windows CE .NETپشتیبانی نشده اند).

این نسخه فشرده دات نت یعنی Microsoft .NET Compact Framework دارای مواردCLR ، کنترلهای Windows Forms و سایر متعلقات و امکانات مربوط به دستگاههای کوچک یا اصطلاحا باهوش می باشد. همچنین دارای زیرمجموعه بزرگی از کتابخانه کلاس های دات نت می باشد که برای دستگاههای هوشمند بهینه شده است.

دقت داشته باشید که نسخه فشرده دات نت بهمراه نسخه اصلی دات نت ارائه نشده بلکه می توانید آن را بصورت جداگانه دریافت کرده یا از طریق Visual Studio .NET 2003 از آن استفاده نمائید.

پشتیبانی از پروتکل اینترنت نسخه ۶ یاIPv6

دات نت نسخه 1.1 بطور کامل از آخرین و جدیدترین پروتکل ارتباطی اینترنت که به نسخه ۶ معروف می باشد و IPv6 نامیده می شود، پشتیبانی می کند. این پروتکل بدین منظور طراحی شده است تا با استفاده از آن بتوان فضای بیشتری در اینترنت را در آینده آدرس دهی نمود.

افزایش ثبات و کارآیی، مستندات بیشتر

علاوه بر موارد مطرح شده در بالا، جهش قابل توجهی از نقطه نظر ثبات (scalibility) و کارآیی (performance)در دات نت نسخه ۱,۱ نسبت به نسخه قبل از آن به چشم می خورد. ارتقاء نسبتا خوبی هم در مستندات دات نت شامل نمونه برنامه های بیشتر و بخش های جدید (شامل راهنمای Secure Code) انجام شده است.

منبع What's New in the .NET Framework 1.1 :

msdn.microsoft.com/netframework/productinfo/overview/whatsnew.asp

۱۰ دلیل مهم جهت ارتقاء به Visual Studio .NET 2003[1]

Visual Studio .NET 2003نسبت به نسخه قبلی خود دارای امکانات و قابلیتهای بیشتری می باشد که امر تولید انواع برنامه و سیستم ها را ارتقاء بخشیده است .

۱ -امکان تولید برنامه های قابل اجرا برروی دستگاههای موبایل

با استفاده از VS .NET 2003 می توانید برنامه های قابل اجرا برای دستگاههای موبایل مانند Pockect PC ، Tablet PCو تلفن های موبایل را به آسانی تولید کنید. این برنامه ها را می توانید با Microsoft Visual Basic .NET 2003یا Microsoft Visual C# .NET 2003 بسازید.

۲ -از آخرین نسخه دات نت استفاده کنید

با استفاده از .NET Framework 1.1 یک دات نت کامل تر را خواهید داشت که از لحاظ امنیت و کارآیی نسبت به نسخه قبل بسیار بهتر شده است.

۳ -جادوگر قویتر!

جادوگر یا همان Wizard محبوب برای تبدیل برنامه های Visual Basic 6.0 به Visual Basic .NET اکنون بسیار بهتر و قویتر شده است و از User Controls و کلاسهای وب پشتیبانی می کند.

۴ -بهشت برنامه نویسان!Java

[1] مدیریت سایت

با دارا بودن Visual J# .NET در VS .NET 2003 برنامه نویسان علاقمند به زبان Java می توانند برنامه ها و سرویس های وب خود را به زبان Java با استفاده از دات نت تولید کنند.

۵ Enterprise Instrumentation Framework -یاEIF

VS .NET 2003با ارائه ابزار Enterprise Instrumentation Framework شما را قادر می سازد تا به آسانی قابلیت های دیده بانی (monitoring) را برای زمان اجرا (run-time) به هر برنامه ای اضافه نمائید.

۶ -پشتیبانی از سرویس های XML وب

VS .NET 2003علاوه بر ارتقاء پشتیبانی خود از سرویس های XML وب، از آخرین استانداردهای سرویس های وب شامل WS-Routing ، WS-Security، WS-Attachmentsو DIME حمایت می کند.

۷ -دیباگر بهتر شده

یک دیباگر (debugger) بهتر شده دید بهتری از داده ها و اطلاعات برنامه به شما داده و به شما کمک می کند تا اشکالات و خطاهای برنامه را بهتر یافته و رفع کنید.

۸ ADO.NET -جدید

یک تامین داده (Data Provider) جدید ADO.NET که امکان ارتباط با بانکهای اطلاعاتی Oracle 8i ، Oracle 7iو منبع داده ODBC را به مجموعه قبلی خود افزوده است.

۹ -سایتهای online انجمن دات نت

علاوه بر یک سیستم پویای یکپارچه بهتر شدهHelp ، می توانید از سایتهای انجمن دات نت بصورت online راهنمائی و کمک بگیرید.

۱۰ -محیط تولید یکپارچه (IDE) جدید

محیط برنامه نویسی VS. NET 2003 نسبت به نسخه قبلی آن بسیار بهتر شده است و قابلیت و امکانات بهتر و هوشمندتری را ارائه نموده است که کار برنامه نویسان را برای نوشتن کد برنامه آسانتر می کند.

منبع : Top 10 Reasons to Upgrade to Visual Studio .NET 2003

msdn.microsoft.com/vstudio/productinfo/vstudio03/topten/upgrade.asp

Threadها در ASP.NET[1]

در این مقاله سعی شده است مفهوم Threading و ارتباط آن با ASP.NET و مفهوم Application Domainبیان گردد .

شاید زمانی که شما با برنامه Visual Studio.NET کار می کنید با مواردی مواجه شوید که مفهوم آنها برای شما گنگ یا جدید باشد. یکی از این موارد که بسیاری از برنامه نویسان با آن مواجه شده اند همین مبحث Threadingمی باشد. احتمالا باید دیده باشید که در نامکدهSystem ، نامکده دیگری با نام Threading موجود است که برای کار با این مبحث اختصاص یافته است. یا احتمالا وقتی قصد داشتید که از Help مربوط به VS.NET استفاده نمایید، در مواردی که مربوط به راهنمای کلاسها می باشد، با یک توضیح با عنوان Thread Safety مواجه شده اید.

در این مقاله سعی شده است تا خواننده با مفهوم Thread و مسائل مربوط به آن آشنا شود. در تهیه این مقاله از مستندات شرکت مایکروسافت و تجربیات نویسنده در این زمینه استفاده شده است.

پروسه چیست؟

قبل از آنکه به بررسی مفهوم Thread پرداخته شود لازم است کمی در مورد پروسه ها (Process) توضیحاتی ارایه گردد. هر برنامه ای که شما اجرا می کنید، تحت نام یک پروسه به اجرا در می آید و این پروسه هست که وظیفه کنترل برنامه های شما را دارد. یکی از مزیت های پروسه ها این هست که شما را قادر می سازد چندین برنامه را در کنار هم به اجرا در آورید (بطور مثال VS.NET و MS Access را در یک زمان اجرا کنید و در همان لحظه توسط WinAmpبه موزیک مورد علاقه خود گوش دهید .(!اساس Windows هم در تمامی نسخه ها بر همین مورد

[1] محمود مروج

استوار بوده است. برای دیدن پروسه های در حال اجرا با فشردن Ctrl + Shift + Esc) در ویندوزهای ۲۰۰۰ به بالا(، Windows Task Managerرا احضار نموده و با انتخاب سربرگ Processes قادر خواهید بود که آنها را مشاهده نمایید. دو پروسه آشنا را می توان بصورت زیر نام برد:

- پروسه devnev.exe که مجری VS.NET می باشد.
- پروسه IExplore.exe که مجری IE مرورگر اینترنت اکسپلورر) می باشد.

البته تمام پروسه ها نشان دهنده یک برنامه آشکار نیستند. مثلا تمام desktop شما تحت پروسه ای با نام Explorerاجرا می گردد. اگر باور ندارید بر روی این پروسه کلیک راست گرفته و گزینه End Process را انتخاب نمایید تا ببینید چگونه کل Desktop شما ناپدید می شود!! به هر حال منظور این است که تمامی برنامه ها و کنترلرهای سیستم عامل تحت پروسه های معینی انجام می شوند.

باید خاطر نشان کرد که IIS شما نیز تحت پروسه ای با نام INETinfo.exe اجرا می گردد. این پروسه نیز باید دائما در حال اجرا باشد تا بتواند کلیه درخواستها برای دیدن صفحات وب را دریافت و آنرا مدیریت کند. برای برنامه های ASP.NET هم پروسه ای جدا از پروسه IIS در نظر گرفته شده تا مدیریت بهتری برای برنامه های ASP.NETوجود داشته باشد. این پروسه ASP_wp.exe نام دارد که مخفف ASP Worker Process می باشد. توجه کنید این پروسه با اولین درخواست برای یک صفحه وب ASP.NET ایجاد می شود .همچنین ممکن است بنا به دلایلی این پروسه اصلا اجرا نگردد. خوب علت آن، این می تواند باشد که (بنا به هر دلیلی) قبلا درخواست شده باشد تا برنامه های ASP.NET هم تحت پروسه IIS اجرا گردند (همان پروسه .(INETinfo.exe خوب این فعلا زیاد فرقی نمی کند و فقط مهم این است که بدانید برنامه های وب شما تحت یکی از دو پروسه ذکر شده اجرا می گردند.

Threadچیست؟

اما یکThread چیست؟ در جواب این سوال می توان گفت کهThread ها منابعی هستند که از طرف سیستم عامل به یک پروسه (برنامه (اختصاص داده می شوند تا از آن برای اجرای محتویات خود استفاده نمایند. به بیانی دیگر شاید بتوان گفت کهThread ها یک نوع زیر پروسه هستند. برای روشن شدن تعریف فوق به بیان مثالی می پردازیم (پیشنهاد می شود که این مثال را همزمان انجام دهید تا درک بیشتری نسبت موضوع پیدا کنید).

در ابتدا پنجره مربوط به Processes را در Windows Task Manager نمایان کنید. سپس از منوی Viewگزینه Select Columns را انتخاب نمایید. از پنجره به نمایش در آمده گزینه Thread Count را انتخاب نمایید و سپس دکمه OK را فشار دهید تا به حالت قبل بازگردید. هم اکنون در پنجره مربوط به Processes یک ستون با نام Threads اضافه شده است. خوب حالا چند پنجره اینترنت اکسپلورر باز نمایید. همانطور که گفته شد، اینترنت اکسپلورر تحت پروسه IExplore.exe به اجرا در می آید. حال به Processes در Windows Task Managerباز گردید. مشاهده می کنید که به ازای هر پنجره باز شده از این نوع یک پروسه IExplore.exe نیز ایجاد شده است. بنابراین به این نتیجه می رسیم که پنجره های اینترنت اکسپلورر هر کدام تحت یک پروسه جداگانه مدیریت می شوند. هر کدام از این پروسه ها نیز بنا به نیازشان خود حاوی چند Thread هستند.

خوب حالا پروسه Explorer.exe را بیابید. به مقدار ستون Threads آن نگاه کنید. می بینید که این برنامه مثلا شانزده Thread دارد. خوب این تعداد را به خاطر بسپارید و سپس سعی نمایید که یک صفحه Windows Explorerرا باز نمایید (با کلیک راست بر روی Start و انتخاب گزینه .Explore)اینبار بر خلاف اینترنت اکسپلورر مشاهده می نمایید که هیچ پروسه جدیدی همنام با Explore.exe ایجاد نمی شود! پس این برنامه چگونه اجرا می گردد؟ مگر نه اینکه هر برنامه باید تحت یک پروسه خاص اجرا شود، پس چرا این برنامه تحت هیچ پروسه ای اجرا نمی گردد؟

یکبار دیگر به مقدار ستون Threads مربوط به پروسه Explore.exe نگاهی بیندازید. خواهید دید که این مقدار افزایش یافته است (اغلب به اندازه تعداد پنجره های Windows Explorer ای که باز کرده اید). با سعی مجدد در باز کردن تعداد صفحات بیشتر خواهید دید که این مقدار نیز افزایش می یابد.

بنابراین می توان نتیجه گرفت که برنامه های Windows Explorer همگی تحت یک پروسه خاص با نام Explorer.exeاجرا می گردند و هر نسخه از آن در یک Thread مجزا قرار می گیرد. در نتیجه می توان گفت Threadها نیز شبیه به پروسه ها هستند و مهم تر آنکهThread ها نیز قابلیت اجرا در کنار هم را دارند. بدین ترتیب می توان گفت که می توان برنامه هایی نوشت که خود همزمان چند زیربرنامه مشابه را در آن واحد به اجرا درآورند. مثلا برنامه ای نوشت که در همان لحظه که در حال محاسبه عملیات زمانبر ریاضی است، به در خواستهای کاربر نیز پاسخ دهد. اینکار بدین صورت انجام می شود که محاسبه آن عملیات را به یک Thread غیر از Thread جاری واگذار نماییم. بدین ترتیب مشاهده می شود که در برنامه ها هم قابلیت اجرای همزمان چند عملیات با هم وجود دارد. در ادامه باید گفت که برنامه ها می توانند تعیین کنند که به چند Thread نیاز دارند. این یک درخواست به سیستم عامل است .

در ضمن توجه نمایید که یک Thread مختص به یک برنامه نیست. مثلا نمی توان گفت Thread شماره ۱۲۰ به فلان برنامه تعلق دارد .Thread ها در سطح سیستم عامل آزاد هستند و پس از درخواست به دریافت یک Thread از سوی یک برنامه، توسط سیستم عامل یک Thread آزاد به آن برنامه تعلق می گیرد. پس از آن، تا رها شدن Threadمربوطه، آن Thread متعلق به آن برنامه می باشد و هیچ برنامه دیگری حق استفاده از آنرا ندارد.

رابطه میان پروسه ها، Threadها وASP.NET

هم اکنون که مفاهیم Thread و پروسه ها معین گردید و متوجه شدید که برنامه های وب شما تحت یکی از دو پروسه ASP_wp.exe یا INETinfo.exe اجرا می گردند، به بیان رابطه موجود میان این موارد سیستم عاملی با برنامه های ASP.NET می پردازیم. اما قبل از آن لازم است با مفهوم جدید Application Domain که توسط ASP.NETارایه شده است آشنا گردید.

Application Domainچیست؟

قبل از بوجود آمدن ASP.NET اجرای برنامه های وب بدین صورت بود که همگی برنامه ها تحت یک پروسه خاص اجرا می شدند و هر درخواست برای هر صفحه به یک Thread معین واگذار می شد و پس از اتمام آن، Threadمربوطه برای استفاده های بعدی آزاد می شد. حال اگر در یک وب سرور ۱۰۰۰ سایت قرار داشت، همگی تحت نام یک پروسه به اجرا در می آمدند. در این میان اگر یکی از سایت ها با مشکلی مواجه می شد ممکن بود که نیاز باشد IIS را Restart کرد تا پروسه از اول اجرا شود و این بدین معنی بود که تمام سایتها برای مدتی قادر به سرویس دهی نبودند و این مسلما خوش آیند نمی باشد. بالاخص اینکه با افزایش تعداد سایتها احتمال بروز مشکل بیشتر می شد.

شاید بگویید که چرا باید پروسه IIS را Restart نمود و چرا Thread مربوطه Restart نمی شود؟ در جواب باید متذکر شد که اگرچه Thread ها شبیه پروسه ها هستند ولی قدرت و محدوده عمل آنها با پروسه ها متفاوت است. مسلما یکسری منابع میان Thread های مورد استفاده یک برنامه مشترک هستند و بنابراین در صورت بروز مشکل در آن موارد راهی جز Restart کردن پروسه(برنامه) نیست.

مایکروسافت این مشکل را متوجه شد و آنرا در محیط جدید خود با مفهوم جدید Application Domain حل نمود. Application Domainها در واقع یک ناحیه کاری برای برنامه می باشند. قبل از هرچیز اینکه Application Domainها مخصوص ASP.NET هستند و در دیگر انواع برنامه های NET. مفهومی ندارند . Application Domainدر واقع محلی است که هر سایت تحت آن اجرا می گردد. Application Domainها. مفهوم بسیار مشابه ای با پروسه ها دارند. برای هر Application Domain منابع جداگانه ای در نظر گرفته می شوند. مثلا یک حافظه کاملا جدا از سایر سایت ها Application Domain .ها تحت یک پروسه هستند. حال در صورت بروز یک مشکل برای یک سایت دیگر نیازی نیست که کل پروسه Restart شود.

در ضمن توجه نمایید که هر Application Domain می تواند چندین Thread را در اختیار بگیرد و بدین صورت نیست که یک تناظر یک به یک بین آنها باشد. در کل می توان مطالب فوق را در یک نمودار قرار داد که در بالاترین سطح پروسه ها هستند، سطح میانی Application Domain و پایین ترین سطح Thread ها می باشند.هر سطح می تواند چندین مورد از موارد زیر سطح را شامل گردد:

- **پروسه ها**
- **Application Domains**
- **Threads**

حال که روابط فوق مشخص شد، کمی هم به نحوه ارتباط و چگونگی استفاده ازThread ها در ASP.NET می پردازیم.

اول آنکه شما می توانید معین نمایید که پروسه ASP_wp.exe حداکثر از چند Thread استفاده نماید. البته تنظیم این مقدار فقط در فایل Machine.Config ممکن است و این مقدار در تگ processModel تنظیم می شود .مقدار پیش فرض آن ۲۵ می باشد.

در ادامه سعی می شود که با ذکر مثالی نحوه چگونگی استفاده ازThread ها، با استفاده از Debugger موجود در .NET Framework SDK شرح داده شود.

در اینجا فرض بر این است که شما کار با این Debugger را می دانید .در غیر اینصورت می توانید با مراجعه به بخش QuickStart از سایت ASP.NET نحوه کار با آنرا بیاموزید.

لینک بخش گفته شده:

samples.gotdotnet.com/quickstart/aspplus/doc/debugcomsdk.aspx

ابتدا نیاز است که شما این برنامه را اجرا نمایید. بنابراین:

•فایل DbgCLR.exe را در کامپیوتر خود یافته و آنرا اجرا کنید،

•یک پروژه ASP.NET که بر روی آن کار می کنید را درنظر گرفته و Code Behind صفحه اول (یا هر صفحه دلخواه دیگر) را باز کنید (مثلا فایل index.aspx.vbیا default.aspx.vb و یا هر فایل مورد نظر دیگر...)،

•در ادامه پروسه مجری ASP.NET را باید به Debugger متصل نمایید. همانطور که در ابتدای مقاله ذکر شد، این پروسه یا ASPNET_wp.exe است و یا .INETinfo.exe به هر حال برای متصل کردن آن به Debugger، منوی Tools | Debug Processes را انتخاب کنید و پروسه ذکر شده را از لیست سمت چپ انتخاب و سپس دکمه Attach را فشار دهید. توجه کنید پروسه INETinfo.exe یک پروسه سیستمی است و برای انتخاب

آن باید قبل از آن گزینه Show System Processes را انتخاب کنید تا این پروسه در لیست مربوطه به نمایش درآید. توجه کنید، در صورت عدم وجود پروسهASPNET_wp.exe ، اقدام به انتخاب این پروسه نمایید . هم اکنون Debugger آماده است. برای دیدنThread ها بدین صورت عمل نمایید:

از منویDebug ، زیرمنویWindows ، گزینه Threads را انتخاب نمایید. یا اینکه به صورت مستقیم دکمه های Ctrl + Alt + H را همزمان فشار دهید.همچنین توصیه می شود که برای بهتر دیدن نحوه استفاده از Threadها صفحه ای که قصد debug آنرا دارید، حتما شامل چند خط کد و برنامه باشد .

هم اکنون در ابتدای روال رویداد Page_Load یک Breakpoint قرار دهید تا به محض شروع به بار کردن صفحه، بتوانید اجرای آنرا در دست بگیرید.

هم اکنون همه چیز برای دیدن نحوه کارThread ها آماده است. حالا یک صفحه اینترنت اکسپلورر باز کنید و یک درخواست برای دیدن صفحه مورد نظر (صفحه ای که Code Behind آن در Debugger باز شده است) انجام دهید. خوب به علت وجود Breakpoint در روال رویدادPage_Load ، اجرای برنامه در خط مورد نظر متوقف می شود. در پنجره Threads چه مشاهده می کنید؟... می بینید که یک خط به جدول آن اضافه شده است. این خط حاوی اطلاعات زیر است :

•شماره ID مربوط به Thread در حال اجرا،

•نامThread ، که معمولا <noname> می باشد،

•Thread Locationکه بیانگر محل اجرای کد می باشد. این مقدار معمولا نشاندهنده نام روال در حال اجرا هست (مثلا در قسمت هایی از اجرا نشان دهنده اجرای روال Page_Load است(

- و موارد دیگر...

در هنگام اجرای این صفحه مشاهده می کنید که یک فلش زرد رنگ کوچک مقابل آن Thread قرار دارد که بیانگر این است که هم اکنون این Thread در حال استفاده است. با فشردن کلید F5 اجازه دهید که برنامه ادامه یابد تا از Debugger خارج شده و حاصل آنرا ببینید. خواهید دید که فلش زرد از جلوی Thread مذکور برداشته شده است. این بدین معنی است که Thread مورد نظر رها شده و بدست سیستم عامل بازگشته است. حال توسط دو صفحه مجزای اینترنت اکسپلورر، دو درخواست به همان صفحه بدهید. در این حالت می خواهیم بررسی کنیم که چگونه درخواست های همزمان (یا تقریبا همزمان) مدیریت می شوند.

پس از اینکه دو درخواست را ارسال کردید، به Debugger بازگردید. خوب احتمالا خواهید دید که یک Threadدر پنجره Threads دیده می شود که نشاندهنده اولین درخواست است. با فشردن F11 (یا F10) یک خط یا چند خط آنرا اجرا نمایید، پس از اجرای این چند خط (بستگی به ساختار برنامه شما دارد) مشاهده خواهید کرد که اجرا به ابتدای روال Page_Load بازخواهد گشت!!... در واقع چنین نیست .یک نگاه به پنجره Threads بیندازید. خواهید دید که یک Thread جدید نیز به این لیست اضافه شده است. در واقع هنگامی که اجرا ظاهرا به خط اول بازگشته، چنین نبوده است، بلکه CLR شروع به اجرای درخواست دوم بصورت همزمان نموده است. حال با فشردن کلیدF11 (یا F10) اجازه دهید چند خط دیگر نیز به اجرا درآید. مشاهده می کنید که کنترل به Thread اول بازگشته است. حال همینطور به اینکار ادامه دهید، می بینید که فلش زرد رنگ کوچک موجود در پنجره Threads بطور متناوب بین Thread های موجود درلیست حرکت می کند. میزان اجرای تعداد کدهای موجود در Thread اول نسبت به Thread دوم در یک لحظه از زمان تقریبا برابر است. بدین معنی که به ازای هر یک خط اجرا از Thread اول، تقریبا یک خط از کد موجود در Thread دوم نیز اجرا می گردد. علت آنکه این مقدار تقریبی بیان شده است، این است که بسته به نوع کدهای موجود، ممکن است در یک لحظه چند کد از یک Thread باهم اجرا گردند. به هر حال از این مثال می توان نتایج زیر را بدست آورد:

•هر Thread بیانگر اجرای یک درخواست است. پس می توان گفت که هر صفحه درخواستی از طرف کاربر در یک Thread اجرا می شود. بنابراین هر صفحه در یک Thread اجرا می شود.

•با این مثال باید بصورت کاملا آشکاری درک کرده باشید که واقعا درخواست ها بصورت همزمان انجام میشوند. بسیاری از برنامه نویسان اینگونه فکر می کنند که درخواستها در یک صف قرار گرفته و به ترتیب اجرا می شوند. در جواب باید گفت که اینگونه نیست. در واقع تا جایی که ممکن باشد، درخواست ها بصورت همزمان انجام می شوند، مگر آنکه تعداد درخواست آنقدر زیاد باشد که از حد تعداد Thread های مجاز برای استفاده یک برنامه بگذرد. در این حالت درخواست های بعدی در یک صف قرار می گیرند.

•این روش به صفحات مشابه محدود نمی شود. بدین معنی که هر تعداد درخواست از صفحات متفاوت، در صورت امکان، همزمان انجام می شود .

•شاید با انجام تعداد درخواست های همزمان بیشتر مشاهده نمایید که مابقی درخواست ها در صف قرار گیرند (مثلا فقط چهار درخواست همزمان اجرا گردند). این بهCLR ، سیستم عامل و شرایط حاکم بر سیستم شما باز می گردد.

خوب هم اکنون باید مفهوم همزمانی برای شما مشخص شده باشد. درک این موضوع در مواردی بسیار حائز اهمیت است. احتمالا این سوال برای شما پیش آمده است که با اجرای همزمان ممکن است بسیاری از مسایل با مشکل مواجه شوند .مخصوصا منابع مشترک در برنامه. اول آنکه باید خاطر نشان کرد که ASP.NET در بسیاری از موارد خود این موضوع را مدیریت می کند و شما را درگیر برنامه های Multi-Threading (همین مفهوم استفاده از چند Threadدر یک زمان را گویند) نمی کند .می توان گفت که مایکروسافت در این زمینه بسیار مناسب عمل کرده است، چون کار کردن با برنامه های MultiThreading از مسایل بسیار پیچیده و پردردسر برنامه نویسی است.

به هر حال مواردی هست که نیاز به کنترل دارد. مثلا هنگامی که شما یک شئ را در Application یا Cacheقرار می دهید و این امکان نیز وجود داشته باشد که هر کاربر آنرا تغییر دهد، باید حتما این موارد را پیش بینی نمایید. در غیر اینصورت امکان تغییر همزمان در یک شئ وجود دارد و در بسیاری از موارد باعث خراب شدن شئ شما خواهد شد .

Thread-Safetyچیست؟

احتمالا این کلمه را هنگامی که به راهنمای کلاسهای موجود در .NET Framework مراجعه کردید بسیار مشاهده نموده اید .معمولا محتوی آن بر این مضمون است که:

"کلیه متدهای static (یا shared در VB.NET) برای عملیاتهای MultiThreaded مطمئن و امن است، ولی برای مابقی متدها تضمینی نیست" .

این بدین معنی است که در صورتیکه شما درخواست های همزمان داشته باشید، متدهایی از نوع فوق خود از بروز مشکل جلوگیری می کنند. مثلا اگر متد insert برای یک شی Thread-Safe بود، بدین معنی است که اگر از این متد استفاده شد، و در همان لحظه در یک Thread دیگر قصد بود که delete استفاده شود (برای همان نمونه ایجاد شده از کلاس)، باعث خراب شدن شی نمی شود و این موضوع را کلاس تضمین می کند .همانطور که قبلا ذکر شد، عملیات هایMultiThreaded ، آنهایی هستند که می توانند در یک لحظه از زمان با هم اجرا گردند (نمونه ای که در مثال مربوط به کار با Debugger نشان داده شد، از همین نوع بود، چراکه چند Thread همزمان اجرا می شدند).

در مقاله بعد سعی خواهم کرد که یک نمونه از عملیات هایی که مجبور به کنترل آن هستید و به اصطلاح Thread-Safeنیستد را شرح دهم که با این مساله و نحوه کنترل آن بیشتر آشنا شوید.

در انتها لازم به ذکر است که در این مقاله سعی شد موارد فوق بصورت ساده ای بیان گردند. بنابراین ممکن است که بعضی موارد فوق در واقعیت آنطور که گفته شد نباشند. ولی از آنجایی که این نوع فرض، به اصل موضوع لطمه ای وارد نمی کند و باعث درک صحیح و آسانی نیز می شود، بدین گونه آورده شده است (مانند بسیاری از فرض هایی که در علومی مانند ریاضی و فیزیک در نظرگرفته می شود تا مسائل بهتر و آسانتر حلاجی شوند، و این در حالی است که ممکن است فرض کاملا صحیح نباشد!!).

نام منابعی که در تهیه این مقاله از آن کمک گرفته شد

(همگی از:.NET Framework Developer's Guide)

- Threads and Threading
- Threading
- Thread Synchronization
- MultiThreaded Applications
- Application Domains and Threads

و همچنین کتاب Professional ASP.NET 1.0 از انتشارات.Wrox Press

ساختار یک صفحه ASP.NET[1]

در این مقاله به زبانی ساده با قسمتهای تشکیل دهنده یک صفحه ASP.NET آشنا می شویم .

صفحات ASP.NET به طور کلی دارای چندین بخش اصلی میباشند. این بخش ها را میتوان در هفت مورد بررسی کرد:

۱ -دایرکتیوها

۲ -بلوکهای اعلان کد

۳ -کنترلهای ASP.NET

۴ -بلوکهای پردازش کد

۵ -توضیحات طرف سرور

۶ -دایرکتیو Include طرف سرور

۷ -متن ساده و تگهای HTML

۱ -دایرکتیوها

دایرکتیوها تنظیمات خاصی را برای صفحات aspx و ascx تنظیم میکنند که نحوه کامپایل شدن صفحه را مشخص میکند. دایرکتیوها با بلوک <%@ %> نشان داده میشوند و در هرکجای صفحه قابل تعریف می باشند (معمولاً در ابتدای صفحه آورده میشوند).

[1] احسان پگاه

در Framework دات نت، هشت نوع دایرکتیو وجود دارد که دو نوع Page و Import از بقیه پرکاربردتر هستند. دایرکتیو Page فقط در صفحات aspx قابل استفاده است و برخی خصوصیات صفحه نظیر زبان برنامه نویسی، نوع محتوا، نوع کدپیج، توضیحات مختصر در مورد صفحه، فعال بودن یا نبودن وضعیت ظاهر و... را مشخص میکند. مثلاً خط زیر به کامپایلر ASP.NET میگوید که زبان برنامه نویسی صفحه VB و نوع محتوا، متن/ایکس ام ال است:

```
<%@ Page Language="VB" ContentType="text/xml" %>
```

دو خصوصیت دیگرPage ، قابلیت ردگیری و قابلیت نمایش پیام های خطای زمان اجراست که برای اشکال زدایی از برنامه بکار می آیند. این دو خصوصیت بدین صورت فعال میشوند:

```
<%@ Page Trace="True" Debug="True" %>
```

دایرکتیوImport ، برای وارد کردن یک فضانام به صفحه بکار میرود. به این ترتیب کلیه کلاسها و اینترفیس های آن فضانام در صفحه قابل استفاده میشود. این فضانام میتواند جزیی از Framework دات نت یا یک فضانام تعریف شده توسط کاربر باشد .صورت کلی این دایرکتیو اینگونه است:

```
<%@ Import namespace="value" %>
```

برای اطلاعات بیشتر در مورد دایرکتیوها میتوانید به MSDN .NET Framework General Referenceو به قسمت Directive Syntax مراجعه کنید.

۲ -بلوکهای اعلان کد

این بلوکها برای تعریف متغیرها و متدها، زیربرنامه ها و توابع و به طور کلی برای تعریف منطق یک صفحه ASP.NETبکار میروند. شکل کلی یک بلوک اعلان کد چنین است:

```
<script runat="server" language="codelanguage" Src="pathname">
   Code goes here
</script>
```

تمامی کدهای یک صفحه ASP.NET در داخل چنین بلوکی قابل تعریف اند و اگر آنها را خارج این بلوک تعریف کنید، با پیغام خطا مواجه میشوید. خصوصیت دلخواهLanguage ، زبان برنامه نویسیِ کدِ داخل بلوک را تعیین میکند. دقت کنید که اگر این خصوصیت تعیین نشود، زبان پیش فرض صفحه که توسط دایرکتیو Page مشخص شده، به عنوان زبان برنامه نویسی صفحه در نظر گرفته میشود. در ضمن زبان مشخص شده در تگ Script با زبان مشخص شده در دایرکتیو Page باید یکسان باشند. در صورتیکه هیچ زبانی انتخاب نشده باشد، زبان پیش فرض، ویژوال بیسیک است.

با استفاده از خصوصیتSrc ، میتوان یک فایل بیرونی را که حاوی محتویات بلوک کد است، به صفحه پیوند زد. به این ترتیب کدهای برنامه از آن فایل خوانده میشوند و محتویاتِ دیگر بلوک اعلان کد، نادیده گرفته میشوند. به مثالی در این زمینه توجه کنید.

```
[VB.NET]
<html>
<script language="VB" runat="server">
   Sub EnterBtn_Click(Src As Object, e As EventArgs)
      Message.Text = "Hi " & Name.Text & ", welcome to ASP.NET!"
   End Sub
</script>

<body>
<form runat="server">
```

```
   Enter your name: <asp:textbox id="Name" runat=server/>
   <asp:button text="Enter" Onclick="EnterBtn_Click" runat="server"/>
   <p>
   <asp:label id="Message" runat=server/>
</form>
</body>
</html>
```

۳ -کنترلهای ASP.NET

کنترلهای ASP.NET را میتوان در کنار محتوای HTML بکار برد. تنها الزام در این مورد، قرار دادن آنها بین دو تگ <form/> و <"form runat="server> است. البته در مورد بعضی تگها نظیر <"span runat="server>و <"asp:Label runat="server> میتوان این الزام را نادیده گرفت. یک محدودیت در این زمینه این است که فقط از یک تگ <"form runat="server> میتوان استفاده نمود. در واقع در صفحات ASP.NETامکان گروه بندی به چند فرم وجود ندارد.

۴ -بلوکهای پردازش کد

بلوکهای پردازش کد برای اجرای کد موجود در محتوای متنی یا HTML یک صفحه ASP.NET بکار میروند. دو نوع بلوک پردازش کد وجود دارد که عبارت اند از: کد درون خطی و عبارتهای درون خطی.

کد درون خطی، دستور یا یکسری از دستورات را اجرا میکند. این کد با تگ های <% و %> مشخص میشود. عبارت درون خطی، مقدار یک متغیر یا متد را نشان میدهد. عبارتهای درون خطی با کاراکترهای <%= و %> مشخص میشوند. مثال زیر کاربرد این دو نوع کد را نشان میدهد:

```
<%@ Page Language="VB" %>
<script runat="server">
   dim strMessage as string
   sub page_load
      strMessage="Hello!"
   end sub
</script>
<html>
   <head>
   </head>
   <body>
      <form runat="server">
         The Value Of strMessage is <%= strMessage %>
         <p>
         <% strMessage="Goodbye!" %>
         The Value Of strMessage is <%= strMessage %>
         </p>
      </form>
   </body>
</html>
```

در مثال بالا توجه شما را به دامنه تعریف متغیرهای اعلان شده جلب میکنم.

۵ -توضیحات طرف سرور

ممکن است بخواهیم مانند تمام زبان های برنامه نویسی دیگر، مابین کدهای برنامه نویسی از توضیحاتی جهت مفهوم تر شدن کد، استفاده کنیم. در ASP.NET این توضیحات را در بین تگهای --%> و <%-- قرار میدهیم. در

اینصورت محتوای بین این تگها، چه کد ASP.NET باشد و چه متن ساده، مورد پردازش قرار نخواهد گرفت. یک نکته اینکه نمیتوان چند توضیح طرف سرور را به صورت تودرتو استفاده کرد.

۶ -دایرکتیو Include طرف سرور

با استفاده از این دایرکتیو میتوان محتویات یک فایل مشخص شده را در هر کجای صفحه ASP.NET قرار داد. شکل کلی این دایرکتیو اینگونه است:

```
<!-- #include file | virtual = filename -->
```

میتوان هم از خصوصیت File و هم از Virtual برای دادن مسیر فایل استفاده کرد. اگر از File استفاده کنیم، فایلی که میخواهیم شامل کنیم، نباید در دایرکتوری بالاتر از دایرکتوری فایلی که دستور Include در آن است باشد. اما درVirtual ، مسیر فایل را میتوان هر کجا که هست، به صورت کامل و نسبی وارد کرد. مثلاً اگر فایل ما در دایرکتوری به اسم MyDir که یکی از زیر شاحه های wwwroot است، باشد؛ باید دستور زیر را بکار برد:

```
<!-- #include virtual="/mydir/includefile.aspx" -->
```

دایرکتیو Include قبل از هر کدی در صفحه اجرا میشود، بنابراین برای تعیین مسیر فایلی که میخواهیم شامل کنیم نباید از متغیرها استفاده کنیم.

۷ -متن ساده و تگهایHTML

آخرین ساختاری که میتوان در یک صفحه ASP.NET بکار برد، محتوای HTML است. قسمتهای ثابت صفحه، با همان تگهای معمولی HTML و متن ساده ساخته میشود. در واقع محتوای HTML صفحه ASP.NET

همراه با بقیه کدها کامپایل میشود. محتوای HTML ای با کلاس LiteralControl عرضه میشود. با خصوصیت Textاین کنترل است که میتوان محتوای HTML صفحه ASP.NET خود را عرضه کرد.

استخراج داده از XML در ASP.NET[1]

در این مقاله با یکی از روشهای ساده استخراج داده از XML به عنوان منبع برای داده های کوچک آشنا می شویم .

یکی از انتخاب های ما برای استخراج داده ها از فایلهای xml استفاده از dataset و یا XMLDocument است که با توجه به منابع زیادی که هر دو اینها استفاده می کنند در بعضی شرایط قابل استفاده نیستند. در این موارد پیشنهاد می شود که از XMLTextReader برای این منظور استفاده شود .XMLTextReader نوعی دسترسی سریع و روبه جلو (ForwardOnly) را برای ما مهیا میکند که در ضمن به خاطر ویژگی non-cached بار زیادی به حافظه اعمال نمی کند.

چند متدی که معمولا بیشتر استفاده می شوند:

Read()که تقریبا شبیه متد هم نام خود در DataReader عمل می کند.

ReadElementString()متن درون المنت را برمی گرداند.

و همچنین ویژگی Item کهAttribute ها و مقادیر آنها را نگهداری میکند. شما با استفاده از HasAttributesمی توانید مطمئن شوید که یک المنت دارای Attribute هست یا خیر.

مثال:

به این فایل XML توجه کنید:

[1] بهنام یوسفی شمالی

```
<?xml version="1.0" encoding="utf-8" ?>
<Root>
  <data>
    <id>1</id>
    <into date="2003-02-18" where="ایران" />
    <comment show="true" title="hi!" />
  </data>
<Root>
```

من برای خواندنAttribute ها از المنت comment و متن المنت id از این فایلXML ای که در بالا آمده اینچنین عمل می کنم:

```
Dim xrdr As New XmlTextReader(path)
Dim s, t, id As String

While xrdr.Read()

  If xrdr.HasAttributes And xrdr.name="comment" Then

    s = xrdr.item("show")

    t = xrdr.item("title")

  ElseIf xrdr.name="id" Then

    id = xrdr.ReadElementString ()
```

```
    End If

End While

xrdr.Close()
```

استفاده از XML به عنوان منابع داده های کوچک می تواند ما را در بسیاری از عملیات برنامه های کاربردی وب کمک کند. به خصوص برنامه هایی که به مصرف منابع حساس هستند.

XML و خواندن اطلاعات از آن[1]

روش تبدیل اطلاعات از بانک اطلاعاتی به XML و نمایش اطلاعات آن

در این مقاله سعی دارم که نحوه نوشتن در XML و خواندن از آن را شرح دهم. برای مثال ما یک وبلاگ داریم که اطلاعات را از بانک اطلاعاتی Access می خواند .ولی من برای اینکه همین اطلاعات را در یک فایل XML هم داشته باشم این کد را به Load صفحه اضافه کردم:

```
Dim DS As New DataSet()
Dim strConnect As New
OleDbConnection(ConfigurationSettings.AppSettings("datasource"))
Dim objSQLAdapter As New OleDbDataAdapter("SELECT * from blog order by id
DESC", strConnect)
objSQLAdapter.Fill(DS, "blog")
Dim doc As XmlDataDocument = New XmlDataDocument(DS)
Dim writer As XmlTextWriter = New XmlTextWriter("D:\www\weblog\weblog.xml",
Nothnig)
writer.Formatting = Formatting.Indented
doc.Save(writer)
```

در این کد با استفاده از شئDataSet ، اطلاعات را از جدول مورد نظر بانک اطلاعاتی خود می خوانیم و آن را در یک فایل XML به نام weblog.xml ذخیره می کنیم. اگر این کد را اجرا کنیم یک فایل XML ایجاد می شود و هر وقت صفحه اول اجرا شود فایل XML رونویسی می شود .

[1] علی حاج شفیعیها

اما حالا یک فایل XML داریم و می خواهیم از آن استفاده کنیم. مخصوصا استفاده از XML برای سایت های خبری یا وبلاگ ها بسیار مناسب است. مثلا فرض کنید که می خواهیم خبرهای موجود در یک سایت دیگر که به شکل XMLارائه می شوند (مانند RSS) را در سایت خودمان نمایش دهیم. برای این کار ابتدا یک فایل ASPX ایجاد کنید و این کد را در قسمت Design یا همان HTML آن بنویسید .

```
<asp:DataList id="theDataList" runat="server" Width="100%" dir=rtl CssClass=text>
  <ItemTemplate>
:<%# DataBinder.Eval(Container.DataItem, "news_date") %><br>تاریخ
: <%# DataBinder.Eval(Container.DataItem, news_title") %><br><br>عنوان
    <%# DataBinder.Eval(Container.DataItem, "news_body") %><br>
    <hr>
  </ItemTemplate>
</asp:DataList>
<br>
```

ما در اینجا یک کنترل DataList ایجاد کردیم و مقادیر مورد نظر را در آن نوشتیم. حالا کدی که در Load این صفحه باید بنویسیم به این شکل خواهد بود:

```
Dim ds As New DataSet()
ds.ReadXml("d:\www\weblog\weblog.xml", XmlReadMode.Auto)
theDataList.DataSource = ds
theDataList.DataBind()
```

اگر این صفحه را اجرا کنید تمام فیلدهای XML در این فایل ASP.NET نمایش داده خواهد شد. در مثال بالا از یک فایل XML موجود بر روی دیسک استفاده شده است حال آنکه می توانیم بجای این آدرس محلی از یک

آدرس موجود در وب که همان آدرس سایت مورد نظر است استفاده کنیم. مثلا اگر بخواهید اخبار سایت مورد نظر را در سایت خود نمایش دهید کافی است اینگونه عمل کنید:

```
Dim ds As New DataSet()
ds.ReadXml("http://www.website.com/weblog.xml",XmlReadMode.Auto)
theDataList.DataSource = ds
theDataList.DataBind
```

گرافیک در ASP.NET[1]

آموزش چگونگی رسم تصاویر بصورت پویا در ASP.NET

برای کار در قسمت گرافیکی ابتدا باید فضانام های System.Drawing و System.Drawing.2D را اضافه کنید سپس یک شئ Bitmap و یک شئ Graphics برای کار کردن تعریف کنید:

```
Dim bmp As New Bitmap (400, 400)
Dim g As Graphics = Graphics.FromImage (bmp)
g.Clear(Color. White)
```

حال می توانیم با استفاده از متدهای Graphics به رسم اشکال مختلف بپردازیم. برای رسم هر شکل تو خالی از یک متد Draw و برای رسم هر شکل تو پر از متد Fill استفاده می کنیم . همیشه هنگام استفاده از Draw باید یک Pen وهنگام استفاده از Fill از یک Brush استفاده کنید:

```
Dim bru As New Solid Brush (Color.Red)
Dim ps As New Pen (Color. Blue)
g.FillRectangle (b, New Rectangle (-10, -10, 320, 220))
Dim rec As New Rectangle (-100, -75, 200, 100)
g.DrawPie (ps, rec, 0, 180)
```

خوب حالا نوبت نمایش عکسی است که تولید کرده ایم برای این کار کافی است Response.contenttype ر ا تنظیم نماییم. به یاد داشته باشید که در انتها لازم است تا با فراخوانی متدDispose ، شئ های Bitmap و Graphicsرا از بین ببریم:

[1] مجتبی کیانی

```
response.contenttype="image/jpeg"
'Send the image to the viewer
(bmp.save (response.outputstream, imageformat.jpeg
bmp.dispose ()
g.dispose ()
```

خوب حالا احتمالا یک مشکل کوچکی وجود دارد آن هم این که تمام یا قسمتی از تصویر ساخته شده قابل دیدن نیست. این مشکل به دلیل قرارگرفتن مبدا مختصات در کناره بالا و سمت چپ تصویر به وجود می آید در حقیقت محور Y ها وارونه قرار گرفته است:

برای رفع این مشکل از کلاس Matrix استفاده می کنیم این کلاس با دو بردار یکه در فضای سه بعدی تعریف می شود که هر یک از این دو بردار در حالت عادی به صورت(۱،۰،۰) هستند. جهت چرخاندن محور Y ها کافیست ۱ را به ۱- تبدیل کنیم برای تغییر دادن محل مبدا نیز از متد Translate ماتریکس استفاده میکنیم. سپس باید این تغییر را در Graphics اعمال کنیم:

```
Dim mt As Matrix = New Matrix (1, 0, 0, -1, 0, 0)
mt.Translate (150, 100)
g.Transform = m
```

دقت داشته باشید که این قسمت باید بلافاصله بعد از تعریف Graphics اضافه شود.

برای رسم اشکال سه بعدی باید با رسم هر یک از وجوه آن اشکال به طور جداگانه آن اشکال را بسازیم. مثلا برای رسم یک مکعب باید کد زیر را وارد کنیم:

```
Dim pf(5) As PointF
'متوازی الاضلاع بالایی----------
pf(0) = New PointF(120, 120)
pf(1) = New PointF(180, 120)
pf(2) = New PointF(150, 90)
Pf(3) = New PointF(120, 90)
pf(4) = New PointF(120, 120)
g.FillPolygon (bru, pf)
'متوازی الاضلاع کناری----------
bru.Color = Color.BlueViolet
pf(0) = New PointF(150, 90)
pf(1) = New PointF(180, 120)
pf(2) = New PointF(180, 180)
pf(3) = New PointF(150, 210)
pf(4) = New PointF(150, 90)
g.FillPolygon(bru, pf)
'مستطیل جلویی----------
bru.Color = Color.SteelBlue
pf(0) = New PointF(90, 90)
pf(1) = New PointF(150, 90)
pf(2) = New PointF(150, 210)
pf(3) = New PointF(90, 210)
```

```
pf(4) = New PointF(90, 90)
g.FillPolygon(bru, pf)
```

تشخیص هویت و تعیین اعتبار در ASP.NET[1]

از موقعی که یک درخواست به وب سرور برای یک فایل که توسط ASP.NET سرویس داده می شود دریافت می گردد و تا موقعی که پاسخ درخواست به کاربر فرستاده می شود یک سری مراحل تشخیص هویت و تعیین اعتبار توسط IIS و ASP.NET صورت می گیرد .

ASP.NET بطور همزمان به همراه IIS و .NET Framework و یک سرویس امنیتی زیرلایه ای که توسط سیستم عامل اعمال می شود، برای ارائه یک سری از انواع تشخیص هویت (Authentication) و تعیین اعتبار (Authorization) کار می کند.

وصیف مکانیسم تشخیص هویت و تعیین اعتبار که توسط IIS و ASP.NET موقع رسیدن یک درخواست صورت می گیرد عبارت است از:

۱- یک تقاضای HTTP از طریق شبکه می رسد، SSL می تواند در این بین برای حفظ امنیت انتقال اطلاعات از کاربران به سرور بکار رود.

۲- IIS بسته به نوع تنظیمات یک از روشهای تشخیص هویت Integrated (NTLM یا Kerberos)، Digest،Basic یا Certificate استفاده می کند. چنانچه سایت شما نیازی به تشخیص هویت ندارد، می توانید IIS را بصورت anonymous که در این حالت (همه به صورت پیش فرض هویت شان تایید می باشد) تنظیم کنید. پس از آن IIS یک بسته اطلاعاتی حاوی هویت کاربر که توسط یکی از روش های بالا تایید هویت شده است یا چنانکه

[1] محمد نظارت

بصورت anonymous تنظیم شده باشد یک بسته اطلاعاتی حاوی تشخیص هویت anonymous (که به صورت پیش فرض "User-Machine"می باشد) می سازد.

۳- IIS کاربر را برای دسترسی به منابع مورد درخواست تعیین اعتبار می کند. که در این مرحله ممکن است ازحقوق امنیتی NTFS که در ACL وصل شده به منبع مورد درخواست برای تعیین اعتبار استفاده شود. و همچنین IIS می تواند برای پذیرفتن درخواست ها از کامپیوترهایی با IP های خاص تنظیم شود.

۴- IIS پس از این اطلاعاتی که تایید شده (که ممکن است anonymous باشد) را به ASP.NET می فرستد.

۵- ASP.NET کاربر را تشخیص هویت می نماید. دراین مرحله اگر ASP.NET برای تشخیص هویت Windows تنظیم شده باشد هیچ گونه تشخیص هویت دیگر صورت نمی گیرد و اگر ASP.NET برای روش تشخیص هویت Forms تنظیم شده باشد مدارک ارائه شده توسط کاربر که از طریق فرم های HTML ارائه می شود با داده های ذخیره شده که بطور معمول در پایگاه داده SQL Server یا دایرکتوری سرویس Active Directory می باشد مقایسه می شود. و اگر ASP.NET برای روش تشخیص هویت Passport تنظیم شده باشد کاربر به سایت Passport مربوطه فرستاده می شود و عمل تشخیص هویت در آنجا صورت می گیرد.

۶- ASP.NET اعتبار کاربر را برای دستیابی به منابع یا پردازش های مورد درخواست بررسی می کند. UrlAuthorization Module (که از ماژول های HTTP ارائه شده توسط سیستم می باشد) از رُل های تعیین اعتبار که در فایل web.config (بویژه تگ<authorization>) تنظیم شده است. برای تایید اعتبار کاربر نسبت به دسترسی به فایلها یا پوشه های مورد درخواست استفاده می کند. FileAuthorization Module (یکی دیگر از ماژول های HTTP) کاربر را برای دسترسی به فایل مورد نظر چک می کند که برای این کار از ACL متصل به منبع

مورد درخواست استفاده می شود. و همچنین رُل های .NET می تواند برای تعیین اعتبار کاربراستفاده شود که می توان آن را در فایل web.config یا بصورت برنامه ای تنظیم کرد.

۷- کد موجود در برنامه شما توسط یک هویت ویژه به منابع محلی یا دور دسترسی پیدا می کند که به صورت پیش فرض ASP.NET هیچگونه تقلید هویتی نمی کند و در نتیجه از حساب تنظیم شده پیش فرض ASP.NET برای تایید استفاده می شود. و انتخاب دیگر استفاده از تائیدیه کاربر اصلی است چنانچه تقلید هویت فعال شده باشد یا یک سرویس تائیدیه تنظیم شده دیگر.

منبع: Building Secure ASP.NET Applications: Authentication, Authorization, and Secure Communication

msdn.microsoft.com/library/default.asp@url=_2flibrary_2fen-us_2fdnnetsec_2fhtml_2fsecnetlpmsdn.asp

مقایسه نحوی ۴ زبان حاضر در ویژوال استودیو ۲۰۰۳[1]

در این مقاله به بررسی و مقایسه اجمالی ۴ زبان J#.NET ، C++.NET، VB.NETو C#.NET پرداخته میشود که این مقایسات به کمک مثال بیان میشود. این مقاله برای افرادی که قصد آموختن یکی از این زبانها را دارند بسیار مفید و کارآمد میباشد .

تعریف متغیرها

```
[VB.NET]
Dim x As Integer
Public x As Integer = 10

[J#.NET]
int x;
int x = 10;

[C++.NET]
int x;
int x = 10;

[C#.NET]
int x;
int x = 10;
```

نوشتن توضیحات در برنامه

[1] حسین پناهی

[VB.NET]

comment '

x = 1 ' comment

Rem comment

[J#.NET]

comment //

multiline */

/* comment

**/

Class Documentation

/**

[C++.NET]

comment //

multiline */

/* comment

[C#.NET]

comment //

multiline */

```
/* comment
```

انتساب

```
[VB.NET]
nVal = 7

[J#.NET]
nVal = 7;

[C++.NET]
nVal = 7;

[C#.NET]
nVal = 7;
```

دستورات شرطی

```
[VB.NET]
If nCnt <= nMax Then
   nTotal += nCnt ' Same as nTotal = nTotal + nCnt.
   nCnt += 1 ' Same as nCnt = nCnt + 1.
Else
   nTotal += nCnt
   nCnt -= 1
End If
```

[J#.NET]

```
if (nCnt <= nMax){
   nTotal += nCnt;
   nCnt++;
}
```

[C++.NET]

```
if(nCnt < nMax) {
   nTotal += nCnt;
   nCnt++;
}
else {
   nTotal += nCnt;
   nCnt--;
};
```

[C#.NET]

```
if (nCnt <= nMax)
{
   nTotal += nCnt;
   nCnt++;
}
else
{
   nTotal +=nCnt;
   nCnt--;
```

```
}
```

دستورات انتخابی

```
[VB.NET]
Select Case n
   Case 0
      MsgBox ("Zero")
      'VB.NET exits the Select at the end of a Case
   Case 1
      MsgBox ("One")
   Case 2
      MsgBox ("Two")
   Case Else
      MsgBox ("Default")
End Select

[J#.NET]
switch(n) {
   case 0:
      System.out.println("Zero\n");
      break;
   case 1:
      System.out.println("One\n");
      break;
   default:
      System.out.println("?\n");
```

```
}

[C++.NET]
switch(n) {
   case 0:
      printf("Zero\n");
      break;
   case 1:
      printf("One\n");
      break;
   case 2:
      printf("Two\n");
      break;
   default:
      printf("?\n");}

[C#.NET]
switch(n)
{
   case 0:
      Console.WriteLine("Zero");
      break;
   case 1:
      Console.WriteLine("One");
      break;
   case 2:
```

```
        Console.WriteLine("Two");
        break;
    default:
        Console.WriteLine("?");
}
```

حلقه For

```
[VB.NET]
For n = 1 To 10
    MsgBox("The number is " & n)
Next

For Each prop In obj
    prop = 42
Next prop

[J#.NET]
for(n=1; n<11;n++)
    System.out.println("The number is " + n);

[C++.NET]
for(int n=1; n<11; n++)
    printf("%d\n",n);

[C#.NET]
for (int i = 1; i <= 10; i++)
```

```
   Console.WriteLine("The number is {0}", i);
foreach(prop current in obj)
{
   current=42;
}
```

حلقهWhile

```
[VB.NET]
While n < 100 ' Test at start of loop.
   n += 1 ' Same as n = n + 1.
End While

[J#.NET]
while (n < 100)
   n++;

[C++.NET]
while(int n < 100)
   n++;

[C#.NET]
while (n < 100)
   n++;
```

انتقال پارامترها با مقدار

```
[VB.NET]
Public Sub ABC(ByVal y As Long)
...
End Sub
ABC(x)
ABC((x))
```

آرگومان y با مقدارش منتقل میشود. اگر زیربرنامه ی ABC مقدار y را تغییر دهد این تغییرات تاثیری بر روی xنخواهد داشت.

شما میتوانید پارامترها را مجبور کنید که با مقدار منتقل شوند بدون توجه به چگونگی تعریف آنها. این کار را می توانید با گذاشتن یک جفت پرانتز () دیگر انجام دهید.

```
[J#.NET]
```

اشیاء همواره با آدرس و انواع داده ی اولیه همیشه با مقدار منتقل میشوند.

```
[C++.NET]
MyMethod(i,j);
[C#.NET]
```

توجه کنید که هیچ راهی برای انتقال انواع مرجع (اشیاء) منحصرا از طریق مقدار وجود ندارد.

یک متد تعریف شده:

```
void ABC(int x)
{
```

```
...
}
```

و فراخوانی این متد:

```
ABC(i)
```

انتقال پارامترها با آدرس

```
[VB.NET]
Public Sub ABC(ByRef y As Long)
End Sub

ABC(x)
```

پارامتر y به صورت ByRef (با آدرس) شده است. اگر ABC مقدار y را تغییر دهد این تغییرات روی x نیز صورت میگیرد.

```
[J#.NET]
[C++.NET]
```

پروتوتایپ تابع ABC که شامل اشاره گری به یک عدد است.

```
int ABC(long *py)
ABC(&VAR)
```

پروتوتایپ تابع ABC که شامل ارجاعی به یک عدد است.

```
int ABC(long &y);
ABC(VAR);
```

[C#.NET]

همچنین توجه داشته باشید که متد C# میتواند مانند C++ شامل اشاره گر باشد. برای مثال این متد:

```
void ABC(ref int x)
{
...
}
ABC(ref i);
```

مدیریت استثنائات ساخت یافته

[VB.NET]

```
Try
   If x = 0 Then
      Throw New Exception("x equals zero")
   Else
      Throw New Exception("x does not equal zero")
   End If
Catch err As System.Exception
   MsgBox("Error: " & Err.Description)
Finally
   MsgBox("Executing finally block.")
End Try
```

```
[J#.NET]
try{
   if (x == 0)
      throw new Exception ("x equals zero");
   else
      throw new Exception ("x does not equal zero");
}
catch (Exception err){
   if (err.getMessage() == "x equals zero")
      System.out.println(err.getMessage());
   //Handle Error Here
}

[C++.NET]
__try{
   if (x == 0)
      throw new Exception ("x equals zero");
   else
      throw new Exception ("x does not equal zero");
}
__catch(Exception e)
{
   Console.WriteLine("Caught Exception");
}
__finally
{
```

```
   Console.WriteLine("Executing finally block");
}

[C#.NET]
// try-catch-finally
try
{
   if (x == 0)
      throw new System.Exception ("x equals zero");
   else
      throw new System.Exception ("x does not equal zero");
}
catch (System.Exception err)
{
   System.Console.WriteLine(err.Message);
}
finally
{
   System.Console.WriteLine("executing finally block");
}
```

رهاسازی ارجاعات اشیاء

```
[VB.NET]
o = Nothing

[J#.NET]
```

```
stringVar = null;

[C++.NET]

[C#.NET]
o = null;
```

نگاهی به ASP.NET Whidbey[1]

نگاه مختصری به نسخه جدید ASP.NET به نام Whidbey و بررسی قابلیتهای آن

مقدمه

امروزه Microsoft ASP.NET بصورت وسیعی در حال گسترش است. بسیاری از سایتهای بزرگ ومعروف مانند DELL، Marrill Lynch، بازار بورس لندن، NASDAQ، هواپیمایی JetBlue و USAToday و بسیاری سایتهای دیگر از این فناوری استفاده کرده اند.

هر روزه هزاران برنامه نویس اقدام به فراگیری ASP.NET کرده و تاکنون بیش از ۱۷۰ عنوان کتاب در مورد ASP.NETچاپ شده است. همچنین وبسایتهای زیادی در قالب سایتهای آموزشی، انجمنها و وبلاگها در این زمینه وجود دارد.

ASP.NET "Whidbey"

نسل جدید ASP.NET که کد-نام آن Whidbey می باشد، گام بلندی است در زمینه افزایش کارایی و نیز قابلیتهای جدیدی که همگی حاصل تجربیات چند ساله جامعه بزرگ متخصصین ASP.NET است. این نسخه‌ی جدید بصورت صد در صد با نسخه قبلی سازگار خواهد بود و تسهیلاتی جهت انتقال برنامه ها از نسخه قدیم به نسخه جدید اندیشیده شده است.

در طراحی Whidbey بطور عمده بر سه هدف تمرکز شده است:

[1] مدیریت سایت iranasp.net

- بهره وری برنامه نویس
- مدیریت و نگهداری
- سرعت و کارایی

بهره وری برنامه نویس

در طراحی Whidbey سعی شده است تا برنامه نویسان بتوانند برنامه های تحت وب حرفه ای را بسیار آسانتر و سریعتر از قبل بسازند. زمان بسیار زیادی صرف گفتگو با انواع برنامه نویسان و بررسی برنامه های موجود شده است تا بتوان نقاط اشتراک میان آنها را مشخص نمود. سپس این نقاط مشترک همگی بصورت بخشی از Whidbey درآمده و به ASP.NET اضافه شده است. برای مثال می توان به موارد جدید و آماده زیر در Whidbey اشاره نمود.

- سیستم شناسایی کاربر (username/ password)
- سیستم شخصی سازی (personalization)
- Master Pagesجهت ایجاد یک قالب واحد برای کلیه صفحات یک وبسایت
- سیستم جدید پیمایش سایت (site navigation) جهت ساخت سریع ساختار لینکهای سایت
- سیستم آمارگیری یا شمارنده برای تعداد بازدیدکنندگان سایت و تعداد صفحات بازدید شده (site counter)
- قالب های آماده جهت ساخت سریع نما و ظاهر سایت (theme)
- بخش جدید ASP.NET Web Part جهت انواع مدلهای چیدمان پورتال(portal)

علاوه بر کلیه موارد فوق، Whidbeyشامل بیش از ۴۵ مورد کنترل (server control) جدید خواهد بود. این کنترلها در زمینه های پایگاه داده، امنیت، تولید تصاویر، پیمایش وبسایت، منو، treeviewو... خواهد بود.

برای ساخت صفحه ای در ASP.NET 1.0 که حاوی یک DataGrid که آن نیز شامل لیست بازشو (dropdownlist)در هر سطر خود می باشد، نیاز بود که بیش از صد خط برنامه نوشت و به دفعات نیز به راهنمای مربوطه (help) مراجعه نمود. اما جالب است که بدانید این کار در Whidbey تنها در یک خط برنامه و ظرف چند ثانیه انجام خواهد شد. همه این کارها در محیط برنامه نویسی جدیدی برای Visual Studio .NET که اتفاقاً کد-نام آن نیز Whidbey است، انجام خواهد شد.

همه کنترلهای استاندارد Whidbey دارای یک واسط کاربری قوی خواهند بود بطوریکه بتوان همه انواع مرورگرها و دستگاههای موبایل را پشتیبانی کرد. همه این کنترلها دارای پیشوند <:asp> خواهند بود و توسط آنها می توان بیش از ۳۰۰ نوع دستگاه موبایل با انواع زبانهای نشانه گذاری مانند XHTML Mobile ، WAP/WMLو cHTMLرا سرویس‌دهی کرد.

در مجموع قابلیتهای جدید Whidbey برای برنامه نویسان بسیار شگفت آور خواهد بود بطوریکه پروژه هایی که هم اکنون ممکن است روزها یا هفته ها زمان ببرند، تنها در عرض چند ساعت اجرا خواهند شد.

مدیریت و نگهداری

در Whidbey هدف این است که مدیران سرورها (administrators) بتوانند ASP.NET را به آن اندازه ای که برنامه نویسان آن را دوست دارند، دوست داشته باشند. این بمعنی ایجاد سهولت بیشتر در نصب، انتقال، نگهداری و مدیریت سرورهای ASP.NET می باشد. در Whidbey ابزارها وAPI های جدیدی ساخته شده است که بتوان به کمک آنها و بصورت برنامه ای، برنامه ها و اسکریپت‌هایی جهت ایجاد، خواندن و بروزرسانی فایلهای Web.config و Machine.configساخت. همچنین یک ابزار مدیریتی جدید بصورت گرافیکی کلیه فایلهای XML مربوط به ساختاربندی (configuration) را مدیریت می کند.

ابزاری همراه Whidbey ارائه خواهد شد تا بتوان به کمک آن و قبل از نصب برنامه‌ی وبسایت، نسخه ای بصورت پیش-کامپایل (pre-compile) تهیه نمود که محتوی همه اجزای برنامه حتی فایلهای .aspx نیز باشد و بدین ترتیب کد فایلهای شما بخصوص فایلهای متنی مانند .aspx و .ascx از چشم دیگران محفوظ مانده و امنیت کد برنامه شما افزایش می یابد.

سرعت و کارایی

امروزه ASP.NET سریعترین فناوری در زمینه برنامه های تحت وب در جهان است و هدف این است که در Whidbeyسرعت آن باز هم بیشتر شود.

Whidbeyاز تکنولوژی ۶۴ بیتی بهره می برد و این بمعنی سود بردن از تمام فضای آدرسی دهی و محاسباتی پردازنده های ۶۴ بیتی است. برنامه نویسان می توانند برنامه های ۳۲ بیتی قدیمی خود را حتی بدون کوچکترین تغییری در کد برنامه بر روی سرور ۶۴ بیتی Whidbey قرار دهند و برنامه های آنها بصورت خودکار به شکل JIT کامپایل شده و بصورت ۶۴ بیتی اجرا خواهد شد.

همچنین Whidbey دارای قابلیت خودکار باطل کردن cache مربوط به سرور پایگاه داده است. این ویژگی برنامه نویسان را قادر خواهد ساخت تا بصورت حریصانه‌ای صفحات مبتنی بر پایگاه داده را cache کرده یا اصطلاحاً از انواع output cache بدون نگرانی استفاده کنند و مسوولیت بروز رسانی آنها را در صورت تغییر داده های پایگاه داده بعهده ASP.NET بگذارند.

دقت داشته باشید که هم اکنون پروژه Whidbey در مرحله آلفا قرار دارد و در حدود دو-سوم تواناییهای فوق به ASP.NET افزوده شده است. در بهار آینده نسخه کامل شده بتای آن ارائه خواهد و ممکن است قابلیتهای بیشتری هم به آن افزوده شود. در آن هنگام می توان Whidbey را دریافت و نصب نمود و با آن عملاً برنامه نوشت.

جهت دریافت اطلاعات بیشتر و تکمیلی به آدرس http://www.asp.net/whidbey مراجعه نمائید.

جهت دریافت اطلاعات در مورد نسخه جدید Visual Studio .NET Whidbey به آدرس http://www.asp.net/whidbey/whitepapers/VSWhidbeyOverview.aspx مراجعه نمائید.

ترجمه و خلاصه از ASP.NET "Whidbey" Overview

www.asp.net/whidbey/whitepapers/aspnetoverview.aspx@tabindex=0&tabid=1

Session State در ASP.NET[1]

در این مقاله شئ Session در ASP.NET بررسی می شود .

ASP.NETتنظیمات مدیریت وضعیت پیشرفته ای نسبت به شئ Session در ASP کلاسیک دارد. این تنظیمات در فایل Web.Config پیکربندی می شوند. در این مقاله تنظیمات پیکربندی موجود برای سرویس مدیریت حالت و رویدادهایی که می توانند در فایل Global.asax اداره شوند و همچنین مثالهایی از استفاده شئ Sessionبرای برقراری وضعیت را بررسی خواهیم کرد.

پیکربندیSession

وضعیت جلسه یا Session State می تواند در قسمت <SessionState> فایل Web.Config پیکربندی شود. قطعه کد زیر که از یک فایل Web.Config انتخاب شده است صفاتی که می تواند به این قسمت اختصاص داده شود را نشان می دهد:

```
<sessionState
mode="Off|InProc|StateServer|SqlServer"
stateConnectionString="tcpip=127.0.0.1:42424"
sqlConnectionString="data source= 127.0.0.1;userid=sa;password="
cookieless="true|false"
timeout="20"
/>
```

[1] احسان ولیزاده

صفت	توضیح
mode	بیان می کند که وضعیت جلسه (Session State) در کجا ذخیره شده است. مقادیر مجاز عبارتند از: Off (وضعیت حالت خاموش است) –Inproc (در حافظه) - StateServer(حافظه سرور) –SqlServer (در پایگاه داده).
stateConnectionString	رشته ارتباط به سرور مدیریت حالت. این صفت زمانی استفاده می شود که صفت modeبرابر SqlServer است.
sqlConnectionString	رشته اتصال به پایگاه داده مدیریت جلسه. این صفت زمانی استفاده می شود که صفت modeبرابر SqlServer است.
cookieless	این صفت یک مقدار Boolean می گیرد و به سایتها اجازه می دهد کاربران آنها از کوکی ها به خاطر بهره گیری از مزایای Session State در ASP.NET استفاده نکنند.
timeout	نشان دهنده دقایقی است که یک جلسه بیکار خاتمه می یابد. پیش فرض ۲۰ دقیقه است.

رویدادهای Session

دو رویداد مربوط به شروع و خاتمه جلسات در فایل Global.asax وجود دارند که می توانید آنها را اداره کنید. این رویدادها Session_Start و Session_End نام دارند. مثلا به کمک آنها می توان تعداد کاربران آنلاین وب سایت را در صفحه نشان داد. برای این کار باید در رویداد Session_Start یک واحد به تعداد کاربران آنلاین اضافه

کنیم و یک واحد از این تعداد را در رویداد Session_End کم کنیم. کد زیر نحوه تعریف کردن رویدادها را در فایل Global.asaxنشان می دهد:

```
<script language="VB" runat="server">

Sub Session_Start(sender As Object, e As EventArgs)

End Sub

Sub Session_End(sender As Object, e As EventArgs)

End Sub

</script>
```

استفاده از شئSession

شئ Session در واقع یک خصوصیت کلاس Page و همچنین یک نمونه از کلاس HttpSessionState است که در فضانام System.Web.SessionState تعریف شده است. استفاده از شئ Session همانند استفاده از یک مجموعه (Collection) است. مثال زیر نام کاربر را به یک متغیر Session در رویداد Page_Load اختصاص می دهد و سپس از آن متغیر برای خوش آمد گویی به کاربر استفاده می کند. صفحات دیگر نیز می توانند آن نام را برای فراهم کردن صفحات شخصی از Session دریافت کنند.

```
<%@ Page Language="VB" %>
```

```
<script language="VB" runat="server">
Sub page_load()

Session["name"] = "James";

End Sub
</script>

<HTML>
<body>
Hello <%=Session["name"]%>!
</body>
</HTML>
```

آمار کاربران سایت در ASP.NET[1]

روزانه کاربران زیادی از سایت شما بازدید می کنند و برای هر مدیر سایت آمار بازدیدها، صفحات بازدید شده، ساعت و تاریخ بازدید، لینک و سایتی که کاربر به واسطه آن از سایت ما بازدید نموده است و . . . از اهمیت ویژه‌ای برخوردار است .

روزانه کاربران زیادی از سایت شما بازدید می کنند و برای هر مدیر سایت آمار بازدیدها، صفحات بازدید شده، ساعت و تاریخ بازدید، لینک و سایتی که کاربر به واسطه آن از سایت ما بازدید نموده است و . . . از اهمیت ویژه‌ای برخوردار است.

برای آمارگیری از سایت‌ها از روشهای مختلفی می توان استفاده نمود:

۱ -استفاده از سایتهای آمارگیری رایگان همانند Nedstat :یا Sitemitter و . . .

۲ -به روش برنامه‌نویسی و بررسی از داخل برنامه

در روش اول به نوعی برای نمایش به کاربران از اعتبار بیشتری برخوردار است اما نقص آن علاوه بر تبلیغ مجانی برای دیگران این است که در بعضی ازFirewall ها به اسکریپتی که اطلاعات کاربر را به این سایتها ارسال می نماید همانند یک تروجان (اسب تراوا یا سیستم جاسوسی) نگاه می کنند و اجازه عبور به آن نمی دهند.

به همین خاطر روش دوم می تواند دقیق تر عمل نماید. از سوی دیگر ذخیره اطلاعات کاربران در یک بانک اطلاعاتی می تواند منشأ گزارشات بسیار جالبی برای مدیران سیستم شود.

[1] رسول هاجری

در نمونه برنامه زیر سعی کرده‌ام اطلاعات نسبتاً جالبی از رفتار کاربران به کمک ASP.NET را به نمایش گذارم. البته مطالب ذیل نمونه‌ای از اطلاعات کاربران سایت می باشد که می شود به سلیقه و نیاز خودتان آنرا تغییر دهید (کلیه متغیرهایی که نوع آنها ذکر نشده است از نوع String می باشد).

الف) نام دستگاه کاربر و IP آن

```
ClientName = Request.UserHostName
ClientIP = Request.UserHostAddress
```

ب) لینکی که کاربر با کلیک بر روی آن سایت را یافته است.

```
REFERER = Request.ServerVariables.Item("HTTP_REFERER")
```

ج) اطلاعاتی در رابطه با سیستم کاربر(Client)

```
LANGUAGE = Request.ServerVariables.Item("HTTP_ACCEPT_LANGUAGE")
AGENT = Request.ServerVariables.Item("HTTP_USER_AGENT")
Platform = Request.Browser.Platform()
```

د) QUERY STRING صفحه حاضر

```
QUERY STRING = Request.ServerVariables.Item("QUERY_STRING")
```

ح) اطلاعاتی در رابطه با Browser کاربر

```
Browser = Request.Browser.Browser()
Browser_Type = Request.Browser.Type()
Browser_Version = Request.Browser.Version()
```

ز) زمان و تاریخ بازدید

```
Dim MyDateTime As New DateTime
MyDateTime = Now()
Dim MyDate As String = MyDateTime.ToString("MM/dd/yyyy")
Dim MyTime As String = MyDateTime.ToString("hh:mm:ss")
```

بدیهی است در صورتی که بخواهیم به محض ورود کاربر به سایت این مجموعه اطلاعات را جمع‌آوری نماییم، می بایست آن را در زیربرنامه Session_Start انجام دهیم(این روتین در Global.asax قرار دارد).

با ارسال محتویات این متغیرها به بانک اطلاعاتی می توان گزارشاتی از جمله تعداد بازدیدها در مقاطع زمانی مختلف، صفحات پر بیننده، پر بیننده‌ترین صفحه امروز و. . . را استخراج نمود .

کارکردن با رشته‌ها و متن‌ها در ASP.NET[1]

اغلب متن‌هایی که کاربران یک سایت توسط فرم‌ها ارسال می‌کنند و یا اطلاعاتی که از یک بانک دریافت می‌شود، نیاز به بررسی و یا اعمال تغییرات دارند. در این مقاله با برخی توابع و امکانات ASP.NETبرای کارکردن با رشته‌ها یا به تعبیری دیگرstring ها آشنا می شویم .

اغلب متنهایی که کاربران یک سایت توسط فرم‌ها ارسال می‌کنند و یا اطلاعاتی که از یک بانک دریافت می‌شود، نیاز به بررسی و یا اعمال تغییرات دارد.

خوشبختانه ASP.NET مجموعه امکانات بسیار خوبی برای کارکردن با رشته‌ها ارائه نموده است. بعبارت دیگر با استفاده از کلاس String برای ذخیره متون در برنامه، امکانات زیادی را برای کار کردن با رشته‌های حرفی خواهیم داشت.

در مثال‌های زیر سعی کرده‌ام مهمترین و پرمصرف‌ترین دستورات کار با رشته‌ها را معرفی کنم.

الف) انتخاب قسمتی از متن(mid)

دستور mid بر اساس فرم زیر قابل استفاده است :

Mid(A, B, C)

A=متغیر متنی یا یک رشته متن

B=شماره کارکتر شروع انتخاب

[1] رسول هاجری

C= تعداد کارکتر انتخاب

```
Dim aString As String = "SomeString"
Dim bString As String
bString = Mid(aString, 3, 3) ' Returns "meS"
Response.Write(bString)
```

ب) انتقال بخشی از یک متن به داخل یک متغیر

Substringبر اساس روش زیر قابل استفاده می باشد :

.Substring(A, B)نام متغیر رشته‌ای

A=شماره کارکتر شروع انتخاب

B= تعداد کارکتر انتخاب

```
Dim aString As String = "A String"
Dim bString As String
bString = aString.Substring(2, 6) ' Returns "String"
Response.Write(bString)
```

ج) جمع نمودن چند رشته با یکدیگر

String.Concatمی‌تواند چند رشته را در کنار هم قرار دهد و یا مجموع اطلاعات یک آرایه را با هم ادغام نماید .

```
Dim myString As String
Dim aString(10) As String
Dim t As Integer
For t = 0 To 9
   aString(t) = CStr(t)
Next t

myString = String.Concat(aString) ' Returns "0123456789"
Response.Write(myString)
```

د) ادغام رشته‌ها به همراه قرار دادن کارکتر جدا کننده

String.Joinاین امکان را فراهم می‌کند تا ضمن ادغام چند رشته، در بین اطلاعات کاراکتر جداکننده نیز قرار دهیم .

```
Dim shoppingItem(2) As String
Dim shoppingList As String
shoppingItem(0) = "Milk"
shoppingItem(1) = "Eggs"
shoppingItem(2) = "Bread"
shoppingList = String.Join(",", shoppingItem) 'Returns "Milk,Eggs,Bread"
Response.Write(shoppingList)
```

ح) اضافه کردن متن جدید به یک رشته(Insert)

برای اضافه کردن متن جدید به یک رشته کافی است از Insert به روش زیر استفاده نماییم :

Insert(A, B).متغیر رشته‌ای

شماره کارکتری که اضافه شدن از آن شروع می‌شود=A

متنی که می‌خواهیم اضافه کنیم =B

```
Dim aString As String = "This is My Stng"
Dim myString As String
myString = aString.Insert(13, "ri") ' Returns "This is My String"
Response.Write(myString)
```

و) جدا نمودن یک رشته بر اساس یک کارکتر تکراری

گاهی اوقات لازم است بر خلاف String.Join متن یک رشته را که به طور مثال با "،" به قطعات مشخص تقسیم شده است را از هم جدا نماییم. برای این منظور همانند مثال زیر عمل نمایید :

```
Dim shoppingList As String = "Milk,Eggs,Bread"
Dim shoppingItem(2) As String
shoppingItem = shoppingList.Split(","c)
Dim s As String
For Each s In shoppingItem
```

```
  If s.Trim() <> "" Then
    Response.Write(s & "<BR>")
  End If
Next s
```

ز) جایگزینی متن در رشته(Replace)

هرگاه نیاز به تعویض و جایگزینی یک متن در یک رشته داشته باشیم می‌توانیم از Replace همانند مثال استفاده کنیم :

```
Dim myString As String = "Shopping List"
Dim aString As String
aString = Replace(myString, "o", "i") ' Returns "Shipping List".
Response.Write(aString)
```

فایل Web.Config را بهتر بشناسیم[1]

بیشتر برنامه‌های کامپیوتری تنظیمات خاص خود را در قالب مفهومی بنام Setup نگهداری می‌کنند. برنامه‌های ASP.NET هم از این قاعده مستثنی نیستند و می‌توان کلیه تنظیمات وب سایت را در قالب یک فایل از جنس XML بنام Web.Config نگهداری کرد .

مقدمه

بیشتر برنامه های کاربردی کامپیوتری دارای بخشی به نام Setup هستند که برای انجام تنظیمات متناسب با استفاده های مختلف کاربران مورد استفاده قرار می گیرد. برنامه های کاربردی مبتنی بر Web نیز که دارای کاربران زیادی هستند از این قائده مستثنی نیستند و احتیاج به انجام تنظیماتی متناسب با کاربردهای مختلف خود دارند.

در ASP.NET اینگونه تنظیمات که مربوط به منابع مورد استفاده برنامه کاربردی، اطلاعات محلی، اطلاعات امنیتی و... می شوند در داخل یک فایل XML ذخیره می شوند که نام آن Web.Config است و طبعا به مانند اسناد XML شامل تعدادی تگ (معادل Tag در HTML) است و به کوچک و بزرگ بودن حروف نیز حساس است (Case Sensitive).

در این مقاله به اختصار به توضیح در مورد تگهای مهم این فایل و چگونگی تنظیم خصوصیات مهم این تگها و کاربردهای آنها می پردازیم. پیش از ادامه مطلب این نکته را متذکر می شویم که این فایل شامل تعداد زیادی تگ با خصوصیات مختلف است که هر یک به جنبه ای از یک برنامه کاربردی تحت Web مربوط هستند. در این مقاله در مورد تگهای پرکاربرد و مهم این فایل و خصوصیات مهم آنها اطلاعاتی ارائه شده است برای به دست آوردن اطلاعات بیشتر در مورد هر یک از تگها و خواص آنها باید به منابع تخصصی مرتبط با هر بخش رجوع کرد.

[1] کیوان نّیری

محل و محدوده عملکرد فایلWeb.Config

به طور طبیعی هر برنامه کاربردی Web در ASP.NET دارای حداقل یک فایل Web.Config در داخل دایرکتوری ریشه خود می باشد که به طور خودکار توسط Visual Studio .NET در زمان ایجاد این Web Applicationایجاد شده و با مقادیر پیش فرض در دایرکتوری ریشه برنامه کاربردی قرار می گیرد. ولی برنامه نویسان می توانند بر حسب نیازها و کاربردهای برنامه های خود تعداد بیشتری از این فایل XML را در داخل دایرکتوریهای مختلف برنامه ایجاد کنند و برای تنظیمات فایلهای داخل همان دایرکتوری یا زیر دایرکتوریهای آن به کار گیرند.

تنظیمات هر فایل Web.Config به فایلهای داخل زیر دایرکتوری‌ای که در آن قرار دارد و همچنین زیر دایرکتوریهای آن اعمال می شود. همچنین برنامه نویس می تواند از طریق کدهای داخل هر فایل Web.Config منابع و دایرکتوریهای مورد نظر را تغییر دهد. برای درک بهتر نحوه ارتباط فایلهای Web.Config در یک برنامه کاربردیWeb ، مثال زیر را بیان می کنیم.

اگر یک دایرکتوری به نام main و دو زیر دایرکتوری به نام Sub1 و Sub2 در برنامه کاربردی موجود باشند که در درون هر کدام از این دایرکتوریها یک فایل Web.Config قرار داشته باشد تنظیمات هر فایل Web.Config بر فایلهای داخل زیر دایرکتوری آن اعمال می شود اما اگر در یک فایل Web.Config که داخل زیر دایرکتوری Sub1یا Sub2 قرار دارد یکی از تنظیماتی که در فایل Web.Config که داخل main قرار دارد صورت نگرفته باشد مقدار تنظیم شده در داخل فایل موجود در main به زیردایرکتوریها به ارث می رسد.

نکته ای که باید در اینجا به آن اشاره کرد این است که کاربران برنامه وب به محتوای فایلهای Web.Config دسترسی ندارند.

ساختار فایلWeb.Config

هر فایل Web.Config شامل یک بخش پایه است که همان دو تگ <configuration> و </configuration>هستند و بقیه محتوا درون این دو تگ قرار می گیرد. پس ساختار کلی هر فایل Web.Configبه صورت زیر است:

```
<configuration>
<!--Some Configurations-->
</configuration>
```

محتوای هر فایل Web.Config شامل دو بخش اساسی است: معرفی و تنظیمات .بخش معرفی داخل دو تگ <configsectons>و </configsections> قرار می گیرد و بخش تنظیمات، تنظیمات مربوط به منابع معرفی شده در این تگها را شامل می شود.

ابتدای فایل Web.Config و بعد از تگ <configuration> تگ <configsections> قرار دارد. داخل این تگ و تگ جفت آن یعنی </configsectons> تگهایی جهت معرفی منابع و اداره کننده قسمتها قرار دارد .حال که با ساختار کلی فایل Web.Config آشنا شدیم در ادامه به توضیحاتی در مورد تگهای مهم این فایل و خصوصیات آنها می پردازیم.

Location

پیشتر اشاره شد که تنظیمات هر فایل Web.Config بر فایلهای داخل ریشه این فایل و زیردایرکتوری های آن اعمال می شوند، حال اگر بخواهیم تنظیمات یک فایل Web.Config را بر دایرکتوری خاصی اعمال کنیم تنها

لازم است این تنظیمات را داخل یک جفت تگ <location> و </location> قرار دهیم یعنی شکل کلی استفاده از این جفت تگ به صورت زیر است:

```
<location path="url">
<!--Configurations-->
</location>
```

اطلاعات محلی

از تگ <globalization> برای تعریف خصوصیات فرهنگی و زبانی برنامه کاربردی استفاده می شود. خواص مهم این تگ عبارتند از:

- culture:فرهنگ برنامه کاربردی را تعیین می کند .
- requestencoding شکل رشته های درخواست شده را تعیین می کند (برای مثال Unicode).
- responseencoding:شکل رشته های پاسخ را تعیین می کند .

تگهای امنیتی

درون فایل Web.Config تگهایی جهت تنظیم مقادیر و خصوصیات امنیتی برنامه کاربردی قابل تعریف هستند. جفت تگهای <authentication> و </authentication>و <authorization> و </authorization>ساختارامنیتی درون فایل Web.Config را می سازند که دارای خصوصیات و مقادیر متفاوتی هستند که توضیح در مورد آنان به مباحث مقدماتی امنیت در ASP. NET بر می گردد ولی در زیر توضیح کوتاهی از هر یک از این جفت تگها ارائه می گردد.

تگ <authentication> : این جفت تگ به تعیین شکل تایید اعتبار در برنامه کاربردی تحت Web می پردازد. از طریق خصوصیت mode که چهار مقدار Form ، Passport، Windowsو None را می پذیرد می توان شکل تایید اعتبار را به ترتیب بر مبنای ساختار امنیتیIIS ، سرویس تایید اعتبار Passport شرکت مایکروسافت، استفاده ازcookie ها و بدون استفاده از سیستم امنیتی تعیین کرد .

تگ <authorization> : این جفت تگ برای تعیین کاربران واجد شرایط و یا فاقد شرایط ورود به محدوده تحت کنترل فایل Web.Config استفاده می شود. با استفاده از دو تگ <allow> و <deny> می توان کاربرانی که اجازه ورود به محدوده تحت حفاظت را دارند و فاقد این اجازه هستند استفاده کرد .

خطای زمان اجرا

تگ <customerrors> جهت تعیین شکل عملکرد برنامه در زمان وقوع یک خطای پیش بینی نشده در زمان درخواست یک صفحه توسط کاربر به کار می رود. این تگ دارای دو صفت مهم است که در زیر توضیح داده شده اند.

mode : این صفت جهت تعیین نوع عملکرد برنامه کاربردی در مواجهه با خطا به کار می رود و یکی از سه مقدار RemoteOnly ، Onو Off را می گیرد. با تنظیم این صفت با مقدار RemoteOnly در صورت بروز خطای زمان اجرا صفحه از پیش تعیین شده در قسمت defaultredirect برای کاربران نشان داده خواهد شد ولی کاربرانی که بصورت محلی از برنامه استفاده می کنند (مقابل سرور نشسته اند) صفحه خودکار ASP.NET و جزئیات خطا را خواهند دید. با تنظیم این صفت با مقدار On در صورت بروز خطای زمان اجرا همواره صفحه از پیش تعیین شده در قسمت defaultredirect نشان داده خواهد شد. با تنظیم این صفت با مقدار Off صفحه خطای خودکار ASP. NET به نمایش در می آید که از لحاظ امنیتی گزینه مناسبی نیست.

defaultredirect: آدرس Url فایلی را که برای نمایش خطا در زمان تنظیم خصوصیت mode با مقدار On یا RemoteOnly بکار می رود تعیین می کند .

وضعیت جلسه کاری

تگ <sessionState> برای تعیین وضعیت جلسه کاری مورد استفاده قرار می گیرد و مدت زمان یک جلسه کاری و همچنین شکل تعیین درخواست کاربران را تعیین می کند. از صفات مهم این تگ یکی Cookiless است که تعیین می کند جلسه کاری کاربران از طریق Cookie ها یا SessionID ها نسبت داده شوند که دو مقدار true یا false را می گیرد true :برای استفاده از SessionID و false برای استفاده از Cookie.

خصیصه مهم دیگر این تگ mode است که یکی از چهار مقدار زیر را می گیرد:

- Off:جلسه کاری غیر فعال
- Inproc:اطلاعات به صورت In-process ذخیره می شوند .
- SQLServer:اطلاعات جلسه کاری توسط پایگاه داده SQL Server مدیریت می شوند .
- StateServer:اطلاعات توسط یک سرویس Out-Of-Process مدیریت می شوند .

خصوصیت مهم دیگر این تگ timeout است که زمان اعتبار جلسه کاری هر کاربر را بر حسب دقیقه تعیین می کند .

کامپایل برنامه کاربردی

تگ <compilation> برای تعیین خصوصیات مربوط به شکل کامپایل برنامه کاربردی مورد استفاده قرار می گیرد که دو خصیصه مهم آن عبارتند از:

- **debug :**یکی از دو مقدار true یا false را می گیرد(false پیش فرض است) و تعیین می کند که اطلاعات اشکال زدایی در اسمبلی های بعد از کامپایل قرار بگیرند یا خیر. با تنظیم این خصیصه با مقدار true این کار صورت می گیرد. و با تنظیم این خصیصه با مقدار false این کار صورت نمی گیرد .

- **defaultlanguage:**زبان برنامه نویسی مورد استفاده جهت کامپایل برنامه کاربردی (به صورت Dynamic)را تعیین می کند .

IrMail، روشی جدید برای ارسال نامه در ASP.NET (بخش اول)[1]

ارسال نامه از طریق سایت بصورت خودکار همواره برای برنامه نویسان وب موضوعی حساس و مهم بوده است. بسیاری از بخش های یک سایت حرفه‌ای مانند صفحه ارتباط با مدیر سایت، نامه‌های یادآوری کلمه رمز کاربران، ارسال خبرنامه و... همگی به نوعی از قابلیت ارسال نامه توسط یک برنامه بهره می برند. این مقاله به معرفی یک شئ جدید بنام IrMail جهت ارسال نامه در ASP.NET با تاکید بر زبان فارسی می پردازد .

تذکر ۱: هر جا در این مقاله کلمه ***"شئ IrMail"*** ذکر شود، منظور همان کامپوننت آن (IrMail Component) است که به منظور رساندن مطلب، از این جایگزین استفاده شده است.

تذکر ۲: شئ IrMail صرفاً برای نسخه .NET 1.1 (و به بعد) طراحی شده است .برای نسخه .NET 1.0 ، با نویسنده مقاله تماس بگیرید.

مطلب را با طرح یک مساله آغاز می کنیم و در ادامه آن به بررسی و یافتن راه حل های مختلف برای آن و مقایسه آنها با هم می پردازیم.

طرح مساله

همانطور که می دانید یکی از مسائلی که برنامه نویسان برنامه های تحت وب (web applications) با آن مواجه هستند، مساله ارسال نامه از طریق سایت به صورت خودکار می باشد. دلایل نیاز به انجام چنین کاری را می توان به صورت زیر خلاصه کرد:

[1] محمود مروج

•ایجاد صفحه "تماس با ما" و امکان ارسال نامه از طریق سایت

•نامه های حاوی اطلاعات کاربر برای تایید عضویت

•نامه های یادآوری Password کاربران

•خبرنامه ها

•و در کل هرگونه نیاز به ارتباط با کاربر از طریق ایمیل به صورت پویا و خودکار

ASP.NETدر مقابل ASP کلاسیک

اگرچه شئ CDONTS به شکل استفاده از یک شئ جنبی این مساله را در ASP کلاسیک تا حدی حل کرده بود، اما کاربران ASP کلاسیک بصورت عمده با این مساله مواجه بودند. این مساله به حدی مهم بود که یکی از مشخصه های اصلی انتخاب یک میزبان (host) مناسب برای میزبانی سایت‌هایشان را همین موضوع در برمی گرفت. بطوریکه تا قبل از تست امکانات آن و مطمئن شدن از آن اقدام به بستن قرارداد نمی کردند و یا در نهایت اقدام به تعویض میزبان مورد نظر می کردند.

با پیدایش .NET Framework عملاً این مشکل حل شد. چرا که:

•وجود شئ ای بصورت ذاتی برای این کار

•امکان ارث بری از کلاسها

•و عدم نیاز به ثبت شدن اسمبلی ها (Dll) در Registry

این مساله بسیاری از توانایی ها و امکانات را در دست برنامه نویسان قرار داد تا حتی در صورت ناقص بودن کلاس های موجود در NET. ، امکان تکمیل و حتی ارایه یک کلاس جدید در این زمینه را داشته باشند .

در ASP کلاسیک از آنجا که اشیاء باید در رجیستری ثبت گردند، عملاً این امکان را از بسیاری از برنامه نویسان عادی سلب می کرد که خود بر روی اشیاء کار کنند و نمونه مورد نظر را ایجاد و استفاده نمایند. دلیل این بود که اکثر میزبانها حاضر به رجیستر کردن شئ شما نیستند و به این ترتیب انحصار کار بر روی این اشیاء برای یک سری شرکتها بدست آمده بود.

اما در NET. با توجه به نکاتی که گفته شد این انحصار از بین رفت و بسیاری از برنامه نویسان عادی نیز اقدام به تهیه اشیائی دراین زمینه کردند بطوریکه در سایت رسمی ASP.NET نیز مکانی برای جمع آوری و معرفی اشیا معرفی شده توسط کاربران فراهم آمد. این اشیا از طریق آدرس

http://www.asp.net/ControlGallery/default.aspx?tabindex=2

قابل مشاهده و حتی دریافت مجانی هستند.

مشکلات شئ موجود در NET.

اما چه مسالهای باعث شد که برنامه نویسان NET. اقدام به تکمیل ویا ایجاد شئ جدیدی در این زمینه کنند؟ همانگونه که راهنمای NET. بیان کرده است، در حقیقت کلاس SmtpMail بصورت یک لفاف یا اصطلاحاً یک wrapperبرای کلاس Collaboration Data Objects for Windows 2000 (CDOSYS) می باشد. در واقع این لفاف تنها قابلیتهای سادهای از این شئ قدرتمند را پشتیبانی کرده است و بسیاری از امکانات آن در صورت

استفاده از این لفاف دیگر قابل استفاه نیست. بعنوان مثال هنگامی که سرویس دهنده SMTP شما نیاز به شناسائی (Authentication)دارد، SmtpMail هیچ امکانی جهت تعیین نام کاربری و کلمه رمز به شما نمی دهد. همچنین شما نمی توانید charset را برای نامه ارسالی خود در حالت HTML تعیین کنید. بعبارت دیگر شئ SmtpMail بسیار ساده و ابتدائی است. ولی ما در برخی مواقع نیاز داریم که قابلیتهای بیشتری از CDOSYS را در اختیار بگیریم و از آنها استفاده کنیم.

توجه : در این مقاله به نحوه کار شئ موجود در .NET Framework پرداخته نمی شود. مقاله های مربوط به کار با این شئ از همین سایت قابل دریافت است.

همین دو مورد و بالاخص مورد اول بسیاری از برنامه نویسان را بر آن داشت که به دنبال راهی برای رفع این مشکلات باشند. از جمله این تلاش ها مقاله ای

www.iranasp.net/articles/showarticle.aspx@articleid=111.htm

است که قبلاً در همین سایت[1] انتشار یافته است. همان مقاله مرا بر آن داشت تا به تکمیل آن اقدام کرده و آنرا در شکلی مناسب تر و قابل استفاده تر ارایه نمایم.

قبل از آنکه به بیان مزایا و توانایی های شئ IrMail بپردازیم، ابتدا با اشاره ای کوتاه، یکی دیگر از معضلات ارسال نامه را بررسی می کنیم.

مشکل ارسال ایمیل به زبان های غیر انگلیسی

این مشکل نیز از جمله مشکلاتی است که گریبانگیر بسیاری از سایت های غیر انگلیسی زبان است. مشکلی که قبل از آنکه به اشیا ارسال نامه مربوط باشد، به عدم وجود استانداردهای لازم در این زمینه بر می گردد. استانداردهایی

[1] www.iranasp.net

که هم مرورگرها ملزم به پشتیبانی از آن باشند و هم سرویس دهندگان ایمیل (منظور آنکه این مشکل به .NET ارتباطی ندارد).

در این مقاله به جای استفاده از کلمه "غیرانگیسی"، از نمونه ای از زبانهای غیر انگلیسی که اینجا فارسی است استفاده می شود. توجه کنید هرکجا که از زبان فارسی نام برده شد، منظور هر زبان غیرانگلیسی است که به عنوان نمونه ای از آن مطرح می شود.

این مشکل از آنجا شروع شد که برخی از برنامه نویسان قصد داشتند نامه های خود را به زبان فارسی ارسال کنند. برنامه نویسان بطور عمده برای حل این مشکل سه راه حل را درپیش داشتند:

- **ارسال نامه به زبان انگلیسی**
- **ارسال تصویری از متن فارسی**
- **و یا نهایتا به صورت پینگلیش (برای فارسی زبانان!)**

در هر سه روش مورد استفاده، عملا مشکل به صورت کامل حل نشد.

آنهایی که قصد ارسال نامه به زبان انگلیسی را داشتند با طیف گسترده کاربران غیر انگلیسی زبان مواجه می شدند و این در حالی بود که نوشتن خود نامه به زبان انگلیسی از دیگر مشکلات بود. در کل نیز ارسال نامه به زبانی غیر از زبان سایت بی سنخیت است. مانند آنکه اداره شما در ایران باشد و کلیه روابط آن بر اساس زبان فارسی و این در حالیکه کلیه مکاتبات شما به زبان انگلیسی صورت گیرد.

در روش ارسال عکس متن، که دیگر کمتر مورد استفاده است، نیز به دلیل حجم بالای نامه و عدم انعطاف پذیری آن (و این روزها وجود ویروس و ...) برای سایت های رسمی و حتی کاربران قابل قبول نیست. ضمن آنکه امکان ایجاد عکس های پویا که فرم مناسبی هم داشته باشند کار سخت و طاقت فرسایی است.

در روش سوم اصل عمل مشکل دارد و بیشتر شبیه شوخی است! شوخی ای که به دلیل عدم آشنایی کاربران به زبان انگلیسی و اینکه سرویس های کامپیوتری بطور عمده به زبان انگلیسی است، به یک جدیت تلخ تبدیل شده است. به هر حال استفاده از این روش برای سایتهایی که عملکرد آنها رسمی و جدی است بی معنی و نامناسب است. این روش شاید تنها مورد استفاده سایت های فانتزی و غیر رسمی باشد.

در این میان برخی نیز تصمیم گرفتند که نامه های خود را به زبان فارسی ارسال کنند و در همان نامه به انگلیسی (یا پینگلیش!) نحوه خواندن نامه به فارسی را ذکر می کردند .چراکه در اکثر مواقع اگرچه نامه فارسی بود ولی در مقصد فارسی دیده نمی شد ونیاز به اعمال روش هایی بود که شخص قادر باشد نامه را به فرم صحیح فارسی ببیند. اگرچه این روش نیز هم اکنون مرسوم است، اما باز هم طیف وسیعی از کاربران عادی نیز با این مورد مشکل دارند.

چرا IrMail مطرح شد؟

شما در این مقاله با دلایل نیاز به یک شئ ارسال نامه آشنا شدید و اهمیت وجود آنرا برای یک برنامه تحت وب دید. سپس روش ها و ابزارهای آماده ای که مایکروسافت در این راستا در اختیار برنامه نویسان قرار داده بود را بررسی نمودیم (البته تا نسخه .NET 1.1). سپس اشاره ای داشتیم به توانایی .NET در رفع این مشکل (اما توسط خود کاربران). نهایتا موضوع ارسال نامه های غیر انگلیسی زبان و روشهای کنونی مواجه با آنرا بررسی کردیم .

هم اکنون به شئ IrMail رسیدیم. اینکه چرا تصمیم بر آن شد که شئ‌ای برای ارسال نامه ایجاد شود، دقیقا به همان دلایل بالا و چند مورد دیگر است. بطوریکه مشکلات فوق در این شئ تا حد زیادی برطرف شد. در بخش پایانی این مقاله شئ IrMail را به طور کامل معرفی و مزیت های آن به طور کامل شرح داده خواهد شد.

IrMail، روشی جدید برای ارسال نامه در ASP.NET (بخش دوم)[1]

ارسال نامه از طریق سایت بصورت خودکار همواره برای برنامه نویسان وب موضوعی حساس و مهم بوده است. بسیاری از بخش های یک سایت حرفه‌ای مانند صفحه ارتباط با مدیر سایت، نامه‌های یادآوری کلمه رمز کاربران، ارسال خبرنامه و... همگی به نوعی از قابلیت ارسال نامه توسط یک برنامه بهره می برند. این مقاله به معرفی یک شئ جدید بنام IrMail جهت ارسال نامه در ASP.NET با تاکید بر زبان فارسی می پردازد .

در قسمت قبل این مقاله به بررسی نیاز به شئ ای برای ارسال نامه ها در برنامه های تحت وب پرداختیم. همچنین نگاهی داشتیم به روشها و امکانات موجود در این زمینه. نهایتا به اینجا رسیدیم که ابزارهای حاضر کامل نیستند و نیاز به تکمیل دارند. در صورتیکه قسمت قبل این مقاله را مطالعه نکرده اید، پیشنهاد می شود قبل از ادامه به خواندن این قسمت، آنرا مطالعه فرماید.

همانطور که اشاره شد، شئ IrMail شئ‌ای است برای ارسال نامه ها از طریق برنامه های تحت دات نت. در زیر مزیت ها و توانایی های این شئ بررسی شده است تا بتوان دیدی صحیح تر نسبت به آن پیدا کرد.

مزیت های شئIrMail

امکان ارسال نامه با قابلیتAuthentication

همانطور که ذکر شد، شئ ذاتی NET. بدلیل سادگی آن، قابلیت Authentication (شناسائی) را به ما نمی دهد. اما برخی اوقات سرویس دهنده های SMTP ما را مجبور می کنند تا قبل از ارسال نامه، خود را معرفی (login)

[1] محمود مروج

کرده و یا به اصطلاح Authenticate شویم. در IrMail این مساله بخوبی پیاده سازی شده است. به صورتی که با تنظیم دو خصیصه username و password عملیات Authentication به سادگی انجام می شود.

امکان ارسال نامه های فارسی بدون نگرانی از نحوه خوانده شدن

این مورد نیز یکی از امکانات بالقوه این شئ می باشد که در نوع خود تقریبا جدید و منحصر بفرد است. این ویژگی که تحت عنوان UTF8AnyWhere ذکر می شود، باعث می شود نامه های شما در هر جایی که UTF8 را پشتیبانی می کند حتما به صورت فارسی دیده شود. از آنجا که امروزه تقریبا تمام سیستم عامل ها از استاندارد یونیکد پشتیبانی می کنند، عدم توانایی خواندن نامه های فارسی بسیار کم می شود. بطور واضح آنکه به همان اطمینانی که شما سایت خود را بصورت یونیکد طراحی می کنید تا فارسی دیده شدن آن تضمین شود، همین تضمین نیز برای درست خوانده شدن نامه های شما بوجود می آید. به بیان ساده تر هرجا سایت شما فارسی دیده شود، این تضمین هست که نامه های شما نیز فارسی دیده شود. نکته دیگر اینکه نامه های شما در هر نوعEncoding ای که قرار داده شده باشد، بازهم فارسی دیده می شود و لزومی نیست که حتما Encoding نامه بصورت UTF8 باشد. بنابر آنچه گفته شد، نامه های شما چه درHotmail ، چه در Yahoo و چه در هر سرویس دهنده دیگری حتما فارسی دیده می شود. استفاده از این ویژگی بسیار ساده است و با True کردن property مربوطه، کلیه عملیات لازم برای این کار به صورت خودکار انجام می شود.

تضمین ارسال تمامی نامه ها حتی در صورت قطع بودن موقتی سرویس دهندهSMTP

مشکل دیگری که کما بیش سایت های مختلف با آن مواجه هستند، قطع شدن موقتی SMTP Server می باشد. مثال زیر را در نظر بگیرید. در بعضی مواقع سرویس دهنده SMTP بنا به دلایل مختلفی برای مدتی (هرچند کوتاه) قطع می شود و از آنجا که اکثر میزبانها برای یک خرابی جزیی خود را لازم به توضیح نمی بینند، بنابراین شما از این وضعیت مطلع نخواهید شد و درنتیجه قادر نخواهید بود که اقدامی را در این زمینه انجام دهید. البته همیشه هم

تقصیر بر گردن میزبانها نیست و گاهی خود سرور با مشکل مواجه می شود. شاید این مشکل در نگاه اول جدی به نظر نرسد. اما اگر در همین زمان یک شرکت بزرگ قصد داشته باشد به شما پیشنهاد همکاری دهد و از طریق صفحه "تماس با ما" و از طریق فرم تهیه شده در آن، قصد ارسال نامه ای به شما را داشته باشد چه رخ خواهد داد؟ خوب عملا این نامه هرگز ارسال نخواهد شد و شاید شما یک موقعیت استثنایی را به همین سادگی از دست بدهید!

در طراحی شئ IrMail این مشکل عملا از بین رفته است. چرا که در صورت عدم امکان ارسال نامه توسط سرویس دهنده SMTP در آن لحظه، نامه مذکور را ذخیره کرده و در تماس بعدی با سرویس دهندهSMTP ،) در صورت فعال بودن آن) سعی در ارسال مجدد نامه های ارسال نشده خواهد کرد. بدین ترتیب حتی اگر SMTP Serverبرای مدت مدیدی هم قطع باشد، دیگر شما نگران از دست دادن هیچ نامه ای نخواهید بود. این مورد برای تمام نامه هایی که از طریق این شئ ارسال و یا دریافت می شوند صدق خواهد کرد. در طراحی این فرآیند، نکات زیر رعایت شده است:

۱ -فایل حاوی نامه های ذخیره شده از نوع XML می باشد. بنابراین حتی اگر درآینده NET. بر روی سرور Linuxو یا حتی هر سرور دیگری ارایه شد، با مشکل مواجه نمی شود. چراکه XML یک استاندارد است که بر فایل های ساده متنی متکی است.

۲ -از آنجا که ظرف ذخیره سازی، یک فایل متنی ساده است، از مدیریت و کنترل دسترسی های همزمان در آن خبری نیست. بدین معنی که اگر SMTP Server فعال نبود و در آن واحد دو کاربر قصد ارسال نامه را داشتند، به احتمال زیاد منبع مورد نظر بهم می ریزد .(Not thread-safe) در اینجا نیز مشکل برطرف شد و این مدیریت با توجه به امکاناتی که NET. در کنترل منابع همزمان فراهم کرده است ایجاد شده است (برای اطلاعات بیشتر به مقاله Threadها در ASP.NET مراجعه کنید). بدین ترتیب دسترسی های همزمان مشابه آنچه در DBMS ها رخ می دهد، امکان پذیر است.

نمونه ای از نحوه استفاده از شئIrMail

از آنجا که این شئ در هر بار ایجاد نیاز به تنظیمات خاصی دارد) مثلا Username، Password، محل دخیره سازی نامه های ارسال نشده و...) که برای هر سایت منحصربفرد است، پیشنهاد می شود که به صورت زیر عمل شود .

یک کلاس از کلاس IrMail.Mail را به ارث برده و تنظیمات خاص خود را در سازنده (Constructor) آن قرار دهید. در هنگام نیاز به ارسال نامه فقط کافی است یک نمونه از کلاس ارث بری شده را ایجاد کنید (نمونه ای از این ارث بری در زیربه زبان VB.NET نشان داده شده است).

```
Public Class MySmtpMail
Inherits IrMail.Mail

Public Sub New()

MyBase.New(HttpContext.Current.Server.MapPath("") )
```

مقدار پارامتر سازنده کلاس IrMail.Mail محل مکانی را نشان می دهد که می خواهید نامه های ارسال نشده در آنجا ذخیره شود. در این مثال محل دخیره سازی را برابر با دایرکتوری ریشه سایت قرار دادیم . شما می توانید جهت امنیت بیشتر، این محل را تغییر و به مکان دیگری منتقل کنید (بطور مثال دایرکتوری محل ذخیرهDB ها که معمولا fpdbمی باشد).

```
Me.SMTPserver = "YourSMTPServer"
Me.PickUpDirectory = "c:\inetpub\mailroot\pickup"
```

خصیصه فوق (PickUpDirectory) مورد نیاز SMTP Server می باشد.

```
Me.UserName = "YourUsername"
Me.Password = "YourPassword"
```

موارد فوق از آن جهت نیاز است که بتوانید نامه های ارسالی خود را از طریق سرویس دهنده هایی که نیاز به Authenticationدارند نیز ارسال نمائید. در صورتیکه سرویس دهنده شما نیازی به Authentication ندارد، نیازی به تعیین نام کاربری و کلمه رمز ندارید.

```
Me.UTF8AnyWhere = True
```

این خصیصه باعث می شود نامه شما در همه جا فارسی دیده شود. در صورتی که نامه شما انگلیسی است، می توانید این خصیصه را false قرار دهید. مقدار پیش فرض آن نیز false است.

```
Me.LoadSharedObject4WebApplication(HttpContext.Current)
```

استفاده از دستور فوق همیشه اجباری است و باید به همین صورت باشد. در صورتی که دستور فوق ذکر نشود، مکن است شئ شما با مشکل مواجه گردد.

```
End Sub
End Class
```

حال هرجا در سایت خود قصد داشتید که نامه ای را ارسال کنید، کافیست یک نمونه از کلاس فوق را ساخته و سایر خصوصیات آنرا تنظیم نمایید. به عنوان مثال:

```
Dim ObjMailer As New MySmtpMail ()
ObjMailer.From = "EveryOne@Server.Com"
ObjMailer.To= "EveryOne@AnotherServer.Com"
ObjMailer.Subject = "New Email Object"
```

```
ObjMailer.Body= "The body of your mail in each language"
ObjMailer.Send()
```

و بدین ترتیب نسبت به ارسال نامه خود اقدام کنید. نمونه برنامه کامل را می توانید از آدرس http://www27.brinkster.com/mahmoud690/irmail/sample.htmدریافت کنید.

توجه: متد IrMailMail.Send دارای یک پارامتر اختیاری است که به صورت پیش فرض true می باشد. اگر آنرا به false تنظیم کنید بدین معنی است که اگر smtp server فعال نبود، لازم نیست آنرا ذخیره کند (غیر فعال کردن ویژگی آخر). این مورد در مواردی کاربرد دارد که ارسال شدن یا نشدن نامه برای شما آنقدر اهمیت نداشته باشد.

نکته آخر: ... مجانی به همراه کد (Free And OpenSource) !

این شئ برای همه به صورت مجانی قابل دریافت و استفاده است. در ضمن کدهای این کلاس به صورت مجانی نیز قابل دریافت است. این بدین معنی است که شما می توانید کدها را مشاهده کرده و آنها را مطابق میل خودتان دستکاری و استفاده نمایید .(OpenSource) این شئ بر روی لیست اشیا مایکروسافت نیز ثبت شده است و امیدوار هستیم این شئ به عنوان اولین شئ ایرانی که در این لیست ثبت شده باعث گردد دوستان دیگر نیز نسبت به ثبت اشیا خود در آن اقدام کنند، چراکه این لیست به عنوان یک کاتالوگ معتبر اشیا NET. در نزد برنامه نویسان مطرح است و مسلما وجود اشیا مختلف از برنامه نویسان ایرانی و با نام ایران، اعتبار سایت و برنامه نویسان ایرانی را در نزد دیگران افزایش خواهد داد.

نهایت آنکه اهداف من از Opensource قرار دادن کلاس فوق به صورت زیر است:

- مسلما این شئ نه آنقدر کامل و بی عیب است و نه دانش من در این زمینه به اندازه کافی کامل و تخصصی است که بخواهم آنرا بصورت تجاری ارایه دهم .ضمن آنکه

امکانات آنرا نیز ندارم.

-با opensource قرار دادن آن، امکان کمک و یاری گرفتن از سایر دوستان را نیز فراهم کرده ام. ضمن آنکه اینگونه برنامه ها امتحان خود را پس می دهند.

-و نهایت آنکه باز کردن کدهای NET. از روی فایل های Exe یا Dll آنها بسیار ساده است!

بطور خلاصه آنکه این شئ متعلق به خود شماست و از همینجا از تمامی دوستانی که در این زمینه فعالیت دارند درخواست دارم که با استفاده از این شئ و یافتن نقاط ضعف و کاستی ها و کمبودهای آن من را در جهت تکمیل آن یاری دهند. در مقابل هم تنها کاری که از دست من در برابر لطف این دوستان برمی آید این است که نام آنها را به عنوان یکی از برنامه نویسان و یاdebugger های آن ذکر کنم .

آدرس سایت انگلیسی: http://www27.brinkster.com/mahmoud690/irmail

آدرس مستقیم Download شئ کامپایل شده:

http://www27.brinkster.com/mahmoud690/irmail/downloadAssembly.htm.htm

آدرس مستقیم Download کدهای برنامه:

http://www27.brinkster.com/mahmoud690/irmail/Download-source.htm

آدرس این شئ در کنترل گالری سایت رسمی:ASP.NET

http://www.asp.net/ControlGallery/ControlDetail.aspx?Control=1918&tabindex=2

تایید و شناسایی کاربران بر اساس فرمهای وب در برنامه‌های ASP.NET[1]

برنامه‌های وب ASP.NET می توانند از سه روش مختلف برای شناسائی و تایید کاربران وب سایت استفاده کنند. این روش ها عبارتند از: Windows Authentication ، Forms-Based Authentication و Passport Authentication. آیا می دانید شناسایی کاربران در سایت Iranasp.net به کدام روش انجام می گیرد؟ یا اگر وارد سایت Iranasp.net شده ولی هنوز شناسایی نشده باشید و بخواهید به یک صفحه امن وارد شوید چه اتفاقی می افتد؟ این مقاله به این نوع سوالات پاسخ می دهد و شما را قدم به قدم با چگونگی شناسایی کاربران آشنا می کند .

لینک دریافت کد: www.iranasp.net/download/evalizadeh005.zip

برنامه های وب ASP.NET می توانند از سه روش مختلف برای شناسائی و تایید کاربران وب سایت استفاده کنند. این روش ها عبارتند از Windows Authentication :. Forms-Based Authenticationو Passport Authentication.آیا می دانید شناسایی کاربران در سایت Iranasp.net به کدام روش انجام می گیرد؟ برای مثال اگر وارد سایت Iranasp.net شده ولی هنوز شناسایی نشده باشید و درخواست یک صفحه امن مانند http://www.iranasp.net/Members/postnews.aspxکه مخصوص کاربران است را بکنید نمی توانید به این صفحه دسترسی داشته باشید. در این حالت شما به صفحه ای که حاوی فرم Login سایت است هدایت می شوید زیرا این سایت کاربران خود را به روش Forms-Based Authentication شناسایی می کند. در این مقاله مراحل لازم برای تایید و شناسایی کاربران یک وب سایت بر اساس فرمهای وب، یعنی Forms-Based Authentication قدم به قدم مورد بررسی قرار می گیرد.

[1] احسان ولیزاده

این مقاله شامل بخشهای زیر می باشد:

•ملزومات

•ایجاد یک برنامه ASP.NET با استفاده از Visual Studio.NET (زبان Visual Basic.NET) :

•پیکربندی تنظیمات امنیتی در فایل Web.config

•ایجاد یک جدول در پایگاه داده برای ذخیره جزئیات و مشخصات کاربران

•ایجاد یک فرم Logon.aspx که محل شناسایی کاربران می باشد

•انجام کدنویسی برای برخی از رویدادها که بتوان کاربران را اعتبارسنجی کرد

•ایجاد یک صفحه default.aspx

•موارد دیگر

•منابع

ملزومات

در ابتدا لازم است ملزومات زیر را بر روی سیستم خود آماده کنید:

• Microsoft Visual Studio .NET
• Microsoft SQL Server
• Microsoft Internet Information Server (IIS) version 5.0 or later

ایجاد یک برنامه ASP.NET با استفاده از Visual Studio.NET زبان(Visual Basic.NET :

۱- Visual Studio .NET را باز کنید.

۲- یک برنامه وب ASP.NET را با زبان Visual Basic.NET ایجاد نمایید.

پیکربندی تنظیمات امنیتی در فایلWeb.config

این قسمت نشان می دهد که چگونه قسمتهای <authentication> و <authorization> فایل Web.configرا تنظیم کنیم تا برنامه ASP.NET ساخته شده در قسمت بالا برای تایید کاربران بر مبنای فرمها پیکربندی شود. برای این کار مراحل زیر را انجام دهید.

۱ -فایل Web.config را از پنجره Project Explorer باز کنید.

۲ -مد authentication را برابر "Forms" قرار دهید.

۳ -تگ <Forms> را قرار دهید و خصوصیات مناسب آنرا تکمیل کنید (برای دیدن لیست کامل این خصوصیات می توانید به مستندات QuickStart مراجعه کنید). قسمت authentication در فایل Web.config باید به شکل زیر باشد. می توانید همین کد را کپی کرده و از منوی Edit در Visual Studio.NET گزینه Paste as HTML را انتخاب کنید .

```
<authentication mode="Forms">
   <forms name=".ASPXFORMSDEMO" loginUrl="logon.aspx"
      protection="All" path="/" timeout="30" />
</authentication>
```

۴ -حال باید با تغییر قسمت authorization به شکل زیر مانع از ورود افراد ناشناس به صفحات شوید:

```
<authorization>
   <deny users ="?" />
   <allow users = "*" />
</authorization>
```

ایجاد یک جدول در پایگاه داده برای ذخیره جزئیات و مشخصات کاربران

این مرحله چگونگی ایجاد یک جدول در پایگاه داده، برای ذخیره نام، کلمه عبور و نقش (وظیفه) کاربران را نشان می دهد.

۱ -برنامه Notepad را باز کنید.

۲ -کد اسکریپت SQL زیر را در آن کپی کنید:

```
if exists (select * from sysobjects where id =
object_id(N'[dbo].[Users]') and OBJECTPROPERTY(id, N'IsUserTable') = 1)
drop table [dbo].[Users]
GO
```

```
CREATE TABLE [dbo].[Users] (
   [uname] [varchar] (15) NOT NULL ,
   [Pwd] [varchar] (25) NOT NULL ,
   [userRole] [varchar] (25) NOT NULL ,
) ON [PRIMARY]
GO
ALTER TABLE [dbo].[Users] WITH NOCHECK ADD
   CONSTRAINT [PK_Users] PRIMARY KEY NONCLUSTERED
   (
      [uname]
   ) ON [PRIMARY]
GO

INSERT INTO Users values('user1','user1','Manager')
INSERT INTO Users values('user2','user2','Admin')
INSERT INTO Users values('user3','user3','User')
GO

INSERT INTO Users values('user1','user1','Manager')
INSERT INTO Users values('user2','user2','Admin')
INSERT INTO Users values('user3','user3','User')
GO
```

۳ -فایل را با نام Users.sql ذخیره کنید.

۴ -SQL Server را باز کنید .فایل Users.sql را در Query Analyzer باز کنید (برای دسترسی به محیط Query Analyzer می توانید آنرا از منوی Tools در SQL Server اجرا کنید). از لیست Databases

پایگاه داده Pubs را انتخاب کنید و اسکریپت (فایل Users.sql) را اجرا کنید. این کار جدولی به نام Users را در پایگاه داده Pubs ایجاد می کند و داده های نمونه ای را در آن برای استفاده در برنامه ASP.NET قرار می دهد. می توانید به جای Pubs از پایگاه داده دیگری هم استفاده کنید که در این صورت، هنگام تعیین Connection String مربوط به آبجکت Connection باید نام آن پایگاه داده را به جای Pubs قرار دهید.

ایجاد یک فرم Logon.aspx که محل شناسایی کاربران می باشد

۱ -یک فرم به نام Logon.aspx را به پروژه اضافه کنید.

۲ -در نمای HTML فرم Logon.aspx کد زیر را بین دو تگ <Form> اضافه کنید:

```
<h3>
<font face="Verdana">Logon Page</font>
</h3>
<table>
  <tr>
    <td>UserName:</td>
    <td><input id="txtUserName" type="text" runat="server"></td>
    <td><ASP:RequiredFieldValidator ControlToValidate="txtUserName"
      Display="Static" ErrorMessage="*" runat="server"
      ID="vUserName" /></td>
  </tr>
  <tr>
    <td>Password:</td>
    <td><input id="txtUserPass" type="password" runat="server"></td>
    <td><ASP:RequiredFieldValidator ControlToValidate="txtUserPass"
```

```
        Display="Static" ErrorMessage="*" runat="server"
        ID="vUserPass" />
      </td>
    </tr>
    <tr>
      <td>Persistent Cookie:</td>
      <td><ASP:CheckBox id="chkPersistCookie" runat="server" autopostback="false"
/></td>
      <td></td>
    </tr>
</table>
<input type="submit" Value="Logon" runat="server" ID="cmdLogin"><p></p>
<asp:Label id="lblMsg" ForeColor="red" Font-Name="Verdana" Font-Size="10"
runat="server" />
```

در این فرم کاربرانی که می خواهند وارد سایت شوند شناسایی می شوند. این فایل را ذخیره کنید.

انجام کدنویسی برای برخی از رویدادها که بتوان کاربران را اعتبارسنجی کرد

این قسمت حاوی کدی است که در فایل code-behind فرم (Logon.aspx یعنی Logon.aspx.vb) قرار می گیرد.

۱ -فایل logon.aspx.vb را باز کنید.

۲ -فضانام های مورد نیاز را در فایل code-behind وارد کنید:

```
Imports System.Data.SqlClient
Imports System.Web.Security
```

۳ -یک تابع به نام ValidateUser ایجاد کنید که مشخصات کاربران را برای اعتبارسنجی در پایگاه داده جستجو می کند. برای این کار می توانید از کد زیر استفاده کنید (Connection String .موجود در این کد باید مطابق با پایگاه داده شما باشد پس در صورت لزوم باید آنرا مطابق با سیستم خودتان تغییر دهید. در اینجا نام کامپیوتر من DOTNET می باشد و SQL Server به صورت Local نصب شده است).

```
Function ValidateUser(uid As string, passwd As string) As Boolean
   Dim cnn As SqlConnection
   Dim cmd As SqlCommand
   Dim dr As SqlDataReader
   Dim retVal As Boolean = False
   cnn = New SqlConnection("Integrated Security=SSPI;Persist Security
Info=False;Initial Catalog=pubs;Workstation                ID=DOTNET;")
   cmd = New SqlCommand("Select * from users where uname = '" & uid & "'", cnn)
   cnn.Open()
   dr = cmd.ExecuteReader()
   While (dr.Read())
      If Strcomp(dr.Item("Pwd"), passwd, 1) = 0 Then
         retVal = True
      End If
   End While
   cnn.Close()
   ValidateUser = retVal
End Function
```

۴ -شما می توانید از یکی از روشهای زیر برای ایجاد کوکی شناسایی بر اساس فرمها استفاده کرده و در رویداد cmdLogin_ServerClickکاربر را به صفحه‌ی مناسب هدایت کنید. یکی از دو قطعه کد زیر را برای رویداد cmdLogin_ServerClickبنویسید:

-متد RedirectFromLoginPage را برای ایجاد خودکار کوکی شناسایی فراخوانی کنید و کاربر را به صفحه ی مناسب هدایت کنید:

```
Private Sub cmdLogin_ServerClick(ByVal sender As Object, ByVal e As
System.EventArgs) _
   Handles cmdLogin.ServerClick
   If ValidateUser(txtUserName.Value,txtUserPass.value) Then
      FormsAuthentication.RedirectFromLoginPage(txtUserName.Value, _
      chkPersistCookie.Checked)
   Else
      lblMsg.Text = "Unknown User! try again."
   End If
End Sub
```

-برچسب شناسایی (authentication ticket) را بسازید و آنرا رمزنگاری کنید. یک کوکی بسازید و آنرا به پاسخ درخواست انجام شده از صفحه اضافه کنید و کاربر را هدایت کنید. این روش امکانات زیادی را در رابطه با نحوه ساخت کوکی به شما می دهد. مثلا می توانید داده های دلخواه خود را به کوکی اضافه کنید:

```
Private Sub cmdLogin_ServerClick(ByVal sender As Object, _
   ByVal e As System.EventArgs) Handles cmdLogin.ServerClick
   If Validateuser(txtUserName.Value,txtUserPass.Value) Then
      Dim tkt As FormsAuthenticationTicket
```

```
    Dim cookiestr As String
    Dim ck As HttpCookie

    tkt = New FormsAuthenticationTicket(1, txtUserName.Value, DateTime.Now(), _
    dateTime.Now.AddMinutes(30), chkPersistCookie.Checked, "your custom data")
    cookiestr = FormsAuthentication.Encrypt(tkt)
    ck = new HttpCookie(FormsAuthentication.FormsCookieName(), cookiestr)
    if (chkPersistCookie.Checked) then ck.Expires=tkt.Expiration
    ck.Path = FormsAuthentication.FormsCookiePath()
    Response.Cookies.Add(ck)

    Dim strRedirect As String
    strRedirect = Request("ReturnURL")
    If strRedirect <> "" Then
       Response.Redirect(strRedirect, True)
    Else
       strRedirect = "default.aspx"
       Response.Redirect(strRedirect, True)
    End If
  Else
    lblMsg.Text = "Unknown User! try again."
  End If
End Sub
```

ایجاد یک صفحه default.aspx

در این قسمت یک صفحه به نام default.aspx می سازیم. این صفحه برای کاربرانی است که شناسایی شده اند و قرار است که به این صفحه هدایت شوند. اگر کاربری بدون این که شناسایی شده باشد این صفحه را درخواست کند به صفحه logon.aspx هدایت می شود.

۱ -یک فرم جدید به نام default.aspx به پروژه اضافه کرده و آنرا در نمای HTML باز کنید. کد زیر را در بین تگ های <Form> اضافه کنید :

```
<input type="submit" Value="SignOut" runat="server" id="cmdSignOut">
```

این دکمه برای خارج شدن (log out) از صفحه default.aspx استفاده می شود. با کلیک بر روی این دکمه کوکی ساخته شده از بین می رود. این فرم را ذخیره کنید.

۲ -فضانام لازم را در فایل code-behind مربوط به فرم default.aspx وارد کنید:

```
Imports System.Web.Security
```

۳ -کد زیر را برای رویداد cmdSignOut_ServerClick اضافه کنید:

```
Private Sub cmdSignOut_ServerClick(ByVal sender As System.Object, ByVal e As
System.EventArgs) _
   Handles cmdSignOut.ServerClick
   FormsAuthentication.SignOut()
   Response.Redirect("logon.aspx", True)
End Sub
```

۴ -اکنون می توانید از این برنامه استفاده کنید. همه فایلها را ذخیره کرده و برنامه را اجرا کنید.

تمام کد مربوط به برنامه همراه با اسکریپت لازم برای ایجاد جدول در پایگاه داده را می توانید از طریق همین صفحه دانلود کنید.

موارد دیگر

•ممکن است شما بخواهید که کلمه های عبور کاربران را به صورت امن و رمزنگاری شده در پایگاه داده ذخیره کنید. در این صورت می توانید از تابعی به نام HashPasswordForStoringInConfigFile که یکی از توابع سودمند کلاس FormsAuthentication می باشد استفاده کنید. به کمک این تابع می توانید کلمه های عبور را قبل از ذخیره کردن در پایگاه داده رمزنگاری کنید.

•ممکن است بخواهید اطلاعات مربوط به SQL Connection را در فایل پیکربندی (Web.config) ذخیره کنید تا راحتتر بتوانید آنرا تغییر دهید.

•ممکن است بخواهید کدهایی را برای جلوگیری از ورود هکرهایی که قصد دارند از ترکیبات مختلف کلمه های عبور برای ورود به سایت استفاده کنند، در نظر بگیرید. برای مثال می توانید منطقی را پیاده سازی کنید که فقط تعداد مشخصی اقدام ورود به سایت را قبول می کند و اگر کاربری در تعداد معینی از انجام عمل ورود نتواند وارد سایت شود، در پایگاه داده با علامت زدن نام این کاربر، دیگر این کاربر غیر فعال شده و با وارد کردن مشخصات درست نیز نتواند وارد سایت شود مگر این که با بازدید از صفحه ای دیگر از سایت یا با تماس با واحد پشتیبانی شما بتواند حساب خود را دوباره فعال کند!

•به دلیل این که کاربر با کوکی تایید اعتبار شناسایی می شود، ممکن است از Secure Socket Layer یا SSLدر برنامه استفاده کنید تا کسی نتواند کوکی و سایر اطلاعات باارزشی که در حال انتقال هستند را بدست آورد.

•تایید اعتبار بر مبنای فرمها نیازمند این است که کاربران سایت کوکی ها را در مرورگر خود فعال کنند.

•پارامتر timeout در بخش پیکربندی <authentication> مدت زمانی که کوکی تایید اعتبار دوباره ایجاد می شود را تایین می کند، شما می توانید هر مقداری را که مناسب می دانید به این پارامتر اختصاص دهید.

منابع

می توانید در آدرسهای زیر مطالب مفیدی را در مورد تایید کاربران بر مبنای فرمها و بحث امنیت در ASP.NETمشاهده کنید:

http://www.gotdot.NETcom/QuickStart/aspplus/default.aspx?url=/quickstart/aspplus/doc/formsauth.aspx

http://msdn.microsoft.com/library/en-us/cpguidnf/html/cpconcookieauthenticationusinganxmlusersfile.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpconaspnetwebapplicationsecurity.asp

http://msdn.microsoft.com/library/dotnet/cpref/frlrfsystemwebsecurity.htm

http://msdn.microsoft.com/library/en-us/cpguide/html/cpconaspnetconfiguration.asp

http://msdn.microsoft.com/library/default.asp?url=/library/en-us/cpguide/html/cpgrfaspnetconfigurationsections.asp

http://msdn.microsoft.com/library/en-us/dnbda/html/authaspdot.NETasp

صفحه نوردی در ASP.NET[1]

حتما تاکنون بعنوان یک برنامه نویس وب به مقوله گزارشگیری برخورد نموده اید. مثلا از شما خواسته اند که لیست فلان اقلام داده ای مانند لیست کالا، لیست سفارشات و... را تهیه کنید .در بسیار از موارد اگر تعداد این اقلام زیاد باشد معمولا یک لیست بلند بالا را بر روی صفحه مانیتور از شما قبول نمی کنند. از طرف دیگر خود شما هم بعنوان یک برنامه نویس حرفه‌ای که برای کیفیت کار خود ارزش قائل هستید نباید براحتی از کنار این مساله بگذرید. در یک کلام منظور این است که آن لیست بلند بالا را صفحه بندی کرده و صفحه به صفحه نشان دهید .

لینک دریافت کد: www.iranasp.net/download/webtech047.zip

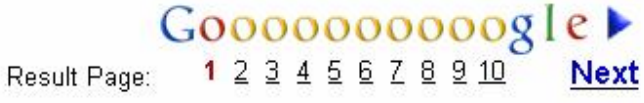
فرض کنید که به سایت گوگل رفته اید و کلمه iranasp را جستجو کرده اید. من این کار را کردم و در زمان نوشتن این مقاله دقیقا تعداد ۱۶۲۰ مورد یافت شد! مقدار زیادی نیست ولی اگر قرار بود که گوگل همه این موارد را تنها در یک صفحه به من نشان دهد و نه ۱۶۲ صفحه (با فرض در هر صفحه ده مورد)، گوگل دیگر به این همه ثروت و شهرت نمی رسید!! شاید فکر کنید من دیوانه شده ام و اصولا چه ربطی میان جستجوی کلمه iranasp و ثروت و قدرت امپراطوری گوگل وجود دارد؟ بله هیچ ربطی ندارد ولی تصور کنید که روزانه میلیونها کاربر برای دفعات متعدد به دنبال کلمه یا کلماتی در گوگل و یا دیگر موتورهای جستجو می گردند و تعداد نتایج آنها به اندازه تعداد نتایج ما ۱۶۲۰ مورد نیست بلکه حاصل نتایج جستجوها بسیار بیشتر از اینها است. مساله این است که نشان دادن همه نتایج در یک صفحه در بیشتر مواقع نه امکان پذیر است و نه به صرفه.

[1] مدیریت سایت

بعبارت دیگر تصور کنید که گوگل می خواست صفحه ای حاوی بیش از یک میلیون نتیجه جستجو را به ما نشان دهد. آیا می دانید حجم این صفحه به بایت چقدر می شد؟ من که نمی دانم ولی هرچه باشد آنقدر هست که حوصله من و شما سر برود و دیگر منتظر نمایش صفحه نتایج نشویم و عطای آن را به لقایش ببخشیم و بر روی آن ضربدر کوچک بالای پنجره ویندوز کلیک کرده و آن را ببندیم و این همان تیر خلاص به قلب گوگل و اصولا هر سایتی است. این یعنی اینکه من و شمای نوعی آن تبلیغات درون صفحه گوگل را نمی بینیم و دیگر بر روی آن لینک که ممکن بود کلیک کنیم دیگر حتما کلیک نمی کنیم و آن کالای تبلیغ شده را که ممکن بود بخریم دیگر حتما نمی خریم و باقی قضایا و این یعنی همان چیزی که در بالا در مورد ثروت و شهرت گوگل گفتم و شما حرف من را جدی نگرفتید! گوگل تنها یک مثال آشنا بود. این مساله را به همه سایتهای وب و احتمالا محصولات نرم افزاری تعمیم دهید، خواهید دید همین "صفحه نورد" کوچک و ساده ما چه نقش مهمی دارد! امیدوارم که مطلب را گرفته باشید!

صفحه نورد

عبارت صفحه نوردی را خودم از روی عبارت Page Navigation ساختم .بیشترین معانی Navigation حول و حوش دریانوردی دور می زند و من هم معادل صفحه نوردی را برای آن انتخاب کردم. شما اگر پیشنهاد بهتری دارید حتما با من تماس بگیرید. و اما صفحه نورد، حتما در پائین صفحات گوگل بعد از جستجوی کلمه ای که نتایج زیادی بهمراه داشته است یک تصویر گوگل با تعداد O های زیاد و نیز در زیر آن تعدادی شماره را دیده اید. اگر ندیده اید(!) به تصویر زیر نگاه کنید .



شماره ها در اصل شماره صفحات نتیجه جستجو هستند و برای اینکه لیست این شماره ها خیلی طولانی نشده و از صفحه بیرون نزند، در هر لحظه تنها تعداد معینی مثلا ده تا یا بیست تا از آنها آورده شده است. شما با کلیک بر روی

شماره ۹ به نهمین صفحه نتیجه می روید و در هر صفحه ای که باشید، با کلیک بر روی کلمه Next یا Prev به ترتیب به صفحه بعد یا قبل از آن می روید.

حالا ما می خواهیم در این مقاله یک صفحه نورد آن هم از نوع فارسی آن را بسازیم. فارسی بودن یعنی اینکه شماره صفحات و نیز عبارت Next و Prev همگی به خط زیبای فارسی باشند. نمونه عملی و کاربردی همین صفحه نورد را در همین سایت IranASP.NET در پائین صفحات خبرها، نتایج جستجو و یا لیست مقالات می بینید.

چرا صفحه نورد؟

شاید برای شما این سوال پیش بیاید که وقتی ما از ASP.NET استفاده می کنیم و از کنترل DataGrid آن برای گزارشگیری استفاده می کنیم که خودش امکان صفحه نوردی را به ما می دهد دیگر چه نیازی به این صفحه نورد شماست؟ جواب این است:

الف) DataGrid کنترل سنگین و حجیمی است و برای صفحاتی که سرعت دریافت در آنها حرف اول را می زند چندان مناسب نیست.

ب) برای استفاده از قابلیت صفحه نوردی DataGrid شما مجبورید در هر رفت و برگشت صفحه هر بار کلیه رکوردهای نتیجه خود را مثلا هر ده هزارتا را هر بار به خورد DataGrid بدهید تا او فقط مثلا ده تا را انتخاب و بر روی صفحه نشان دهد و بقیه را دور می ریزد. برای اینکار شما یا هر بار باید به سراغ پایگاه داده خود رفته و تقاضای خود را برای دریافت ده هزار رکورد مورد نظر بدهید و این یعنی مصرف بالای منابع، کندی دسترسی به اطلاعات و طولانی شدن زمان نمایش صفحه و در اکثر مواقع از کار افتادن برنامه یا سایت شما در حالاتی که حجم داده خیلی بالا بوده و افراد زیادی به سایت شما آمده اند. اگر هم خوش شانس باشید بگونه‌ای که همه کاربران سایت شما قرار است تنها به یک سری رکورد مشخص همواره نیاز داشته باشند و شما بخواهید آن تعداد را در حافظه سرورتان اصطلاحا Cacheکنید، بازهم ممکن است حافظه زیادی مصرف کنید و دچار مشکلات دیگری شوید.

یکی از بهترین و کارآ ترین کنترلهای ASP.NET برای نمایش اطلاعات هم جنس، کنترل Repeater می باشد. این کنترل ضمن سبکی و نیاز کمتر به پردازش سمت سرور، دست برنامه نویس را برای هر نوع طراحی صفحه در نمایش اطلاعات باز می گذارد. تنها مشکل این است که خودتان باید مساله صفحه نوردی آن را حل کنید. چیزی که این مقاله به دنبال است. لازم بذکر است که این صفحه نورد مورد نظر تنها برای استفاده در کنار Repeater طراحی نشده است و می توان آن را در کنار سایر تکرار کننده ها مانند DataGrid و DataList نیز استفاده نمود. بعبارت دیگر این صفحه نورد هیچگونه وابستگی به هیچ نوع کنترلی در ASP.NET ندارد و می توان در هرکجا از آن استفاده کرد.

و اما اصل مطلب...

خوب تا اینجا هرچه گفتیم مقدمه چینی و حرف بود و احتمالا حوصله شما هم سر رفته است و باید زودتر برویم سراغ اصل مطلب یعنی کد، والا آن ضربدر کوچک بالای پنجره، ما را هم تهدید می کند!

این صفحه نورد بصورت یک کنترل کاربری یا یک User Control در نظر گرفته شده است تا بتوان براحتی در جاهای مختلف یک وبسایت از آن استفاده کرد. کد HTML این کنترل به صورت زیر است:

```
<%@ Control Language="c#" AutoEventWireup="false"
Codebehind="PageNavigation.ascx.cs" Inherits="YourNamespace.PageNavigation"
TargetSchema="http://schemas.microsoft.com/intellisense/ie5" %>
<P>
  <TABLE cellSpacing="1" cellPadding="3" width="100%" border="0">
    <TR>
      <TD align="middle" dir="rtl">
        <asp:Label id="lblNav" runat="server"></asp:Label>
      </TD>
```

```
    </TR>
  </TABLE>
</P>
```

همانگونه که می بینید اصل کد بالا عبارت است از تعریف یک کنترل از نوع Label و سایر متعلقات بنا به سلیقه شما قابل تغییر است. بخش اصلی این کنترل در قسمت کدبرنامه یا codebehind آن است. جهت اختصار همه کدبرنامه آن را اینجا نمی آورم و می توانید کد کامل آن را از بالای همین مقاله دریافت کنید. تنها به معرفی چند متد و متغیر مهم آن می پردازیم.

- CurrentPage :متغیرنشان دهنده شماره صفحه فعلی .این متغیر توسط برنامه استفاده کننده جهت انتقال به روال ذخیره شده‌ی پایگاه داده، بکار می رود.
- PageSize :متغیر نشان دهنده تعداد رکورد در هر صفحه. این متغیر توسط برنامه استفاده کننده جهت انتقال به روال ذخیره شده‌ی پایگاه داده، بکار می رود.
- TotalRecords :متغیر نشان دهنده تعداد کل رکوردهای یافته شده. این متغیر توسط روال ذخیره شده‌ی پایگاه داده مقدار دهی می گردد.
- InitNav :از این متد جهت مقداردهی اولیه صفحه نورد پیش از ارسال متغیرهای فوق به روال ذخیره شده پایگاه داده استفاده می گردد.
- ShowNav: این متد جهت نمایش صفحه نورد در آخرین مرحله، فراخوانی می شود.
- CssClass: از این متغیر جهت تعیین کلاس CSS این کنترل استفاده می شود.

- InActiveCssClass :از این متغیر جهت تعیین کلاس CSS موارد غیرفعال این کنترل استفاده می شود (مانند کلمه "قبلی" هنگامی که در صفحه اول هستیم).

نحوه استفاده

این کنترل، همانند یک User Control معمولی بکار می رود. اگر در Visual Studio .NET هستید، در پنجره Solution Explorer برروی آن کلیک کرده و بصورت drag-and-drop آن را در صفحه مورد نظر قرار دهید. همچنین می توانید کد زیر را بصورت دستی در صفحه مورد نظر تایپ کنید:

```
<%@ Register TagPrefix="uc1" TagName="PageNavigation"
Src="PageNavigation.ascx" %>
...
<uc1:PageNavigation id="pnPageNavigate" runat="server"></uc1:PageNavigation>
```

حال جهت تعیین مقادیر PageSize و CssClass و یا InActiveCssClass کد فوق را بصورت زیر تغییر دهید:

```
<uc1:PageNavigation id="pnPageNavigate" PageSize="10" runat="server"
CssClass="yourclassname"
InActiveCssClass="anotherclassname"></uc1:PageNavigation>
```

البته متغیرهای فوق بترتیب دارای مقادیر ۱۰ و "" و "" می باشند و لزومی به مقداردهی آنها نیست. همچنین این مقادیر را می توانید از درون کدبرنامه استفاده کننده هم مقداردهی کنید.

حال لازم است تا شئ ای از نوع کلاس صفحه نورد یعنی PageNavigation در کدبرنامه استفاده کننده تعریف نمائیم تا بتوانیم به این کنترل از درون کدبرنامه استفاده کننده دسترسی داشته باشیم. برای این کار بصورت زیر عمل می کنیم:

```
protected YourNamespace.PageNavigation pnPageNavigate;
```

حال با استفاده از این متغیر مرجعی، به متغیرهای اساسی کنترل صفحه نوردمان دسترسی داریم. در کد زیر نحوه بکارگیری آن را می بینیم.

```
this.pnPageNavigate.InitNav ();
DataSet ds = GetPagedAllNews (this.pnPageNavigate.CurrentPage ,
                this.pnPageNavigate.PageSize,
                out this.pnPageNavigate.TotalRecords);
this.repNews.DataSource = ds;
this.repNews.DataBind();
this.pnPageNavigate.ShowNav();
```

در کد فوق نحوه استفاده از متدهای InitNav و ShowNav را می بینیم. همچنین نحوه ارسال متغیرهای CurrentPageو PageSize و TotalRecords مشخص شده است. تا اینجا، کار ما با کنترل صفحه نورد تمام شده است. مساله مهم دیگر نحوه بازیابی اطلاعات از پایگاه داده و استفاده از مقادیر متغیرهای فوق در قالب متد کمکی GetPagedAllNewsاست. این متد را بصورت زیر نوشته ایم:

```
private DataSet GetPagedAllNews (int CurrentPage, int PageSize, out int
TotalRecords)
{
   DataSet ds = new DataSet ();
```

```
    // Create Instance of Connection and Command Object
    SqlConnection myConnection = new SqlConnection(ConnectionString);
    SqlCommand myCommand = new SqlCommand("procGetPagedAllNews",
myConnection);
    // Mark the Command as a SPROC
    myCommand.CommandType = CommandType.StoredProcedure;
    myCommand.Parameters.Add ("@CurrentPage", CurrentPage);
    myCommand.Parameters.Add ("@PageSize", PageSize);
    myCommand.Parameters.Add (new SqlParameter ("@TotalRecords",
SqlDbType.Int));
    myCommand.Parameters ["@TotalRecords"].Direction = ParameterDirection.Output;
    // Create a new SqlDataAdapter object for the News table
    SqlDataAdapter da = new SqlDataAdapter(myCommand);
    // Populate the ds with the data
    da.Fill(ds);
    TotalRecords = Convert.ToInt32 (myCommand.Parameters
["@TotalRecords"].Value);
    return ds;
}
```

بخش مهم دیگر در اینجا همان کد روال ذخیره شده procGetPagedAllNews می باشد که وظیفه واکشی رکوردهای مورد نظر را بصورت صفحه به صفحه دارد. از آنجا که شرح این روال در این مقال نمی گنجد تنها به آوردن کد آن اکتفا کرده ام. ابتدا یک جدول در SQL Server بنام News ساخته و دو عدد فیلد بصورت زیر برای آن تعریف کنید:

```
NewsID int
NewsTitle nvarchar (256)
```

سپس یک روال ذخیره شده بصورت زیر در پایگاه داده تان بسازید:

```
CREATE PROCEDURE dbo.procGetPagedAllNews
(
   @CurrentPage int,
   @PageSize int,
   @TotalRecords int output
)
AS

SET NOCOUNT ON

--Create a temp table to hold the current page of data
--Add an ID column to count the records
CREATE TABLE #TempTable
(
   ID int IDENTITY PRIMARY KEY,
   NewsID int,
   NewsTitle nvarchar (256),
)
--Fill the temp table with the News data
INSERT INTO #TempTable
(
```

```
    NewsID,
    NewsTitle,
)
SELECT NewsID, NewsTitle
FROM News

--Create variable to identify the first and last record that should be selected
DECLARE @FirstRec int, @LastRec int
SELECT @FirstRec = (@CurrentPage - 1) * @PageSize
SELECT @LastRec = (@CurrentPage * @PageSize + 1)
--Select one page of data based on the record numbers above
SELECT *
FROM
#TempTable
WHERE
ID > @FirstRec
AND
ID < @LastRec
--Return the total number of records available as an output parameter
SELECT @TotalRecords = COUNT(*) FROM #TempTable
```

حال سایر تنظیمات مانند ConnectionString در کدبرنامه صفحه اصلی را مطابق پایگاه داده خود قرار دهید و سایر تنظیمات را انجام داده و برنامه را کامپایل و اجرا کنید.

تولید صفحات استاتیک به کمک XML و ASP.NET (بخش اول)[1]

شاید برای شما جالب باشد که بدانید سایت هایی بزرگی چون CNN یا news.com چگونه صفحات خود را تولید می کنند و اینکه چرا پسوند صفحات آنها مثلا بجای asp یا aspx یا php وغیره، html است؟ این مقاله را بخوانید تا با یکی از متداول‌ترین و مناسب‌ترین راه‌های تولید صفحات وب بصورت استاتیک برای سایت‌های با محتوای دینامیک مانند سایت های خبری یا مقاله‌ای آشنا شوید .

اشاره:

این مقاله به تشریح یکی از متداول‌ترین و مناسب‌ترین راه‌های تولید صفحات وب برای سایت‌های با محتوای دینامیک از جمله سایت‌های ناشر خبر و مقاله به کمک XML می‌پردازد. روشی که در این مقاله توضیح داده شده برای تمام فناوری‌های ایجاد صفحات دینامیک از جمله ASP.NET ، PHP، JSPو ColdFusion قابل اجراست اما در این مقاله تنها به روش پیاده سازی تکنیک مورد بحث با استفاده از ASP.NET اشاره شده است. برای اینکه بتوانید از این مقاله حداکثر استفاده را داشته باشید، لازم است قبلا با مبانی XML و XSL آشنایی نسبی داشته باشید ولی برای آندسته از خوانندگان که اطلاعات خیلی کمی دراین باره دارند، توضیحات مقدماتی آورده شده است.

صورت مساله

اگر یک طراح و برنامه‌نویس صفحات وب باشید، حتما تاکنون متوجه شده‌اید که یکی از روش‌های متداول برای نمایش محتوای اطلاعاتی در سایت‌های خبری اینست که به ازای هر خبر یا مقاله‌ی قابل نمایش در صفحات سایت، یک رکورد متناظر در پایگاه داده‌های (پایگاه داده) سایت قرار دهیم و اطلاعات درون فیلدهای هر رکورد را به محض درخواست بازدیدکننده، داخل یک صفحه Template از جنس ASP یا ASP.NET و غیره نمایش دهیم. اما اگر

[1] بهروز نوعی پور

احیانا به ساختار سایت‌های خبری بزرگ مانند CNN و news.com توجه کرده باشید، ممکن است از خود پرسیده باشید> چطور تعداد صفحات وب دینامیک این سایت بسیار اندک یا در حد صفر است؟ <چرا بعضی از سایت خبری بزرگ برای نمایش اخبار و مقالات خود از صفحات html به جای asp و aspx و php و jsp استفاده می‌کنند؟ چه رازی بر این شیوه حاکم است و اصولا این روش چه مزیتی دارد؟

به زبان دیگر، سوال اساسی اینست که آیا برای نمایش محتوای دینامیک در یک سایت وب، حتما مجبور به استفاده از صفحات دینامیک مانند ASP.NET و PHP و غیره هستیم؟ یا اینکه ترفند دیگری هم وجود دارد؟

بررسی موضوع

در واقع پاسخ مثبت است. بسته به اینکه منطق شما برای تولید صفحات خبر و مقاله چگونه باشد، تکنیک‌های متفاوتی برای این منظور وجود دارد که تنها یکی از آنها بکارگیری صفحات دینامیک است. روشهای دیگری هم برای نمایش محتوای دینامیک در سایت وجود دارند که ما یکی از آنها را که بهتر به نظر می‌رسد، بررسی خواهیم کرد .اما شاید یادآوری این نکته بد نباشد که ببینیم روش مرسومی که غالبا سایت‌های کوچک و متوسط خبری بکار می‌برند (و مورد نظر این مقاله هم نیست) چگونه عمل می‌کند .

در روش متداول که آسان ترین روش نیز هست، شما به ازای تمام مقالات و اخباری که فرمت مشابهی دارند، فقط و فقط یک صفحه نمایش دهنده خبر یا مقاله، آنهم از جنس ASP.NET یا ASP می‌سازید و به کمک آن، محتوای خبر یا مقاله مورد نظر را نمایش می‌دهید. شما در حقیقت از روش رندرکردن (Render) صفحه در زمان نمایش استفاده می‌کنید .

مهمترین کاری که باید در این تکنیک صورت گیرد، انتقال مقدار فیلد کلیدی هر رکورد (معمولا فیلد id) به صفحه Template مذکور و خواندن اطلاعات همان رکورد از طریق یک فرمان SQL می‌باشد. به عنوان مثال با اجرای URLزیر در یک سایت فرضی: article.aspx?id=102632

مقدار id مساوی عدد ۱۰۲۶۳۲ به عنوان کلید رکورد به صفحه ASP.NET انتقال یافته و در داخل صفحه، یک جا به کمک فرمانی مانند عبارت:

SELECT * FROM Articles WHERE ID=@id

اطلاعات فیلدهای مربوط به این رکورد را می‌خوانیم و سپس در طول صفحه از طریق کدهایی مانند عبارت زیر:

```
<#% DataBinder.Eval( Container, “DataItem.ArticleTitle” ) %>
```

داخل صفحه Template نمایش می‌دهیم.

چنانکه ملاحظه می‌کنید، در این روش هربار که کاربری قصد مشاهده خبر یا مقاله مورد نظر (با کد ۱۰۲۶۳۲) را داشته باشد، یکبار فرآیند فوق اجرا شده و صفحه دینامیک رندر می‌شود و سپس برای بازدیدکننده نمایش داده میشود.

در روش دیگری که مقصود این مقاله است، ما مایل هستیم به ازای هر خبر یا مقاله، یک صفحه استاتیک متمایز داشته باشیم، طوری که کاربر برای مشاهده مطلب مورد نظر، به عنوان مثال با کلیک کردن روی صفحه‌ای با نام 102632.htmبتواند آن را مشاهده کند. در این روش باید تکنیکی را پیدا کنیم که قبلا به ازای تمام رکوردهای خبر و مقاله درون بانک اطلاعاتی، یک صفحه وب تولید و روی server ذخیره کرده باشیم.

مزایای تولید صفحات استاتیک از روی رکوردهای پایگاه داده

قبل از هرچیز می‌خواهیم بدانیم اصولا فایده روش مذکور چیست و درحالی که خیلی ساده می‌توانیم از طریق ترفند article.aspx?id=102632 یک خبر یا مقاله خاص را به سادگی نمایش دهیم، چرا باید به ازای هر خبر یک صفحه جداگانه وب تولید کنیم؟

۱ -صفحات استاتیک سریعتر اجرا می‌شوند.

مهمترین دلیل برای روی آوردن به این روش آنست که به این ترتیب سرعت نمایش صفحات سایت شما بالا می‌رود زیرا صفحات دینامیک مانند ASP.NET و PHP نیاز به رندر شدن دارند درحالی که صفحات HTML برای نمایش روی مرورگر کاملا آماده هستند. گذشته از اینها اگر یک سایت خبری پربیننده باشد، تعدد پردازش‌های دینامیک، فشار زیادی را روی سرور ایجاد می‌کند که در مجموع راندمان آن را پایین می‌آورد. حتی اگر سایت شما بیننده اندکی داشته باشد، در صورتی که از روش Shared Hosting استفاده کنید، توانایی سرور برای همان تعداد اندک بازدیدکننده چندان بالا نخواهد بود و شما در همان میزان اندک نیز شاهد کاهش راندمان سایت متناظر با افزایش تعداد بازدیدکنندگان خواهید بود.

۲- Loadروی پایگاه داده کم می‌شود.

حسن بزرگ دیگری که از بکارگیری روش توصیه شده در این مقاله حاصل می‌شود آنست که فشار روی پایگاه داده کم می‌شود. به عنوان یک توصیه کلی و یک اصل تجربه شده همواره به خاطر داشته باشید که هرچه استفاده از پایگاه داده ها در یک سایت کمتر باشد بهتر است! این هنر نیست که شما برای تکمیل هر فرآیند کوچکی در سایت خود از پایگاه داده استفاده کنید زیرا افزایش میزان استفاده از بانک اطلاعاتی موجب کند شدن سرعت نمایش صفحات از یک سو و نیز بالا رفتن میزان منابع سیستمی مورد نیاز برای پردازشهای دینامیک (resources) از سوی دیگر می‌شود. بنابراین همواره در جستجوی روش هایی باشید که از پایگاه داده‌کمتر استفاده شود. استفاده موجز و مختصر و موثر از پایگاه داده یک روش صد در صد توصیه شده از سوی برنامه نویسان باتجربه است.

۳ -وابستگی سایت شما به پایگاه داده کم می‌شود.

به طور کلی وقتی تعداد صفحات دینامیک یک سایت کمتر می‌شود، وابستگی آن به پایگاه داده نیز کمتر می‌شود و این به معنی کاهش منابع اشکال زا در هنگام وقوع خطاست. به عنوان مثال سایتی که شدیدا به پایگاه داده ها وابسته است در صورتی که برای چند دقیقه پایگاه داده دچار مشکل یا down شود، ممکن است دهها بازدیدکننده را با خطا یا exception مربوط به در دسترس نبودن بانک اطلاعاتی مواجه کند. سایتی که از صفحات استاتیک بیشتر استفاده می‌کند کمتر در این مواقع دچار اشکال و نارسایی می‌شود.

۴ -جستجوی صفحه برای ماشین‌های جستجو آسان می‌شود.

دقت کنید که برخی از موتورهای جستجو، چه آنهایی که روی کل اینترنت کار می‌کنند و چه آنهایی که روی یک سایت مشخص عمل می‌کنند، با برخی پارامترهای انتقال داده شده به یک صفحه دینامیک مشکل دارند. بهترین و سریع‌ترین روش ممکن برای ایندکس شدن صفحات وب یک سایت در یک موتور جستجو اینست که آن صفحه اساسا به صورت استاتیک باشد.

۵ -استناد به URL یک خبر یا مقاله خاص برای مردم آسانتر می‌شود.

فراموش نکنید که وقتی آدرس یک خبر یا مقاله درسایت شما به صورت article.aspx?id=102632&cat=214&sessionid=423442 و مشابه اینها (یا در اغلب اوقات حتی پیچیده‌تر) است، به خاطر سپاری آدرس و یا درج آن در کتاب‌های و مجلات و روزنامه‌ها سخت‌تر می‌شود. درحالی که آدرس یک صفحه به صورت 102632.htm خیلی راحت‌تر است و احتمال اینکه موقع نوشتن یا تایپ یا حتی به خاطر سپردن آدرس، اشتباهی صورت بگیرد کمتر است.

۶ -آرشیو کردن اخبار و مقالات بهینه‌تر می‌شود.

یادتان نرود که اگر قرار باشد سایت شما برای مدت چند سال فعالیت کند آنوقت باید حجم عظیمی از اطلاعات را به صورت دینامیک داخل پایگاه داده نگهداری کنید. اینکار هم هزینه شما را بالا می‌برد (بخصوص اگر از SQL Serverیا MySQL یا ORACLE وDBMS های دیگر استفاده کرده باشید) و نگهداری‌اش سخت می‌شود. اگر مهمترین کاربرد بانک اطلاعاتی در سایت شما نگهداری محتوای اخبار و مقالات است، بهتر است فقط آخرین x رکورد را داخل پایگاه داده دم دست نگه دارید و بقیه رکوردها را از بانک اطلاعاتی خارج، و به طریقی که صلاح می‌دانید (مثلا ذخیره کردن به صورت صفحات استاتیک وب) آرشیو کنید. به این ترتیب رکوردهای قدیمی شما خیلی ساده از طریق جابجا کردن صفحات وب، قابل انتقال خواهند بود و اگر زمانی تصمیم بگیرید Host خود را عوض کنید، برای جابجا کردن داده‌های درون پایگاه داده مکافات نخواهید داشت!

۷ -برای نمایش صفحات سایت به Host ارزان‌تری نیاز خواهید داشت.

به کمک روش تولید مکانیزه صفحات استاتیک می‌توانید تمام یک سایت را با صفحات معمولی HTML بسازید، بدون اینکه نیازمند استفاده از قابلیت‌های یک web server خاص باشید. مثلا اگر از یک میزبان ارزان قیمت استفاده می‌کنید که به شما اجازه استفاده از امکانات پیشرفته‌ای چون پایگاه داده را نمی‌دهد، می‌توانید سایت خود را به طور کامل به صورت استاتیک بسازید. به این ترتیب که روی یک دستگاه خانگی سایت خود را شبیه‌سازی می‌کنید و صفحات استاتیک را با روش توضیح داده شده در این مقاله تولید می‌کنید و سپس حاصل کار را در دوره‌های زمانی مشخص به میزبان سایت اصلی خودتان upload می‌کنید. البته اگر سایت خود را تمام استاتیک بسازید، امکان پیاده سازی قابلیت‌هایی مانند جستجو (منظور خود صفحه جستجوگر است نه صفحاتی که مورد جستجو قرار می‌گیرند) را نخواهید داشت.

تولید صفحات استاتیک به کمک XML و ASP.NET (بخش دوم)[1]

شاید برای شما جالب باشد که بدانید سایت هایی بزرگی چون CNN یا news.com چگونه صفحات خود را تولید می کنند و اینکه چرا پسوند صفحات آنها مثلا بجای asp یا aspx یا php وغیره، html است؟ این مقاله را بخوانید تا با یکی از متداول‌ترین و مناسب‌ترین راه‌های تولید صفحات وب بصورت استاتیک برای سایت‌های با محتوای دینامیک مانند سایت های خبری یا مقاله‌ای آشنا شوید .

لینک دریافت کد: www.iranasp.net/download/behrooznp002.zip

پس از بررسی مزایای تولید صفحات استاتیک به کمک XML و ASP.NET در قسمت اول این مقاله، به چگونگی انجام این کار می پردازیم.

تولید مکانیزه صفحات استاتیک به کمکXML

اکنون روش مورد نظر در این مقاله را بررسی می‌کنیم. در این روش به جای اینکه یک صفحه دینامیک از جنس ASPیا ASP.NET بسازیم که محتوای هر خبر یا مقاله را نمایش بدهد، یک صفحه Template از جنس XSL می‌سازیم و محل قرارگرفتن داده‌های فیلدهای هر رکورد را داخل صفحه مشخص می‌کنیم. برای ساختن صفحات Templateمذکور باید ابتدا صفحات وب را به فرمت XHTML تبدیل کنیم. سپس با قراردادن فیلدهای موردنظر، صفحه XHTML را به XSL تبدیل کنیم.

تبدیل HTML بهXHTML

[1] بهروز نوعی پور

ساختن صفحات XHTML بسیار ساده است. کافی است صفحه معمولی خود را بسازید و محل قرارگرفتن فیلدها را مشخص کنید (مثلا از طریق تایپ کردن نام هر فیلد) و سپس به کمک یک مبدل HTML به XHTML صفحه وب خود را به فرمت سازگار با XHTML تبدیل کنید. برنامه‌ای مانند Dreamweaver به کمک یک فرمان ساده، این کار را به راحتی آب خوردن انجام می‌دهد!

تفاوت یک HTML معمولی با یک HTML سازگار با فرمت XHTML چندان زیاد نیست. درواقع کنترلی باید روی صفحه انجام شود تا مطمئن شویم که‌tag های صفحه در قیاس با قواعد XML اصطلاحا well-form هستند. مثلا در فرمت XML هر tag که شروع می‌شود باید حتما پایانی داشته باشد. مثل <table></table> اگر بعضی از tag ها قرار است که تنها باشند مثل (
) این tag ها باید به صورت مثلا </br> اصلاح شوند. چند تغییر جزئی دیگر نیز باید صورت گیرد تا صفحه کاملا با فرمت XHTML سازگار باشد. حتی اگر یک کاراکتر هم ناسازگار با قواعد XML باشد، امکان استفاده از روش توضیح داده شده در این مقاله وجود نخواهد داشت. ضمنا، کلاس و شئ XML در دات‌نت بعضی از کاراکترها و ترکیب‌های کاراکتری را نیز نمی‌پذیرد. مثلا   که نمایانگر یک spaceیا فاصله است، از دید مفسر XML در دات نت مجاز نیست و به جایش باید همان کاراکتر space را تایپ کنید. نگران نباشید. محدودیت‌هایی که در این زمینه وجود دارند بسیار اندک هستند و تقریبا صفحه خود را با حدود یکی دو درصد تغییر می‌توانید به یک XHTML قابل استفاده برای کلاس XML تبدیل کنید.

یادآوری مهم : ضروری است که صفحات فارسی خود را از جنس یونی‌کد و با کاراکترست utf-8 بسازید. زیرا شئ xml در دات نت تنها در این صورت می‌تواند اطلاعات فارسی را بدون مشکل پردازش کند.

تبدیل XHTML به XSL

این قسمت از کار مهمترین قسمت ساختن Template مورد نظر است. برای اینکار باید با استاندارد XSL و تکنیک‌ها و زبان آن آشنا شوید .XSL بحث مفصلی دارد و برای آموختن آن کتاب‌های متعددی چاپ شده است که لازم است حداقل یکی از آنها را به عنوان مرجع دم دست داشته باشید. من برای سهولت کار، چند تکنیک ساده و پرکاربرد XSL را که در این مقاله نیاز داریم، داخل فایل مثالی که ضمیمه مقاله است، پیاده کرده‌ام که می‌توانید از آنها به عنوان الگو استفاده کنید .XSL به معنی eXtensible Stylesheet Language است XSL .در دنیای فناوری XML مشابه CSS در دنیای HTML است .اگر با CSS ها آشنایی داشته باشید، درک کاری که XSL انجام می‌دهد چندان برایتان مشکل نخواهد بود. همانطور که از CSS برای فرم دادن به صفحات وب استفاده می‌کنیم، از XSL نیز برای فرم دادن به محتویات فایلهای XML استفاده می‌کنیم .

تکنیک‌هایی که برای پردازش فایل‌های XSL و قراردادن داده‌های XML استفاده می‌شود اصطلاحا XSLT نام دارد (به انتهای عبارت XSL کلمه Transformation را بیفزایید).

قلب فرآیندی که اتفاق می‌افتد به این شرح است:

-داده‌های شما، یعنی اطلاعات رکوردهای خبر و مقاله باید به فرمت XML درآیند.

-کد برنامه، فایل XSL شما را باز می‌کند و آن را برای یافتن tag های مخصوص نمایش فیلدهای اطلاعاتی جستجو می‌کند و به محض پیدا کردن محل فیلد، اطلاعات آن را جایگزین می‌کند.

به عنوان مثال عبارت <"xsl:value-of select="ArticleTitle> معادل گزینه‌ای است که چند پاراگراف بالاتر به عنوان روش نمایش فیلد ArticleTitle در صفحه ASP.NET شرح دادم. به عنوان نمونه دیگر، در زبان XSLبا استفاده از syntax زیر می‌توانیم یک حلقه بسازیم:

```
<xsl:for-each select="//Table">
...
</xsl:for-each>
```

که حلقه روی فیلدهای جدول Table تکرار می‌شود.

کاری که ما در برنامه خود انجام می‌دهیم اینست که ابتدا رکوردهای بانک اطلاعاتی را با استفاده از شئ xml در دات نت به XML تبدیل می‌کنیم سپس با استفاده از کلاس xslt یک فایل XSL را بازمیکنیم و داده ها را درون آن میریزیم و حاصل را به صورت یک HTML معمولی ذخیره می‌کنیم. همین!

کد برنامه تولید مکانیزه صفحات وب با استفاده از ASP.NET

کاری که ما باید انجام دهید اینست که یک صفحه ASP.NET بسازیم که عملیات تولید صفحه استاتیک را به طور خودکار انجام دهد تا بتوانیم حاصل فرآیند را خیلی ساده روی سایت publish کنیم. البته می‌توانیم ASP.NETرا طوری بنویسیم که خودش در محل مشخص شده توسط ما صفحات HTML تولید شده را save کند. در اینصورت باید میزبان شما قابلیت استفاده از فناوری ASP.NET را در اختیار شما قرار دهد وگرنه مجبورید صفحات را روی دستگاه خودتان تولید کنید و بعد حاصل کار را به صورت دستی یا به هر روش دیگر upload کنید.

و حالا سورس کد:

(من از VB.NET استفاده کرده ام ولی خودتان می‌دانید که چقدر تبدیل آن به C# آسان است)

۱ -برای اجرای برنامه لازم است کلاسهای مورد نیاز را import کنیم:

```
Imports System.Text
Imports System.Data
Imports System.Data.SqlClient
Imports System.IO
Imports System.IO.Path
Imports System.Xml
Imports System.Xml.Xsl
```

۲ -من فرض را براین می‌گذارم که شما یک datagrid خودتان ساخته‌اید و می توانید در این صفحه ASP.NETرکوردهای اخبار و مقالاتتان را مدیریت کنید. من در این مقاله روش اینکار را توضیح نمی‌دهم چون روشش خیلی آسان و سرراست است و در همه سایت‌های مربوط به ASP.NET میتوانید یک دوجین مقاله درباره روش نمایش رکوردهای بانک اطلاعاتی توسط datagrid پیدا کنید. شما باید همچنین در این datagrid امکان selectکردن یک رکورد (جهت انجام پردازشها مورد نیاز این مقاله) را اضافه کنید. این کار نیز آسان است و من قصد آموزش آن در اینجا را ندارم.

۳ -همه عملیات را یک تابع بسیار ساده و جادویی انجام می‌دهد☺

```
Function GenerateOneHTML(ByVal RecordID As Integer) As Boolean

   'single node----------------------
   Dim myData As DataSet = GetDataSet()
   Dim doc As XmlDataDocument = New XmlDataDocument(myData)
   Dim FileName As String
   Dim ln As Integer
   Dim node As XmlElement = doc.DocumentElement.SelectSingleNode("//Table[ID="
+ RecordID.ToString + "]")
```

```
    If Not node Is Nothing Then
        Dim doc2 As XmlDocument = New XmlDocument
        doc2.LoadXml(node.OuterXml)

        'Generate HTML file name
        FileName = RecordID.ToString
        Dim HTMLsPath As String = Server.MapPath("\") + "Articles\" +
Trim(doc2.SelectSingleNode("//Category").InnerText) + "\"

        'Transform
        Dim xslt As XslTransform = New XslTransform
        xslt.Load(HTMLsPath + "temp.html")
        Dim writer As XmlTextWriter = New XmlTextWriter(HTMLsPath + FileName +
".htm", System.Text.Encoding.UTF8)
        'writer.Formatting = Formatting.Indented
        'writer.Indentation = 2
        xslt.Transform(doc2, Nothing, writer, Nothing)

        writer.Close()
    End If

    Return True
End Function
```

از جزئیات این تابع نترسید. خیلی ساده در سه مرحله، یعنی در سه سوت (!) کار را تمام می‌کند.

من این تابع را بعد از یک ماه تلاش مداوم به این صورت خلاصه و مفید نوشتم و خیالتان راحت باشد که صد در صد کار می‌کند. فقط توجه کنید که نام فیلدهای استفاده شده در این تابع با توجه به نامگذاری فیلدهای پایگاه داده خودم صورت گرفته است و شما باید این جزئیات تابع را مطابق نیاز خود تغییر دهید. مراحل سه گانه فوق به این شرح هستند :

۱ -خواندن xml از رویdataset

۲ -تولید نام فایل خروجی

۳ -تبدیل(Tarnsform)

سومین مرحله که مهترین مرحله است، درواقع همان عملیات XSLT است. در اینجا باید گفت دست طراحان معماری دات نت درد نکند که با چهار خط کد می‌توان یک فرآیند XSLT را کامل کرد!

مرحله اول:

در این مرحله از روی dataset یک xml می‌سازیم:

```
'single node----------------------
Dim myData As DataSet = GetDataSet()
Dim doc As XmlDataDocument = New XmlDataDocument(myData)
Dim FileName As String
Dim ln As Integer
Dim node As XmlElement = doc.DocumentElement.SelectSingleNode("//Table[ID=" +
RecordID.ToString + "]")
```

توجه کنید که من روش ساختن dataset را توضیح نداده‌ام ولی شما خودتان می‌توانید تابع GetDataSet را مطابق نیازتان بنویسید. این کار یکی از آسان‌ترین و ابتدایی ترین روشهای کار با رابط برنامه‌نویسی ADO.NET است (لابد می‌دانید که دو روش عمده برای خواندن اطلاعات از بانک اطلاعاتی وجود دارد، یکی شئ DataReader و دیگری شئ DataSet) و من در اینجا برای اجتناب از طولانی شدن مقاله، از توضیح دادن آن صرف‌نظر کردم.

شئ جادوئی XmlDataDocument کارش تبدیل dataset به XML است. خط آخر این مرحله یک شئ XmlElementاز روی همان رکوردی که می‌خواهیم اطلاعاتش را بخوانیم می‌سازد. دقت کنید که چطوری نام جدول پایگاه داده و id رکورد را که به صورت input وارد تابع کرده‌ام، به این شئ خورانده می‌شود.

مرحله دوم:

حالا باید از روی خروجی xml شئ مذکور یعنی node.OuterXml یک شئ XmlDocument بسازیم و datasetرا کنار بگذاریم (توجه کنید که در سورس اصلی تابعی که نوشته‌ام، مرحله دوم و سوم را داخل یک شرط if قرار داده‌ام تا اگر رکورد مورد نظر پوچ بود این دو مرحله اجرا نشوند):

```
Dim doc2 As XmlDocument = New XmlDocument
doc2.LoadXml(node.OuterXml)

'Generate HTML file name
FileName = RecordID.ToString
Dim HTMLsPath As String = Server.MapPath("\") + "Articles\" +
Trim(doc2.SelectSingleNode("//Category").InnerText) + "\"
```

در واقع یک XmlDocument در کد برنامه دات نت، معرف یک xml است. و در تمام این مقاله هرجا گفتم کلاس (یا شئ xml در دات نت) بیشتر مقصودم همین شئ بود. البته کل اشیاء بکار رفته در این مقاله از کلاس مادر System.xmlو System.xml.xsl مشتق شده اند .

در ادامه این مرحله من کمی با اطلاعات فیلد RecordID بازی کرده ام تا یک نام دلخواه و یک path مناسب برای دخیره کردن فایل HTML نهایی جور کنم. همچنین اگر خواستید روی داده‌های رکورد مورد نظر، پیش از قرارگرفتن در XSL تغییری انجام دهید، مثلا تاریخ میلادی را به شمسی تبدیل کنید، جایش همین مرحله است. با استفاده از syntax زیر می‌توانید به محتویات یک فیلد از این رکورد دسترسی داشته باشید (قابل خواندن و نوشتن):

```
doc2.SelectSingleNode("//FieldName").InnerText
```

مرحله سوم:

در این مرحله عملیات جادویی تبدیل صورت می‌گیرد!

خط اول یک XslTransform معرفی کرده ام. این شئ یک تابع بدقلق اما فوق‌العاده نیرومند دارد که حدود ده تا تعریف overload دارد. یعنی می‌توان این تابع را به چندین روش فراخوانی کرد. پیدا کردن حالت مناسب برای این تابع خودش مکافاتی بود (!) ولی حالا نتیجه کار بسیار ساده از آب درآمده است. من فایل XSL که به روش توضیح داده شده ساختم را به اسم temp.html در همان مسیری (path) که می‌خواهم فایل خروجی را ذخیره کنم قرار دادم. ابتدا با استفاده از تابع load این فایل را باز می‌کنم. سپس با استفاده از یک شئ جادویی دیگر به نام XmlTextWriterیک مجرای خروجی (یک stream) برای تابع Transform فراهم می‌کنم:

```
'Transform
Dim xslt As XslTransform = New XslTransform
xslt.Load(HTMLsPath + "temp.html")
```

```
Dim writer As XmlTextWriter = New XmlTextWriter(HTMLsPath + FileName +
".htm", System.Text.Encoding.UTF8)
'writer.Formatting = Formatting.Indented
'writer.Indentation = 2
xslt.Transform(doc2, Nothing, writer, Nothing)

writer.Close()
```

ذکر چند نکته در اینجا ضروری است:

۱ -آدرسی که در مرحله دوم ساختم را به XmlTextWriter می‌دهم.

۲ -فرمت یونی‌کد را نیز برای XmlTextWriter مشخص می‌کنم.

۳ -اگر دلم خواست، از property فرمتینگ نیز استفاده می‌کنم:

```
writer.Formatting = Formatting.Indented
writer.Indentation = 2
```

این property خیلی جالب و در عین حال دردسرساز است. کارش اینست که خروجی HTML شما را برای خواندن سورس آن فرم بدهد. اگر اینکار را نکنید، XmlTextWriterتمام فایل HTML شما را در یک خط (!) می‌نویسد. هنگام نمایش HTML هیچ اتفاق یا مشکل خاصی پیش نمی‌آید اما اگر بروید روی مرورگر گزینه view sourceرا بزنید متوجه می‌شود که خواندن این فایل واقعا مکافات است!

Propertyفوق این مشکل را برطرف می‌کند و خروجی HTML را طوری تولید می‌کند که بشود سورس آن را (مثل آدم!) خواند. فقط اشکالش اینست که در برخی Layout ها به دلیل انداختن یک سری space اضافه، صفحه را از ریخت می‌اندازد. در مورد سایت من چنین مشکلی پیش آمد و من بیخیال فرمتینگ شدم .Property دوم که

Indentationنام دارد میزان تاثیر گذاری فرمتینگ را مشخص می‌کند .عدد ۲ نرمال است. اگر بیشتر بدهید، nestingسورس HTML را بیشتر می‌کند، خودتان امتحان کنید، متوجه خواهید شد.

سرانجام فرمان آسمانی Transform انجام می‌شود. همه مقاله را به خاطر این تابع پرتشریفات نوشتم. می‌خواستم از اول بگویم این تابع چه تابع خوبی است، توی رودربایستی افتادم و این مقاله را نوشتم !

فراموش نکنید که مجرای XmlTextWriter را باید پس از اتمام کار بست. حالا اگر به پوشه‌ای که می‌خواستید فایلتان آنجا ذخیره شود مراجعه کنید می‌بینید که یک صفحه وب تر و تمیز حاوی مقاله یا خبر شما آنجا نشسته و اسم دلخواه شما را نیز به خود گرفته است (در اینجا همان کدid).

باید datagrid خودتان را طوری بسازید که بشود از طریق یک checkbox کنار هر رکورد، آنهایی که مایلید صفحه خبر یا مقاله‌اش را بسازید، مشخص کرد. سپس یکی یکی id این رکوردها را به تابع مورد بحث می‌دهید تا به ازای هرکدام از آنها یک صفحه بسازد. توجه کنید که تکرار اجرای این فرمان روی یک رکورد موجب overwrite شدن فایل قبلی می‌شود. یعنی می‌توانید به سادگی فایل را update کنید.

اگر خواستید Layout سایت تان را تغییر دهید، کافی است فایل XSL خود را یکبار دستکاری کنید. اینکار خیلی ساده است. اگر XSL را با پسوند htm یا html ذخیره کنید، برنامه‌ای مانند Dreamweaver آن را مانند یک صفحه وب معمولی باز می‌کند تا بتوانید ویرایشش کنید. بعد از تغییرXSL ، یکبار دیگر فرمان Transformation را روی رکورهای مقالات و اخبار تکرارکنید تا صفحات جدید را بدست آورید.

نتیجه‌گیری:

اگر به صورت مساله اول مقاله برگردیم، حالا متوجه می‌شویم این سایت‌های خبری بزرگ دنیا چطوری بدون استفاده از صفحات دینامیک، سایتهای فعالی دارند. در واقع کلکش خیلی آسان بود .

پی‌نوشت:

من خودم از این تکنیک برای ساختن صفحات بخش مقالات سایتم [1]استفاده کرده‌ام. البته در بخش وبلاگ سایتم از روش متداول و دینامیک بیشتر استفاده کردم ولی اگر به بخش مقالات مراجعه کنید، متوجه می‌شوید که حتی یک صفحه دینامیک هم آنجا وجود ندارد.

[1] www.behrooznp.com/default.htm

چگونگی ساخت Setup برای برنامه های .NET که از Crystal Report استفاده می کنند[1]

برنامه نویسان حرفه ای معمولاً بعد از تولید برنامه‌ی خود، با استفاده از امکانات موجود در Visual Studio .NET برای آن یک Setup یا اصطلاحاً یک برنامه‌ی نصّاب می سازند. اما اگر برنامه شما جهت گزارشگیری از ابزار Crystal Report برای .NET استفاده کرده باشد، احتمالاً برنامه نصّاب شما به هنگام نصب به مشکلی برخورد خواهد کرد. این مقاله را بخوانید تا راه حل این مشکل را بیابید .

احتمالا تاکنون برنامه هایی با .NET نوشته و آن را روی سیستمی که Visual Studio.NET روی آن نصب نباشد امتحان کرده اید. این کار بدون هیچ مشکلی انجام می شود، کافی است برای برنامه خود Setup درست کنید و بعد از نصب .NET Framework روی سیستم مورد نظر برنامه خود را نصب و اجرا کنید.

اما اگر در برنامه خود گزارشهایی توسط Crystal Report نوشته باشید حتی در صورت طی مراحل بالا هنگام اجرای برنامه و فراخوانی گزارشهای مربوطه با یک پیغام خطا روبرو خواهید شد و گزارش مورد نظر شما نشان داده نخواهد شد دلیل آن این است که نحوه ساخت Setup برای این گونه برنامه ها کمی متفاوت است.

مراحل ساخت Setup برای اینگونه برنامه ها به این شرح ذیل است:

ابتدا مراحل اولیه ساخت Setup را مانند برنامه های عادی طی کنید.

پس از ساخت پروژه Setup در Solution Explorer روی آن کلیک راست کرده و از منوی Add گزینه Merge Moduleرا انتخاب کرده و آیتمهای زیر را اضافه کنید:

[1] لطفعلی خوش نفس

Crystal_Database_Access2003.msm

Crystal_Database_Access2003_enu.msm

Crystal_regwiz2003.msm

در Solution Explorer روی گزینه Crystal_regwiz2003.msm کلیک راست کرده و Properties را انتخاب کنید.

در پنجره Properties گزینه MergeModuleProperties را باز کرده و در قسمت License Key کد مربوطه را وارد کنید.

توجه:

برای بدست آوردن Licence Key گزینه About از منوی Help در Visual Studio .NET را انتخاب کنید سپس کدی که در مقابل Crystal Reports for Visual Studio .NET نوشته شده را کپی کرده و در قسمت مربوطه وارد کنید.

چند زبانی در .NET[1]

یکی از برتری‌های .NET نسبت به رقیب این است که شما می توانید از چندین زبان برنامه نویسی در یک برنامه استفاده نمائید. حسن این قابلیت حداقل در این است که افراد تیم برنامه نویسی شما مجبور نیستند که همگی تنها یک زبان را بدانند و شما هم بهتر و زودتر می توانید تیم خود را تشکیل دهید .

احتمالا به این مطلب برخورد کرده اید که یکی از ادعاهای Microsoft در ارائه .NET چند زبانه بودن آن است. در این مقاله می خواهیم این موضوع را بررسی کنیم.

چند زبانه بودن به این معنی است که ما می توانیم قسمتهای مختلف یک برنامه را به زبانهای مختلف بنویسیم و آنها بدون هیچ مشکلی در کنار هم کار کنند. در .NET بدلیل استفاده زبانها از کتابخانه کلاس .NET تفاوت آنها فقط در نحوه نوشتاری دستورات می باشد مثلا زبان C# هیچ برتری خاصی نسبت به VB.NET ندارد و انتخاب بین یکی از این زبانها به سلیقه و علاقه برنامه نویس بستگی دارد (البته من خودم VB را به دلیل دارا بودن Syntax واضح تر ترجیح می دهم).

اما اگر می خواهید همزمان از چند زبان استفاده کنید مثال ساده زیر شاید بتواند راهنمایی برای شما باشد. در مثال زیر می خواهیم از زبانهای VB.Net و C# در یک برنامه Windows Application استفاده کنیم:

- ابتدا یک پروژه VB با نام MyvbProj در .NET ایجاد کنید.
- سپس در Solution Explorer روی نام Solution کلیک راست کرده و از منوی Addگزینه New Project را انتخاب کنید.
- یک پروژه C# با نام MycsProj اضافه کنید.

[1] لطفعلی خوش نفس

- حال در Solution Explorer روی پروژه Mycsproj کلیک راست کرده و Propertiesرا انتخاب کنید.
- در پنجره باز شده Output Type را از Windows Application به Class Libraryتغییر دهید.
- سپس در Solution Explorer روی نام Solution کلیک راست کرده و گزینه Solution Buildرا انتخاب کنید.
- حال در قسمت Referencese پروژه Myvbproj کلیک راست کرده و گزینه Referencese Addرا انتخاب کنید.
- در پنجره باز شده به قسمت Projects رفته و Mycsproj را انتخاب و Select را انتخاب کنید و Ok را بزنید.

حال می توانید در داخل پروژه VB از اجزای پروژه C# استفاده کنید مثلا می توانیم با قرار دادن یک کنترل Button در فرم مربوط به پروژه VB و نوشتن کد زیر برای رویداد کلیک آن فرم مربوط به پروژه C# را نشان دهیم:

```
Dim Frm as new Mycsproj.Form1
Frm.ShowDialog(Me)
```

به همین روش شما می توانید در یک برنامه از چندین زبان استفاده کنید این روش زمانی مفید است که یک تیم برنامه نویس روی یک برنامه کار می کنند و هر کدام از اعضا مایل به برنامه نویسی با یک زبان خاص می باشد.

مدیریت نقش ها با استفاده از Principle Generic و FormsAuthentication[1]

مدیریت در سایت های دارای بخش کاربران و کلا سایت هایی که به نوعی احتیاج به طبقه بندی دسترسی ها به بخش های مختلف سایت دارند، با این بحث آشنا هستند.در این مقاله به شکل عملی به شرح نحوه ی استفاده از امکانات ASP.NET برای کار با نقش ها می پردازیم .

برای مدیریت نقش ها در این روش نخست باید در استفاده از FormsAuthentication مسلط باشید.

با ساخت یک WebApplication شروع می کنیم. صفحه Logon یادتان نرود! مسیر کار به این شرح است:

۱- Web.Configرا برای FormsAuthentication تغییر می دهیم.

۲- تهیه‌ی یک Ticket برای شناسایی کاربر

۳-ساخت اشیاء GenericPrincipal وFormsIdentity

۴-آزمایش برنامه

۱-در Web.Config بخش <authentication> را به شکل زیر تغییر دهید:

```
<"authentication mode="Form>
<forms name="TestCookie " loginUrl= "login.aspx" protection="All" timeout="30"
```

[1] بهنام یوسفی شمالی

```
path="/"></forms>
</authentication>
```

المنت <authorization> زیر را پایین بخش <authentication> اضافه کنید.

```
<authorization>
<deny users="?" />
<allow users="*" />
</authorization>
```

۲-ساخت یک Ticket شناسایی برای کاربر

Ticketشناسایی در واقع نوعی کوکی است که ماژول FormsAuthentication برای شناسایی استفاده می کند.

در مرحله اول باید اطلاعات کاربر (شناسه و کلمه عبور) را تایید کنید .روش این کار کاملا سلیقه ای است. می توانید از پایگاه داده یا از Web.Config استفاده کنید. در هر صورت شما به یک Function مانند زیر نیاز دارید:

```
Function IsAuthenticated(ByVal uid As String,ByVal pwd As String)As Boolean
  . . .
  Return true
End Function
```

مطمئنا وقتی می خواهید نقش ها را مدیریت کنید، برای هر کاربر در پایگاه داده ها نقش ها را نیز مشخص کرده اید. پس یک Function هم باید وظیفه ی بازگرداندن نقش ها را به عهده داشته باشد. مانند زیر:

```
Function GetRoles(ByVal uid As String,ByVal pwd As String)As Boolean
  . . .
  Return "Admin|Moderator|Manager"
End Function
```

در صفحه‌ی Logon.aspx درون کد Button این کدها را بنویسید:

```
If dp.Authenticate(Trim(txtuid.Text), Trim(txtpwd.Text)) Then
Dim strRoles As String = dp.GetRoles(Trim(txtuid.Text) , Trim(txtpwd.Text))
Dim Roles() As String = Split(strroles,"|")
Dim authTicket As New FormsAuthenticationTicket(1, Trim(txtuid.Text), Now(),
Now.AddMinutes(60), False, Roles)
Dim encryptedTicket As String = FormsAuthentication.Encrypt(authTicket)
Dim authCookie As New HttpCookie(FormsAuthentication.FormsCookieName,
encryptedTicket)
Response.Cookies.Add(authCookie)
Response.Redirect(FormsAuthentication.GetRedirectUrl(Trim(txtuid.Text), False))
```

۳-ساخت اشیاء GenericPrincipal وFormsIdentity

صفحه‌ی کدهای Global.asax را باز کنید و در قسمت بالای کد این قسمت را اضافه کنید:

```
Imports System.Web.Security
Imports System.Security.Principal
```

در بخش Application_AuthenticateRequest کد های زیر را اضافه کنید.

```
Dim cookieName As String = FormsAuthentication.FormsCookieName
Dim authCookie As HttpCookie = Context.Request.Cookies(cookieName)

If authCookie Is DBNull.Value Then
  ' There is no authentication cookie.
  Return

End If
```

بخش زیر را برای رمزگشایی کوکی اضافه کنید:

```
Dim authTicket As FormsAuthenticationTicket
Try
  authTicket = FormsAuthentication.Decrypt(authCookie.Value)
Catch ex As Exception
  ‘ Log exception details (omitted for simplicity)
  Return
End Try

If authTicket Is DBNull.Value Then
  ‘ Cookie failed to decrypt.
  Return
End If
```

این بخش را برای بازیافت نقش ها اضافه کنید:

```
Dim roles As String() = authTicket.UserData.Split("|")
```

این بخش را برای ساخت FormsIdentity با شناسه ای که از کوکی گرفته شده و ساخت GenericPrincipalکه شامل این Identity است، اضافه کنید:

```
' Create an Identity object
Dim id As New FormsIdentity(authTicket)
' This principal will flow throughout the request.
Dim principal As New GenericPrincipal(id, roles)
' Attach the new principal object to the current HttpContext object
Context.User = principal
```

۵-برای آزمایش برنامه این کد ها را در صفحه Test.aspx بنویسید:

در ابتدا

```
Imports System.Security.Principal
```

و سپس در کد Load صفحه:

```
Dim p As IPrincipal = HttpContext.Current.User
Response.Write("Authenticated Identity is: " + p.Identity.Name)
Response.Write("<p>")
If p.IsInRole("admin") Then
  Response.Write("RoleName is: Admin")
End If
```

منابع :

Building Secure Microsoft ASP.NET Applications کتاب

کاهش حجم خروجی در ASP.NET[1]

مساله سرعت بارگذاری صفحات چه به صورت ایستا طراحی شده باشند و چه به صورت پویا از مباحث مهم در طراحی حرفه ای صفحات وب است این مساله در کشور ما با توجه به سرعت نسبتا پایین خطوط اینترنت اهمیت مضاعفی می یابد از طرفی در سایتهای بزرگ که طبعا حجم بیشتری هم دارند مساله پهنای باند مصرفی نیز مورد توجه قرار می گیرد که رابطه مستقیمی با سرعت بارگذاری صفحات دارد چرا که سرعت بارگذاری صفحات نیز رابطه مستقیمی با حجم صفحات دارد .

مساله سرعت بارگذاری صفحات (چه به صورت Static طراحی شده باشند چه به صورت Dynamic) از مباحث مهم در طراحی حرفه ای صفحات وب است این مساله در کشور ما با توجه به سرعت نسبتا پایین خطوط اینترنت اهمیت مضاعفی می یابد از طرفی در سایتهای بزرگ که طبعا حجم بیشتری هم دارند مساله پهنای باند مصرفی نیز مورد توجه قرار می گیرد که رابطه مستقیمی با سرعت بارگذاری صفحات دارد چرا که سرعت بارگذاری صفحات نیز رابطه مستقیمی با حجم صفحات دارد.

در واقع عامل اصلی در سرعت بارگذاری صفحات حجم کدهای HTML صفحات است. صفحات چه به صورت Staticطراحی شده باشند چه به صورت Dynamic در نهایت به صورت کدهای HTML در سمت Client به نمایش در می آیند. اگر چه در صفحات Dynamic میزان محاسبات روی Server نیز نقش دارند ولی با توجه به رشد قدرت سخت افزار و نرم افزار این عامل روز به روز کم رنگ تر می شود.

حال بحث را به ASP.NET محدود می کنیم. اگر با ساختار پاسخ گویی به درخواستهای کاربران توسط IIS در ASP.NET آشنا باشید، می دانید که پس از پردازش کدهای صفحات و تعیین کدهای HTML آنها را به سمت

[1] کیوان نیّری

Clientمی فرستد. حال برای حل مشکل سرعت صفحات حجیم راه حل هایی ایجاد شده است که ریشه در فشرده سازی و کاهش حجم کدهای HTML ارسالی به سمت کاربر و بازگشایی کدهای HTML در سمت Client دارد.

روشهای بسیار قوی و توانایی در این زمینه وجود دارند که می توانند تا متوسط هشتاد درصد در سرعت و پهنای باند مصرفی صرفه جویی کنند. و بسته های نرم افزاری آنها در سایتهای داخلی و خارجی موجودند(مانند بسته نرم افزاری ASP.JET [1].(ولی سوالی که مطرح می شود اینست که بازگشایی کدهای فشرده شده در سمت Client بر چه اساسی صورت می گیرد؟ پاسخ این است که در استاندارد HTTP 1.1 الگوریتمهای به صورت پیش فرض برای فشرده سازی صفحات در نظر گرفته شده اند که فشرده سازی نیز توسط همین الگوریتمها صورت می گیرد.

مقاله پیش روی شما به روشی می پردازد که جدا از مساله بازگشایی کدهای فشرده سازی شده است. پس در استاندارد HTTP 1.0 نیز قابل استفاده است ولی درصد کمتری از فشرده سازی را نسبت به روشهای مذکور انجام می دهد (بین ۷ تا ۳۰ درصد).

می دانیم در طراحی کدهای HTML در اکثر نرم افزارهای طراحی Editor ها برای خوانایی کدها و خطایابی، آنها را با فواصل و شکست خطوط مناسبی به نمایش گذاشته و به همین صورت نیز ذخیره می کند. این فواصل اضافی توسط مرورگرها در نظر گرفته نمی شوند ولی حجم زائدی را اشغال می کنند .در این مقاله با استفاده از یک کلاس نوشته شده در Visual Basic.NET این فواصل را پیش از ارسال کدهای HTML به سمت Client از بین می برد و از فواید این روش این است که برای سایتهای کوچکتر و وبلاگها به سهولت قابل استفاده است و در صورتی که در سایتهای بزرگ Client ها از استاندارد HTTP 1.0 استفاده کنند می توانند با استفاده از این روش خلا موجود را تا حدی پوشاند.

[1] www.softtool.info/aspjet/fa/default.htm

در اینجا ذکر این نکته لازم است که الگوریتمهای مورد استفاده در فشرده سازی صفحات نمی توانند کدهای حاصل از روش موجود در این مقاله را فشرده کنند و حاصل کدهایی خواهد بود که در مرورگر به نمایش در نمی آید زیرا این روشها بر مبنای شکست خطوط و فواصل پیش فرض واقعند. در صورتی که قصد استفاده از این روشها را در کنار هم دارید لازم است روش زیر را توسط شروط لازم به استاندارد HTTP 1.0 محدود کنید.

نکته قابل توجه دیگر این است که کلاس به کار رفته در این مقاله تعدادی از فواصل زائد موجود در Tag ها و کدهای معمول را حذف می کند و شما به سادگی می توانید مطابق الگوی به کار رفته Tag ها و کدهای دیگری را به این کلاس اضافه کنید.

روش استفاده:

این مقاله از فضای نام HttpResponse.Filter برای کاهش حجم خروجی استفاده می کند. برای استفاده از این الگوریتم لازم است که کد کلاس WhitespaceFilter موجود در زیر را در یک فایل به نام WhitespaceFilter.vb ذخیره نموده و به پروژه خود اضافه کنید و خط زیر را به متد Application_BeginRequest در فایل Global.asax اضافه کنید:

```
    Sub Application_BeginRequest(ByVal sender As Object,
ByVal e As EventArgs)

            ' Fires at the beginning of each request

            Response.Filter = New
WhitespaceFilter(Response.Filter)

    End Sub
```

در صورتی که هدف شما استفاده از این روش در صفحات خاصی است کافیست به جای عمل فوق کد بالا را در متد Load صفحات هدف اضافه کنید با کد موجود در فایل Global.asax را مانند مثال زیر با شروط لازم تغییر دهید:

```
    Sub Application_BeginRequest(ByVal sender As Object,
ByVal e As EventArgs)

        ' Fires upon attempting to authenticate the use

        If
Request.Url.PathAndQuery.ToLower.IndexOf("makethumbnail") =
-1 Then

            Response.Filter = New
WhitespaceFilter(Response.Filter)

        End If

    End Sub
```

بهتر است برای مقایسه حجم صفحات HTML خود قبل از استفاده از این روش یکی از صفحات خود را در مرورگر مشاهده کرده و توسط یک Editor مانند NotePad محتوای آن را بررسی کنید و آن را به صورت یک فایل متنی ذخیره کنید.سپس بعد از استفاده از این روش همان صفحه را مجددا بررسی کرده و به صورت فایل دیگری ذخیره سازی کنید.بعد از آن حجم دو فایل را با هم مقایسه کنید تا برای هر صفحه به درصد فشرده سازی صفحات پی ببرید.

بررسی کدهای کلاس WhitespaceFilter:

حال به بررسی کدهای مربوط به این Class می پردازیم:

```
Imports System.IO

Imports System.Text.RegularExpressions
```

این فیلتر از ایجاد فضاهای خالی غیر ضروری د رخروجی جلوگیری می کند:

```
' This filter gets rid of all unnecessary whitespace in
the output.


Public Class WhitespaceFilter

    Inherits Stream


    Private _sink As Stream

    Private _position As Long


    Public Sub New(ByVal sink As Stream)

        _sink = sink

    End Sub 'New
```

این قسمت از کد در صورت ایجاد هر گونه تغییر نباید تغییر کند.

```
    #Region " Code that will most likely never change from
filter to filter. "
```

این اعضا از Stream مهم هستند.

```
        ' The following members of Stream must be
overridden.

        Public Overrides ReadOnly Property CanRead() As
Boolean

            Get

                Return True

            End Get

        End Property



        Public Overrides ReadOnly Property CanSeek() As
Boolean

            Get

                Return True

            End Get
```

```
        End Property


        Public Overrides ReadOnly Property CanWrite() As
Boolean
            Get
                Return True
            End Get
        End Property


        Public Overrides ReadOnly Property Length() As Long
            Get
                Return 0
             End Get
        End Property


        Public Overrides Property Position() As Long
            Get
                Return _position
```

```
            End Get

            Set(ByVal Value As Long)

                _position = Value

            End Set

        End Property



        Public Overrides Function Seek(ByVal offset As
Long, ByVal direction As System.IO.SeekOrigin) As Long

            Return _sink.Seek(offset, direction)

        End Function 'Seek



        Public Overrides Sub SetLength(ByVal length As
Long)

            _sink.SetLength(length)

        End Sub 'SetLength



        Public Overrides Sub Close()

            _sink.Close()
```

```
        End Sub 'Close



        Public Overrides Sub Flush()

            _sink.Flush()

        End Sub 'Flush



        Public Overrides Function Read(ByVal MyBuffer() As
Byte, ByVal offset As Integer, ByVal count As Integer) As
Integer

            _sink.Read(MyBuffer, offset, count)

        End Function



    #End Region
```

متد اصلی انجام عمل فیلتر است. Write

```
        ' Write is the method that actually does the
filtering.
```

```
        Public Overrides Sub Write(ByVal MyBuffer() As
Byte, ByVal offset As Integer, ByVal count As Integer)

            Dim data(count) As Byte

            Buffer.BlockCopy(MyBuffer, offset, data, 0,
count)
```

اینجا ASCII Encoding استفاده نکنید چون IDE برخی ار کاراکترها مثل ® را با کاراکترهای UTF-8 عوض می کنند اگر از ASCII Encoding استفاده کنید B را به جای علامت ® دریافت می کنید.

```
            ' Don't use ASCII encoding here.  The .NET IDE
replaces some characters, such as &reg;

            ' with a UTF-8 entity.  If you use ASCII
encoding, you'll get B. instead of the registered

            ' trademark symbol.

            Dim s As String =
System.Text.Encoding.UTF8.GetString(data)
```

جا به جا کردن کاراکترهای موجود در کنترلها با فضاهای خالی.

بررسی Semi-colon در این قسمت به علت وجود این کاراکتر در توضیحات Java Script موجود در کامپوننت است.

با این راه ما مقدار را حفظ می کنیم.

```
            ' Replace control characters with either spaces
or nothing



            ' The funky semi-colon handling is there
because of a JavaScript comment in a component.

            ' This way, we keep the carriage returns that
actually matter.

            s = s.Replace(ControlChars.Cr,
Chr(255)).Replace(ControlChars.Lf,
"").Replace(ControlChars.Tab, "")

            s = s.Replace(";" & Chr(255), ";" &
ControlChars.Cr)

            s = s.Replace(Chr(255), " ")
```

حذف فضاهای خالی اضافی

```
            ' Eliminate excess whitespace.

            Do

                s = s.Replace("  ", " ")

            Loop Until s.IndexOf("  ") = -1
```

حذف توضیحات پیش فرض و شناخته شده.

ما در قالبهای خود از سه توضیح استفاده می کنیم که در هر صفحه ای از سایت وارد می شوند.بنا بر این در زمان خروج صفحات مط توانیم انها را حذف کنیم.با این راه توضیحات برای نگهداری و عیب یابی موجودند ولی در زمان انتشار صفحات حذف می شوند.

```
            ' Eliminate known comments.



            ' We use three comments in our template.  These
comments go on every single page on the site.

            ' Obviously, we can kill them when they are
going out.  This way, the comments stay in for

            ' maintenance, but are trimmed before release.

            s = s.Replace("<!-- Page Content Goes Above
Here -->", "")

            s = s.Replace("<!-- Page Content Goes Below
Here -->", "")

            s = s.Replace("<!-- Do not get rid of this
  on data pages -->", "")
```

حذف برخی فضاهای خالی وارد شده که از بین بردن آنها مجاز است.به دلایلی یک فضای خالی قبل از هدایت کننده DOCTYPE وارد می شود.

```
            ' Eliminate some additional whitespace we can
kill
```

```
            ' For some reason, a single space gets emitted
before each of our DOCTYPE directives.

            s = s.Replace(" <!DOCTYPE", "<!DOCTYPE")
```

اینها فضاهای خالی معمولی هستند که می توانیم حذف کنیم:

```
            ' These are the most common excess whitespace items
we can remove.

            s = s.Replace("<li> ", "<li>").Replace("</td> ",
"</td>").Replace("</tr> ", "</tr>").Replace("</ul> ",
"</ul>").Replace("</table> ", "</table>").Replace("</li> ",
"</li>")

            s = s.Replace("<LI> ", "<LI>").Replace("</TD> ",
"</TD>").Replace("</TR> ", "</TR>").Replace("</UL> ",
"</UL>").Replace("</TABLE> ", "</TABLE>").Replace("</LI> ",
"</LI>")

            s = s.Replace("<td> ", "<td>").Replace("<tr> ",
"<tr>")

            s = s.Replace("<TD> ", "<TD>").Replace("<TR> ",
"<TR>")

            s = s.Replace("<P> ", "<P>").Replace("<p> ", "<p>")

            s = s.Replace("</P> ", "</P>").Replace("</p> ",
"</p>")
```

```
            s = s.Replace("style=""display:inline""> ",
"style=""display:inline""">")

            s = s.Replace(" <H", "<H").Replace(" <h",
"<h").Replace(" </H", "</H").Replace(" </h", "</h")

            s = s.Replace("<UL> ", "<UL>").Replace("<ul> ",
"<ul>")

            s = s.Replace(" <TABLE", "<TABLE").Replace("
<table", "<table")

            s = s.Replace(" <li>", "<li>").Replace(" <LI>",
"<LI>")

            s = s.Replace(" <br>", "<br>").Replace(" <BR>",
"<BR>").Replace("<br> ", "<br>").Replace("<BR> ", "<BR>")

            s = s.Replace(" <ul>", "<ul>").Replace(" <UL>",
"<UL>")
```

جا به جایی Tag های بلند با معادل کوتاه آنها:

```
            ' Replace long tags with short ones

            s = s.Replace("<STRONG>",
"<B>").Replace("<strong>", "<b>")

            s = s.Replace("</STRONG>",
"</B>").Replace("</strong>", "</b>")
```

جا به جایی برخی نمادهای HTML با کدهای کاراکتری درست معادل آنها:

```
            ' Replace some HTML entities with true
character codes

            s = s.Replace("&brkbar;", "|")

            s = s.Replace("&brvbar;", "|")

            s = s.Replace("&shy;", "-")

            s = s.Replace(" ", Chr(160))

            s = s.Replace("&lsquor;", "'")

            s = s.Replace("&ldquor;", """")

            s = s.Replace("&lsquo;", "'")

            s = s.Replace("&rsquor;", "'")

            s = s.Replace("&rsquo;", "'")

            s = s.Replace("&ldquo;", """")

            s = s.Replace("&rdquor;", """")

            s = s.Replace("&rdquo;", """")

            s = s.Replace("&ndash;", "-")

            s = s.Replace("&endash;", "-")
```

اگر این کار را انجام ندهیم JavaScript روی سایت عمل نمی کند:

```
            ' If we don't do this, JavaScript horks on the
site

            s = s.Replace("<!--", "<!--" & ControlChars.Cr)

            s = s.Replace("}", "}" & ControlChars.Cr)
```

آخرین شانس برای از بین بردن فضاهای خالی:

```
            ' Last chance to eliminate excess whitespace

            Do

                s = s.Replace("  ", " ")

            Loop Until s.IndexOf("  ") = -1
```

در نهایت کاری که انجام داده ایم را خارج می کنیم:

```
            ' Finally, we spit out what we have done.

            Dim outdata() As Byte =
System.Text.Encoding.UTF8.GetBytes(s)

            _sink.Write(outdata, 0, outdata.GetLength(0))



        End Sub 'Write
```

```
    End Class
```

منبع: http://www.codeproject.com/aspnet/WhitespaceFilter.asp

ساخت یک خروجی RSS برای سایت شما[1]

RSSیک راه استاندارد برای به اشتراک گذاشتن محتوای سایت ها با دیگران است .RSS یا همان Really Simple Syndicationچیزی نیست به جز یک نشانه گذاری استاندارد شده XML که محتوایی که قصد به اشتراک گذاشتن آن را دارید را توصیف می کند. در این مقاله قصد داریم چگونگی ساخت یک فایل RSS را در ASP.NET بررسی کنیم .

فرمتRSS

همان طور که قبلا اشاره شد RSS چیزی به جز یک XML نشانه گذاری شده با تگ های مخصوص نیست. کد زیر یک نمونه ساده RSS را نشان می دهد.

```
<rss version="2.0">
   <channel xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance">
   <title>IranASP.NET New Articles RSS Feed</title>
   <language>fa-IR</language>
   <copyright>(c) 2004 by IranASP.NET</copyright>
   <pubDate>Sun, 08 Aug 2004 06:11:37 GMT</pubDate>
   <lastBuildDate>Sun, 08 Aug 2004 06:11:37 GMT</lastBuildDate>
   <generator>IranASP.NET rss generator</generator>
   <item>
 ASP.NET</title> کاهش حجم خروجی در       <title>
```

[1] حامد سعیدی فرد

```
    <link>http://www.iranasp.net/articles/showarticle.aspx?articleid=154</link>
    <pubDate>Sun, 25 Jul 2004 19:30:00 GMT</pubDate>
  </item>
  </channel>
</rss>
```

اجازه بدهید به تگ های این سند کمی دقیق تر نگاه کنیم

<rss> : تشکیل دهنده تگ ریشه سند است که شامل نسخه سند RSS هم می باشد. آخرین نسخه RSS نسخه 2 است .

<channel>: گره ریشه سند RSS می تواند تگ <channel> هم داشته باشد. این تگ هم به نوبه خود می تواند شامل گره های <tile>,<link> و <item> باشد.

<title> :مشخص کننده عنوان سند RSS است .

<link>: این تگ URL سایت ارایه دهنده RSS را مشخص می کند .

<description> :جزییات و اطلاعات بیشتر مربوط به این سند را مشخص می کند .

<copyright>: اطلاعات مربوط به کپی رایت سند را مشخص می کند .

<generator> :این تگ Application ای که این سند را ساخته معرفی می کند .

علاوه بر تگ های بالا، یک سند RSS می تواند یک یا چند تگ <item> داشته باشد. تگ <item> مشخص کننده همان محتوایی است که می خواهید به اشتراک بگذارید مثلا یک مقاله یا یکی از پست های یک وبلاگ. هر تگ <item>به علاوه شامل زیرگره های زیر است.

<title>:این تگ عنوان آیتم را مشخص می کند مثلا عنوان مقاله .

<link>:این تگ URL آن آیتم را نشان می دهد مثلا آدرس مقاله در وب .

<description>:این تگ آیتم را توصیف می کند، در مورد یک مقاله می تواند شامل خلاصه مقاله باشد .

<pubDate>:این تگ تاریخ انتشار آن آیتم را نشان می دهد. فرمت معمول برای نمایش این تاریخ به صورت Sun, 25 Jul 2004 12:00:00 GMTاست .

چگونه یک سند RSS بسازیم ؟

حالا که با ساختار یک فایل RSS آشنا شدید، به سراغ دات نت می رویم. چگونه در دات نت یک فایل RSS بسازیم؟ دات نت مجوعه ای غنی ای از کلاس های مرتبط با XML دارد. برای ساختن فایل RSS ما از کلاس XML Text writerاستفاده می کنیم. ولی می خواهیم یک راه حل کلی داشته باشیم که با هر وب سایتی کار کند. معنی این حرف این است که کد ما باید مستقل از یک جدول یا یک فیلد خاص در دیتابیس سایت باشد. برای این کار ما یک کلاس در VS.NET می سازیم. منبع تگ های <item> سند XML ما هم یک دیتاست است که از جدول دلخواه ما در دیتابیس ساخته می شود. کلاسی که قصد نوشتن آن را داریم ویژگی ها (properties) و متدهای (methods) زیر را دارد:

ویژگی ها (properties)

Outputstream:یک شی جریان (Stream) که خروجی سند در آن نوشته می شود .

RssTitle:مقدار تگ <title> را در زیر تگ <channel> مشخص می کند .

PublisherUrl:مقدار تگ <link> را در زیر تگ <channel> مشخص می کند .

Description:مقدار تگ <description> را در تگ <channel> مشخص می کند .

Copyright:مقدار تگ <copyright> را در زیر تگ <channel> مشخص می کند .

Generator:مقدار تگ <generator> را در زیر تگ <channel> مشخص می کند .

ItemSource:تعیین کننده شی دیتاستی است که سطرهای آیتم را شامل می شود .

ItemTitleField نام Datacolumn (ستون داده ای در دیتاست) که مقدار تگ <title> از تگ <item> را نشان می دهد .

ItemURLField:نام Datacolumn که مقدار تگ <link> از تگ <item> را مشخص می کند .

ItemDescriptionField:نام Datacolumn که مقدار تگ <description> از تگ <item> را معرفی می کند .

ItemPublicationDateField:نام Datacolumn که مقدار تگ <pubDate> از تگ <item> را نشان می دهد .

متدها

PublishRss:این متد سند نشانه گذاری شده RSS را در Outputstream می نویسد .

در زیر کد کامل کلاس RSS را می بینید، توجه کنید که برای سادگی و نمایش سریع به جای properties از متغیرهای عمومی (Public) استفاده شده است.

```
Public Class Rss
    Public OutputStream As Stream
    Public RssTitle As String
    Public PublisherUrl As String
    Public Description As String
    Public Copyright As String
    Public Generator As String
    Public ItemSource As DataSet
    Public ItemTitleField As String
    Public ItemUrlField As String
    Public ItemDescriptionField As String
    Public ItemPublicationDateField As String
    Public Shared Function PublishRss(ByVal r As Rss)
        Dim writer As New XmlTextWriter(r.OutputStream,
        System.Text.Encoding.ASCII)
        writer.WriteStartElement("rss")
        writer.WriteAttributeString("version", "2.0")
        writer.WriteStartElement("channel")
        writer.WriteElementString("title", r.RssTitle)
        writer.WriteElementString("link", r.PublisherUrl)
        writer.WriteElementString("description", r.Description)
        writer.WriteElementString("copyright", r.Copyright)
        writer.WriteElementString("generator", r.Generator
```

```
        For Each row As DataRow In r.ItemSource.Tables(0).Rows
            writer.WriteStartElement("item")
            writer.WriteElementString("title", row(r.ItemTitleField))
            writer.WriteElementString("link", row(r.ItemUrlField))
            writer.WriteElementString("description",
            row(r.ItemDescriptionField))
            writer.WriteElementString("pubDate",
            CType(row(r.ItemPublicationDateField),
            DateTime).ToString("ddd, dd MMM yyyy 12:00:00 tt G\MT"))
            writer.WriteEndElement()
        Next
        writer.WriteEndElement()
        writer.WriteEndElement()
        writer.Flush()
    End Function
End Class
```

در کد کلاس بالا متد PublishRss مهمترین اتفاقی است که در کلاس می افتد. ابتدا ما یک نمونه از کلاس System.XML.XmlTextwriterساخته ایم. این کلاس سریع ترین راه برای نوشتن اسناد XML است. بعد OutputStreamرا به تابع پاس کرده و encoding سند را مشخص کردیم. در مورد این کد از ASCII استفاده شده است. بعد از این شروع به نوشتن بخش های مختلف سند RSS خودمان کردیم. در کلاس XmlTextwriterما از متدهای زیر برای نوشتن سند خودمون استفاده کردیم:

WritestartElement:این متد تگ شروع را برای یک تگ خاص می نویسد (هر تگ شامل یک تگ شروع هست و یک تگ پایان مثلا <channel> تگ شروع و </channel> تگ پایان می باشد).

WriteAttributeString:خواص تگ باز فعلی را می نویسد.

WriteElementString:مقدار هر تگ که بین تگ شروع و تگ پایان قرار دارد را می نویسد .

WriteEndElement:تگ پایان را برای تگی که اخیرا باز بوده می نویسد. نیازی نیست که تگ های داخلی تر را مشخص کنید .

Flush:این متد، تمام خروجی های بافرشده را به مقصد منتقل می کند .

توجه کنید که شما باید متدهای WriteStartElement و WriteEndElement را برای بهتر شکل دادن به سند خودتان استفده کنید.

ساختن وب فرمASP.NET

حالا که یک کلاس کلی نوشتیم، می تونیم از این کلاس در وب فرم خودمان استفاده کنیم. یک ساختار که به عنوان نمونه می توانید در بانک اطلاعاتی از آن استفاده کنید به شکل زیر می باشد. از اطلاعات این ساختار در بانک اطلاعاتی برای ساخت سند RSS استفاده می گردد .

Article_ title - Varchar (255)
Article_ Description -Varchar (1000)
Article_ url - Varchar (255)
Article_ pubdate - DateTime

ما به کمک یک تابع به نام GetDataSet اطلاعات را به دیتاست می دهیم، این دیتاست به عنوان منبع برای تولید فایل RSS مورد استفاده قرار می گیرد. برای پر کردن دیتاست از یک DataAdapter استفاده می کنیم:

```
Function GetDataSet() as DataSet
   Dim cnn as New SqlConnection("connection string here")
   sql = "select * from sometable order by Article_pubdate desc"
   Dim da As New SqlDataAdapter(sql,cnn)
   Dim ds as New DataSet()
   da.Fill(ds,"MyArticles")
   Return ds
End Function
```

برای ایجاد یک خروجی RSS از محتوای سایت، باید یک نمونه از کلاس Rss که نوشتیم بسازیم، سپس ویژگی های مختلف آن را مقداردهی کنیم و سپس متد GetDataSet را فراخوانی کنیم. بعد از اینکه دیتاست را بدست آوردیم فقط کافی است که خصوصیت ItemSource شئ گرفته شده از کلاس Rss را به همین دیتاست مقدار دهیم. به علاوه دو خصوصیت از شئ response یعنی ContentEncoding و ContentType را هم مقدار می دهیم .

```
Private Sub Page_Load(ByVal sender As System.Object,
   ByVal e As System.EventArgs) Handles MyBase.Load
   Dim r As New Rss
   Dim ds As DataSet = GetDataSet()
   r.OutputStream = Response.OutputStream
   r.RssTitle = "DotNetBips.com Latest Articles"
   r.PublisherUrl = Request.Url.Host
   r.Description = "DotNetBips.com - Applying.NET"
   r.Copyright = "Copyright (C) DotNetBips.com."
   r.Generator = "DotNetBips.com RSS Generator"
   r.ItemSource = ds
   r.ItemTitleField = "Article_title"
```

```
    r.ItemDescriptionField = "Article_Description"
    r.ItemPublicationDateField = "Article_pubdate"
    r.ItemUrlField = "Article_url"
    Response.ContentEncoding = System.Text.Encoding.UTF8
    Response.ContentType = "text/xml"
    Rss.PublishRss(r)
    Response.End()
End Sub
```

حالا تابع PublishRss را فراخوانی می کنیم و شئ گرفته شده از کلاس Rss را به این تابع پاس می کنیم.

تمام شد! حالا یک خروجی XML بر اساس استاندارد RSS داریم .

مبنع: http://www.dotnetbips.com

کوچ از ASP.NET 1.x به ASP.NET 2.0 - قسمت اول[1]

نسل اول ASP.NET در قالب نسخه‌های ۱,۰ و ۱,۱ ارائه گشت. گذشت زمان و کسب تجربیات متعدد، پیدایش نیازهای جدید، ضعف های موجود و نیز به دلیل ایجاد راه حل‌های تکراری ولی پرکاربرد توسط برنامه نویسان جهت حل برخی مسائل که همواره وقت زیادی را از آنها می گرفت، همگی سبب شدند تا مایکروسافت به فکر گسترش تکنولوژی .NET و ASP.NET به عنوان جزء مهمی از آن باشد. حاصل کار پیدایش نسخه بهتر شده دات نت و مهمتر از آن ASP.NET 2.0 جهت ارائه در سال ۲۰۰۵ بود .

مقدمه

زبانهای کامپیوتری و چارچوبهای کاری با توجه به نیازهای روز طراحان سیستمهای نرم افزاری روز به روز تغییییر کرده و گسترده تر می شوند ASP.NET .مهمترین راه حل مایکروسافت برای طراحان صفحات پویای وب نیز از این قاعده مستثنی نیست. این قابلیت مهم تکنولوژی .NET که به اعتقاد بسیاری از طراحان و کارشناسان بهترین در نوع خودش می باشد هر روز به منبعی کارآمد تر برای ساخت سایتهای بزرگ تبدیل می شود.

بعد از نسخه اوّل و دوم ASP.NET در نسخه های ۱,۰ و ۱,۱ در سالهای ۲۰۰۲ و ۲۰۰۳ که تفاوت چندانی با هم نداشتند نیازهای جدید، ضعف های گذشته و راه حل های متداول که همواره وقت زیادی را از طراحان می گرفت باعث شد مایکروسافت به فکر گسترش تکنولوژی .NET و ASP.NET به عنوان جزء مهمی از آن باشد. نتیجه مجموعه‌ی Visual Studio .NET Whidbey بود که در سال ۲۰۰۴ به بازار آمد.

[1] کیوان نیّری

این نسخه تفاوتهای اساسی و زیادی با نسخه های قبلی دارد که با به بازار آمدن این نسخه لازم است طراحان به فکر یادگیری قابلیت ها و خصوصیات جدید باشند لذا بر آن شدم تا این مقاله را برای ساده شدن راه طراحان ایرانی به رشته تحریر در آورم. به قولی که حضرت مولانا می فرماید:

وقت آن شد که به زنجیر تو دیوانه شویم (جناب بیل گیتس)بند را بر گسلیم از همه بیگانه شویم!!!

البته مفهوم این شعر چیز دیگری است!! ولی تمامی علاقه مندان و طراحان برای رشد سریع تر این تکنولوژی در ایران باید آن را زودتر فرا گیرند.

این مقاله بر اساس موضوع بندی و شیوه نگارش مقاله ای دیگر با همین عنوان در MSDN به طور خلاصه ولی با حفظ تمام مفهوم و در جهت ساده سازی با استفاده از کدهای مثال در مقاله اصلی نگاشته شده است.

اهداف ASP.NET 2.0

طراحان از ساخت ASP.NET 2.0 چهار هدف اصلی را دنبال می کردند:

۱ -افزایش اعتبار و کاربردی بودن برنامه های تحتWeb

در گذشته بیشتر برنامه های کاربردی تحت وب روی Microsoft Information Services (IIS) 5.0 اجرا می شدند ASP.NET 2.0 .بر اساس IIS 6.0 ساخته شده است که مدل پردازشی جدیدی را برای برنامه های کاربردی ایجاد می کند. مهمترین خصوصیت جدید آن این است که هر برنامه کاربردی در پروسه منفرد مخصوص به خود پردازش شده و نمی تواند برنامه های کاربردی دیگر را تحت تاثیر قرار دهد. ساده تر اینکه برنامه های کاربردی نمی توانند برنامه های دیگر را مختل کنند. هر برنامه به طور کامل جدا از بقیه اجرا می شود. پس اگر یک برنامه مختل شد بقیه تاثیر نمی پذیرند.

۲ -کاهش خطوط کد لازم برای اعمال متداول

ASP.NET 2.0شامل Wizard ها و کنترلهایی است که به شما در طراحی اعمال متداول کمک می کنند (برای مثال دسترسی به داده ها)

۳ -خصوصیات جدید کاربری برای شخصی سازی برنامه های کاربردیWeb

ASP.NET 2.0شامل کنترلهای ساخته شده برای کمک به مدیریت Account کاربران و شخصی سازی محتوا و لایه بندی برنامه های کاربردی است. سرویس عضو گیری به شما در مدیریت کاربران کمک می کند. کنترلهای جدید Login به همراه سرویس عضو گیری ساخته شده اند تا ایجاد Account و Login کاربران را بدون نیاز به هیچ کدی خودکار سازی کنند.

۴ -افزایش خصوصیات طراحی برای عمومی تر کردن ظاهر برنامه ها و لایه های آنها

ASP.NET 2.0شاملMaster Page ها، Themeها وSkin ها برای ساخت برنامه های کاربردی بزرگ وب با یک فرم طراحی ثابت برای صفحات است.

در ادامه به مرور جزئی تر این اهداف و نحوه پیاده سازی آنها می پردازیم.

کوچ ازASP.NET 1.x

اگر شما برنامه کاربردی در حال ساخت با نسخه ASP.NET 1.x دارید هیچ نگران نباشید چون ASP.NET 2.0به طور کامل با ASP.NET 1.x هماهنگ است. این به این معنا است که برنامه های کاربردی ASP.NET

1.xبدون هیچ مشکلی روی ASP.NET 2.0 اجرا می شوند ولی خواهید دید که لازم است برای بهتر شدن برنامه های کاربردی خود آنها را بروز کنید به طور کلی مدل گسترش ASP.NET 2.0 تغییر کرده است.

تغییرات در معماری

اساس معماری ASP.NET همیشه بر انعطاف و توسعه پذیری استوار بوده است ASP.NET 2.0 .نیز این روش را با مدل های Provider جدید که پشتیبان کننده خصوصیات جدید هستند دنبال کرده استUtility .ها و API های جدید برای گسترش کارایی و تنظمیات سایت ها اضافه شده اند. تمام اینها برای افزایش سرعت و سادگی در گسترش برنامه های کاربردی ASP.NET 2.0 طراحی شده اند.

مدلProvider

بسیاری از خصوصیات جدید در ASP.NET 2.0 بر مبنای ارتباط بین برنامه کاربردی تحت Web و منبع داده ای قرار گرفته اند بنابراین برای دسترسی به این هدف ASP.NET 2.0 از مجموعه ای از تامین کنندگان استفاده می کند.

مدل Provider در واقع ابزاری است که به گسترش دهندگان در چارچوب ASP.NET 2.0 برای رسیدن به منابع داده و رفع نیازهای لازم برای نیل به این هدف طراحی شده است. برای مثال طراح می تواند برای سیستم ذخیره اطلاعات مشخصات از یک منبع داده استفاده کند. بیشتر تامین کنندگان برای کار با بانکهای اطلاعاتی در سیستمها طراحی شده اند پس برنامه نویس مختار است کهMethod ها و کلاسهای لازم برای هر تامین کننده را انتخاب کند:

ASP.NET 2.0 Providers

مدل Provider مجموعه ای از محیط های کاربری را تعریف می کند و به منابع داده ای لازم برای فرامین خاص مرتبط می سازد. ASP.NET از Provider های گسترده و گوناگونی استفاده می کند که بعضی از مشهورترین و مهمترین آنها عبارتند از:

Membership:این سیستم عضویت تعیین اعتبار و مدیریت کاربران را پشتیبانی می کند.

Profile:اعمال مربوط به اطلاعات کاربری مخصوص به هر Profile را پشتیانی می کند.

Personalization:این Provider تامین کننده تنظیمات هر بخش شبکه و لایه های مخصوص هر کابر را پشتیبانی می کند.

Site Navigation:این Provider نقشه فیزیکی محل ذخیره صفحات ASP.NET را تامین می کند و برای تعیین انواع تغییرات در لینک های مورد استفاده در کنترلهای به کار رفته در صفحات به کار می رود.

Data Provider:ابزار ADO.NET همیشه از این مدل Provider برای ارتباط بین یک Database (بانک اطلاعاتی) و API خود استفاده کرده است ASP.NET 2.0 .نیز مانند گذشته از این قابلیت مهم تکنولوژی .NETبهره می گیرد و در این راه از خصوصیات جدید و پیش فرض در ADO.NET استفاده می کند.

هر یک از این Provider ها به طور مستقل از سایرین عمل می کنند پس شما می توانید Provider مخصوص Profileرا با Profile مخصوص Membership بدون هیچ مشکلی جا به جا کنید. در ادامه این مقاله مثال های خاصی در مورد چگونگی استفاده از این تامین کنندگان در بخش های مختلف ASP.NET 2.0 خواهید دید.

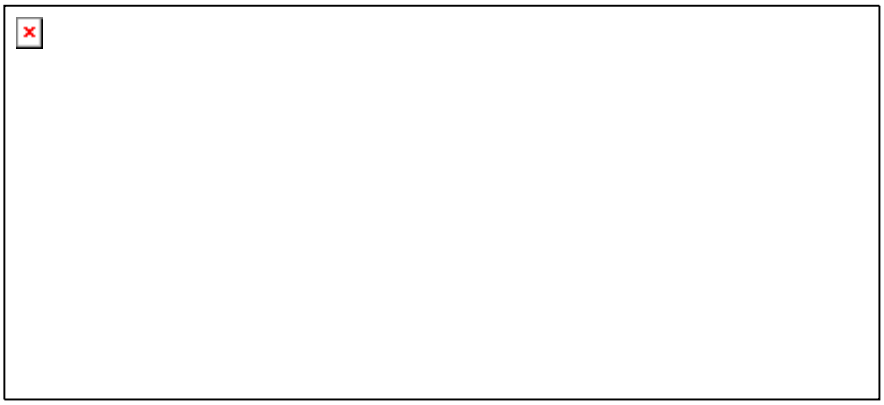
مدل کد نویسی ASP.NET 2.0

در ASP.NET 1.x شما می توانید یک صفحه ASP.NET را به دو شیوه طراحی کنید: اولین راه نوشتن مستقیم خطوط کد در بین کدهای HTML اصطلاحا (Code Inline) و دومین راه نوشتن کدهای برنامه به صورت جداگانه Code Behind بود .تکنیک دوم تکنیکی جدید و مهم بود چرا که در روش اوّل که از گذشته مورد استفاده بوده کدهای برنامه نویسی با کدهای HTML در کنار یکدیگر باعث ناخوانایی و مشکل در تغییرات چه در ظاهر و چه در متن کدهای برنامه بودند.

مدل Code Behind یک کلاس خارجی برای نگهداری کدها ایجاد می کرد در حالیکه ASPX کدهای HTMLو Tag های آنرا نگهداری می کرد. این تکنیک کدها را از محتوا جدا می کرد ولی مشکلات دیگری را ایجاد می کرد و آن هم نیاز برنامه نویس به نگهداری دو فایل مجزا برای هر صفحه بود ASP.NET 2.0 .هنوز هر دوی این مدلها را با تغییراتی نسبتا اساسی پشتیبانی می کند.

Code Inline

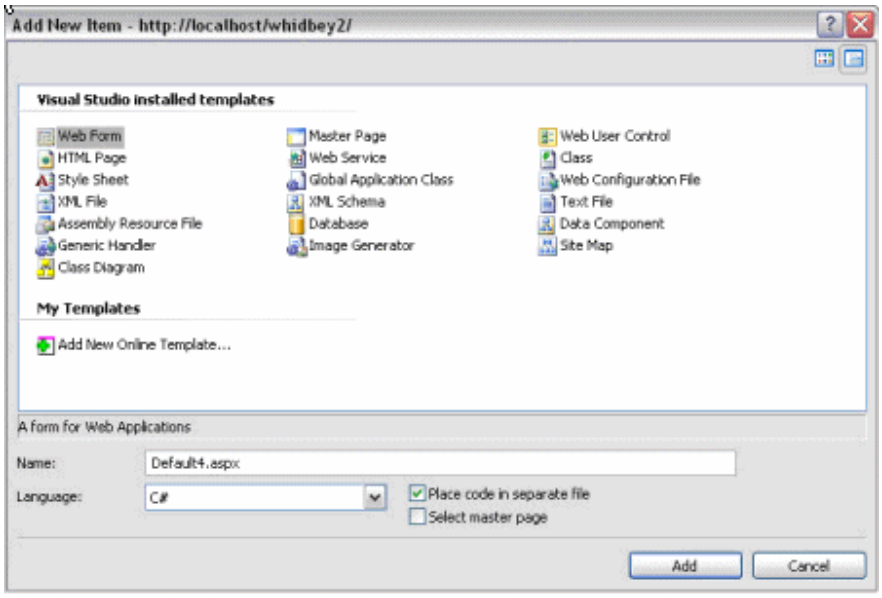
این مدل کد نویسی حالا مدل اصلی و پیش فرض برای Visual Studio 2005 است و جای مدل Code Behindرا گرفته، هر کدی که شما به صفحه اضافه کنید به طور خودکار به بلاک <Script> در فایل ASPX به جای کلاس Code Behind اضافه می شود Visual Studio 2005 .هنوز کد را در حالت نمایش مخصوص خود نشان می دهد به عبارتی دیگر شما در استفاده از Visual Studio تغییری نمی دهید به جز اینکه کدهای شما به طور خودکار و مستقیم داخل صفحه ASPX به جای کلاسی مجزا وارد می شود.



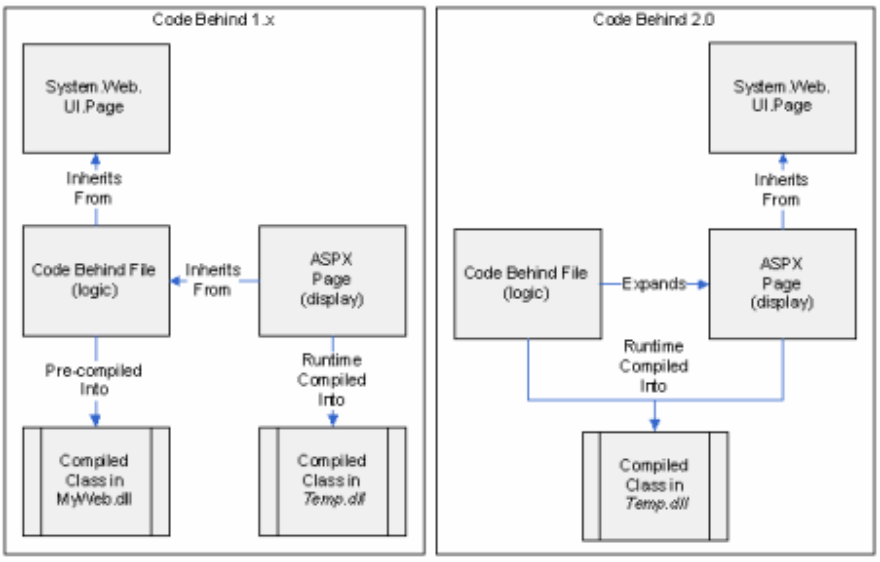
نکته قابل توجه این است که هنوز کدها از محتوا از طریق بلاکهای <Script> جدا هستند ولی در یک فایل قرار می گیرند. پس شما به عنوان یک برنامه نویس نباید نگران دو فایل باشید و فقط با یک فایل در گیر هستید.

Code Behind

اگر شما می خواهید با یک فایل کد جدا گانه کار کنید لازم است شما در زمان ایجاد یک صفحه ASPX این کار را انجام دهید خوشبختانه این کار به سادگی کلیک رو یک Check Box در زمان ایجاد فایل صفحه است.



تفاوت اساسی بین Code Behind در ASP.NET 1.x و ASP.NET 2.0 این است که فایل Code Behind اکنون به عنوان Partial Class به جای یک Class کامل محسوب می شود. یک Partial Class یک ساختار جدید .NET است که به شما اجازه می دهد یک کلاس منفرد را در فایلهای منبع مختلف استفاده کنید. در ASP.NET 2.0 یک Partial Class کاملا برای فایلهای Code Behind مناسب است به گونه ای که از روابط Inherit که در مدل قبلی Behind Code استفاده می شد بی نیاز است.



دو Partial Class یعنی Code Behind و ASPX در زمان Compile شدن به درون یک کلاس منفرد فراخوانی می شوند، آنگاه فایل Code Behind از تمام تعاریف کنترلها و فراخوانیهایی که با مدل قبلی Code Behind به اشتراک گذاشته شده اند مجزا است. مهمترین تفاوت اساسی قابل مشاهده در فایل Code Behind این است که دیگر شامل تمام کدهای خودکار و عمومی اضافه شده به هر صفحه برای مشخص کردن ارتباطات، منابع و کنترلها نیست. یک فایل Code Behind قدیمی شامل ناحیه Initialization برای تعریف هر کنترل قرار داده شده در هر صفحه است:

```
public class WebForm1 : System.Web.UI.Page
{
   protected System.Web.UI.WebControls.Label Label1;
   private void Page_Load(object sender, System.EventArgs e) { }
   #region Web Form Designer generated code
   override protected void OnInit(EventArgs e)
   {
      InitializeComponent();
      base.OnInit(e);
   }
   private void InitializeComponent()
   {
      this.Load += new System.EventHandler(this.Page_Load);
   }
   #endregion
   void Page_Load(object sender, EventArgs e)
   {
      Label1.Text = "Hello ASP.NET 2.0";
```

```
    }
}
```

از زمانی که تعاریف کنترلها به صفحه ASPX وارد شده اند فایل جدید تمام این کدها را حذف کرده است:

```
namespace ASP {
    public partial class Webform1_aspx
    {
        void Page_Load(object sender, EventArgs e)
        {
            Label1.Text = "Hello ASP.NET 2.0";
        }
    }
}
```

می توانید ببینید که فایل Code Behind جدید ساده تر است. فایل Code Behind به طور خودکار به تمام کنترلهای اضافه شده به هر صفحه ASP.NET دسترسی دارد.

/Code directory

آخرین تغییر بزرگ در مدل کدنویسی دسترسی مستقیم به یک مساله در ASP.NET است. بیشتر برنامه های کاربردی تحت Web به یک یا چند کلاس پشتیبانی کننده نیاز دارند. متد قبلی ایجاد یک پروژه جداگانه برای هر کلاس و اضافه کردن یک Reference به پروژه پشتیبانی کننده از درون برنامه کاربردی تحت Web بود.

حتی اگر شما پروژه جدید ایجاد نمی کردید لازم بود یک Reference به فضای نام (Name Space) مورد استفاده در برنامه های خود ایجاد کنید. نگهداری چند پروژه جداگانه برنامه ها را مشکل تر و بزرگتر می کردند که معمولا برای طراحانی که از یک یا دو کلاس ساده استفاده می کردند مناسب نبود.

ASP.NET 2.0روش جدیدی را با اضافه کردن کدهای پشتیبانی ارائه کرده است و آن یک Directory جدید با نام Code/ است که به طور خودکار به برنامه های کاربردی اضافه می شود و به عنوان یک منبع توسط برنامه کاربردی ASP.NET شناخته می شود. این به این معنی است که هر کلاسی که داخل این Directory اضافه شود به طور خودکار توسط صفحات ASPX در برنامه های کاربردی شما قابل دسترسی هستند.

کامپایلر Visual Studio .NET 2005 و ASP.NET به طور خودکار اسمبلی های لازم را از این کلاسها ایجاد می کنند و آنها را در اسمبلی های برنامه های کاربردی تحت وب شما قرار می دهند.

تنظیمات و نگهداری سایت

یکی از مشکلات بزرگ طراحان ASP.NET در گذشته، تنظیم مشخصات المانهای فایل Web.Config بود که متنوع و Case Sensitive بودند در ASP.NET 2.0 و Visual Studio .NET 2005 شما خصوصیات جدیدی دارید که در اجرای فرامین شما را یاری می دهند.

Intellisenseدر Web.Config

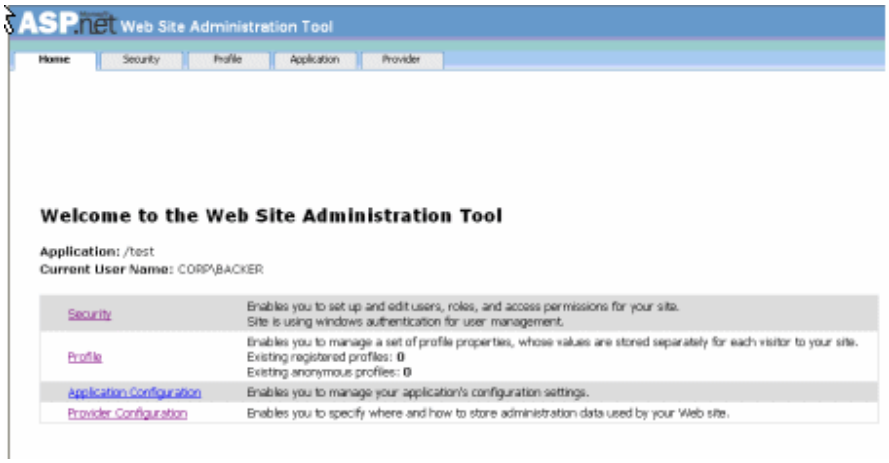
متاسفانه معادل فارسی مناسبی برای Intellisense به ذهنم نرسید. ولی این قابلیت مهم زبانهای Visual همان پنجره ای است که خصوصیات و متدهای اشیا مختلف را در زمان تایپ کدها به نمایش می گذارد.

اول از همه خصوصیت جدید و مهم Intellisense این است که برای هر فایل XML که دارای فرم تعریف شده ای است عمل می کند. برای فایل Web.Config این به آن معناست که شما کمک کامل Intellisense را برای ویرایش این فایل از طریق Visual Studio در اختیار دارید.

Intellisenseبرای کاهش شانس تنظیمات اشتباه در فایل کمک زیادی به شما می کند .از طرفی ASP.NET 2.0همچنین شامل یک Administrative Web site و کنسول مدیریتی مایکروسافت است که کارها را آسانتر می کند.

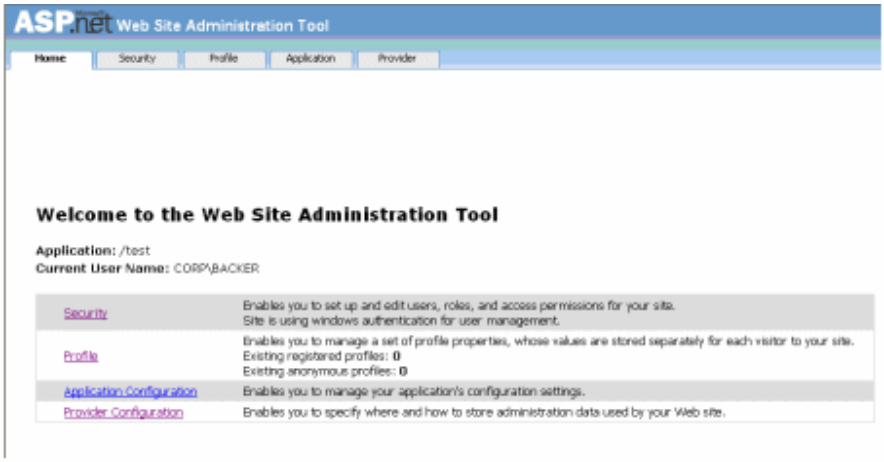
Administrative Web site

برای پیاده سازی پروسه مدیریت کاربران، ASP.NET 2.0یک ابزار تنظمیات وب سایت در اختیار شما می گذارد این ابزار یک وب سایت ساده است که تنها از طریق یک ارتباط امن یا به طور مستقیم از Local host قابل دسترسی است به وسیله این ابزار یک Administrator می تواند یک برنامه کاربردی را با تنظیم خدماتی مثل مدیریت کاربر، امنیت وProfile ها مدیریت کند. این ابزار همچنین اجازه می دهد که به سادگی شمارنده ها، خطایابی و Trace کردن اطلاعات برای برنامه را تنظیم کنید.



کنسول مدیریتی مایکروسافت

ASP.NET 2.0یک کنسول مدیریتی مایکروسافت (MMC) برای IIS ارائه می کند که به شما کمک می کند تصمیم بگیرید برنامه ها باید با چه نسخه ای از .NET Framework. اجرا شوند.



همچنین MMC IIS به شما کمک می کند نسخه ای که برنامه کاربردی شما در ASP.NET از آن استفاده می کند را مشخص کنید و محل فایل Web.Config را نشان می دهد. همچنین این کنسول دارای یک Botton مخصوص Edit Configuration برای تنظیم مشخصات فایل Web.Config بدون نیاز به دسترسی مستقیم به فایل Web.Config است.

تنظیمات اضافه شده به API

همچنین شما کلاس System.Configuration.Configuration را در اختیار دارید. این API به شما کمک می کند به فایلهای تنظیمی XML از طریق کدهای برنامه دسترسی داشته باشید.این به شما اجازه می دهد ابزارهای جدیدی ایجاد کنید. کد زیر روش فعال کردن Authentication را برای برنامه کاربردی روی Local Machineنشان می دهد:

```
[C#]
Configuration cfg = Configuration.GetConfigurationForUrl("/Application_name");
Response.Write( cfg.Web.Authentication.Mode.ToString() );

[VB.NET]
Dim cfg As Configuration =
Configuration.GetConfigurationForUrl("/Application_name")
Response.Write( cfg.Web.Authentication.Mode.ToString() )
```

کد زیر نیز حالت authentication Forms-Based را برای یک برنامه کاربردی فعال می کند:

```
[C#]
Configuration cfg = Configuration.GetConfigurationForUrl("/MyApp");
cfg.Web.Authentication.Mode = HttpAuthenticationMode.Forms;
cfg.Update();

[VB.NET]
Dim cfg As Configuration = & _
```

```
Configuration.GetConfigurationForUrl("/MyApp")
cfg.Web.Authentication.Mode = HttpAuthenticationMode.Forms cfg.Update()
```

تمام خصوصیات بالا به شما کمک می کنند که مقدار وقت و تلاشتان را برای انجام تنظیمات برنامه های کاربردی کاهش دهید.

تغییرات در Development

ASP.NET 2.0 و Visual Studio 2005 همچنین مقدار زیادی از تغییرات مورد نیاز برای گسترش برنامه های کاربردی وب در طول زمان را تغییر داده اند. در این بخش نگاهی به برخی از خصوصیات جدید می اندازیم که در ASP.NET 1.x نیز موجود بوده اند و اکنون کاملتر شده اند یا تغییر کرده اند.

ارتباط با Server

در ASP.NET 1.x و نسخه های قدیمی Visual Studio .NET شما برای ارتباط با یک IIS باید از Microsoft FrontPage Server Extensions استفاده می کردید. به عنوان یک طراح شما همچنین باید برای ایجاد سایتهای جدید به IIS دسترسی می داشتید که این برای شرکتهای بزرگ ایجاد مشکلاتی می کرد، پس این فرآیند تسهیل شد:

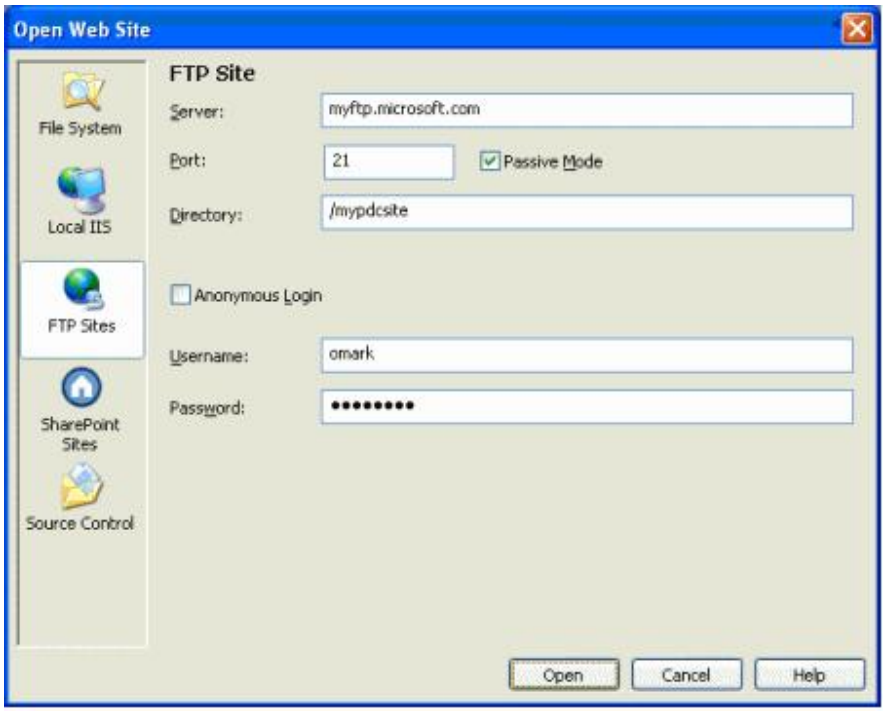
The Development Server

اول برای گسترش Visual Studio .NET 2005 یک Server ساخته شده به بازار آمده این Web Server فقط از طریق درخواستهای محلی قابل دسترسی است. بنابر این از نظر امنیتی قابل اطمینان نیست. این Server خصوصیات جدید خطایابی را پشتیبانی می کند و می تواند به عنوان یک ابزار گسترش برای برنامه های

کاربردی جدید وب به کار رود. زمانی که شما آماده انتقال و نصب نهایی برنامه های خود بودید به سادگی می توانید آنها را به IIS منتقل کنید.

The Production Server

اکنون زمانی که شما به IIS وصل می شوید انتخابهای متعددی دارید. زمانی که یک برنامه کاربردی وب را در Visual Studio .NETباز می کنید شما برای انتخاب متد Connection مورد سوال قرار می گیرید.



شما می توانید Microsoft FrontPage Server Extensions ، FTP، Share Pointیا مکانیسم های گوناگون دیگر را نیز برای انتقال فایلها به Server مورد استفاده قرار دهید.

کامپایل کردن برنامه های کاربردیASP.NET 2.0

تغییر بزرگ دیگر در ASP.NET 1.x مربوط به راه هایی است که برنامه های کاربردی وب می توانند کامپایل شوند. در ASP.NET 1.x برنامه ها در زمان اولین درخواست یا حالت Batch در زمان Start Up کامپایل می شدند. هر دو روش مزایا و معایبی داشتند. مهمترین عیب آنها این بود که شما باید کدهای کامپایل نشده را به Server می فرستادید که این می توانست بسته به نوع مدیریت امنیتی شما برای نیازهای امنیتی خطر ساز باشد.

ASP.NET 2.0متد جدید کامپایل کردن را در اختیار شما می گذارد که همان Pre-Compile کدهای داخلی به اسمبلی های باینری برای نصب (deployment) است که در نتیجه برنامه ها روی اسمبلی هایی که می توانند برای مدت طولانی نامگذاری و علامت گذاری شوند و فایلهای منبع گوناگون مثل تصاویر بنا می شوند.پس برنامه کاربردی Pre-Compile شده دارای امنیت بیشتری نسبت به برنامه های عادی ASP.NET است.

کوچ از ASP.NET 1.x به ASP.NET 2.0 - قسمت دوم[1]

نسل اول ASP.NET در قالب نسخه‌های ۱,۰ و ۱,۱ ارائه گشت. گذشت زمان و کسب تجربیات متعدد، پیدایش نیازهای جدید، ضعف های موجود و نیز به دلیل ایجاد راه حل‌های تکراری ولی پرکاربرد توسط برنامه نویسان جهت حل برخی مسائل که همواره وقت زیادی را از آنها می گرفت، همگی سبب شدند تا مایکروسافت به فکر گسترش تکنولوژی .NET و ASP.NET به عنوان جزء مهمی از آن باشد. حاصل کار پیدایش نسخه بهتر شده دات نت و مهمتر از آن ASP.NET 2.0 جهت ارائه در سال ۲۰۰۵ بود .

Navigation در سایت

یکی از مشکلات عمده و معمول در ASP.NET 1.x کار با کنترلهایی است که نیاز به آدرس نسبی صفحات دیگر در سایت را دارند مثل کنترل HyperLink که برای اشاره به به صفحه دیگری روی سایت آدرس نسبی آن صفحه را می پذیرد .حال اگر این صفحه در یک شاخه قرار داشته باشد و این کنترل به صفحه ای در شاخه دیگری اشاره کند با جابه جا شدن صفحه اول این آدرس نسبی نیز باید تغییر کند ASP.NET 1.x .راه حل مناسبی برای این مشکل ارائه نکرده بود و شما باید کدهای HTML داخل کنترل ها را تغییر می دادید ASP.NET 2.0 . خصوصیات جدید مناسبی برای بهبود Navigation در سایت و کاهش نیاز به تغییر و عملیات درصفحات وب را ارائه می کند.

ASP.NET 2.0 به شما اجازه می دهد برنامه های کاربردی خود را بر مبنای یک ساختار منطقی (Logical Structure)تعریف کنید. استفاده از ساختار منطقی برای Navigation به شما کمک می کند تا یک مسیر

[1] کیوان نیّری

Navigationبرای برنامه های کاربردی خود با استفاده از رابطه بین صفحات وب به جای مبنا قرار دادن آدرس هر یک از صفحات وب به طور منفرد تعریف کنید. با این روش شما می توانید صفحات خود را دچار تغییر ساختار کنید بدون آنکه نیازی به تغییر کدها باشد. با استفاده از ساختار منطقی همچنین نیازهای مربوط به Navigation مثل Menuها ، Tree Viewها و ... نیز برطرف می شود.

فایلWeb.SiteMap

کلاس SiteMap در ASP.NET 2.0 ساختار منطقی را برای وب سایت شما برآورده می کند. شما می توانید این ساختار را با استفاده از XML یا تامین کننده Navigation که در ASP.NET 2.0 موجود است تعریف کنید یا می توانید از تمام ساختارهای دیگر داده و تامین کننده دلخواه استفاده کنید. بعد از اینکه یک بار کلیت کار را تعریف کردید می توانید از ساختار Navigation در راه های مختلفی استفاده کنید. یک ساختار منطقی (Logical Structure)می تواند با دو گام ساده ساخته شود:

۱ -ایجاد یک فایل XML که با نام Web.SiteMap اجرا می شود که صفحات موجود در سایت را در یک روش سلسله مراتبی قرار می دهد Visual Studio.NET 2005 .به شما اجازه می دهد فایل های جدید با پسوند .Siteرا با انتخاب گزینه Add New Item و انتخاب قالب SiteMap ایجاد کنید و در زمان کار با این نوع فایل ها Intellisenseبه شما کمک می کند.

```
<?xml version="1.0" encoding="utf-8" ?>
<siteMap>
  <siteMapNode title="Home" url="default.aspx">
    <siteMapNode title="Article 1" url="~/articles/demoarticle1.aspx" />
    <siteMapNode title="Article 2" url="~/articles/demoarticle2.aspx" />
    <siteMapNode title="Article 3" url="~/articles/demoarticle3.aspx" />
```

```
  </siteMapNode>
  <siteMapNode title="Picture Gallery" url="~/PhotoAlbum/PhotoAlbums.aspx">
    <siteMapNode title=
"Meetings"url="~/PhotoAlbum/PictureAlbum.aspx?albumid=1"/>
    <siteMapNodetitle="Activities"url="~/PhotoAlbum/PictureAlbum.aspx?albumid=2
"/>
    <siteMapNode title=
"Training"url="~/PhotoAlbum/PictureAlbum.aspx?albumid=3"/>
  </siteMapNode>
</siteMap>
```

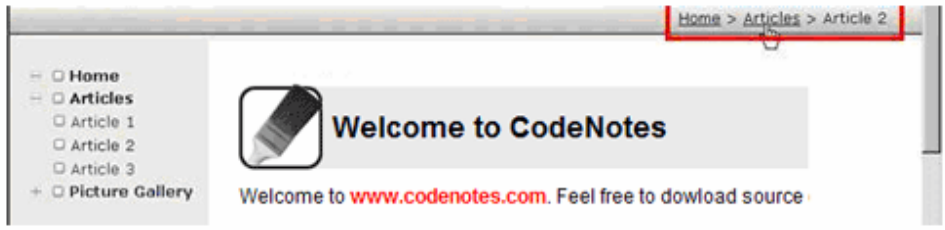
۲ -انتخاب کنترل SiteMap DataSource از ToolBox و وارد کردن آن به صفحه (Drap & Drown آن). این کنترل به طور خودکار به ساختار منطقی سایت که شامل Web.SiteMap می شود مقید می گردد تا زمانی که کنترل SiteMap DataSource روی صفحه است شما می توانید به سادگی آنرا به کنترلهای دیگری مثل Tree Viewو Menu ها مقید کنید.

```
<%@ page language="VB" master="~/Mysite.master" %>
<asp:content id="Content1" contentplaceholderid="LeftSideContent">
  <H2>Navigation </h2>
  <asp:treeview id="Navigation tree" runat="server" datasourceid="NavSource"/>
</asp:content>
```

کنترلهای Navigation

ASP.NET 2.0همچنین مجموعه ای از کنترلهای Navigation که می تواند با سرویس Navigation در سایت استفاده شوند به شما ارائه می کند. با استفاده از کنترلهای <asp:Menu> و <asp:TreeView> شما می

توانید یک ساختار Navigation همیشگی برای برنامه خود ایجاد کنید. کنترل جدید <asp:SiteMapPath> برای عمومی کردن یک ساختار Navigation برای یک Bread Crum Trail استفاده شود.



نقشه گذاری URL

خصوصیت جدید دیگر ASP.NET 2.0 برای Navigation نقشه گذاری URL است. سایت ها معمولا نیاز به URL های طولانی، سخت و پیچیده دارند (به URL ها MSDN نگاه کنید!!). اگر شما بخواهید URL های خود را در یک برنامه کاربردی ASP کلاسیک مخفی کنید شما باید یک ISAPI Handler مناسب برای پردازش دوباره درخواست ها بنویسید. ASP.NET 2.0 تنظیمات URL Mapping را به فایل Web.Config اضافه کرده است که با استفاده از آن یک برنامه نویس می تواند یک نقشه گذاری برای مخفی کردن یک URL به URL های مناسب برای کاربر ایجاد کند.

```
<urlMappings enabled="true">
   <add url= "~/Home.aspx"mappedUrl="~/Default.aspx?tabid=0" />
</urlMappings>
```

تنظیمات نشان داده شده در بالا باید تمام درخواستها برای صفحه Home.aspx را نقشه گذاری (و Redirect)کند به صفحه Default.aspx?tabid=0

دسترسی به داده ها و منابع داده

دسترسی به داده ها در ASP.NET 1.x بر مبنای Connection ها و Command هایی است که اطلاعاتی را که می توانند به کنترلهای مختلف ASP.NET مقید کنند.

ASP.NET 2.0ارتباط با ADO.NET را با تامین کردن منابع داده ای که کدها را برای باز کردن کپسولهای Connection ها و Command در یک فایل XML در داخل فایل Web.Config به کار می گیرد. مثل ASP.NET 1.x شما می توانید یک Wizard را در Visual Studio 2005 را برای ایجاد منابع داده ای به کار گیرید ASP.NET 2.0 .با منابع داده ای زیر می تواند مرتبط شود:

Microsoft Access:منبع داده ای Access می تواند به طور خودکار به یک پایگاه داده ای Access متصل شده و Query های مناسب را ایجاد کند.

اشیا (Objects):کنترل ObjectDataSource در بالای قسمت مرکزی ردیف لایه داده ها یا دیگر مجموعه اشیا قرا دارد. شما می توانید از این شئ برای ارتباط دادن یک یا چند مجموعه داده ها که می توانند به کنترلهای مقید سازی داده ها مرتبط شوند استفاده کنید.

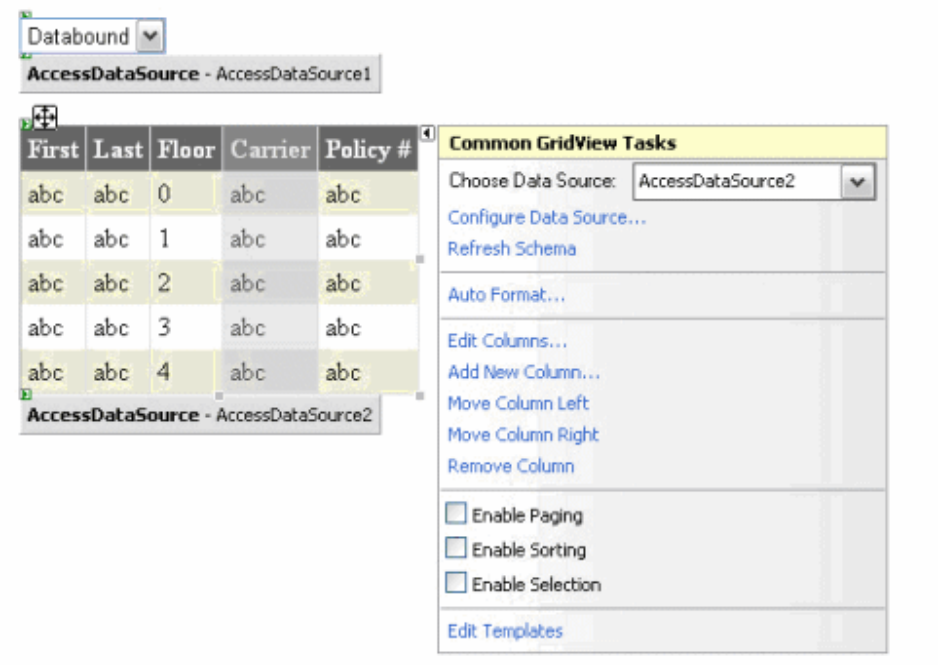
DataSet:منبع DataSetDataSource می تواند به داده های فهرستی مثل فایلهای مجزا شده با کاما یا فایلهای XML مرتبط شود.

XML:منبع XMLDataSource به داده های سلسله مراتبی در یک فایل XML متصل می شود. تفاوت اولیه بین XMLDataSource و DataSetDataSource در XMLDataSource است که برای مقید سازی به کنترلهای سلسله مراتبی مثل TreeView مناسب تر است. همچنین DataSetDataSource برای کنترلهای فهرستی مثل DataGrid مناسب تر است.

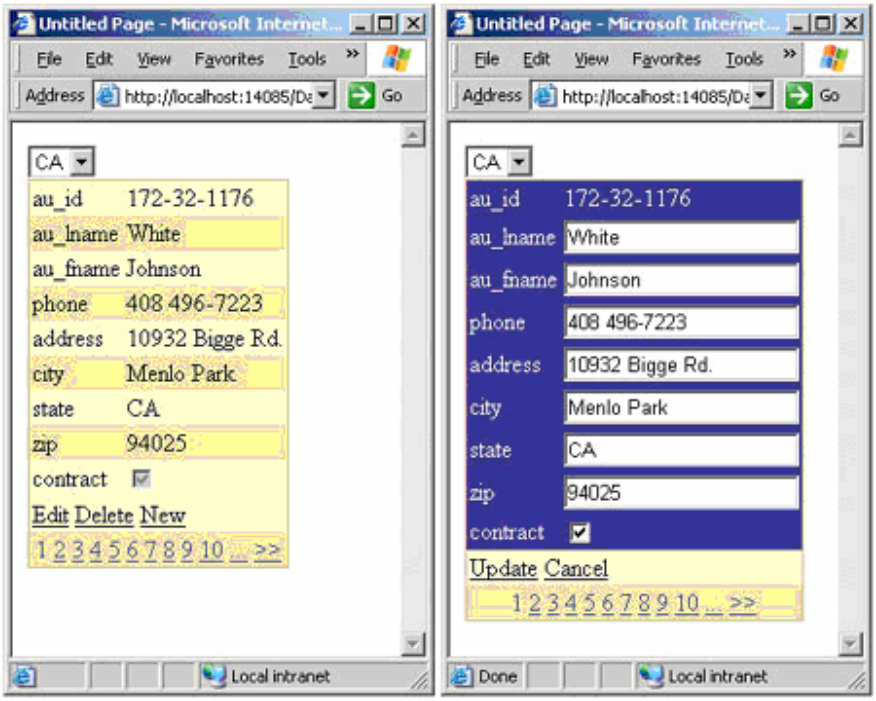
نقشه سایت SiteMapDataSource :یک منبع داده ای برای کنترلهای Navigation در سایت فراهم می کند.

کنترلهای جدید مقیدسازی داده ها

ASP.NET 2.0همچنین شامل دو کنترل جدید مقید سازی داده ها است. کنترل اول GridView است که روی DataGrid با تامین کدهای قابل تنظیم برای ویرایش خانه های جدول نشان دادن سطرها در صفحات مختلف، مرتب سازی، پاک سازی، انتخاب و بقیه اعمال معمول بسط یافته است.



کنترل دوم DetailsView است که بررسی جزئیات مربوط به رابطه مکمل GridView و DataGrid را بر عهده دارد.



کنترل DetailView یک رکورد را در هر زمان به نمایش می گذارد که کنترل بیشتری را برای نمایش، ویرایش، پاک سازی و یا ایجاد رکوردها به شما می دهد.

برنامه های کاربردی Web برای دستگاه هایMobile

امکان دارد شما به عنوان یک برنامه نویس با برنامه های کاربردی ASP.NET با Microsoft Mobile Internet Toolkitیا MMIT و کنترلهای خاص موبایل مواجه شده باشید. در ASP.NET 2.0 کنترلهای موبایل و MMIT به عنوان یک چارچوب کاری اصلی نقش ایفا می کنند. به طور معمول کنترلهای ASP.NET 2.0 مدل بازگشت توافقی که دارای رابطه مشابهی با کنترلهای قدیم Mobile است تامین می کنند. تمام این اعمال فقط در قالب یک بسته قال انعطاف و ساده برای استفاده در دسترس هستند.

خصوصیات جدید درASP.NET 2.0

معمولا هر بروز رسانی کلی در یک چارچوب کاری با اضافه شدن خصوصیات جدید و کلی همراه می شود که در این بخش به بررسی این خصوصیات در ASP.NET 2.0 می پردازیم.

Master Pages

این خصوصیت جدید به شما در طراحی و گسترش برنامه های کاربردی بر اساس یک ظاهر ثابت کمک می کند. این خصوصیت به شما کمک می کند تا یک قالب ثابت را برای چندین صفحه وب طراحی کرده و در آنها استفاده کنید.

اولین هدف Master Pages جلوگیری از نیاز به طراحی یک قالب و تکرار کد آن در صفحات مختلف است. فایده دیگر این خصوصیت این است که اگر شما نیازی به تغییر قالب صفحات داشته باشید به جای نیاز به تغییر تمام صفحات تنها یک صفحه را تغییر می دهید. این خصوصیت مشابه تکنیک Visual Inheritance در فرمهای Microsoft Windowsاست.

Master Page ها مشابه صفحات دیگر ASP.NET هستند و تفاوت آنها تنها در محیط (که پسوند این صفحات master. به جای aspx.) و برخی کنترلهای خاص است Master Pages .می توانند شامل یک یا چند کنترل <asp:ContentPlaceHolder> باشند. این کنترلها نواحی مورد نیاز برای تغییر در محتوا را شامل می شوند. اساساً هرچه در ContenetPlaceHolder نیست در تمام صفحات به نمایش در می آید.

Visual Studio 2005به طور خودکار بیشتر این کد را برای شما ایجاد می کند پس شما نیازی به نوشتن کدهای پیچیده HTML ندارید. یک Master Page می تواند شامل قطعه کد زیر باشد:

```
<%@ master language="C#" compilewith="site.master.cs"
classname="ASP.site_master" %>
<html>
<head runat="server"><title>Untitled Page</title></head>
```

```
<body>
  <form runat="server">
    <asp:contentplaceholder id="ContentPlaceHolder1" runat="server">
    </asp:contentplaceholder>
  </form>
</body>
</html>
```

جدای از این تغییرات کلیدی هر Master Page می تواند شامل هر کنترلی که در صفحات عادی ASP.NETموجود است نیز باشد.

اگر چه Master Page ها و Frame ها دارای اهداف یکسانی هستند ولی Master Page ها وظیفه مندی خاصی را ارائه می کنند. بر خلافFrame ها استفاده از Master Page ها به شما قابلیت های زیر را می دهد.

۱ -یک بار طراحی کنید و به تعداد دلخواه استفاده نمائید. بعبارت دیگر در این حالت ساختار و نمای کلی یک صفحه از Master Page و محتوای آن از صفحه مربوطه گرفته می شود و در نهایت و برخلاف Frame ما تنها یک صفحه عادی وب خواهیم داشت.

۲ -امکان استفاده آسان و راحت از انواع کنترلها به جای تگ های دست و پا گیر HTML در مقایسه به Frameها و نحوه ساخت آنها. همه این کارها توسط Visual Studio برای شما به آسانی انجام می گردد.

۳ -سازگاری کامل صفحات محتوا با .Master Pages پس شما نباید ترسی از این داشته باشید که ورود کدهای HTML محتوا به صفحه لایه های دیگر را تحت تاثیر قرار دهد.

خلاصه اینکه Master Page ها به طرز بسیار جالبی فرآیند ساخت برنامه های کاربردی وب بر مبنای ظاهر ثابت را بدون توجه به تعداد صفحات ساده سازی کرده است.

صفحات محتوا(Content Pages)

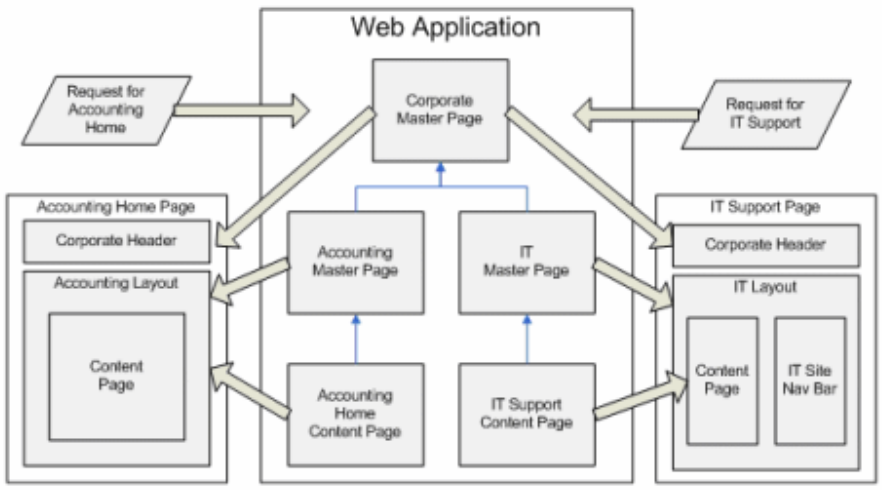
صفحات محتوا به Master Page ملحق می شوند و یک محتوا را برای کنترلهای ContentPlaceHolder در Master Page ها تعریف می کنند. صفحه محتوا شامل کنترلهای <asp:content> است که به منبع کنترل <asp:contentPlaceHolder>در Master Page از طریق ContentPlaceHolder ID پیوند خورده است. صفحات محتوا و Master Page ها در یک دسترسی به صورت یک جا به هم می پیوندند.

```
<%@ page language="VB" master="~/Mysite.master" %>
<asp:content id="Content1" contentplaceholderid="LeftSideContent">
   <H2>Navigation </h2>
   <asp:treeview id="Navigation tree" runat="server" datasourceid="NavSource"/>
   </asp:content>
   <asp:content id="Content1" contentplaceholderid="RightSideContent">
   <asp:label runat="server">Support section</asp:label>
</asp:content>
```

یک بار دیگر: کاربر حقیقتا یک درخواست را برای صفحه محتوا ارسال می کند ولی ASP.NET به طور خودکار این محتوا را با Master Page آمیخته و به کاربر ارسال می کند.

Master Pageهای تو در تو

در مثالهای مجزاMaster Page ها باید لایه بندی شوند تا دسترسی به کنترل روی سایت محقق شود. برای مثال ممکن است شرکت شما یک سایت داشته باشد که یک قسمت بالایی و پایینی ثابت برای هر صفحه دارد ولی بخش حسابداری شما قالب متفاوتی با بخش IT داشته باشد.



با استفاده از Master Page های لایه بندی شده در Master Page های سطح بالای شرکت، شما می توانید کنترل کاملی روی محتوایی که به نمایش در می آید داشته باشید. نکته اساسی این است که به یاد داشته باشید در نهایت یک صفحه توسط یک کاربر درخواست شده و یک صفحه توسط او دریافت می گردد اگر چه ممکن است چند صفحه Master Page و محتوا مورد پردازش قرار گیرند.

باطل کردنMaster Page ها

هدف اساسیMaster Page ها ایجاد یک ظاهر ثابت برای همه صفحات در برنامه کاربردی شماست اگر چه ممکن است حالتهایی پیش آید که ممکن است شما بخواهید محتوای متفاوتی را به صفحه خود وارد کنید. که برای حل این مشکل به سادگی از یک کنترل محتوایی استفاده می کنید.

برای باطل کردن خصوصیات یک Master Page به طور پویا شما باید بخشهای عمومی (Public) کلاس Master Pageرا برنامه ریزی کنید که شامل آن چیزهایی است که شما ایجاد کرده اید. بنابراین اگر می خواهید

خصوصیات <title> و <keyword> از بخش <head> را در صفحات HTML باطل کنید شما باید تابعی بسازید که خصوصیات پیش فرض را در Master Page شما ویرایش کند:

```
<head runat="server" id="Head1">
   <title><% =m_HtmlTitle %></title>
   <meta name="keywords" content="<% =GetKeywords() %>" >
   <meta name="description" content="A newsletter focused on meeting the needs of
.NET developers, providing cool tips for the advanced programmer, and keeping you
informed on what's happening with .NET!" >
   <LINK href="<% =Request.ApplicationPath %>/portal.css" type="text/css"
rel="stylesheet">
</head>
```

Skinsو Themes

می دانید که یک طراح برای زیبا سازی و تغییر ظاهر صفحات خود نیاز به استفاده از فایلهای CSS یا استفاده از CSSبه صورت ضمنی دارد .این مساله منجر به استفاده از یک شکل ثابت در صفحات می شود. برای حل این مشکل نیز در ASP.NET 2.0 راه حلهایی در نظر گرفته شده است.

Themes

Themeها مشابهCSS Style ها هستند چرا که هر دو برای تعریف خصوصیات و صفات معمول در یک یا چند صفحه به طور ثابت استفاده می شوند ولی به هر شکل Theme ها تفاوتهایی نیز با Style Sheet ها دارند:

۱- Theme ها می توانند خصوصیات زیادی را برای هر کنترل تعریف کنند در حالیکه Style ها فقط مجموعه ای از صفات معمول را تعریف می کنند.

۲- Theme ها می توانند شامل فایلهای مختلفی باشند در حالیکه CSS Style ها نمی توانند.

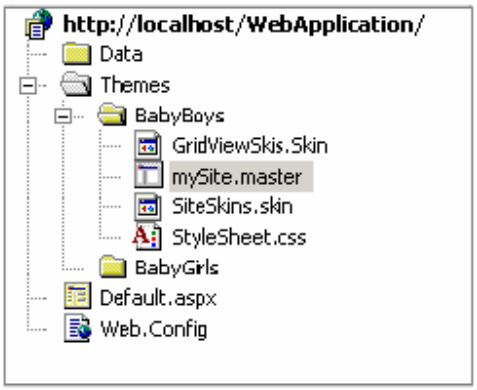
۳- Theme ها به مانند Style Sheet ها از شیوه نامه آبشاری استفاده نمی کنند.

۴- Theme ها می توانند شامل Refrence به Style Sheet ها باشند.

هر برنامه کاربردی یک شاخه Themes دارد .هر Theme خاص زیر شاخه مخصوص به خود را دارد که شامل فایلهای Skin و بقیه فایلهای خود است.

تعریف یکTheme

شما می تونید Theme ها و Skin ها را در زیر شاخه هایی که از شاخه Themes مشتق می شوند تعریف کنید.هر زیر شاخه شامل یک Theme است. هر زیر شاخه شامل فایلهای .skins است که به مانند دیگر منابع مورد استفاده توسط Theme ها مانند Style Sheet ها و تصاویر قرار گرفته اند.



تعریف معمولی و نرمال Skin در فایلهای .Skin موجود است و بسیار شبیه به تگهای مورد استفاده در نمونه کنترلهای فایلهای ASPX است. برای مثال، در شاخه Themes یک زیر شاخه با نام Pink ایجاد کنید. برای ایجاد یک فایل .skin برای Theme خود به سادگی کد زیر را اضافه کنید و در شاخه Pink ذخیره کنید:

```
<asp:DropDownList runat="server" BackColor="hotpink" ForeColor="white" />
<asp:DataGrid runat="server" BackColor="#CCCCCC" BorderWidth="2pt"
BorderStyle="Solid" BorderColor="#CCCCCC" GridLines="Vertical"
HorizontalAlign="Left">
   <HeaderStyle ForeColor="white" BackColor="hotpink" />
   <ItemStyle ForeColor="black" BackColor="white" />
   <AlternatingItemStyle BackColor="pink" ForeColor="black" />
</asp:DataGrid>
```

آنگاه Theme می تواند در تمام صفحات برنامه کاربردی شما مورد استفاده قرار گیرد .بنابر این رنگ پس زمینه Data Gridشما به رنگ صورتی تیره در می آید.

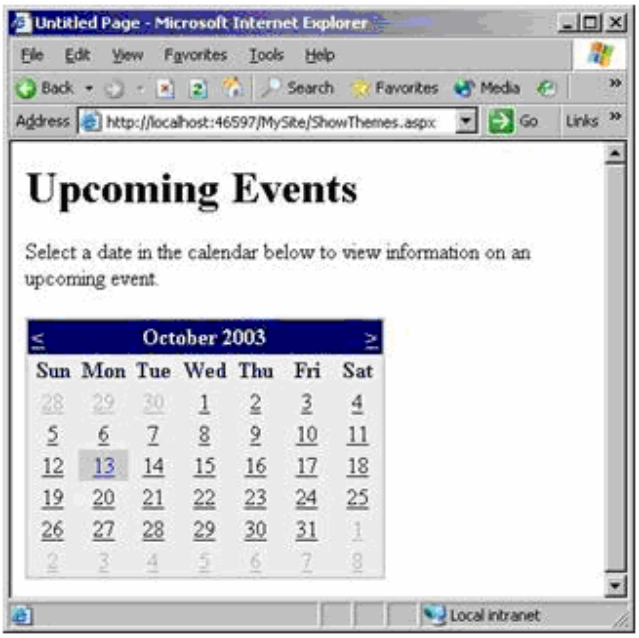
استفاده از یکTheme

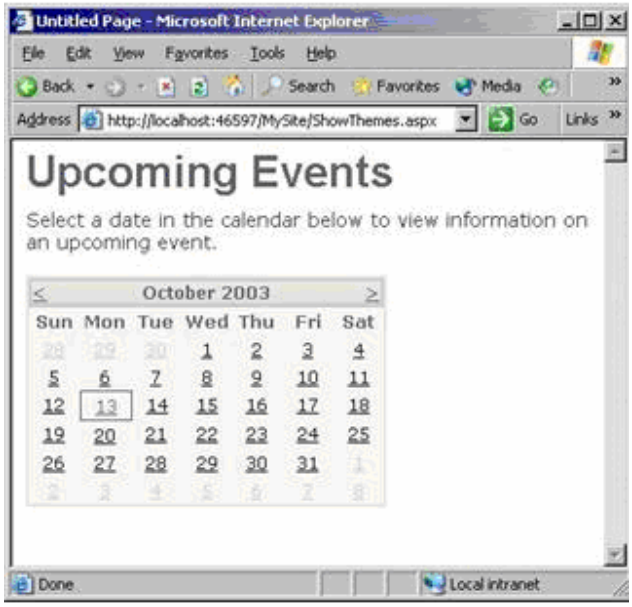
Themeها می توانند به صورتهای زیر به کار روند:

۱ -در صفحه با استفاده از تگ منفرد<% "page language="VB" theme="Pink @%>

۲ -در تمام سایت با استفاده از المانهای تنظیمی فایل Web.config بصورت <"pages theme="Pink>/

فایل Web.Config فایل اصلی تنظیمات برای هر برنامه کاربردی است. تاثیر استفاده از Theme ها را می توانید در کنترلهای خود ببینید.





Skins

یک Skin مجموعه ای از قالبها و خصوصیات است که می تواند برای استاندارد سازی اندازه فونت و دیگر صفات کنترلها روی یک صفحه به کار رود. شما می توانیدSkin ها را برای ایجاد یک ظاهر مشخصی برای کنترلها به کار گیرید که این تنظیمات می تواند توسط شما یا کاربر در زمان دسترسی به صفحه صورت گیردSkin .ها می توانند با استفاده از خصوصیت SkinID در کنترلها در آنها به کار روند.

```
<!- Default Skin -->!>
<asp: label runat="server" Font-names="verdana, arial" font-size="10pt" ForeColor=
"#000066" BackColor="transparent"/>
<!- Title Skin -->!>
<asp: label runat="server" id="foo" skinid="Title" Font-names="verdana, arial" font
size="18pt" ForeColor= "#000066" BackColor="transparent" font-bold="true" font-
underline="true"/>
```

وقتی یک بار یک Skin را تعریف کردید می توانید آنرا در کنترلهای مختلف خود که دارای SkinID هستند به کار گیرید.

به کار گیری یک Skin برای یک کنترل

اگر یک Skin پیش فرض موجود باشد و با استفاده از یک Theme تعریف شده باشد Skin پیش فرض به طور خودکار در یک کنترل استفاده می شود. اگر شما خصوصیت SkinID را برای کنترل تعریف کنید، Skinپیش فرض با Skin خاصی که در خصوصیت SkinID به آن اشاره شده جا به جا می شود. این خصوصیت می تواند در زمان گسترش یا اجرا تنظیم شود.

عضویت(User Membership)

یکی از مسائل در گسترش ASP مساله سیستم عضویت است که باید از ابتدا توسط برنامه نویس طراحی شود. این کار مشکل نیست ولی وقت گیر است. سیستم عضویت و کنترلهای Login این مساله را تا حد زیادی بهبود بخشیده است.

به طور خلاصه باید گفت این سیستم یک ساختار کامل برای عضویت کاربران و مدیریت آن را برای شما ایجاد کرده است. این سیستم اطلاعات را در یک پایگاه داده به صورت کاملا امن ذخیره می کند. این سیستم به طور پیش فرض از Microsoft Access و Microsoft SQL Server پشتیبانی می کند. ولی شما می توانید نیازهای دلخواه خود را از طریق ایجاد تنظیمات در فایل Web.Config ایجاد کنید .همچنین شما می توانید از کنترلهای جدید Login در ASP.NET 2.0 برای خودکار سازی سیستم ورود و خروج و ثبت نام کاربران استفاده کنید.

کنترلهای Login

ASP.NET 2.0کنترلهای Login را برای کمک به ایجاد و مدیریت سیستم کاربری در اختیار شما می گذارد بدون آنکه نیازی به نوشتن حتی یک خط کد داشته باشید .شما می توانید به سادگی کنترل <asp:CreateUserWizard>را به صفحه خود وارد کنید و از آن برای ثبت نام کاربران به صورت مرحله مرحله استفاده کنید.

همچنین این کنترلهای جدید محیط مناسبی را در اختیار شما می گذارند تا ظاهر آنها را با برنامه کاربردی خود منطبق کنید. خصوصیاتی مثل Label و Value و Error message validation نیز برای هر خصوصیت موجود

است. با استفاده از کنترلهای LoginName و LoginStatus شما می توانید به هر کاربر یک پیغام خوش آمد مناسب ارائه کنید بدون آنکه نیازی به حتی یک خط کد داشته باشید.

کنترل LoginView به شما کمک می کند تا مشخص کنید چه محتوایی به هر کاربر نشان داده می شود (بدون نوشتن یک خط کد). محتوا با توجه به نقش و سطح دسترسی هر کاربر تعیین می شود. در برنامه های کاربردی ASPشما باید کدهایی را برای انجام این اعمال می نوشتید تا پیامهای مناسب هر کاربر را نمایش دهد و محتوای مناسب هر کاربر را به او ارائه کند.

Profileها

خصوصیات مربوط به Profile در ASP.NET 2.0 به شما اجازه می دهند تا اطلاعات هر کاربر که با سایت به اشتراک گذاشته را تعریف و ذخیره کنید. در برنامه های ASP شما باید کد نوشته شده خود را برای جمع کردن اطلاعات کاربر گسترش می دادید و برای session کاربر ذخیره می کردید و در پایگاه های داده ای لازم ذخیره می کردید Profile .ها مستقیما از طریق شئ Profile در هر صفحه ASPX قابل دسترسی اند.

تعریف یکProfile

از طریق فایل Wen.Config یا Machine.Config و مقادیر <property> می توانید اطلاعات کاربری مربوط به Profile مثل نام، آدرس و آدرس پست الکترونیک و... را برای هر کاربر تعریف کنید.

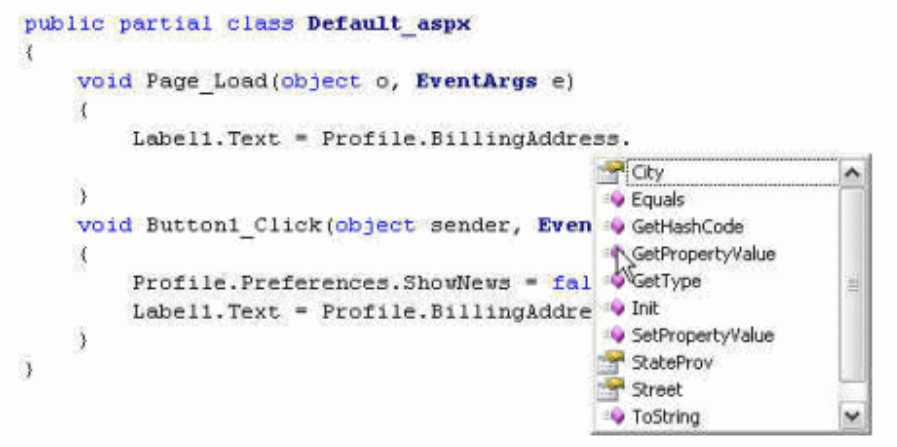
```
<profile>
  <group name="BillingAddress">
    <add name="Street" type="System.String" />
```

```
    <add name="City" defaultValue="Toronto" type="System.String" />
    <add name="StateProv" type="System.String" />
    <add name="ZipPostal" type="System.String" />
  </group>
</profile>
```

بعد از ایجاد این خصوصیات ASP.NET 2.0 به طور خودکار از محتوای آنها محافظت می کند.

استفاده از Profile ها

زمانی که یک Profile را تعریف کردید Visual Studio 2005 به طور خودکار اطلاعات آن را از طریق شئ Profileدر اختیار شما می گذارد.



Visual Studio 2005همچنین به طور کامل Intellisense را برای هر Profile پشتیبانی می کند. هر زمانی که شما تغییری را در تعریف یک Profile انجام دادید Visual Studio به طور خودکار Intellisense را به درستی برای شما تغییر داده و به سرعت بعد از اعمال تغییرات در فایل Web.Config در اختیار شما می گذارد.

بهترین راه یادگیری ASP.NET[1]

همواره ایمیل های زیادی بدستم رسیده است که می پرسند: بهترین راه یادگیری ASP.NET چیست؟ یا اینکه برای یادگیری ASP.NET از کجا باید شروع کنیم؟ و سوالاتی از این قبیل. در این مقاله سعی کرده ام که تجربیات خودم را در این زمینه با شما در میان بگذارم و احتمالاً بتوانم جوابی کاربردی و عملی به این سوال بدهم. سوالی که احتمالاً شما هم به دنبال جواب آن هستید .

نظر من به تعداد انسانها، راه برای یادگیری ASP.NET هست! شوخی نمی کنم، جدی می گویم. به نظر من هر شخصی روش یادگیری خودش را دارد. فرقی هم نمی کند که بخواهید ASP.NET یاد بگیرید یا آشپزی یا هر چیز دیگری. شما همانطور که اگر بخواهید آشپزی یاد بگیرید ممکن است از کتاب، کلاس، سعی و خطا، تجربه و حتی از وب استفاده کنید، برای ASP.NET و اصولا هر چیز دیگری هم ممکن است از همین ابزار و راهها استفاده کنید.

چیزی که من می خواهم بگویم توضیح واضحات است. اما چون خیلی ها سوال می کنند ناچارم اشاره ای به این موضوع داشته باشم. به نظر من بیشتر، افراد مبتدی و تازه کار هستند که نیاز دارند که پاسخ این سوال را بدانند چون کسانی که به نوعی ASP.NET را یاد گرفته اند یا با آن آشنا هستند خود می دانند که چه کار باید بکنند و چگونه یاد بگیرند.

کتاب

خیلی از افراد سوال می کنند که مثلا کدام کتاب خوب است؟ به عقیده من برای یک فرد تازه کار که می خواهد با الفبای کار آشنا شود، هر کتابی در زمینه آموزش مقدماتی و کلیات ASP.NET خوب و مفید است. حتی بسیاری از کتابهای تخصصی هم فصلهای اولیه خود را به آموزش و مرور مقدمات ASP.NET تخصیص می دهند. پس دنبال

[1] مدیریت سایت

کتاب خاصی نباشید. اولین کتابی را که بدست آوردید بنشینید و بخوانید. بالاخره هر کتابی هر چند سطح پائین هم که باشد آنقدر مطلب دارد که یک تازه کار را برای مدتها به خود مشغول کند.

حرفه‌ای ترها هم که خود می دانند دنبال چه هستند و چه کتابی به کارشان می خورد. دقت داشته باشید که متاسفانه سطح کتابها جهت استفاده حرفه‌ای ها بسیار پائین است به گونه‌ای که آنها معمولاً کتاب مورد نظر خود را نمی توانند پیدا نمی کنند. لابد نویسندگان فکر می کنند که حرفه‌ای ها نیازی به کتاب ندارند و آنقدر توانا هستند که یا خودشان فکر کنند یا مطلب شان را در وب پیدا کنند. چیزی که بحث بخش بعدی این مقاله است.

به هرحال آنقدر در کتابهای آموزشی ASP.NET در مورد فرم های وب و نحوه استفاده از ADO.NET برای ارتباط با بانک اطلاعاتی و XML گفته اند که هیچگاه در مورد چگونگی طراحی ساختار و لایه های خود برنامه یا سایت چیزی گفته نشده است. مقوله ای که بدون آن معماری را می ماند که آجر و سنگ و سیمان دارد اما نقشه ندارد.

سایتهای آموزشی

ابزار بعدی، استفاده از سایتهای آموزشی موجود در وب هست. باور کنید که برخی مطالب ذکر شده در سایتهای آموزشی را هیچ وقت در هیچ کتابی نخواهید یافت و اصولا آنها هیچوقت چاپ نمی شوند .بخصوص زمانی که دارید وارد قلمرو حرفه ای می شوید. با بکارگیری گوگل، این گونه سایتها هیچوقت برای شما مخفی یا ناشناخته نمی مانند. به شرطی که در انتخاب کلید واژه‌ی جستجوی خود دقت کنید .

سایتی مثل W3Schools [1] و بخش آموزش سریع یا QuickStart [2] مایکروسافت می توانند منابع خوبی از جنس دوره های آموزشی باشند. اما انتظار نداشته باشید که در دیگر سایتهای آموزشی/مقاله‌ای ASP.NET بتوانید دوره های منظم و مرحله به مرحله‌ی کلاسیک و جزوه ای پیدا کنید. بیشتر سایتها پر هستند از مقاله های رنگارنگ و

[1] www.w3schools.com/aspnet/default.asp
[2] samples.gotdotnet.com/quickstart/default.htm

متنوع که لزوما وحدت موضوعی نداشته و صرفاً برحسب نیاز خود برنامه نویسان تهیه شده اند. من اینگونه سایتها و مطالب را برای دوران پس از مقدماتی توصیه می کنم.

بیشتر این سایتها دارای خبرنامه الکترونیکی رایگان می باشند. با عضویت در آنها و دریافت مرتب تیتر مطالب شان، آنها را زیر نظر بگیرید و اگر مقاله‌ای داشتند که به کار شما می خورد، به سراغش رفته و استفاده کنید. این مساله سبب می شود که شما مجبور نباشید که همواره به آن سایتها سر بزنید یا همه مطالب شان را بخوانید و یا احتمالا مقاله‌ی مفیدی را از دست بدهید. از این رو ست که همواره بر داشتن خبرنامه‌ی الکترونیکی برای سایتها و عضویت افراد در آنها تاکید شده است. خوشبختانه سایت IranASP.NET هم از همان روزهای آغازین این مهم را سرلوحه کار خود قرار داده است. دقت داشته باشید که شما هر چه جلوتر بروید و حرفه‌ای تر شوید، وقت کمتری برای مطالعه و یادگیری خواهید داشت. بخصوص اگر به حدی برسید که درگیر یک کار یا پروژه هم بشوید!

کلاس

کلاس های آموزشی هم می توانند مفید باشند. به گونه ای که در مدت زمان کوتاهی سرنخ مطالب جدیدی را به شما می دهند و گاهاً ممکن است مطلبی را مطرح کنند که شما هیچ وقت در حالت خودآموزی به آن برخورد نکنید. به کلاسهای آموزشی صرفا به عنوان نقطه شروع نگاه کنید و نه چیز دیگر. هیچ کلاسی نمی تواند ادعا کند که شما را یک ماهه یا بیشتر به یک برنامه نویس حرفه ای وب تبدیل می کند که اگر ادعا کرد خود معیار خوبی است جهت میزان صداقت گردانندگان آن کلاس و سطح آن .

معمولاً شما نمی توانید قبل از تشکیل یک کلاس به میزان مفید بودن آن برای یادگیریتان پی ببرید. هر چقدر هم از گردانندگان آن سوال کنید، آنها آنقدر کلمات فنی و قلمبه سلمبه بلد هستند که شما فکر کنید بیسواد مطلق هستید. نمی گویم که همه‌ی کلاس ها اینگونه هستند و یا اینکه شما بدون تحقیق و بررسی به هر کلاسی بروید. اما می گویم که اندکی هم ریسک پذیر باشید چون ممکن است از هزینه و وقت تان آنگونه که انتظار داشته باشید در پایان

کلاس بهره نبرده باشید. این را کاریش نمی توان کرد و از ملزومات کار است که البته اشکالی هم ندارد و همین مساله خود برای شما نکته‌ی آموزشی خواهد شد اگرچه در زمینه ASP.NET هم نباشد!

از دیگر فواید کلاس می توان به امکان پیدا کردن دوست و یا همکار و مهم تر از همه پروژه اشاره نمود. خیلی از همکارهای کاری و پروژه ای می تواند از درون همین کلاس ها رقم بخورد چه برای دانشجویان چه برای شخص مدرس!

زبان زبان زبان

بله، زبان البته از نوع انگلیسی آنقدر مهم است که سه بار در عنوان این بخش تکرار شود. این را من برای شما که فارسی زبان هستید می گویم. شما نمی توانید یک برنامه نویس وب باشید اما زبان انگلیسی بخصوص در زمینه خواندن یا همان Reading را بلد نباشید. چون شما نمی توانید همه مطالب خود که هیچ بلکه حتی کسری از آن را هم به زبانی غیر از انگلیسی پیدا کنید.

شما برای خواندن و یادگیری به زبان نیاز دارید. همه‌ی کتابها انگلیسی هستند و آنهایی هم که مثلاً ترجمه فارسی شده اند را خیلی جدی نگیرید. آنها بیشتر بدرد ناشرشان می خورد و کمی هم مترجم شان و نه شما. شما باید اصطلاحات را به شکل انگلیسی شان یاد بگیرید و نه ترجمه های من در آوردی و عجیب و غریب فارسی. کتابهای فارسی در این زمینه را اغلب کسانی ترجمه می کنند که به نوعی ماشین ترجمه هستند و حتی ممکن است ندانند که آنچه را ترجمه کرده اند به چه دردی می خورد.

خوشبختانه مطالب فنی کامپیوتری نمی توانند دارای متون پیچیده و سخت انگلیسی باشند. عمده‌ی کلمات که اسامی خاص و اصطلاحات فنی هستند و بقیه هم افعال و کلمات ساده و معمولی زبان می باشند. پس شما قرار نیست که نمایشنامه‌ی شکسپیر را بخوانید.

همچنین شما برای برنامه نویسی تان و انتخاب اسامی متغیرها، کلاس ها و متدهای مورد نیازتان باید بتوانید اسامی با مسمایی را انتخاب کنید. شما باید بتوانید شرح یا کامنتی هر چند کوچک در لابلای خطوط برنامه تان بنویسید تا کد برنامه شما به یک غول دست نیافتنی تبدیل نشود. شما باید بتوانید خیلی از برنامه های از پیش نوشته شده را بخوانید و بفهمید. معنا و مفهموم اسامی انتخاب شده برای کلاس ها و متدها از دید زندگی روزمره به فهم بیشتر شما از آن برنامه کمک می کند. همه‌ی اینها با دانستن زبان انگلیسی میسر است.

پس لطفاً قبل از یاد گرفتن ASP.NET و یا اینکه چگونه باید از ADO.NET استفاده کرد از زبان خود مطمئن شوید. اگر ضعیف هستید اول آن را تقویت کنید و بعداً بیائید.

پیش نیاز

توجه داشته باشید که جهت یادگیری ASP.NET که یک تکنولوژی جهت ساخت برنامه های تحت وب است و نه یک زبان برنامه نویسی، شما به پیش نیازهای زیاد و متنوعی نیاز دارید. باز تکرار می کنم که ASP.NET تکنولوژی است نه زبان برنامه نویسی. مفاهیم و مسائل زیادی جهت به کارگیری این تکنولوژی مورد نیاز است .بدون تسلط و نه صرفاً اطلاع از آنها شما نمی توانید ASP.NET را یاد بگیرید. مواردی که به ذهنم می رسد را در زیر فهرست وار به اطلاع شما رسانده ام. اگرچه ممکن است لیست کاملی نباشد ولی حداقل مهم ترها را دارد.

-شما باید یک برنامه نویس کامپیوتر باشید.

-شما باید بر مفاهیم شئ گرایی مسلط باشید.

-شما باید یکی از زبانهای خانواده C++ یا Visual Basic را بدانید.

-شما باید بانک اطلاعاتی و زبان SQL را بدانید.

-شما باید مفاهیم وب و برنامه نویسی وب را بدانید.

-شما باید کار با محیط های Visual را بدانید.

-شما باید IIS را بشناسید و فرق وب سایت و Virtual Directory و تنظیمات آن را بدانید.

-شما باید یک ویندوز-سرور کار خوب باشید.

-کمی هم شناخت و ذوق هنری در طراحی صفحات و گرافیک وب داشته باشید.

- ...

-شما باید عاشق مایکروسافت باشید!

بهترین راه یادگیری ASP.NET

حالا می خواهم به اصل سوال پاسخ دهم. شما دارید از یک فوت کوزه گری نه چندان مخفی مطلع می شوید. راه های مختلف را گفتم اما حالا می خواهم بهترین آن را بگویم. بهترین راه یادگیری ASP.NET چیست؟

پاسخ: کار

شما از موارد فوق تنها می توانید مقدمات را یاد بگیرید یا اینکه گوش و چشم تان را با واژه هایی آشنا کنید. برای ملکه شدن دانسته هایتان و حرفه‌ای بودن تان باید کار کنید. یعنی اینکه باید برنامه بنویسید. باید درگیر کار یا پروژه ای شوید. باید زور بزنید و درگیر باشید. باید نیازمند باشید. اگر از سر سیری یا صرفاً برای پُز دادن دنبال چیزی هستید مطمئن باشید چیزی یاد نمی گیرید مگر به اندازه همان پُز دادن.

شما نیاز دارید که همه چیز را حتی مقدمات را هم عملاً و شخصاً تجربه کنید. این می توانید از طریق تعریف پروژه های من درآوردی شخصی باشد یا یک کار تجاری واقعی. شما باید در حین کار شلاق بخورید تا فولاد آبدیده شوید! هیچ چیز بجز یک کار جدی و واقعی نمی تواند به شما چیز یاد بدهد. این چیزی است که خود من به شخصه تجربه کرده ام. در این چند سال کتاب و سایتهای فراوانی را خواندم و مقالات زیادی را نوشتم اما هیچ یک به اندازه‌ی

یک پروژه جدی سنگین به من آموزش نداد. بعبارت دیگر همه آن خواندنها لازم هستند ولی کافی نیستند. دوباره دقت کنید: لازم هستند. بیهوده نیستند. اما کافی هم نیستند.

جهت یادگرفتن ASP.NET شما باید پرحوصله، وقت دار، سمج، فعال، علاقه مند و پیگیر باشید.

بررسی خطاها در ASP.NET - قسمت اول[1]

هنگامی که در یک صفحه ASP.NET خطایی رخ می دهد، ASP.NETاطلاعاتی راجع به خطا به Clientمی فرستد. در این مقاله به نحوه راه اندازی و اعلام خطاها به کاربر،نمایش یا عدم نمایش خطاها برای کاربران، سفارشی کردن صفحات خطا و نوشتن خطاها در Event Log می پردازیم .

- سفارشی کردن (Customize)صفحات خطا
- بررسی خطاها بوسیله برنامه نویسی
- نوشتن در گزارش رویداد(Event Log)
- خلاصه بخش

هنگامی که در یک صفحه خطایی رخ می دهد، ASP.NETاطلاعاتی راجع به خطا به Client می فرستد.

خطاها به چهار دسته تقسیم می شوند:

- خطاهای وضعیتی :(Configuration errors)هنگامی رخ می دهد که ترکیب و یا ساختار فایل Web.config در سلسله مراتب ترکیبی (Configuration)نادرست باشد.
- خطاهای مفسر :(Parser errors)هنگامی رخ می دهد که اصطلاحا Syntax یا ترکیب ASP.NET در یک صفحه ناقص باشد.
- خطاهای زمان کامپایل :(Compilation errors)هنگامی رخ می دهد که جملات (Statements)در یک صفحه در زبان (Language)تعیین شده، نادرست باشد.

[1] نسرین قاسمپور ملکی

•خطاهای زمان اجرا :(Run-time errors)هنگام اجرای یک صفحه رخ می دهد، و این خطاها در زمان کامپایل قابل شناسایی نیستند.

در حالت پیش فرض، اطلاعاتی که جهت خطای زمان اجرا نمایش داده می شود بصورت فراخوانی از پشته هستند (زنجیره ای از فراخوانی های رویداد که وقفه را راه اندازی می کند). هنگامی که مد اشکالزدایی فعال باشد، ASP.NETشماره خطی از کد که خطای زمان اجرا در آن واقع شده نشان می دهد. مد اشکالزدایی یک ابزار مفید جهت اشکالزدایی برنامه است. مد اشکالزدایی در سطح صفحه توسط کد زیر فعال می شود:

```
<%@ Page Debug="true" %>
```

مد اشکالزدایی در سطح برنامه نیز با استفاده از فایل Web.config در شاخه اصلی بصورتی که در مثال نشان داده شده فعال می شود.

```
<configuration>
  <system.web>
    <compilation debug="true"/>
  </system.web>
</configuration>
```

نکته:اجرا در مد اشکالزدایی موجب کاهش محسوس کارایی برنامه می شود. قبل از انتشار نسخه پایانی برنامه خود حتما این مد را غیرفعال کنید.

سفارشی کردن صفحات خطا

بسته به شرایط محیط نیاز به بررسی خطاهای برنامه به روشهای مختلفی می باشد.برای مثال، زمان برنامه نویسی (Development)احتمالا نیاز به مشاهده جزئیات صفحات خطا که ASP.NET جهت راهنمایی شناسایی و

تصحیح مشکلات فراهم کرده، می باشد. اما اگر برنامه در مرحله استفاده توسط کاربر باشد احتمالا نیاز به نمایش جزئیات خطا برای کاربر نیست که می توان از قابلیت عدم نمایش خطا برایclient های محلی (Local)در ASP.NETاستفاده کردclient) های موجود در همان کامپیوتر سرور). برای راهنماییclient ها می توان هنگام بروز خطا آنها را به صفحه خطایی هدایت کرد.

مثال سفارشی کردن خطاها در فایل Web.config برای برنامه ها:

```
<configuration>
  <system.web>
    <customErrors defaultRedirect="genericerror.htm" mode="RemoteOnly" />
  </system.web>
</configuration>
```

توسط کد زیرclient های محلی می توانند صفحه پیش فرض جزئیات خطا ASP.NET را ببینند اما clientهای دیگر (Remote)به صفحه سفارشی genericerror.htm هدایت می شوند. این صفحه میتواند یک صفحه aspx. نیز باشد ASP.NET.مسیر صفحه ای که خطا در آن رخ داده را به شکل یک آرگومان QueryString به صفحه خطا می فرستد. توجه شود که اگر اجرای صفحه خطا هم خطایی تولید کند یک صفحه خالی به client غیر محلی (Remote)ارسال می شود.

```
<%@ Page Language="C#" Description="Error page"%>

<html>
<head>
  <title>Error page</title>
</head>
```

```
<body>
   <h1>Error page</h1>
   Error originated on: <%=Request.QueryString["ErrorPage"] %>
</body>
</html>
```

نکته :تنها فایلهایی که در aspnet_isapi.dll در IIS نگاشته شده اند، اینگونه خطاها را ایجاد می کنند. فایلهایی که در aspnet_isapi.dll بکار نرفته اند، توسط ASP.NET انجام نمی شوند و خطای IIS ایجاد می کنند. برای اطلاعاتی راجع به وضعیت (Configuration)خطاهای سفارشی IIS به مستندات IIS مراجعه کنید.

در زیر وضعیتattribute ها و مقادیر برچسب <customErrors> را تشریح می کند.

Attribute	توضیح
mode	نشاندهنده فعال یا غیر فعال بودن خطاهای سفارشی و یا نمایش و عدم نمایش آن برای کامپیوترهای راه دور (Remote)می باشد . مقادیر: **On، Off، RemoteOnly** (پیش فرض).
defaultRedirect	نشاندهنده آدرس پیش فرضی که جستجوگر باید هنگام بروز خطا به آن رجوع کند، می باشد.این خاصیت اختیاری است .

خاصیت Mode مشخص می کند که آیا خطاها برایclient های محلی و یاclient های راه دور و یا هر دو نمایش داده شوند. نتایج هر کدام از تنظیمات در زیر شرح داده شده است.

Mode	Local host request	Remote host request
On	صفحه خطای سفارشی.	صفحه خطای سفارشی.
Off	صفحه خطای ASP.NET.	صفحه خطای ASP.NET.
RemoteOnly	صفحه خطای ASP.NET.	صفحه خطای سفارشی.

مثال زیر نشان می دهد چگونه ترکیب <customErrors> استفاده می شود.

```
[Web.config]
<configuration>
   <system.web>
      <customErrors defaultRedirect="genericerror.htm" mode="On"/>
   </system.web>
</configuration>


[Genericeroor.htm'
<html>
<head>
   <title>An Error Has Occured</title>
</head>
<body bgcolor="beige">
```

```
   <font face="verdana">
      <h4>We're Sorry...</h4>
       An error has occured on the page you were requesting. If this problem persists,
please contact the site administrator.
      <p>
      <hr>
   </font>
</body>
</html>
```

[Custom1.aspx]

```
<html>

<script language="C#" runat="server">

void Error_500(Object sender, EventArgs e) {
   String foo = null;
   Response.Write(foo.ToString());
}

</script>

<body>
<form runat="server">
   <h4><font face="verdana">Cause an Error to Occur...</font></h4>
   <asp:button text="500 Server Error" OnClick="Error_500" width="150"
```

```
runat="server"/><p>
</form>
</body>

</html>
```

علاوه بر هدایت به سمت صفحه عمومی برای همه خطاها، می توان برای کد خاصی از خطاها، صفحات خاصی را مشخص نمود. ترکیب <customErrors> از برچسب داخلی <error>که کدهای HTTP را به صفحات خطای سفارشی مرتبط می کند، پشتیبانی می کند. برای مثال:

```
<configuration>
  <system.web>
    <customErrors mode="RemoteOnly" defaultRedirect="/genericerror.htm">
      <error statusCode="500" redirect="/error/callsupport.htm"/>
      <error statusCode="404" redirect="/error/notfound.aspx"/>
      <error statusCode="403" redirect="/error/noaccess.aspx"/>
    </customErrors>
  </system.web>
</configuration>
```

جدول زیرattribute ها و مقادیر برچسب <error>را شرح می دهد.

Attribute	توضیح
StatusCode	کد خطایHTTP ، که صفحه خطای سفارشی باید آن را استفاده کند.مثال: *403، 404 Not Found، Forbidden یا ۵۰۰ Internal Server Error.*

Redirectآدرسی که جستجوگر client در صورت بروز خطا باید به سمت آن هدایت شود.

مثال زیر نشان می دهد که چگونه از برچسب <error>استفاده کنیم. توجه شود که مثال یک صفحه aspx را برای خطای"File Not Found" مشخص کرده است و آدرس صفحه مفقود شده ازطریق QueryString ارسال می شود.

```
[Custom2.aspx]
<html>

<script language="C#" runat="server">

void Error_404(Object sender, EventArgs e) {
   Response.Redirect("nowhere.aspx");
}

void Error_500(Object sender, EventArgs e) {
   String foo = null;
   Response.Write(foo.ToString());
}

</script>

<body>
   <form runat="server">
      <h4><font face="verdana">Cause an Error to Occur...</font></h4>
      <asp:button text="404 Not Found" OnClick="Error_404" width="150"
```

```
runat="server"/><p>
      <asp:button text="500 Server Error" OnClick="Error_500" width="150"
runat="server"/><p>
   </form>
</body>

</html>
```

[Notfound.aspx]

```
<%@ Page Language="C#" %>
<html>
<head>
   <title>An Error Has Occured</title>
</head>
<body bgcolor="ccffcc">
   <font face="verdana">
      <h4>We could not locate the page you requested...</h4>
      <% if ( Request.QueryString["aspxerrorpath"] != null ) { %>
      <%=HttpUtility.HtmlEncode(Request.QueryString["aspxerrorpath"])%>
      <% } %>
      <p>
      Perhaps you mis-typed the URL? Please try again, or visit our search page for help.
      <p>
      <hr>
   </font>
```

```
</body>
</html>
```

[Web.config]

```
<configuration>
   <system.web>
      <customErrors defaultRedirect="genericerror.htm" mode="On">
         <error statusCode="404" redirect="notfound.aspx"/>
      </customErrors>
   </system.web>
</configuration>
```

بررسی خطاها در ASP.NET - قسمت دوم[1]

هنگامی که در یک صفحه ASP.NET خطایی رخ می دهد، ASP.NETاطلاعاتی راجع به خطا به Clientمی فرستد. در این مقاله به بررسی سفارشی کردن صفحات خطا، بررسی خطاها بوسیله برنامه نویسی و نوشتن خطاها در Event Log می پردازیم .

بررسی خطاها بوسیله برنامه نویسی

خطاها را میتوان در هر دو سطح صفحه و برنامه توسط کد راه اندازی کرد. کلاس پایه Page دارای متد Page_Errorاست، که میتوان از خاصیت override آن در صفحات خود بهره برد. این متد هر زمان که اتفاق بی دلیلی در زمان اجرا رخ دهد، فراخوانی می شود.

```
<script language="C#" runat="server">

void Page_Error(Object source, EventArgs e) {
   String message = "<font face=verdana color=red>"
      + "<h4>" + Request.Url.ToString() + "</h4>"
      + "<pre><font color='red'>"
      + Server.GetLastError().ToString() + "</pre>"
      + "</font>";

   Response.Write(message);
```

[1] نسرین قاسمپور ملکی

```
}

</script>
```

مثال زیر متد Page_Errorرا نشان میدهد.

```
[error2.aspx]
<html>

<script language="C#" runat="server">

void Error_500(Object sender, EventArgs e) {
   String foo = null;
   Response.Write(foo.ToString());
}

void Page_Error(Object sender, EventArgs e) {
   String message = "<font face=verdana color=red>"
      + "<h4>" + Request.Url.ToString() + "</h4>"
      + "<pre><font color='red'>" + Server.GetLastError().ToString() + "</pre>"
      + "</font>";

   Response.Write(message);
   Server.ClearError();
}

</script>
```

```
<body>
<form runat="server">
   <h4><font face="verdana">Cause an Error to Occur...</font></h4>
   <asp:button text="500 Server Error" OnClick="Error_500" width="150"
runat="server"/><p>
</form>
</body>

</html>
```

می توان برخی اقدامات مفید ازقبیل ارسال یک ایمیل به مسئول سایت برای گفتن اینکه صفحه بدرستی اجرا نشده، انجام داد ASP.NET.مجموعه کلاسهایی در فضانام System.Web.Mail دقیقا به همین منظور داراست. برای فراخوانی این فضانام از دستور @Import در بالای صفحه خود به شکل زیر استفاده می کنیم:

```
<%@ Import Namespace="System.Web.Mail" %>
```

میتوان از اشیاء MailMessage و SmtpMail جهت ارسال برنامه وار ایمیل استفاده نمود.

```
MailMessage mail = new MailMessage();
mail.From = "automated@www.contoso.com";
mail.To = "administrator@www.contoso.com";
mail.Subject = "Site Error";
mail.Body = message;
mail.BodyFormat = MailFormat.Html;
SmtpMail.Send(mail);
```

مثال زیر نشان میدهد که چگونه ایمیلی در پاسخ به خطای صفحه ارسال می شود.

نکته :این مثال در حقیقت تا زمانی که سرویس Mail SMTP بر روی سیستم فعال نشده باشد، ایمیلی ارسال نمی کند. برای اطلاعات بیشتر در رابطه با سرویسMail SMTP ، در مستندات IIS جستجو شود.

```
[error3.aspx]
<%@ Import Namespace="System.Web.Mail" %>

<html>

  <script language="VB" runat="server">

  Sub Error_500(sender As Object, e As EventArgs)
    Dim foo As String = Nothing
    Response.Write(foo.ToString())
  End Sub

  Sub Page_Error(Sender As Object, E As EventArgs)
    Dim message As String = "<font face=verdana color=red>" _
      & "<h4>" & Request.Url.ToString() & "</h4>" _
      & "<pre><font color=red>" & Server.GetLastError().ToString() & "</pre>" _
      & "</font>"

    Response.Write(message)
    Response.Write("An error has occured on this server, and the administrator of the
site has been notified.")
```

```
        Dim mail As New MailMessage
"automated@www.contoso.com        mail.From = "
"administrator@www.contoso.com        mail.To = "
        mail.Subject = "Site Error"
        mail.Body = message
        mail.BodyFormat = MailFormat.Html
        SmtpMail.Send(mail)

        Server.ClearError()
    End Sub

    </script>

<body>
    <form runat="server">
        <h4><font face="verdana">Cause an Error to Occur...</font></h4>
        <asp:button text="500 Server Error" OnClick="Error_500" width="150"
runat="server"/><p>
    </form>
</body>

</html>
```

علاوه بر راه اندازی خطاها در سطح صفحه، ممکن است نیاز به بررسی خطاها در سطح برنامه باشد. به همین منظور، از رویداد Application_Error در Global.asax استفاده می شود. این رویداد برای هر پیشامد غیرمنتظره ای در برنامه اجرا می شود.

```
void Application_Error(Object sender, EventArgs e) {
//...Do something here
}
```

نوشتن در گزارش رویداد(Event Log)

فضانام System.Diagnostics کلاسهایی را جهت نوشتن گزارش رویداد ویندوز مهیا کرده است. برای استفاده از این فضانام در صفحات، در ابتدا باید فضانام به شکل زیر فراخوانی شود:

```
<%@ Import Namespace="System.Diagnostics"%>
```

کلاس EventLog گزارش را در خود ذخیره می کند. همچنین متدهای ایستایی برای بازیابی و یا ایجاد گزارش فراهم ساخته و می تواند جهت نوشتن گزارش ارسالی از طریق کد، معرفی شود. مثال زیر مراحل را طی متد Application_Errorاز Global.asax نشان می دهد. هرگاه وقفه غیرمنتظره ای در برنامه اتفاق بیافتد، یک ورودی شامل پیغام خطا و ردیابی پشته در گزارش برنامه ایجاد می شود.

```
void Application_Error(Object sender, EventArgs e) {

  String Message = "\n\nURL:\n http://localhost/" + Request.Path
    + "\n\nMESSAGE:\n " + Server.GetLastError().Message
    + "\n\nSTACK TRACE:\n" + Server.GetLastError().StackTrace;

  // Create event Log if it does not exist

  String LogName = "Application";
  if (!EventLog.SourceExists(LogName)) {
```

```
    EventLog.CreateEventSource(LogName, LogName);
  }

  // Insert into event log
  EventLog Log = new EventLog();
  Log.Source = LogName;
  Log.WriteEntry(Message, EventLogEntryType.Error);

}
```

کد کاملی برای مثال بالا در زیر موجود است. توجه شود که این کد به جهت جلوگیری از ورود به گزارش رویداد ویندوز سیستم شما غیرفعال شده و بنابراین نمی تواند اجرا شود. اگر مایل به مشاهده اجرای این کد باشید یک ریشه مجازی IIS که دربرگیرنده شاخه شامل این فایل است، ایجاد کنید.

```
[Global.asax]
<%@ Import Namespace="System.Diagnostics" %>

<script language="C#" runat="server">

void Application_Error(Object sender, EventArgs e) {

  String Message = "\n\nURL:\n http://localhost/" + Request.Path
    + "\n\nMESSAGE:\n " + Server.GetLastError().Message
    + "\n\nSTACK TRACE:\n" + Server.GetLastError().StackTrace;

  // Create Event Log if it does not exist
```

```
    String LogName = "Application";
    if (!EventLog.SourceExists(LogName)) {
        EventLog.CreateEventSource(LogName, LogName);
    }

    // Insert into Event Log
    EventLog Log = new EventLog();
    Log.Source = LogName;
    Log.WriteEntry(Message, EventLogEntryType.Error);

}

</script>
```

خلاصه

۱-خطاها به چهار رسته تقسیم می شوند:خطاهای وضعیتی،خطاهای پارسر،خطاهای زمان کامپایل، و خطاهای زمان اجرا.

۲-بطور پیش فرض، اطلاعاتی که برای خطای زمان اجرا نمایش داده می شوند،بصورت فراخوانی از پشته می باشد(زنجیره ای از فراخوانی های رویداد که وقفه را راه اندازی می کند). هنگامی که مد اشکالزدایی فعال باشد، ASP.NETشماره خطی از کد که خطای زمان اجرا در آن واقع شده نشان می دهد .

۳-قابلیت تعیین نمایش یا عدم نمایش خطاها بهclient های محلی، clientهای راه دور و یا هر دو در ASP.NETوجود دارد. بطور پیش فرض، خطاها فقط برایclient های محلی(client های موجود در همان

کامپیوتر سرور) نمایش داده می شوند. همچنین می توان برای خطاهایی که بروز می کند ،کاربر را به صفحه خطای سفارشی هدایت کرد.

۴-علاوه بر هدایت به سمت صفحه عمومی برای همه خطاها، می توان برای کد خاصی از خطاها، صفحات خاصی را مشخص نمود. ترکیب <customErrors> از برچسب داخلی <error>که کدهای HTTP را به صفحات خطای سفارشی مرتبط می کند،پشتیبانی می کند.

۵-خطاها را میتوان در هر دو سطح صفحه و برنامه توسط کد راه اندازی کرد.کلاس پایهPage دارای متد Page_Errorاست، که میتوان از خاصیت override آن در صفحات خود بهره برد.این متد هر زمان که اتفاق بی دلیلی در زمان اجرا رخ دهد،فراخوانی می شود.

۶-فضانام System.Web.Mail کلاسهایی را جهت ارسال ایمیل از طریق برنامه در اختیار قرار می دهد که می تواند ابزار مفیدی جهت اعلام بروز خطا به مدیر برنامه باشد.

۷-علاوه بر راه اندازی خطاها در سطح صفحه ، از رویدادApplication_Error در Global.asaxبرای بررسی خطاها در سطح برنامه استفاده می شود.این رویداد برای هر پیشامد غیرمنتظره ای در برنامه اجرا می شود .

۸-فضانامSystem.Diagnostics کلاسهایی را جهت نوشتن در گزارش رویداد ویندوز مهیا کرده است.

به رمز درآوردن اطلاعات[1]

تا به حال به این فکر افتاده اید که یک سری اطلاعات را به صورت رمز در آورید و از دید کاربر پنهان کنید؟ برای مثال یک QueryString یا اطلاعات ذخیره شده داخل یک کوکی. گاهی هم بد نیست بعضی از اطلاعات داخل بانک اطلاعاتی خود را به صورت رمز ذخیره کنید تا در صورت سرقت آن، حداقل برخی اطلاعات حساس شما تا حدی محفوظ بماند. در این مقاله سعی میکنیم یک روش استاندارد برای به رمز در آوردن رشته ها را با هم مرور کنیم .

برای اینکه همراه ما پیش بروید در VS.net یک پروژه جدید C# Application به صورت ASP.net Web Applicationایجاد کنید WebForm1.aspx .به صورت پیش فرض در مقابل شما قرار میگیرد، از Toolbox یک Label داخل WebForm1 درگ کنید، به صورت پیش فرض نام آن Label1 خواهد بود.

معرفیBase64String

Base64String نوعی از String است (و با خصوصیات یک string معمولی) که کاربر نمیتواند آن را تشخیص دهد(اگر کسی را میشناسید که میتواند به من هم بگویید!) برای مثال جمله"I LOVE ASP.net" ، در Base64Stringمیشود "SSBMT1ZFIEFTUC5uZXQ=" . اما چطور؟

```
string String = "I LOVE ASP.net";
byte [] Byte = System.Text.Encoding.ASCII.GetBytes(String);
Label1.Text = Convert.ToBase64String(Byte);
```

[1] حسین روزنامه‌چی

در خط اول ما یک string معمولی را ساختیم در خط بعدی آن را به byte تبدیل کردیم و Byte نامیدیم، در خط سوم Byte را به صورت Base64String در آوردیم و توسط Label1 به نمایش در آوردیم. فکر خوبی است که عملیات رمز سازی را همین جا تمام کنیم چون Base64String برای کاربر معمولی قابل مشاهده نیست! اما اگر چند خط بالا را با این خط ها عوض کنیم چه میشود؟

```
string MyBase64String = "SSBMT1ZFIEFTUC5uZXQ=";
byte[] MyByte =Convert.FromBase64String(MyBase64String);
Label1.Text = System.Text.Encoding.ASCII.GetString(MyByte);
```

حدس بزنید نتیجه چه میشود؟! بله !"I LOVE ASP.net" در حقیقت ما همان راهی را که رفته بودیم برگشتیم! ابتدا Base64String را به byte تبدیل کردیم و سپس byte را به string معمولی. پس اگر برای ما مهم باشد که اطلاعات رمز شده ما برای کسی قابل بازگشت نباشد هنوز کارمان تمام نشده است. حالا که یاد گرفته اید با Base64Stringکار کنید بگویید:

"VHJ5IHRvIGZpbmQgd2hhdCB5b3UgbmV2ZXIgbG9vc2U= "چیست ؟! من جوابش را پیدا کردم و به صورت رمز در آوردم " jcUKVu8D4dlpy4BHw6bgefaVMWW9x0qV" جواب است ، امتحان کنید آن را به روش قبلی به string تبدیل کنید و ببینند آیا به جواب منطقی می رسید؟ برای رسیدن به جواب اصلی باید بتوانید آن را رمز گشایی کنید!

Cryptographyبا استفاده از TripleDES وMD5

در بالا با Base64String آشنا شدیم حالا وقت آن است که با یکی از روشهای معمول رمز سازی در دات نت آشنا شویم این روش فقط یکی از روشهای معمول رمزسازی در دات نت است برای آشنایی با روشهای دیگر از MSDNراهنمایی بگیرید . بیایید ابتدا با عبارتهای تازه‌ی این مبحث آشنا شویم:

TripleDES:برای رمز سازی سه بار از الگوریتم DES استفاده میکند.

DES:مخفف Data Encryption Standard.

MD5:یک الگوریتم کار آمد رمز سازی است که در بسیاری از زبانهای برنامه نویسی دیگر نیز و جود دارد . MD5عضوی با نام ComputeHash دارد که یک تابع Hash یا (Hash function) است.

توابع Hash یا (Hash functions) یک binary string با طول دلخواه را به یک binary string با طول کوتاه و ثابت! تبدیل میکند، و این خاصیت را دارد که هرگز برای دو ورودی مجزا یک خروجی یک یکسان وجود نخواهد داشت (به عبارت دیگر توابع Hash تضمین میکنند که هر ورودی خروجی منحصر به فردی را داشته باشد). خوب برای به رمز در آورن کافی است مانند مثال زیر عمل کنیم (فعلا آن را اجرا نکنید!).

```
byte[] IV = new byte[8] {240, 32, 45, 29, 0, 76, 173, 59};
string cryptoKey = "All you need is Love";
string MyString = "I LOVE ASP.net";
byte[] buffer = System.Text.Encoding.ASCII.GetBytes(MyString);
TripleDESCryptoServiceProvider des = new TripleDESCryptoServiceProvider();
MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider();
des.Key =
MD5.ComputeHash(System.Text.ASCIIEncoding.ASCII.GetBytes(cryptoKey));
des.IV = IV;
byte[] CodedBuffer =
des.CreateEncryptor().TransformFinalBlock(buffer,0,buffer.Length);
Label1.Text = System.Convert.ToBase64String(CodedBuffer,0,CodedBuffer.Length);
```

ابتدا بگذارید ببینیم IV و cryptoKey چیستند؟

DESCryptoServiceProvider(و به دنبال آن TripleDESCryptoServiceProvider) دارای یک Keyو یک IV مخفف Initialization Vector هستند که برای رمز کردن داده ها از آن استفاده میشود (همان طور که لازم نیست بدانیم کی برد و موس چگونه دستورات ما را به سخت افزار کامپیوتر منتقل میکنند تا از آنها استفاده کنیم، لازم نیست به دنبال این باشیم که Key و IV در DESCryptoServiceProvider دقیقا چگونه کار میکنند فقط کافی است بدانیم از آنها چطور استفاده کنیم). باید توجه داشته باشیم اگر فراموش کنیم IV را مقدار دهی کنیم در هنگام اجرا IV یک مقدار تصادفی خواهد گرفت که برگردان داده های رمز شده را غیر ممکن میکند!

در مثال بالا IV را از جنس byte تعریف کردیم و cryptoKey از جنس string به صورت "All you need is Love"مقدار دهی کردیم که در حقیقت "All you need is Love" کلید رمز ما خواهد بود MyString . همانstring ای است که باید به صورت رمز در بیاید. (اگر کمی از C# سر در بیاورید با چند خط بعدی مشکلی نخواهید داشت).

des.Key:حالا وقت آن شده است که Key را مقدار دهی کنیم، اما باید آن را توسط MD5.ComputeHashبه صورتی مناسب برای Key بودن تبدیل کنیم و فراموش نکنیم که MD5.ComputeHash، از string به عنوان ورودی نمیتواند استفاده کند و باید قبل از آن cryptoKey خود را به حالت byte تبدیل کرده باشیم.(System.Text.ASCIIEncoding.ASCII.GetBytes(cryptoKey))

و حالا وقت خوبی است تا CreateEncryptor برای رمز سازی اقدام کند و TransformFinalBlock مقدار Hash شده‌ای برای منطقه مشخص شده ای از آرایه های byte مشخص شده حساب میکند. اما توجه داشته باشید که ما byte رمز شده لازم نداریم بلکه string رمز شده میخواهیم پس در قدم آخر باید یک Base64String تولید کنیم.

سوالی که پیش می‌آید این است که TripleDESCryptoServiceProvider و MD5CryptoServiceProviderکجا هستند؟ فضای نام System.Security.Cryptography پاسخ این سوال است پس حالا میتوانید پروژه‌ای را که ایجاد کرده بودیم بیاورید و به جای کد های قبلی کدهای مثال بالا را در آن قرار دهید و فراموش نکنید که namespace جدید را به صورت using System.Security.Cryptography به بالای کدهای خود اضافه کنید .

این روش بر خلاف بعضی از روشهای یک طرفه، قابلیت بازیابی اطلاعات رمز شده را دارا میباشد، فقط کافی است از CreateDecryptorاستفاده کنیم:

```
byte[] IV = new byte[8] {240, 32, 45, 29, 0, 76, 173, 59};
string cryptoKey = "All you need is Love";
string CodedString64 = "zNvKahC6S/+8xMX3o658wQ==";
byte[] buffer = Convert.FromBase64String(CodedString64);
TripleDESCryptoServiceProvider des = new TripleDESCryptoServiceProvider();
MD5CryptoServiceProvider MD5 = new MD5CryptoServiceProvider();
des.Key =
MD5.ComputeHash(System.Text.ASCIIEncoding.ASCII.GetBytes(cryptoKey));
des.IV = IV;
byte[] CodedBuffer =
des.CreateDecryptor().TransformFinalBlock(buffer,0,buffer.Length);
Label1.Text = System.Text.Encoding.ASCII.GetString (CodedBuffer);
```

خوب حالا میتوانید به من بگویید " jcUKVu8D4dlpy4BHw6bgefaVMWW9x0qV" (که همان جواب معمای ما هم بود) چیست ؟ صد در صد خیر! مگر اینکه من به شما IV و Key آن را بگویم IV !مانند مثال و Keyنیز "red rose" است! حالا اگر مطالب را دنبال کرده باشد احتمالا به راحتی میتوانید رمزگشایی کنید !

چند نکته امنیتی

آیا این روش، روش قابل اطمینانی است؟ باید بگویم تا وقتی نخواهید اطلاعات چند صد هزار دلاری یا اسناد محرمانه ای را به رمز تبدیل کنید احتمالا میتوانید روی آن حساب کنید مخصوصا که TripleDES سه بار با الگوریتم DESآن را به رمز تبدیل کرده است (سه بار رمز شکستن به راحتی سه بار رمز کردن نیست!). شما میتوانید برای سخت تر کردن کار رمزشکن از Key های مختلفی استفاده کنید (البته سعی کنید برای رمز کردن کلمه عبور، نام کاربری یا email را به عنوان کلید رمز قرار ندهید! چون شما جزو اولین ۱۰۰۰۰ نفری نیستید که این کار را کرده اند!). حفاظت اطلاعات شما در گروی امن بودن server ای است که سایت شما روی آن قرار دارد، همینطور از قرار دادن Key در Web.config به عنوان یک کار خلاقانه! جدا خودداری کنید .

جستجوی نام های دامنه (Whois) در ASP.NET[1]

اگر به سایتهای ثبت دامنه مانند www.Register.com سر زده باشید، دیده اید که این سایتها امکاناتی برای جستجوی نام دامنه در اختیار شما می گذارند که به شما وجود یا عدم وجود آن نام را اطلاع می دهند و اطلاعاتی نیز درباره فرد یا سازمان صاحب نام می دهد. در این مقاله نحوه ی انجام این کار را با ASP.NETشرح می دهیم .

برای انجام عمل جستجو ابتدا باید یک ارتباط با سرورهایی که بانکهای اطلاعاتی آنها این مشخصات را دارند ارتباط برقرار کرد و با یکQuery ، اطلاعات مورد نظر را درخواست نمود. این ارتباط از نوع TCP و از طریق پورت ۴۳ انجام می گیرد. لیست کاملی از این سرورها را اینجا[2] می توانید ببینید.

مشخص کردن فضا نامها

ابتدا فضا نامهای مورد نیاز را مشخص می کنیم:

```
Using System. NET.Sockets;
Using System.Text;
Using System.IO
Using System.Text.RegularExpressions;
```

با استفاده از TcpClient در .NetFramework می توانیم این ارتباط را با سرور مورد نظر برقرار نماییم. برای استفاده از این کلاس باید فضانام System. NET.Sockets را وارد کنیم. از دو فضانام دیگر نیز برای فرمت ورودی و خروجی خود استفاده می کنیم .

[1] احسان صادقی

[2] www.mit.edu/afs/athena/contrib/potluck/net-services/whois-servers.list

ایجاد و ارسال Query

ابتدا متغیرهای مورد نیاز را تعریف می کنیم:

```
string StrSvr,StrDomain, Resp;
TcpClient TcpClt;
byte[] ArrDomain;
Stream TcpStr;
StreamReader TcpStrRdr;
```

حال باید رشته ای را که می خواهیم به عنوان Query به سرور ارسال نماییم، ایجاد کنیم. برای این کار باید نام درخواستی کاربر را همراه پسوند مورد نظر (Org،Com، ...)را به سرور ارسال نماییم. توجه داشته باشید که هر کدام از سرورهای موجود در لیست فوق الذکر برای جستجوی پسوند خاصی می باشد. برای مثال whois.internic.com تنها برای دامنه های .NET و .com و .edu می باشد و اگر دامنه ای با پسوند .Info را با این سرور جستجو نمایید، جواب مطلوب را نخواهید گرفت.

بنابراین ابتدا باید پسوند مورد نظر کاربر و سپس سرور مربوطه را مشخص نماییم . در اینجا برای انتخاب پسوند از DropDownList استفاده کرده ام:

```
if (DDLstSuffix.SelectedItem.Value == ".COM" ||
DDLstSuffix.SelectedItem.Value == ".NET" || DDLstSuffix.SelectedItem.Value ==
".EDU")
{
   StrSvr = "whois.internic.net";
   StrDomain = TxtDomainName.Text.Trim() + DDLstSuffix.SelectedItem.Value+
"\r\n";
```

```
}
else if (DDLstSuffix.SelectedItem.Value == ".ORG")
{
   StrSvr = "whois.publicinterestregistry.net";
   StrDomain = TxtDomainName.Text.Trim() + DDLstSuffix.SelectedItem.Value+
"\r\n";
}
else if (DDLstSuffix.SelectedItem.Value == ".BIZ")
{
   StrSvr = "whois.neulevel.biz";
   StrDomain = TxtDomainName.Text.Trim() + DDLstSuffix.SelectedItem.Value+
"\r\n";
}
else if (DDLstSuffix.SelectedItem.Value == ".INFO")
{
   StrSvr = "whois.afilias.info";
   StrDomain = TxtDomainName.Text.Trim() + DDLstSuffix.SelectedItem.Value+
"\r\n";
}
else if ((DDLstSuffix.SelectedItem.Value == ".IR") || (DDLstSuffix.SelectedItem.Value
== ".CO.IR") || (DDLstSuffix.SelectedItem.Value == "..NETIR") ||
(DDLstSuffix.SelectedItem.Value == "ID.IR"))
{
   StrSvr = "whois.nic.ir";
   StrDomain = TxtDomainName.Text.Trim() + DDLstSuffix.SelectedItem.Value+
"\r\n";
}
```

همانطور که می بینید، رشته ای که برای جستجو ارسال می شود به "r\n\" ختم می شود .اگر این رشته را در انتهای نام دامنه قرار ندهید، سرور نمی تواند جواب مطلوب را برگرداند.

اتصال به سرور

حال یک ارتباط باید با سرور مورد نظر برقرار نماییم. کلاسTcpClient ، متدهایی برای اتصال، دریافت و ارسال اطلاعات دارد. به دو صورت می توانیم با یک سرور ارتباط برقرار کنیم:

۱-ایجاد یک نمونه از کلاس TcpClient بدون هیچ پارامتری و سپس استفاده از متدConnect :

```
TcpClient TcpClt = new TcpClient();
TcpClt.Connect(StrSvr,43);
```

۲-ایجاد یک نمونه از کلاس TcpClient با پارامترهای آدرس سرور و پورت مقصد که می خواهید با آن ارتباط برقرار کنید. این Constructor ارتباط را به صورت اتوماتیک ایجاد می کند:

```
TcpClient objTCPC = new TcpClient(StrSvr, 43);
```

پس از آن با متد GetBytes کلاس Encoding (فضا نام System.Text) رشته StrDomain را به یک آرایه از نوع بایت می نویسیم.

```
ArrDomain = Encoding.ASCII.GetBytes(strDomain);
```

با استفاده از متد GetStream کلاس TcpClient ، یک Stream ایجاد می کنیم و از طریق آن به ارسال و دریافت اطلاعات می پردازیم:

```
TcpStr = TcpClt.GetStream();
```

با استفاده از این Stream می توانیم با متد Write یک آرایه از Byte را در Stream جاری نوشت:

```
TcpStr.Write (ArrDomain,0,StrDomain.Length);
```

دریافت و نمایش نتیجه جستجو

یکی از راههای دریافت اطلاعات از سرور، ایجاد یک نمونه از کلاس StreamReader و سپس خواندن اطلاعات بر اساس یک Encoding مشخص می باشد.

```
TcpStrRdr = new StreamReader(TcpClt.GetStream(),Encoding.ASCII);
```

قدم بعد، استفاده از متد ReadToEnd برای دریافت جواب سرور می باشد. این رشته شامل کاراکترهای خط جدید "\n" می باشد که باید با "
" جایگزین شود. برای این کار از متد Replace کلاس RegEx استفاده می کنیم ، سپس برای راحتی جستجو این رشته را به حروف کوچک تبدیل می کنیم:

```
Resp = Regex.Replace(TcpStrRdr.ReadToEnd(),"\n","<br>");
Resp = Resp.ToLower();
```

اگر در این رشته، زیر رشته هایی مانند "no match" یا "not found" یا "no entries found" وجود داشته باشد، نام دامنه مورد نظر آزاد می باشد. در غیر اینصورت آن دامنه ثبت شده است. حال با استفاده از این مساله، نتیجه را به کاربر اعلام می کنیم. برای جستجوی این زیر رشته ها از متد IsMatch کلاس RegEx استفاده می کنیم. اگر در این متد زیر رشته مورد نظر در رشته موجود باشد True و در غیر اینصورت False برگشت داده می شود.

```
    if (Regex.IsMatch(Resp,"no match") || Regex.IsMatch(Resp,"not found") ||
Regex.IsMatch(Resp,"no entries found"))
{
   SearchRes = " دامنه مورد نظر موجود می باشد ";
   PnlOrder.Visible = true;
   PnlOrderOk.Visible = false;
}
else
{
   SearchRes = " دامنه مورد نظر موجود نمی باشد ";
   PnlOrder.Visible = false;
   PnlOrderOk.Visible = false;
}
```

در انتها نیز ارتباط را می بندیم.

```
TcpClt.Close();
```

منبع: ***Build a WHOIS Lookup in ASP.NET***

www.sitepoint.com/article/build-whois-lookup-asp-net

جلوگیری از دزدیده شدن محتویات وب سایت[1]

موضوع این مقاله کلیه فایلهای موجود در سایت از قبیل JPG ، DOC، ...PDFمی‌باشد. اگر از فایلهای خود محافظت نکنید به راحتی هر کسی میتواند به آنها دسترسی پیدا کند چون فقط کافیست از URLآن مطلع شود تا به راحتی بتواند آن را دانلود کند و یا با استفاده از یک Spider کل اطلاعات سایت شما را دریافت کند همانطور که مدتهاست گوگل در حال انجام اینکار است و کلیه تصاویر موجود در هر سایت را ذخیره می کند. خوب راه حل جلوگیری از اینکار چیست؟

هر نوع اطلاعاتی که بتوان فقط با یک URL به آن دسترسی پیدا کرد دیگر مال همه است و همه میتوانند به آن دسترسی داشته باشند و سایتهایی مثل گوگل در ظرف چند دقیقه کل اطلاعات شما را به این صورت دریافت میکنند.

بعضی اوقات شما دوست ندارید همه به اطلاعات شما دسترسی داشته باشند و میخواهید فقط به بعضی افراد اجازه دسترسی به بعضی از فایلها را بدهید خوب به نظر شما راه حل چیست؟ اگر فقط یک سیستم مدیریت کاربر بگذارید و کاربران پس از ورود به آن URL خاص دسترسی پیدا کنند خیلی جالب نخواهد بود زیرا فقط کافیست آن آدرس URL لو رود تا مجددا افراد بدون مجوز بتوانند به آن دسترسی پیدا کنند. پس عملا Authentication ساده ASP.netقادر به امن کردن فایلهای شما نیست و شما باید تدابیر دیگری بیاندیشید.

حتی در مورد این فکر نکنید مثلا اسم فایل را بگذارید ۳۲sdf43ef45.pdf چون این کار ابلهانه‌ای است و در اصل مسئله تفاوتی ایجاد نمی کند. یک Spider و یا یک Hacker خوب میتواند به راحتی به فایلهای شما دسترسی پیدا کند.

[1] امین رودکی

راه حل در استفاده از HttpHandler است.

مروری بر HttpHandler

HttpHandlerشامل یکسری API هایی جهت انجام امور اشیاء Request/Response که با استفاده از آن میتوانید انتقال اطلاعات را کنترل کنید. در اینجا ما باید یک Handler ایجاد کنید که در زمانی که کسی درخواست یک فایل مثلا .doc میدهد اجرا شود.

خوب بهتر است مقداری کد نویسی کنیم تا بهتر موضوع مشخص شود و متوجه شوید چطور میتوان در یک سایت E-Commerce اطلاعات را امن کرد بدون استفاده از تکنیک های FTP و نیاز به مرورگرهای خاص.

تغییر Web.Config

مرحله اول: باید تگ زیر را در web.config موجود در فولدری که میخواهید آن را Secure کنید وارد کنید.

```
<HttpHandlers>
  <add verb="*" path="*.doc" type="pdfIntercept.pdfHandler, pdfInterceptX" />
</HttpHandlers>
```

با اینکار در واقع به IHttpHandler میگوئیم که ما میخواهیم درخواستهایی که آخر آنها doc است را کنترل کنیم .

مشخصه verb میتواند حالتهای POST یا GET یا HEAD داشته باشد که در اینجا ما نوشته ایم * یعنی تمامی حالتهای تقاضا را مورد پوشش قرار می دهد.

مشخصه Path در واقع آدرسی که باید مورد بررسی قرار گیرد که با نوشتن *.doc گفته ایم کلیه آدرس های منتهی به doc

مشخصه Type در واقع کلاس دات نت است که باید تقاضا را Handle کند. باید نام کامل کلاس را به صورت زیر بنویسیم.

[NAMESPACE].[CLASS], [ASSEMBLY NAME]

اضافه کردن یک پسوند دلخواه Custom Extension

باید توجه کرد که باید پسوند مورد نیاز را در IIS هم اضافه کنیم. درضمن سپس تنظیمات دایراکتوری محتوی فایلها را به صورت زیر انجام دهید:

Read : False
Write: False
Directory Browsing: False

حال IIS Manager را اجرا کنید و سپس Properties/Edit ...را انتخاب کنید آنگاه به بخش Home بروید و دکمه Configuration را بزنید. سپس باید پسوند مورد نیاز خود مثلا doc, pdf را در صفحه تنظیمات نرم افزار اضافه کنید. برروی دکمه add بزنید و اطلاعات مورد نیاز را پر کنید کار بسیار ساده ای است.

بعد از اینکه اینکارها را کرده اید حال آماده هستید تا کد مربوط به HttpHandler را بنویسید.

کد HttpHandler

در زیر کد مربوط به HttpHandler ارائه شده است. البته شما خود باید کد مربوط به Authentication مربوط به سایت خود را در آن قرار دهید.

```
using System;
using System.Web;
using System.IO;

namespace pdfIntercept
{

  public class pdfHandler : IHttpHandler
  {

    //Notice ProcessRequest is the only method
    //exposed by the IHttpHandler
    public void ProcessRequest(HttpContext context)
    {
      try
      {
        string strString = "yes";
        HttpRequest oRequest = context.Request;
        HttpResponse oResponse = context.Response;

        //ADD YOUR CUSTOM AUTHENICATION HERE
        //ADD YOUR CUSTOM AUTHENICATION HERE
```

```
//ADD YOUR CUSTOM AUTHENICATION HERE
//ADD YOUR CUSTOM AUTHENICATION HERE

if (strString == "yes")
{

   //Since they've made it this far, they've been validated
   //by your system…
   //We'll fire up a FileStream object
   FileStream MyFileStream;
   long FileSize;

   //Map the path to the .doc file
   //You might need to parse out the Request path to figure out
   //what resource they're actually requesting…
   string strMapPath = context.Server.MapPath("book1.doc");
   MyFileStream = new FileStream(strMapPath, FileMode.Open);
   FileSize = MyFileStream.Length;

   //Allocate size for our buffer array
   byte[] Buffer = new byte[(int)FileSize];
   MyFileStream.Read(Buffer, 0, (int)FileSize);
   MyFileStream.Close();

   //Do buffer cleanup
   context.Response.Buffer = true;
   context.Response.Clear();
```

```
            //Add the appropriate headers
            context.Response.AddHeader("content-disposition",
               "attachement filename=x.doc");

            //Add the right contenttype
            context.Response.ContentType = "application/doc";

            //Stream it out via a Binary Write
            context.Response.BinaryWrite(Buffer);
        }
        else
        {

            //It's a bogus request and they weren't validated.
            context.Response.Write("<b>DENIED</b>");
        }
    }
    catch (System.Exception err)
    {
        err.ToString();
    }
}

//By calling IsReusable, an HTTP factory can query a handler to
//determine whether the same instance can be used to service
//multiple requests
```

```
    public bool IsReusable
    {
      get
      {
        return false;
      }
    }
  }

}
```

اگر میخواهید پیچیدگی مربوط به binary stream را کم کنید میتوانید از متد Response.WriteFile به صورت زیر استفاده کنید.

```
if (strString == "yes")
{
  context.Response.Buffer = true;
  context.Response.Clear();
  context.Response.AddHeader("content-disposition",
    "attachement; filename=x.doc");
  context.Response.ContentType = "application/doc";
  context.Response.WriteFile("pp.doc");
}
else
{
  context.Response.Write("<b>DENIED</b>");
}
```

مراحل کد ارائه شده به صورت زیر است:

۱- HttpHandlerدرخواست مربوط به یک فایل doc را دریافت میکند.

۲-سپس شما یکسری چک کردن با دیتابیس برای اینکه متوجه شوید کاربر مجاز به استفاده است انجام میدهید.

۳-اگر درخواست مجاز بود یک Binary Stream فایل را برای کاربر ارسال میکند.

۴-اگر درخواست غیر مجاز بود برروی صفحه نوشته میشودDenied

HttpHandlerیک ابزار بسیار قوی جهت کنترل دسترسی به محتویات موجود برروی وب سایت شما است.

از این روش میتوان در سایتهای E-Commerce و در جاهایی که لازم دسترسی به یک یا چند فایل محدود شود استفاده نمود. با این روش دیگر هیچ URL در کار نخواهد بود که کسی بتواند با منتشر کردن آن اطلاعات شما را بردارد.

دستکاری تصاویر در ASP.NET[1]

در دات نت همیشه کارهای بزرگ را می توان با چند خط کوتاه، برنامه‌ی ساده انجام داد. یکی از این کارها ترسیم و دستکاری تصاویر است .

لینک فایل: www.iranasp.net/download/hrooznamechi002.zip

در دات نت همیشه کارهای بزرگ را می‌توان با چند خط کوتاه، برنامه‌ی ساده انجام داد. یکی از این کارها ترسیم و دستکاری تصاویر است. در این مقاله سعی می‌کنیم دستکاری تصاویر با دات نت را با هم مرور کنیم. یاد آور می‌شوم که مقاله‌ای در باب ترسیم، با نام گرافیک در ASP.NET توسط دوست خوبمان آقای مجتبی کیانی قبلا در سایت منتشر شده است که مطالعه آن به شما پیشنهاد می‌کنم.

در دات نت ترسیم یک تصویر همیشه با استفاده از شیئی از کلاس Graphics روی می‌دهد. و می‌دانیم متد DrawImageاز این کلاس، متدی غنی برای اجرای کارهای گیج کننده گرافیکی به صورت خیلی ساده است. متد دیگری نیز با نام DrawImageUnscaled وجود دارد که این متد فقط وقتی مورد استفاده قرار می‌گیرد که مستطیل‌های منبع و مقصد همسان باشند.

قبل از هرچیز برای ترسیم باید بدانیم:

۱ -همیشه یک مستطیل مقصد وجود دارد (در حقیقت مساحتی که تصویر داخل آن کپی میشود).

[1] حسین روزناه چی

۲ -همیشه یک مستطیل منبع وجود دارد که ممکن است تمام یا بخشی از تصویر اولیه را در بر بگیرد (در حقیقت مساحتی از تصویر اولیه که می‌خواهیم کپی شود).

۳ -مستطیل مقصد ممکن است کوچکتر یا بزرگتر مستطیل منبع باشد (در این صورت اندازه تصویر ایجاد شده با اندازه تصویر اولیه فرق خواهد کرد).

همینطور دانستن نکات زیر برای ترسیم به ما کمک میکند:

۱ -به هنگام ترسیم تصویر می‌توان با استفاده از ImageAttributes در تصویر تغییراتی داد، در این صورت بسیاری از عملیات پیچیده ترسیم به راحتی ممکن می‌گردد.

۲ -شیئی از کلاس Graphicsمی‌تواند برای تغییر اندازه نیز استفاده شود، همینطور این شیئ دارای فیلترهای الحاقی میباشد که بر کیفیت و سیمای تصویر اثر میگذارند.

در مثال‌های زیر می‌بینیم که چطور متد DrawImage مورد استفاده قرار میگیرد "myGraphics" .شئی از کلاس graphics و "img" تصویری که در حافظه ذخیره شده است فرض شود. شما می‌توانید عملکرد هرقسمت را به ترتیب در مثال نمونه همراه این مقاله مشاهده کنید (توجه داشته باشید که چون این نمونه مثال روی دستگاه شما فایل result.jpg را می‌نویسد، ممکن است هنگام اجرا با ایراد امنیتی روبرو شوید، در این صورت به سند readme.docدر بین فایل‌های مثال رجوع کنید).

۱ -یک کپی مستقیم از تصویر:

```
myGraphics.DrawImage(img,<point>)
```

نقاط X و Y مختصات گوشه بالا سمت چپ مستطیل منبع را بیان می‌کند. در این مثال مستطیل مقصد ما هم اندازه مستطیل منبع ما بود. اما اگر بخواهیم مستطیل مقصد بزرگ تر باشد (و در نتیجه تصویر بزرگ تر از حالت اولیه خود باشد) می‌توانید به صورت زیر عمل کنید توجه داشته باشد که در این حالت ممکن است تصویر کیفیت مطلوب خود را از دست بدهد (همینطور می‌توان مستطیل مقصد را کوچکتر از مستطیل منبع تعریف کرد که واضح است تصویر کوچکتر از تصویر اولیه خواهد بود) .

۲ -کپی با اندازه های متفاوت:

```
myGraphics.DrawImage(img,BiggerRectangle)
```

در این حالت مستطیل منبع کل تصویر اولیه خواهد بود (واضح است که BiggerRectangle مستطیل مقصد می‌باشد) اما اگر بخواهیم فقط قسمتی از تصویر اولیه را استفاده کنیم .مستطیل منبع باید قسمتی از تصویر اولیه باشد .

۳ -انتخاب قسمتی از عکس اولیه:

```
DistRectangle=new Rectangle(100,100,10,10)// مقصد مستطیل
myGraphics.DrawImage(img, DistRectangle,75,40,10,10,GraphicsUnit.Pixel)
```

این مثال نشان می‌دهد که میتوان مستطیل منبع (یا مقصد) را مستقیما در پارامترهای DrawImage تعریف کرد. در ضمن در موردGraphicsUnit ، به یاد داشته باشیم که همیشه واحد گرافیک Pixel است. در مثال بالا مستطیل مقصد نیز به اندازه مستطیل منبع است .اما اگر بخواهیم اندازه قسمت کپی شده را تغییر دهیم باید مستطیل مقصد را با اندازه‌ای متفاوت از مستطیل منبع تعریف کنیم.

۴ -انتخاب قسمتی از عکس اولیه و اعمال تغییر اندازه:

```
DistRectangle =new Rectangle(10,20,120,40) )// مقصد مستطیل
SourRectangle =new Rectangle(75,40,10,10) )// منبع مستطیل
myGraphics.DrawImage(img, DistRectangle, SourRectangle,GraphicsUnit.Pixel)
```

چرخاندن (Rotating) و تغییر صورت (transformation) یک تصویر نیز توسط شیئی از Graphics کلاس مقدور است. برای transformation باید یک ماتریس تعریف کنید. در این مثال، ما تصویر را به اندازه ۳۰ درجه در جهت خلاف عقربه های ساعت می‌چرخانیم.

۵ -چرخاندن با مقیاس درجه:

```
Matrix mx = new Matrix();
mx.Rotate(-30);
myGraphics.Transform = mx;
myGraphics.DrawImage(img,new Point(100,50));
```

یا می‌توانید به شکل زیر تغییر اندازه و چرخاندن را انجام دهید.

۶ -چرخاندن با مقیاس طول و عرض:

```
Matrix mx = new Matrix();
mx.Translate(20.0F, 10.0F);
myGraphics.MultiplyTransform(mx);
myGraphics.RotateTransform(20.0F);
```

در این مثال F ۲۰,۲ و F ۱۰,۰ به ترتیب X و Y ای هستند که به صورت ماترس ترجمه می‌شوند.

روش دیگری برای ایجاد تغییر زاویه وجود دارد و آن استفاده از مختصات نقاط به هنگام تعریف مستطیل‌ها است (تصویر حاصل از این روش حالتی آیینه‌ای نسبت به تصویر اولیه دارد). برای انجام این عمل مستطیل مقصد به صورت معمولی تعریف میشود اما به هنگام تعریف مستطیل منبع به شکل زیر عمل میکنیم.

۷ -ایجاد حالت آینه‌ای:

```
Rectangle DistRectangle = new Rectangle(50,50,320,240);
Rectangle SourRectangle= new Rectangle(0, img.Height, img.Width,- img.Height);
myGraphics.DrawImage(img,DistRectangle,SourRectangle,GraphicsUnit.Pixel);
```

ترسیم تصویرهای شفاف توسط کلاس ImageAttributes انجام می‌شود. چنین شیئی میتواند یک ColorMatrixدر خود ذخیره کند که توسط آن میتوانalpha ی یک تصویر را هنگام ترسیم تغییر داد. از میان تمامproperty ها میتوان به ColorMatrix.Matrix33 اشاره کرد که شما را قادر میکند کاملا شفافیت یک تصویر را تنظیم کنید.

۸ -ترسیم های شفاف:

```
myGraphics.DrawImage(img1,new Point(0,0));
ImageAttributes ia = new ImageAttributes();
ColorMatrix cm = new ColorMatrix();
cm.Matrix33=0.5f;
ia.SetColorMatrix(cm);
myGraphics.DrawImage(img2,new Rectangle(0, 0, img2.Width, img2.Height ), 0, 0,
img2.Width, img2.Height, GraphicsUnit.Pixel, ia);
```

۹ -ایجاد Thumbnail از یک تصویر:

مطلب دیگری که خوب است در باره آن مختصری بحث شود ایجاد Thumbnail از یک تصویر است. اگرچه در این روش از شئی از Graphics استفاده نمی‌شود اما بجاست که استفاده از این روش برای کوچک کردن عکس ها با روش (شماره ۱۰) مورد مقایسه قرار گیرد. تصویرهای کوچکی هستند که معمولا در گالری‌های تصویر Thumbnail ها برای پیش نمایش با اندازه کوچک و به تعداد زیاد در یک صفحه نمایش داده می‌شوند. برای تولید آن از روش زیر استفاده می‌کنیم:

```
System.Drawing.Image.GetThumbnailImageAbort myCallBack = new
System.Drawing.Image.GetThumbnailImageAbort(ThumbnailCallback);
System.Drawing.Image imgResizedImage =
MainPic.GetThumbnailImage(360,270,myCallBack,IntPtr.Zero);
```

پارامترها:

به ترتیب از چپ به راست: (عرض، طول، callbackو callbackData) هستند که callback یک نماینده برای Image.GetThumbnailImageAbort است. در نگارش ۰، GDI+1، نماینده عملا مورد استفاده قرار نمی‌گیرد. اما شما حتماً باید یک نماینده بسازید و آن را به یک مرجع منتصب کنید (برنامه نمونه را نگاه کنید).

callbackDataنیز همیشه IntPtr.Zero است.

اگر تصویر اولیه، در خود Thumbnail به صورت ذخیره شده داشته باشد این متد آن را به اندازه خواسته شده تغییر اندازه می‌دهد و در غیر این صورت این متد با کوچک کردن سایز تصویر اصلی Thumbnail می‌سازد .برای ایجاد تصویرهایی با اندازه کوچک (برای مثال ۱۲۰ در ۱۲۰) این متد بسیار کار آمد است اما در صورتی که بخواهیم تصویر را در اندازه های بزرگ تری کوچک کنیم، احتمال افت کیفیت برای تصویرها وجود خواهد داشت. در این حالت

توصیه می‌شود که تصویر اصلی را با استفاده از DrawImageکوچک کنید. روش زیر برای تغییر اندازه تصویرها در اندازه‌های بزرگتر روشی کارآمدتر است.

۱۰ -تغییر اندازه:

```
System.Drawing.Image NewImage = new System.Drawing.Bitmap(300,300);
System.Drawing.Graphics NewImageGraphics =
System.Drawing.Graphics.FromImage(NewImage);
Rectangle DistRectangle = new Rectangle(0,0,NewImage.Width,NewImage.Height);
Rectangle SourRectangle = new Rectangle(0,0,my1stImage.Width,my1stImage.Height);
NewImageGraphics.DrawImage(my1stImage,DistRectangle,SourRectangle,GraphicsUn
it.Pixel);
```

در این روش یک Image جدید با نام NewImage را از نوع Bitmap و با اندازه های دلخواه تعریف کردیم و در حقیقت یک کپی از تصویر اولیه را با مستطیل منبع هم اندازه تصویر اولیه روی تصویر جدید با اندازه مستطیل مقصد هم اندازه تصویر جدید ترسیم کردیم.

تمام مباحث به ترتیب در برنامه‌ای گرد آورده شده است. این برنامه دو تصویر را از روی دستگاه شما (سرور) می‌خواند و حاصل هر مرحله را در تصویری با نام result.jpg ذخیره میکند و آن را نمایش می‌دهد. برای اجرای این برنامه کافی است IIS و .NET Framework روی دستگاه شما نصب شده باشد.

کوکی ها در ASP.NET[1]

این مقاله توضیح می دهد که چگونه می توان کوکی‌ها را با استفاده از ASP.NET ساخت، خواند و یا حذف کرد .

کوکی ها برای ذخیره مقدار کمی اطلاعات بر روی دستگاه مشتری (CLIENT) استفاده می شوند. یک کوکی می تواند حداکثر تا ۴ کیلو بایت را ذخیره کند. بطور کلّی کوکی ها برای ذخیره داده هایی که اغلب کاربر تایپ می کند استفاده می شوند، از قبیل اسم کاربری و پسورد برای لاگین در سایت.

دو نوع کوکی وجود دارد: زمانی (session) و دائمی (persistent) .کوکی های زمانی کوکی های موقتی نیز خوانده می شوند که در حافظه مرورگر ذخیره می شوند و زمان حیات آنها به مرورگر بستگی دارد. هنگامی که شما مرورگر را می بندید این کوکی ها هم می میرند. از طرف دیگر کوکی های پایدار بر روی هارد دیسک با اطلاعات تاریخ ذخیره می شوند و برای زمان درازی می توانند زنده بمانند. هنگامی که شما کوکی های پایدار را ایجاد می کنید شما می توانید زمان زندگی کوکی را نیز تعیین کنید. اگر چه مدتی که شما تعیین می کنید تا کوکی زنده بماند قطعی نیست. همچنین اگر چه کوکی ها شئ‌های مفیدی در دنیای وب هستند امّا آنها برخی محدودیت هایی هم دارند. مثلاً کوکی ها نمی توانند اطلاعات زیادی را در خود ذخیره کنند.

همچنین کوکی ها وابسته به مرورگرها هستند به این معنی که کوکی کاربردی در برخی مرورگرها نمی تواند کار کند. شما می توانید زمان زندگی یک کوکی را تعیین کنید اما نمی توان تضمین کرد که کوکی در آنجا برای مدت طولانی در دسترس باشد .بنابراین ذخیره اطلاعات مهم در کوکی ها ایده خوبی نیست.

ساختن و خواندن کوکی‌ها

[1] محمد علویان

کلاس HttpCookie در فضانام System.Web تعریف شده است که کوکی ها را نمایش می دهد. خواص کوکی‌ها مثل Response و Request می تواند در خواندن تمام کوکی ها به کار رود بطوریکه شئ HttpCookieCollectionمی تواند همه کوکی ها را نمایش می دهد.

همانند دیگر کلاس ها کلاس HttpCookieCollection اضافه کردن حذف کردن و خواندن کوکی ها را از مجموعه برای اعضا فراهم می کند. شما می توانید یک کوکی توسط کلاس HttpCookie با مشخص کردن نام و مقدار کوکی به صورت رشته ای ایجاد کنید. یا شما می توانید از خواص Name و Value برای نام و مقدار کوکی استفاده کنید .خاصیت Expires کلاس HttpCookie یک کوکی پایدار می سازد و زمانی را که کوکی از بین خواهد رفت مشخص می کند. در قطعه کد زیر دوتا کوکی می سازد به نام UID و .PASSشما در این کد می توانید ببینید که ما کوکی ها را به مجموعه توسط تابع Response.Cookies.Add اضافه می کنیم:

Listing 1. Creating cookies

```
'ساختن اسم کاربری و پسورد کوکی ها
'دادن مقدار به آنها و اضافه کردن به مجموعه
Dim cookie As HttpCookie = New HttpCookie("UID")
cookie.Value = "myid"
cookie.Expires = #9/28/2002#
Response.Cookies.Add(cookie)
cookie = New HttpCookie("PASS")
cookie.Value = "mypass"
cookie.Expires = #9/28/2002#
Response.Cookies.Add(cookie)
```

شما می توانید کوکی ها را با استفاده از صفت Request.Cookies بخوانید. قطعه کد زیر کوکی را از مرورگر می خواند و آنها را به کنترل ListBox اضافه می‌کند:

Listing2 . Read cookies

```
'خواندن کوکی ها
Dim cookieCols As New HttpCookieCollection()
cookieCols = Request.Cookies
Dim str As String
'خواندن و اضافه کردن تمام کوکی ها به لیست باکس
For Each str In cookieCols
   ListBox1.Items.Add("Cookie: " + str)
   ListBox1.Items.Add("Value:" & _
   Request.Cookies(str).Value)
Next
```

شما می توانید از توابع Clear و Remove در HttpCookieCollection برای حذف کوکی خاص یا همه کوکی ها استفاده کنید. قطعه کد زیر با استفاده از تابع Remove کوکی‌ها را حذف می کند:

Listing3 . Deleting cookies

```
Dim cookieCols As New HttpCookieCollection()
cookieCols = Request.Cookies
Dim str As String
'خواندن و حذف تمام کوکی ها از لیست باکس
```

```
Request.Cookies.Remove("PASS")
Request.Cookies.Remove("UID")
```

یک کوکی همچنین می تواند چندین مقدار را ذخیره کند. این نوع کوکی، کوکی دیکشنری نامیده می شود. شما می توانید از صفت Values برای ایجاد و خواندن این نوع کوکی ها استفاده کنید. کد زیر یک کوکی دیکشنری می سازد.

Listing4 . Creating a dictionary cookie

```
Dim cookDict As HttpCookie = New HttpCookie("dict")
cookDict.Values("fname") = "first name"
cookDict.Values("lname") = "last name"
cookDict.Values("Address") = "address"
Response.Cookies.Add(cookDict)
```

منبع: Asp.Heaven

فرمهای وب

کنترل های HTML[1]

دریافت اطلاعات ورودی و بررسی کنترل های HTML در ASP.NET

لینک دریافت کد: www.iranasp.net/download/htmlcontrols.zip

در یک صفحه ASP.NET می توان دو نوع کنترل داشت: کنترل های HTML و کنترل های Web. کنترل های HTML جهت ایفای نقش همان تگ های HTML معمولی طراحی شده اند. بعبارت دیگر اگر یک صفحه HTMLمعمولی داشته باشیم، با تبدیل تگ های مربوطه به کنترل های HTML می توانیم سریعا آن را به یک صفحه ASP.NET تبدیل کنیم.

بعنوان مثال صفحه ساده HTML زیر را در نظر بگیرید. این صفحه حاوی یک فرم است که شما را قادر می سازد تا مثلا رنگ دلخواه خود را در آن وارد کنید.

```
<html>
<head><title>SimpleHTML.htm</title></head>
<body>

<form method="post" action="SimpleHTML.htm">

Enter your favorite color:
<br>
<input name="favColor" type="text">
<p>
<input type="submit" value="Submit!">
```

[1] مدیریت سایت

```
</form>

</body>
</html>
```

اگر رنگ دلخواه خود را وارد کرده و دکمه ارسال فرم را بزنید، اتفاق خاصی نخواهد افتاد چرا که فرم تنها اطلاعات وارد شده را به خود برمی گرداند و از آنجا که پس از ارسال فرم، پردازشی روی اطلاعات ورودی انجام نمی شود، ورودی مذکور از بین می رود.

در مثال زیر صفحه فوق به یک صفحه ASP.NET تبدیل شده است.

```
<html>
<head><title>SimpleASPX.aspx</title></head>
<body>

<form runat="server">

Enter your favorite color:
<br>
<input id="favColor" type="text" runat="server">
<p>
<input type="submit" value="Submit!" runat="server">

</form>
```

```
</body>
</html>
```

همانگونه که مشاهده می کنید چهار تغییر اساسی و مهم نسبت به صفحه HTML مثال قبل صورت گرفته است:

- پسوند فایل به aspx تغییر کرده است.
- مشخصه runat="server" به همه عناصر فرم موجود اضافه شده است.
- بجای مشخصه name از مشخصه id در تگ <input> استفاده شده است.
- مشخصه های Method و Action از تگ <form> حذف شده اند زیرا فرم های ASP.NET بصورت پیش فرض به همان صفحه ارسال می شوند و دیگر نیازی به مشخصه های فوق نیست.

اگر مثال فوق را اجرا کنید خواهید دید که پس از ارسال فرم و نمایش مجدد صفحه فوق، رنگ مورد نظرتان در قسمت ورودی همچنان موجود است. بعبارت دیگر برعکس HTML معمولی، کنترل های HTML در ASP.NET مقادیر خود را بصورت خودکار در استفاده های بعدی حفظ می کنند.

کنترل های وب در ASP.NET[1]

دریافت اطلاعات از ورودی و بررسی کنترل های وب در ASP.NET

لینک دریافت کد: www.iranasp.net/download/webcontrols.zip

HTMLدر طول زمان تکمیل شده و به شکل امروزی رسیده است و تاکنون چندین بار ترمیم شده است. لذا اگر دقت کنید متوجه می شوید در برخی موارد، HTMLآن حس واقعی از کنترل های واسط کاربری را به انسان نمی دهد. بعبارت دیگر در HTML با استفاده از یک تگ مانند <select> می توان دو نوع واسط کاربری متفاوت مانند ListBoxو ComboBox ساخت، در حالیکه برای ساخت دو نوع واسط کاربری از یک جنس مانند InputLine و TextBoxنیاز به دو تگ متفاوت <input> و <textarea> داریم.

در ASP.NET سعی شده است که این نقیصه بنوعی برطرف گردد. در ASP.NET جهت ساخت واسط کاربری، عناصر متعددی در نظر گرفته شده است که کنترل های وب نام دارند. این کنترل ها در مرحله برنامه نویسی کاملا با تگ های HTML متفاوتند اما در مرحله اجرا، این کنترل ها به تگ های معادل و معمولی در HTML تبدیل می شوند. در زیر برخی کنترل های وب رایج و پایه ای در ASP.NET معرفی می گردد:

- TextBox:جهت دریافت ورودیهای متنی (تایپی) بصورت تک خط یا چند خط و نیز کلمه عبور.
- Label :جهت نمایش برچسب و عبارات متنی ثابت و نمایشی.
- CheckBox :جهت ورودی های از نوع فعال / غیرفعال.
- RadioButton :جهت ورودی های انتخابی از میان چندین گزینه.
- HyperLink :جهت نمایش و ساخت پیوند یا اتصال) لینک) به دیگر صفحات.

[1] مدیریت سایت

- Button :جهت ساخت یک دکمه یا کلید برای ارسال یک فرم HTML جهت پردازش به سرور یا انجام عمل خاصی.
- LinkButton :جهت ساخت پیوندی که با فشردن آن یک فرم HTML جهت پردازش به سرور ارسال می گردد.
- ImageButton :جهت ساخت یک دکمه تصویری که با فشردن آن یک فرم HTML جهت پردازش به سرور ارسال می گردد.

کاربرد این کنترل ها در یک صفحه مانند تگ های HTML است. یعنوان مثال، در صفحه ASP.NET زیر، دو کنترل از نوع TextBox و یک کنترل از نوع Button در یک فرم تعریف شده اند.

```
<html>
<head><title>Guestbook.aspx</title></head>
<body>

<h3>Please Sign Our Guestbook!</h3>

<form runat="Server">

Your Name:
<br>
<asp:TextBox
ID="username"
runat="Server" />
<p>
```

```
Comments:
<br>
<asp:TextBox
ID="comments"
TextMode="Multiline"
Columns="50"
Rows="4"
runat="Server" />

<p>
<asp:Button
Text="Submit!"
Runat="Server" />

</form>

</body>
</html>
```

در نگاه اول شاید ترکیب استفاده از کنترل های وب در یک صفحه برای شما نامأنوس باشد. اما اگر دقت کنید متوجه می شوید که استفاده از این کنترل ها چندان هم مشکل نیست. بعنوان مثال اولین کنترل TextBox جهت دریافت اسم کاربری بصورت زیر تعریف شده است.

```
<asp:TextBox
ID="username"
runat="Server" />
```

دقت داشته باشید که برای تعریف صحیح یک کنترل وب باید سه مورد زیر را در نظر گرفت:

- تعیین نوع کنترل (TextBox ، Label ، ...)
- یک نام منحصربفرد در صفحه با استفاده از مشخصه ID
- عبارت runat="Server" جهت تعیین اینکه این کنترل از نوع کنترل های وب است و هنگام اجرا باید ابتدا برروی سرور به تگ معادل HTML تبدیل شود.

رسیدگی به رویدادهای کنترلی در ASP.NET[1]

نحوه پاسخگوئی به اعمال یا درخواستهای کاربر در یک صفحه ASP.NET

لینک دریافت کد: www.iranasp.net/download/eventhandle.zip

هنگامی که در یک فرم HTML یک دکمه (Button) فشرده (کلیک) می شود، یک رویداد (Event) رخ می دهد یا هنگامی که یک لیست بازشو (ComboBox) را باز می کنیم باز هم یک رویداد اتفاق افتاده است. بنابراین یک رویداد در یک صفحه وب عبارت است از تغییر وضعیتی که در آن صفحه رخ می دهد خواه این تغییر وضعیت، خودکار باشد یا بصورت غیرخودکار توسط کاربر رخ دهد.

از رویدادهای خودکار می توان به رویدادهای Init ، Loadو Unload اشاره کرد. از موارد غیرخودکار می توان به فشردن یک دکمه در یک فرم یا خاموش / روشن کردن یک گزینه انتخابی (checkbox) اشاره نمود.

همانگونه که می دانیم دو نوع کنترل HTML و وب در ASP.NET وجود دارد .هر دو نوع این کنترل ها می توانند سبب ایجاد رویدادی گردند. رسیدگی به یک رویداد (Event Handling) عبارت است از انجام عمل خاصی به ازای رخ دادن آن رویداد. جهت رسیدگی به یک رویداد می توان یک روال نوشت که مثلا هنگام فشرده شدن دکمه ای در یک فرم، داده های ورودی در فرم درون یک جدول در یک بانک اطلاعاتی ذخیره شوند.

رسیدگی به رویدادهای کنترل هایHTML

فشردن یک دکمه HTML در یک صفحه ASP.NET سبب رخ دادن رویداد ServerClick می شود. هنگامی که فرم مذکور به روی سرور ارسال می شود، این رویداد بر روی سرور رخ می دهد.

[1] مدیریت سایت

مثال زیر نحوه رسیدگی به رویداد ServerClick را بخوبی نشان می دهد .حاصل اجرای برنامه زیر نمایش وارونه متنی است که در فرم وارد شده است.

```
<Script runat="Server">

Sub submitText( s As Object, e As EventArgs )
mySpan.InnerHtml = StrReverse( myTextArea.Value )
End Sub

</Script>

<html>
<head><title>HtmlControlEvents.aspx</title></head>
<body>

<form runat="server">

Enter some text:
<br>
<textarea
id="myTextArea" cols="30" rows="3"
runat="Server"></textarea>

<p>
<input type="submit" value="Submit Text!"
runat="server" onServerClick="submitText">
```

```
<p>
<span id="mySpan" runat="server"></span>

</form>

</body>
</html>
```

در مثال فوق دکمه HTML بصورت زیر تعریف شده است:

```
<input type="submit" value="Submit Text!"
runat="server" onServerClick="submitText">
```

به مشخصه onServerClick="submitText" توجه کنید. هنگامی که بر روی دکمه مذکور کلیک شود، مشخصه onServerClick سبب اجرای روال submitText می گردد.

روال submitText در یک بلوک برنامه ای در بالای صفحه تعریف شده است .این روال متن وارد شده در textboxرا گرفته و آن را بصورت وارونه به کنترل منسوب می کند.

روال submitText دارای دو نوع پارامتر است: یکی از نوع object و دیگری از نوع .EventArgs پارامتر از نوع object حاوی کنترلی است که رویداد مربوطه را صادر کرده است. در مثال فوق کنترل صادر کننده رویداد، دکمه روی فرم است. پارامتر EventArgs حاوی اطلاعات مخصوصی است که ممکن است همراه رویداد ارسال گردد. در مثال فوق اطلاعاتی ارسال نمی شود.

رسیدگی به رویدادهای کنترل های وب

رسیدگی به رویدادهای کنترل های وب همانند رسیدگی به رویدادهای کنترل های از نوع HTML است. برای مثال کنترل Button یا دکمه یک رویداد از نوع Click را صادر می کند که می توان در برنامه به آن رسیدگی نمود.

مثال زیر همانند مثال قبل می باشد با این تفاوت که این مثال برای کنترل های وب نوشته شده است.

```
<Script runat="Server">

Sub submitText( s As Object, e As EventArgs )
myLabel.Text = StrReverse( myTextBox.Text )
End Sub

</Script>

<html>
<head><title>WebControlEvents.aspx</title></head>
<body>

<form runat="server">

Enter some text:
<br>
<asp:TextBox
id="myTextBox"
TextMode="Multiline"
Columns="30"
Rows="3"
Runat="Server" />
```

```
<p>
<asp:Button
Text="Submit Text!"
runat="server"
onClick="submitText" />
<p>
<asp:Label
id="myLabel"
runat="server" />
</form>
</body>
</html>
```

در مثال فوق کنترل Button بصورت زیر تعریف شده است:

```
<asp:Button
Text="Submit Text!"
runat="server"
onClick="submitText" />
```

مشخصه onClick جهت برقرار نمودن ارتباط رویداد Click مربوط به کنترل Button به روال submitTextمی باشد. هنگامی که بر روی کنترل Button کلیک شود، این روال بر روی سرور اجرا می گردد.

روال submitText متن وارد شده در کنترل Textarea را دریافت و پس از وارونه کردن، آن را در کنترل Labelنشان می دهد.

کنترل Label یا برچسب[1]

نمایش متن یا برچسب با استفاده از کنترل Label و تعیین رنگ، فونت و سایر مشخصات آن

لینک دریافت کد: www.iranasp.net/download/labelcontrol.zip

جهت نمایش متون ثابت در یک صفحه ASP.NET از طریق برنامه می توان از کنترل Label یا برچسب استفاده نمود. برای این منظور متنی که به صفت Text این کنترل منسوب گردد توسط این کنترل نمایش داده می شود. مثال زیر نحوه مقدار دهی یک کنترل برچسب با تاریخ روز در هنگام بارگذاری یک صفحه با استفاده از رویداد Page-Loadرا نشان می دهد.

```
<Script runat="Server">

Sub Page_Load
   lblDate.Text = Now.ToString( "D" )
End Sub

</Script>

<html>
<head><title>ShowDate.aspx</title></head>
<body>

<asp:Label
   id="lblDate"
```

[1] مدیریت سایت

```
   Runat="Server" />

</body>
</html>
```

جهت نمایش متن مورد نظر با مشخصات دلخواه مانند رنگ، پس زمینه، فونت و غیره می توان از کلیه خواص پایه ای کنترل های وب در این زمینه بهره جست. بعنوان مثال صفحه زیر تاریخ روز را با رنگ آبی در پس زمینه زرد با شکل فونت Verdana در اندازه p ۲۴ نشان می دهد.

```
<Script runat="Server">

Sub Page_Load
   lblDate.Text = Now.ToString( "D" )
End Sub

</Script>

<html>
<head><title>LabelFormat.aspx</title></head>
<body>

<asp:Label
   id="lblDate"
   ForeColor="Blue"
   BackColor="Yellow"
   Font-Name="Verdana"
   Font-Size="24pt"
```

```
   Runat="Server" />

</body>
</html>
```

الحاق یک ListBox به شئ ArrayList[1]

این مقاله نحوه ترکیب و فراخوانی داده ها را از طریق شئ ArrayList نشان می دهد .

لینک دریافت کد: www.iranasp.net/download/shahoo03.zip

در ASP.NET معمولا محتویات یک ListBox از درون کد رویداد Page_Load استخراج می گردد. در ASP.NETمی توان داده های یک کنترل را از یک منبع داده (Data souce) تامین کرد. در این حالت کنترل بصورت خودکار مقادیر موجود در منبع داده را به خود می گیرد. برای این منظور می توان از یکی از اشیاء رایج و پرکاربرد در ASP.NET که همان ArrayList است بهره برد.

اغلب برای ساختن و فراخوانی مجموعه ای از اشیاء از ساختاری مشخص تبعیت می شود که این ساختار با عنوان مجموعه اشیاء (Objects Collection) شناخته می شود. بعنوان مثال ArrayList یک مجموعه اشیاء است. در مجموع، الحاق یک کنترل ListBox به یک Datasource ساده است. برای اینکار کافیست که خاصیت Datasourceیک کنترل معلوم شود. سپس با استفاده از متد DataBind مربوط به آن کنترل، ارتباط آن با ListBoxمشخص می شود. در مثال زیر ابتدا یک ArrayList از نام گلها ساخته می شود و سپس کنترل ListBoxدر اینجا (lstFlowers) برای فراخوانی داده به ArrayList متصل می شود. مثال زیر چگونگی انجام کار را نشان میدهد.

```
<% @Page Language="VB" Debug="True" %>
<html><head><title>Databing Demo</title></head>
<body>
<form runat="server">
Select a flower, and then click the submit button please:
```

[1] شاهو طوفانی

```
<br>
<asp:ListBox id="lstFlowers" runat="server" rows="3"
   AutoPostBack="True" onSelectedChanged="showSelection"/>
</asp:ListBox>
<br><br>
<asp:Label id=lblMessage runat="server"></asp:Label></p>
</form>
</body></html>
<script language=vb runat="server">
sub Page_Load(source As Object, e As EventArgs)
   if Not Page.IsPostBack Then
      Dim myArrayList As New ArrayList
      myArrayList.Add("Azalea")
      myArrayList.Add("Tulip")
      myArrayList.Add("Rose")
      lstFlowers.DataSource=myArrayList
      lstFlowers.DataBind()
      lstFlowers.SelectedIndex=0
   end if
end sub
sub showSelecttin(Sender As Object, e As EventArgs)
   lblMessage.Text="You have selected " +lstFlowers.SelectedItem.Text
end sub
</script>
```

کنترل HyperLink یا پیوند[1]

نشان دادن پیوند با استفاده از HyperLink وخصوصیات آن

لینک دریافت کد: www.iranasp.net/download/shahoo02.zip

جهت نشان دادن پیوند (لینک) در یک صفحه ASP.NET میتوان از کنترل HyperLink استفاده کرد. برای این منظور عبارت پیوند را میتوان در خاصیت Text پیوند نشان داد و بوسیله خاصیت NavigateUrl آدرس پیوند را مشخص کرد همچنین بوسیله خاصیت ImageUrl از کنترل عبارت پیوند را با تصویری جایگزین کرد که هنگام کلیک روی تصویر ما به صفحه ای که در خاصیت NavigateUrl مشخص شده است هدایت خواهیم شد. مثال زیر نحوه عملکرد این کنترل را نشان می دهد:

)باید توجه داشت در این مثال شما به تصویری با عنوان hyper.gif در پوشه برنامه نیازدارید(.

```
<%@ Page Language="VB" Debug="True" %>
<html>
<head>
<title>HyperLink Control</title>
</head><body>
<asp:HyperLink id="hl1" runat="server"
   NavigateUrl="http://www.iranasp.net"
   Text=" Go to IranASP.NET" />
<br>
<asp:HyperLink id="hl2" runat="server"
   NavigateUrl="http://www.iranasp.net/forum"
   ImageUrl="hyper.gif"
```

[1] شاهو طوفانی

```
   Text="Go to IranASP.NET Forum"/>
<br>
<br>
</body>
</html>
```

آشنائی با فرم های وب[1]

فرم های وب به عنوان یکی از مهمترین اجزای ASP.NET ، جهت ایجاد فرم های ورود اطلاعات کاربر بکار می روند .

به عنوان یک برنامه نویس ویژوال بیسیک شما می توانید برنامه های تحت اینترنت نیز بنویسید. به طور معمول برنامه نویسان ویژوال بیسیک به سمت ASP که یک تکنولوژی از مایکروسافت است متمایل هستند. دلیل این امر هم شباهت میان VB و VBScript می باشد. بزرگترین ایرادی که ASP کلاسیک دارد نداشتن یک محیط ویژوال مانند فرم های معمولی بیسیک است. مایکروسافت با Visual InterDev سعی کرد این کمبود را جبران کند اما چندان موفق نبود. بالاخره در ویژوال بیسیک دات نت ترکیبی از InterDev و ویژوال بیسیک وجود دارد و امکانات فرم های ویژوال بیسیک را برای اینترنت نیز فراهم می کند.

Web Formها یکی از اجزای تکنولوژی ASP.NET است که به برنامه نویس های اکثر زبان ها این امکان را می دهد که یک قالب ویژوال با HTML و یک محیط برنامه نویسی تحت سرور با کدهای پیشرفته داشته باشند.

Web Fromها در عمل

بهترین راه برای فراگیری این تکنولوژی یک مثال عملی از آن است. پس از مثال معروف Hello World برای شروع استفاده می کنیم.

آماده سازی محیط

[1] حامد بنایی

قبل از شروع ابتدا باید نرم افزارهای مورد نیاز را از روی لیست زیر نصب کنید. اگر ویژوال استودیو دات نت را به شکل کامل و بر روی ویندوز ۲۰۰۰ یا اکس پی نصب کردید احتیاج به مراحل زیر ندارید .

•سیستم عامل شما باید حتماً از نوع ان تی باشد، ویندوز ۲۰۰۰ سرور یا(Professional) ، ویندوز اکس پی Professional و یا ان تی سرور ۴.

•باید NET Framework.بر روی سروری که می خواهید با آن کار انجام دهید یا برنامه شما بر روی آن اجرا خواهد شد نصب شده باشد. اگر ویژوال استودیو دات نت را نصب کرده اید مشکلی در این مرحله ندارید.

•مایکروسافت توصیه کرده که سیستم فایل هارد دیسک سرور شما بهتر است NTFS باشد، هم به دلیل مسائل امنیتی و هم سرعت بیشتر.

Hello World

در فرم مخصوص ایجاد یک پروژه جدید ASP.NET Web Application را انتخاب کنید و نام آن را HelloWorldقرار دهید. دقت کنید که مکان ذخیره پروژه http://localhost باشد.

سپس بر روی کلید OK کلیک کنید تا یک solution جدید ایجاد گردد. به طور قراردادی ویژوال استودیو یک Web Formبا نام WebForm1.aspx ایجاد می کند. دقت کنید که پسوند فایل چه تغییری کرده است.

وقتی بر روی کلید OK کلیک می کنید چند عمل در پشت صحنه انجام می شود .به غیر از ایجاد کردن یک شاخه در دایرکتوریVisual Studio Projects ، ویژوال استودیو یک web application نیز در سروری که انتخاب کرده اید ایجاد می کند. بر روی سرور، ویژوال استودیو دات نت :

•یک دایرکتوری با نام پروژه در شاخه inetpub/wwwroot ایجاد می کند.

•این دایرکتوری را به عنوان یک IIS Application معرفی کرده و اجازه اجرای script را بر روی آن می دهد.

•اگر FrontPage Server Extensions را نصب کرده باشید یک FrontPage Web ایجاد می کند تا با FrontPageهم بتوانید به آن دسترسی داشته باشید.

می توانید همانگونه که با فرم های معمولی ویژوال بیسیک کار می کردید از Web Form ی که جلوی شما است استفاده کنید، یعنی به شما امکان استفاده از toolbox و استفاده از کامپوننت های درون آن بر روی web formداده شده است. یک Label را از toolbox برداشته و بر روی قسمت بالای فرم قرار دهید و خاصیت text آن را به Hello World تغییر دهید.

برای این مثال تمام کاری که لازم بود انجام شود را انجام دادیم. حالا می توانیم برنامه را اجرا کنیم. قبل از اینکار از toolbar و در قسمت Solution Configuration به جایdebug ، releaseرا انتخاب کنید. حالا بوسیله کلید F5برنامه را اجرا کنید. اگر هیچ مشکلی در سیستم نباشد باید صفحه مرورگر باز شود و فایل WebForm1.aspx نمایش داده وبر روی آن Hello World نوشته شود.

بر روی صفحه مرورگر کلید سمت راست موس را بزنید و View Source را انتخاب کنید تا ببینید چه مطالبی در سورس این صفحه آمده است. همانطور که می بینید کدهایی به HTML است که بوسیله فایل aspx در زمان اجرا ایجاد شده است.

همانطور که می بینید یک HTML Form در این متن دیده می شود در حالی که ما چنین چیزی را اضافه نکرده بودیم، درباره این مساله در ادامه توضیح خواهم داد. Label ی که اضافه کرده بودیم در تگِ span قرار دارد. تگِ span مانند یک container برای Label ما است و اطلاعات آن را در خود نگهداری می کند. به ویژوال استودیو دات نت باز می گردیم.

همانطور که دیدید Web Form ها خیلی شبیه فرم های معمولی ویندوز هستند. در Web From جدیدی که می سازیم این خاصیت را بیشتر امتحان می کنیم. در برنامه Hello World که ایجاد کردیم تنها یک Web Formداشتیم: WebForm1.aspx. یک Web Form دیگر می سازیم تا کارهای بیشتری با آن انجام دهیم.

منوی Project | Add Web Form را انتخاب کنید. در فرمی که باز می شود Web Form را انتخاب کنید و مطمئن شوید که نام آن WebForm2.aspx است. (قبل ازاین کار ویژوال استودیو را از حالت اجرای برنامه خارج کنید)

بر روی Open کلیک کنید تا WebForm2.aspx در solution ایجاد شود. بر روی WebForm2.aspx در Solution Explorer دو بار کلیک کنید تا مطمئن باشید که فرمی که تازه ایجاد کرده اید فعال است. مانند مثال قبلی یک Label بر روی فرم قرار دهید، سپس یک Button در زیر آن قرار دهید و اندازه هر دو را یکسان کنید. بر روی Label کلیک کنید و از پنجره Properties خاصیت ID را انتخاب کنید و آن را به lblText تغییر دهید. سپس بر روی کلید کلیک کنید و ID آن را به btnSubmit تغییر دهید. بر روی کلید یک بار کلیک کنید، سپس کلید Enter را بزنید تا به قسمت نوشتن کد برای این کلید وارد شوید.

در ASP.NET هر کدام از کنترل ها، کدی در پشت صحنه برای خود دارند .همانطور که مشاهده می کنید یک روال با نام btnSubmit_Click وجود دارد که هنگامی که بر روی کلید کلیک می شود اجرا می شود. کدی که در این روال نوشته شده باشد در سرور اجرا می شود و نه در مرورگر کامپیوتر کاربر. کد زیر را در روال مورد بحث بنویسید:

```
lblText.Text = "Hello World"
```

همانطور که مشاهده کردید IntelliSense وارد عمل شده و وقتی بعد ازlblText ، نقطه را تایپ کردید لیستی از خواص و متد های مربوط به Label را به شما نمایش داد. این خاصیت در InterDev هم وجود دارد ولی در ویژوال استودیو دات نت از امکانات بیشتر و لیست پرمحتواتری برخوردار است.

پنجره کد را ببندید و به قسمت طراحی Web Form بروید، خاصیت ِText کلید را به Submit تغییر دهید. حالا برنامه را امتحان می کنیم. اگر سعی کنید تا برنامه را بوسیله کلید F5 اجرا کنید دوباره WebForm1.aspxنمایش داده خواهد شد، زیرا که این فرم، فرم ابتدایی در پروژه ما است. برای اینکه WebForm2.aspxبه فرم ابتدایی تبدیل شود در پنجره Solution Explorer بر روی WebForm2.aspx کلید سمت راست موس را بزنید و سپس Set As Start Page را انتخاب کنید. حالا می توانید برنامه اجرا کنید.

Web Formجدید، WebForm2.aspxدر مرورگر اینترنت نمایش داده می شود در حالی که بر روی آن یک Label و یک کلید وجود دارد. بر روی کلید کلیک کنید تا متنی که تایپ کرده بودید در Label نمایش داده شود. همانطور که می بینید برنامه مانند فرم های معمولی ویندوز اجرا می شود.

مراحل مختلف اجرای یک فرم وب در ASP.NET[1]

ASP.NETمانند ASP کلاسیک به شکل تفسیری اجرا نمی شود، بلکه کامپایل شده و هر بار فایل کامپایل شده با سرعت بالایی اجرا می شود .

برنامه نویسی در محیط وب تا قبل از دات نت حتی در ASP احتیاج به ساختن یک صفحه با HTML و قرار دادن کدهای مورد نیاز در آن داشت. یک صفحه ASP تا قبل از ASP.NET یک متن ساده بود که از بلاک های کد ASPو بلاک های کد HTML تشکیل شده بود. در هنگام فراخوانی توسط کاربر صفحه های ASP توسط سرور خوانده شده و خط به خط دستورات آن اجرا می شد، در نهایت یک صفحه HTML به مرورگر اینترنت کاربر تحویل داده می شد.

اما فرم وب مانند یک برنامه کامل اجرا (Execute) می شود و نتیجه آن یک HTML است. همانطور که قبلاً مشاهده کردید فرم های وب با پسوند aspx و به صورت یک فایل متنی هستند. در یک سرور دات نت (هر IIS Serverی که NET Framework.بر روی آن نصب شده باشد) ، وقتی که یک مرورگر، درخواستی برای دیدن یک aspx می دهد runtime مربوط به ASP.NET آن صفحه را تجزیه و تحلیل کرده و آن را کامپایل می کند. این مرحله شبیه به نحوه کار ASP کلاسیک است، به غیر از اینکه نتیجه این کار در یک کلاس از نوع دات نت ذخیره می شود. این کد، یک کد کامپایل شده است و مانند ASP کلاسیک نیست که به صورت تفسیری در هنگام لزوم اجرا شود. این روش اجرا سرعت را در هنگام فراخوانی مجدد aspx بالا می برد.

به طور کلی یک پروژه برنامه اینترنتی در دات نت (بیسیک) حداقل یک فایل aspx و یک فایل aspx.vb دارد، که در اولی کد های HTML مانند ASP کلاسیک وجود دارد و در فایل دوم کدهایی که برای هر کدام از اجزا نوشته می شود قرار می گیرد.

[1] حامد بنایی

به علاوه در این نوع پروژه یک فایل Global.asax قرار دارد که مشابه Global.asa در ASP کلاسیک است. همچنین فایلی با نام Web.Config وجود دارد که با ساختار XML ذخیره شده است و اطلاعاتی درباره پروژه را در خود ذخیره می کند.

این فایلها در دو مکان قرار می گیرند. مکان اصلی سرور اینترنتی است که برنامه باید بر روی آن اجرا شود، همچنین یک نسخه از آن در قسمت cache ویژوال استودیو ذخیره می شود. هنگامی که در ویژوال استودیو تغییری بر روی پروژه می دهید آن تغییر در هر دو مکان اعمال می شود.

وقتی بوسیله ویژوال استودیو می خواهید برنامه خود را انتقال بدهید، ویژوال استودیو مانند یک برنامه معمولی با پروژه رفتار می کند. تمامی کدهای آن را به فایل DLL تبدیل می کند و هیچ سورسی انتقال نمی یابد. البته فایلهای aspxبه همان شکل انتقال می یابند. در هنگام نصب نیز فایلهای DLL ی که باید در سرور اینترنت قرار گیرند در آن جا کپی می شوند. در هنگام اجرا فایل aspx و DLL بعد از تلفیق با هم به مرورگر درخواست کننده ارسال می شود.

مثال HelloWorld را در هنگامی که فایل WebForm2 فایل اصلی بود به خاطر بیاورید. پروژه از فایلهای زیر تشکیل شده است:

WebForm2.aspx
WebForm2.aspx.vb
Global.asax
Global.asax.vb
Web.Config

دقت کنید که ما فایل Global.asax را نساخته ایم، بلکه خود ویژوال استودیو دات نت بود که این کار را به شکل خودکار انجام داد. وقتی با کلید F5 برنامه را اجرا می کردیم فایلهای WebForm2.aspx و Global.asax مستقیماً، بدون تغییر بر روی سرور اینترنت قرار می گرفتند. اما فایلهای WebForm2.aspx.vb و

Global.asax.vbکامپایل می شوند و نتیجه در فایل HelloWorld.dll قرار می گیرد و در سرور در کنار دو فایل قبلی کپی می شود.

وقتی بوسیله مرورگر فایل aspx فراخوانده می شود، ASP.NETبه صورت پویا یک فایل cls برای آن ایجاد می کند، سپس این فایل به فایل dll کامپایل می شود .این فایل dll در نهایت فایل HelloWorld.dll را صدا می زند و نتیجه اجرا به HTML تبدیل شده و به مرورگر باز گردانده می شود.

به یاد داشته باشید که هر برنامه ای در .NET Framework از System استفاده می کند. برنامه های اینترنتی و برای مثال aspx نیز از کلاس System.Web.UI.Page مشتق شده است. این امر سبب می شود که در هنگامی که aspx به کلاس تبدیل شود، کلاس ایجاد شده از کلاس اصلی (system.web.ui.page) مشتق شود. این کلاس مفاهیمی همچون Response ، Requestو... را فراهم می کند.

در ظاهر این مراحل، اجرای یک صفحه را بسیار کند می کند، ولی تمام اینها فقط یک بار و در هنگام اولین درخواست دیدن صفحه اتفاق می افتد. فایل کلاسی که ساخته شد در مکانی ذخیره شده و تا هنگامی که آن را تغییر نداده باشید ثابت می ماند .بنابراین در دفعات بعدیِ فراخوانی صفحه، سرعت اجرا و نمایش آن خیلی زیاد تر می شود.

تشریح ساختمان فرم های وب[1]

فرم وب در هر کدام از زبان های دات نت از یک بخش کد html و یک بخش کد به زبان مورد نظر تشکیل شده است .

فرم وب بوسیله ایجاد یک محیط ویژوال و قابلیت وجود رویدادها برای اجزای فرم، شکاف بین برنامه نویسی ویژوال بیسیک و ASP کلاسیک را پر می کند.

یک فرم وب از دو قسمت تشکیل شده است: اجزای ویژوال که آنها را در هنگام طراحی می توانید ببینید و دیگری کدهایی که در پشت کنترل ها و صفحه ها می باشند. اجزای ویژوال در مرورگر اینترنت کاربر دیده می شود و کدهای کنترل ها و رویدادهای آنها در سرور اجرا می شوند و نتیجه به کاربر اعلام می شود. در ویژوال استودیو دات نت برای اجزای ویژوال از فایل های aspx و برای کد های آن از فایلهای) vb یا cs و کلا بسته به زبان مورد استفاده) استفاده می شود.

بوسیله تقسیم اجزای فرم وب در فایلهای مختلف و در ویژوال استودیو بوسیله نمایش آنها در پنجره های متفاوت، فرم وب یک محیطی شبیه به برنامه های معمولی ویژوال بیسیک پیدا می کند. در بیسیک قدیمی ابتدا بوسیله اجزای ویژوال فرم های خود را نقاشی می کردید و سپس کد مربوط به هر جز را می نوشتید، در فرم وب نیز ابتدا اجزای مورد نیاز خود را بر روی صفحه قرار می دهید سپس برای آنها کد می نویسید.

بخش کد

همانطور که ذکر شد کد هایی که برای aspx می نویسید در فایلی با همان نام و با پسوند vb ذخیره می شود، برای دیدن این فایلها در قسمت بالای Solution Explorer بر روی Show All Files کلیک کنید. در زیر

[1] حامد بنایی

مجموعه فایلهای aspx فایلهای vb را نیز می توانید مشاهده کنید. بر روی فایل WebForm2.aspx.vb کلیک کنید، همانطور که مشاهده می کنید این فایل حاوی یک کلاس با نام WebForm2 است. این کلاس مشتق شده از System.Web.UI.Pageاست، این امر باعث می شود که صفحه از آبجکت های , Request , Response Session , Serverو Application استفاده کند. برای فعال شدن رویداد های کنترل های مورد استفاده در صفحه نیز از WithEvents استفاده شده است.

یک مثال پیچیده تر

فرض کنید بخواهیم که تقویم این ماه را برای روی یک صفحه وب نمایش دهیم. ساختن یک تقویم پویا با استفاده از ASP کلاسیک حداقل به ۵۰ تا ۱۰۰ خط برنامه نویسی احتیاج دارد. می بایست یک جدول بسازید تا روزها را در آن نشان دهید و یا سال و ماه را در بالای آن. باید پیدا می کردید که کدام روز، روز اول این ماه است و این ماه چند روز دارد. تمامی روزهای یک هفته را باید در یک ردیف جای دهید و خیلی کارهای دیگر که برای نمایش تقویم ماه جاری باید انجام دهید. قبل از بوجود آمدن فرم وب یک راه دیگر برای انجام این کار وجود داشت و آن استفاده از ActiveXاست که مشکلات خاص خود را دارد، مانند تفاوت در مرورگرها، لزوم استفاده از سیستم عامل ویندوز، لزوم نصب شدن ActiveX و غیره. فرم وب راحتی استفاده از ActiveX و همچنین خروجی به HTML را همزمان داراست.

در پروژه HelloWorld یک فرم وب جدید اضافه کنید. به صورت قرار دادی نام، WebForm3.aspxاست، همان نام را بپذیرید و این فرم وب را برای طراحی آماده کنید. در toolbox به دنبال Calendar بگردید و آن را بر روی صفحه قرار دهید. همین !! تمامی کارهایی را که با ASP کلاسیک می بایست انجام دهید اینجا با یک Drag & Dropانجام دادید!!

قبل از اینکه برنامه را اجرا کنیم ابتدا کمی تغییرات در شکل تقویم می دهیم تا زیبا تر شود! تغییرات زیر را در خواص تقویمی که اضافه کردید اعمال کنید:

خاصیت مقدار

ID myCal

BackColor A light yellow (#ffffc0)

BorderColor Black

BorderStyle Solid

BorderWidth 1px

DayHeaderStyle , BackColor Light orange (#ffc080)

DayHeaderStyle , BorderColor Dark red (#c00000)

DayHeaderStyle , BorderStyle Inset

DayHeaderStyle , BorderWidth 1px

DayHeaderStyle , Font , Bold True

DayNameFormat Short

Font , Name Verdana

Font , Size X-Small

NextPrevFormat ShortMonth

OtherMonthDayStyle , ForeColor Silver

SelectedDayStyle , BackColor Dark blue (#0000c0)

SelectedDayStyle , ForeColor White

فراموش نکنید که WebForm3.aspx را به فرم اول پروژه تبدیل کنید. تمام کاری که انجام دادید استفاده از یک کامپوننت از toolbox و تغییر تعدادی از خاصیت های آن برای زیبایی بیشتر است. این صفحه هم در Internet Explorerدیده می شود و هم در Netscape 4 به بعد.

کنترل های موجود در فرم های وب[1]

در دات نت به غیر از کنترلهایی که می توانند توسط برنامه نویس ایجاد شوند، سه نوع کنترل دیگر با نام های HTML Server Control ، ASP.NET Server Controlو Validation Control وجود دارد .

کنترل هایی که در یک فرم وب استفاده می شوند با آنهایی که در فرم های معمولی ویندوز مورد استفاده قرار می گیرند تفاوت دارد. چهار نوع کنترل برای استفاده در فرم وب وجود دارد:

- HTML Server Control
- ASP.NET Server Control
- Validation Control
- User Control

قبل از اینکه نگاهی به این کنترل ها داشته باشیم بهتر است مفهوم Server-Side Control را مشخص کنیم.

مفهومServer-Side Control

مانند ASP کلاسیک می توانید کدهای ASP را با <% و %> از HTML جدا کنید. با استفاده از این تگ ها شما باید تمامی اطلاعاتی را که کدهای HTML و دیگر کدهای ASP لازم دارند را به درستی تهیه کنید. برای مثال فرض کنید یک فرم با یک جعبه متن و یک کلید وجود دارد. همچنین فرض کنید وقتی کلید زده شود مقدار جعبه متن بدون تغییر دوباره نوشته شود. از nm برای نام جعبه متن استفاده کردیم. همانطور که می بینید شما به عنوان برنامه نویس باید متن مورد نظر را در جعبه متن قرار دهید. این کار را بوسیله استفاده از آبجکت Request و استفاده

[1] حامد بنایی

از خاصیت value از جعبه متن انجام دادید. وقتی برای اولین بار این صفحه دیده می شود جعبه متن خالی است ، اما هنگامی که کاربر نامی را تایپ کرد و بر بروی کلید کلیک کرد ، نام تایپ شده در جعبه متن آورده می شود.

در فرم های وب ، مایکروسافت مفهوم جدیدی را ایجاد کرده است که مسئول نگه داشتن مقادیر صفحه ها به شکل خودکار می باشد و دیگر احتیاجی به استفاده از کدهای اضافه برای این کار نیست. در فرم وب می توانید بگویید که کدام کنترل وقتی که کاربر کلید submit را فشار داد به صورت خودکار مقدار خود را نگه دارد. این کار بوسیله خاصیت runat="server" در کنترل ها و فرم ها اجرا می شود. متن زیر با استفاده از runat="server" و کمی تغییرات دیگر مشکل برنامه بالا را حل می کند:

```
<HTML>
<HEAD>
<title>Untitled</title>
</HEAD>
<BODY>
<Form Action="testform.aspx" runat="server">
What is your name ?
<asp:textbox name="nm"
value="" size="40"
MAXLength="40" runat="Server"/><br>
<input Type="submit" Name="cmd" Value=" Submit ">
</FORM>
</BODY>
</HTML>
```

کد بالا را با پسوند aspx ذخیره کنید تا ASP.NET آن را اجرا کند .بدون اینکه کد اضافی نوشته باشیم صفحه حالت قبلی خود را حفظ می کند. تغییراتی که این کد با مورد مشابه در ASP کلاسیک دارد در درجه اول در خود پسوند aspx است. تفاوت بعدی در تگ فرم است که خود از خاصیت server= استفاده می کند. این خاصیت به ASP.NETمی گوید که باید کارهای اضافه تری نسبت به یک فرم معمولی انجام دهد. این کار بوسیله یک فیلد مخفی که به فرم اضافه می شود انجام خواهد شد.

اگر کد HTML حاصل اجرای برنامه فوق را در مرورگر خود ببینید مشاهده خواهید کرد که در قسمت form یک ID و Name اضافه شده ، همچنین فیلد مخفی با نام __VIEWSTATE برای ذخیره مقدار کنترل روی فرم اضافه شده است.

HTML Server Control

HTML Server Controlاجزایی هستند که بوسیله خاصیت runat="server" متمایز می شوند. در ویژوال استودیو دات نت اینگونه کنترل ها در قسمت فرم وب در جعبه ابزار (Toolbox) قرار دارند. کنترل های معمولی HTML در بخش HTML در جعبه ابزار هستند. کنترل های HTML Server مشابه مدل کلاسیک خود هستند به غیر از اینکه خاصیت runat="server" دارند و می توان برای آنها در فرم وب برنامه نوشت.

HTML Server Controlکنترل هایی هستند که معمولاً استفاده زیادی در صفحه های HTML دارند مانند Form , Table , TextBox , ListBox , CheckBox و غیره. هر کدام از این کنترل ها که قبلاً تعریف شده اند دارای خواص عمومی و طبعاً خواص مخصوص به خود هستند.

در محیط ویژوال استودیو دات نت برای قرار دادن یک کنترل HTML معمولی بر روی فرم کافی است آن را از جعبه ابزار انتخاب کرده و بر روی فرم قرار دهید. در حالت عادی کنترل هایHTML معمولی حالت

"runat="serverندارند ، برای اینکه آنها دارای این خاصیت بشوند می توان از پنجره Design و با زدن کلید سمت راست موس بر روی آن کنترل و انتخاب Run As Server Control ، این کنترل را به کنترلِ سروری ارتقا داد.

کنترل های HTML در دات نت از کلاس System.Web.UI.HTMLControls مشتق می شوند. با تبدیل یک کنترل HTML معمولی به HTML Server Control می توانیم اعمال زیر را به راحتی برای کنترل جدید انجام دهیم:

•نوشتن کد برای رویدادهایی که بر روی فرم وب اتفاق می افتند در حالتی که این کدها در سرور اجرا می شوند ، مانند رویداد کلیک که برای کلید اتفاق می افتد.

•نوشتن کد برای رویدادهایی که بر روی فرم وب اتفاق می افتند در حالی که این کدها در client اجرا می شوند مانند حرکت موس.

•نگه داری حالت قبلی کنترل بدون نیاز به نوشتن کد اضافی.

HTML Server Controlها به این دلیل وجود دارند تا با برنامه های ASP کلاسیک سازگاری وجود داشته باشد. آنها کمک می کنند تا برنامه های ASP کلاسیک خیلی راحت به ASP.NET ارتقا یابند. در هر حال تمام کارهایی که با HTML Server Control می توان انجام داد را با ASP.NET Server Control نیز می توان انجام داد.

ASP.NET Server Controls

HTML Server Controlفقط در محدوده کنترل های معمولی HTML هستند و تقریباً هیچ خاصیت اضافی برای کنترل وضعیت خود در برنامه نویسی ندارند. اما ASP.NET Server Controls امکانات بسیار زیادی دارند ، نمونه آنها تقویم بود که قبلاً مثال آن دیده شد.

VB.NETدارای حدود بیست کنترل ASP.NET Server است که از حالت ساده ای مانند TextBox شروع می شود و تا حالتهای پیشرفته مثل DataGrid ادامه می یابد. در پشت صحنه ، این کنترل ها با تگ :asp> شروع می شوند. برای مثال وقتی در WebForm1.aspx یک Label بر روی فرم قرار دادیم و مقدار Text آن را به HelloWorldتغییر دادیم ، ویژوال استودیو بوسیله کد زیر آن را معرفی می کند:

```
<asp:label id="Message" text="Hello World" runat="server"/>
```

می توانید سورس صفحه را با زدن کلید سمت راست موس در هر جای آن و انتخاب View HTML Source ببینید. همچنین وقتی که از یک کلید بر روی فرم استفاده کردیم به جای اینکه از input type="Submit" استفاده شود کد زیر نوشته شد:

```
<asp:button text="Go!" runat="server"/>
```

توجه کنید که این کد یک کنترل معمولی HTML نمی سازد ، بلکه یک کنترل از نوع .ASP.NET Server در هنگام اجرا این کد با توجه به نوع مرورگر تبدیل به یک HTML ساده می شود.

Validation Controls

کنترل های اعتباری مانند دو دسته کنترلی که ذکر شد نیستند و حالت ویژوال آنها مانند قبلی ها نیست. بلکه برای تایید درستی مقدار موجود در کنترل های دیگر در سرور یا در Client به کار می روند. برای مثال فرض کنید که کاربر باید در یک جعبه متن حالت خاصی از تاریخ را بنویسد ، برای کنترل مقدار آن جعبه متن می توان از این دسته از کنترل ها استفاده کرد.

برای استفاده از این دسته کنترل ها باید کنترلی را که می خواهد مقدارش چک شود را از طریق خاصیت ControlToValidateانتخاب کنید. با این کنترل ها می توانید مقادیر را به شکل های زیر تایید کنید:

- ورود اطلاعات اجباری است
- شکل خاصی از اطلاعات باید تایپ شود
- اطلاعات وارد شده باید بین دو مقدار خاص باشد

ویژوال بیسیک دات نت کنترل های زیر را در بخش فرم وب دارد:

RequiredFieldValidator:اطمینان از اینکه کاربر یک جعبه متن را حتماً پر کرده باشد.

CompareValidator:مقایسه مقدار تایپ شده توسط کاربر با مقدار پیش فرض.

RangeValidator:اطمینان از اینکه مقداری که کاربر تایپ کرده است بین دو مقدار بیشتر و کمتر قرار دارد.

RegularExpressionValidator:اطمینان از اینکه مقدار تایپ شده با شکلی که برنامه نویس مد نظر داشته است مطابقت دارد.

CustomValidatorمقایسه مقداری که کاربر تایپ کرده است در برابر مقدار منطقی که برنامه نویس در نظر داشته.

بهترین راه برای دیدن قدرت این ابزار استفاده عملی از آنهاست.

یک فرم وب جدید به پروژه HelloWord اضافه کنید. یک جعبه متن بر روی آن قرار دهید و خاصیت ID آن را به txtName تغییر دهید. یک کنترل RequiredFieldValidator از جعبه ابزار بردارید و در کنار جعبه متن قرار دهید. سپس یک کلید زیر جعبه متن قرار داده و خاصیت Text آن را به Submit تغییر دهید.

برای تنظیم کردن کنترل تایید کننده ابتدا ID آن را برابر rfvTxtName قرار دهید. این نام را به این دلیل انتخاب می کنیم که از نام کنترل مشخص باشد که مربوط به کدام کنترل دیگر است. از خاصیت ControlToValidate نام txtName را انتخاب کنید. سپس خاصیت ErrorMessage را به Required Field. Please enter your name تغییر دهید. این متنی است که می خواهیم به کاربر نمایش داده شود.

WebForm4.aspxرا به عنوان فرم اول پروژه انتخاب کنید و سپس پروژه را اجرا کنید. مرورگر اینترنت باز می شود و یک صفحه با یک جعبه متن و کلید در آن وجود دارد. بدون اینکه چیزی درون جعبه متن تایپ کنید بر روی کلید کلیک کنید تا حالتی که کاربر فیلد لازم را پر نکرده است بوجود آید. همانطور که می بینید متنی که به عنوان پیغام خطا نوشته بودیم نمایش می دهد.

اگر کد HTML صفحه را در مرورگر ببینید مشاهده می کنید کدی به زبان جاوااسکریپت نوشته شده است تا عمل کنترل مقدار جعبه متن را انجام دهد. برنامه نویس احتیاج به فکر کردن درباره طرز کار این کد ندارد ، فقط به راحتی از کنترل اعتباری در فرم وب استفاده می کند.

User Controls

این دسته از کنترل ها آنهایی هستند که خود شما می نویسید. مانند ویژوال بیسیک کلاسیک که امکان ساختن اجزای ویژوال برای وب به برنامه نویس می داد.

رویدادها در فرم های وب[1]

بررسی رویدادها در فرم های وب و نحوه رسیدگی به آنها و نشان دادن عکس العمل مناسب به کاربر

در دنیای ویندوز رویدادهای در سه حالت امکان رخداد دارند. وقتی که کاربر از خود حرکتی نشان داده مانند حرکت موس، استفاده از صفحه کلید و غیر. نوع بعدی توسط خود سیستم اتفاق می افتد، مانند لود شدن یک صفحه و نوع سوم رویدادهایی است که بر اثر گذر زمان اتفاق می افتند. در دنیای وب به خاطر استفاده از پروتکل HTTP خیلی از این رویدادها متفاوت هستند. بعنوان مثال:

- وقتی که یک مرورگر یک صفحه را درخواست می کند.
- وقتی یک وب سرور کد یک صفحه را خط به خط اجرا می کند.
- خروجی عملیات سرور به مرورگر بوسیله HTML بازگشت داده می شود.
- در این مرحله آن صفحه دیگر در سرور وجود ندارد.
- کاربر عملیاتی بر روی صفحه انجام می دهد.
- اگر سرور احتیاج به عملیاتی در برابر کاربر دارد صفحه دوباره باید به سرور ارسال شود.
- و این چرخه همچنان ادامه می یابد.

فرم های وب امکان استفاده از رویدادها را برای کنترل هایی که بر روی فرم هستند به برنامه نویس می دهد. ولی این رویدادها در کامپیوتر کاربر اجرا نمی شوند، بلکه به سرور بازگشت داده شده و در آنجا کدهای مربوط به آن اجرا می شود .رویدادهایی که معمولاً استفاده می شود رویدادهایی از نوع کلیک هستند. بعنوان مثال رویدادی مانند حرکت موس را نمی توان با هر بار حرکت آن به سرور فرستاد و منتظر یک جواب بود.

[1] حامد بنایی

طول عمر یک فرم وب

یکی از مسائلی که برنامه نویسِ ویژوال بیسیک در راه استفاده از فرم وب با آن روبرو است طول عمر یک فرم وب است. در نظر بگیرید در ویژوال بیسیک ۶ وقتی یک فرم در نمایشگر نمایش داده می شد و دوباره حذف می شد پنج رویداد رخ می داد، Form_Initialize، Form_Load، Form_QueryUnload، Form_Unload و Form_Terminate.

در مورد فرم های وب، وقتی یک مرورگر درخواست یک صفحه را می کند، در مرحله اول فرم وب بارگذاری می شود. در مرحله بعدی رویدادهای آن بررسی می شود و در مرحله بعدی قبل از اینکه کد HTML به مرورگر فرستاده شود از حافظه حذف می شود. هر بار که یک صفحه توسط مرورگر درخواست شود این عملیات تکرار می شود. اجازه دهید نگاهی به هر کدام از مراحل فوق بیندازیم:

•**بارگذاری:** این مرحله شباهت زیادی به Form_Initialize و Form_Load در ویژوال بیسیک ۶ دارد. وقتی این مراحل انجام شد رویداد Page_Load اتفاق می افتد .این رویداد اولین مکانی است که برنامه نویس می تواند برای صفحه خود برنامه بنویسد .مثلاً اتصال به بانک اطلاعاتی و تنظیم کنترل ها را در این رویداد می توان نوشت.

•**بررسی رویدادهای دیگر:** اگر این اولین باری باشد که مرورگر این صفحه را درخواست می کند هیچ رویدادی برای اجرا شدن وجود ندارد. اما اگر قبلاً این صفحه بارگذاری شده باشد و قرار باشد رویدادی اتفاق بیافتد در این مرحله کدهای مربوطه اجرا می شود.

•**از بین رفتن:** این مرحله، مرحله پایانی طول عمر یک فرم وب در حافظه سرور است. این مرحله معادل Form_Unload و Form_Terminate در ویژوال بیسیک قدیمی است. در این رویداد است که باید اتصال ها به

بانک اطلاعاتی بسته شود و هر گونه شیٔای که در حافظه بار شده است دوباره پاک شود. در این مرحله، رویدادِ Page_Unload اتفاق می افتد.

انواع رویداد

انواع رویداد در فرم وب را می توان به اجزای زیر تقسیم بندی کرد:

- رویدادهای ذاتی
- رویدادهای طرف کاربر و رویدادهای طرف سرور
- رویدادهای Postback وnon-Postback
- رویدادهای حبابی
- رویدادهای Application وSession

رویدادهای ذاتی

این دسته رویدادهایی هستند که در اکثر کنترل های فرم وب دیده می شوند، مانند کلیک.

رویدادهای طرف Client و رویدادهای طرف سرور

کنترل های ASP.NET Server فقط دارای رویدادهایی هستند که در سرور اجرا می شود و در مقابل کنترل های HTML دارای رویدادهایی هستند که در نزد کاربر اجرا می شوند. رویدادهایی مانند کلیلک در سرور رسیدگی می شوند ولی رویدادهایی مانند MouseMove در نزد کاربر.

رویدادهای Postback وnon-Postback

رویدادهایی که در کنترل های ASP.NET اتفاق می افتد به سرور ارسال می شوند و در آنجا اجرا می شود. برای مثال رویداد OnChange برای یک جعبه متن در زمانی که کاربر متن را تغییر می دهد اتفاق نمی افتد بلکه منتظر می ماند تا صفحه به سرور ارسال شود، سپس این رویداد اتفاق می افتد .این روش در مقابل روشی که رویداد در نزد کاربر اتفاق بیافتد قرار دارد. هر یک از این روشها دارای معایب و مزایای مخصوص به خود هستند.

رویدادهای حبابی

کنترل هایی مانند DataGrid می توانند در خود کنترل های دیگری مانند ردیف داشته باشند و خود DataGridبه عنوان یک container برای کنترل کوچکتر باشد. این گونه کنترل های کوچکتر (فرزند کنترل اصلی) می توانند دارای رویداد باشند. رویدادهایی که در آنها اتفاق می افتد به container فرستاده شده و container آن رویداد را به وسیله رویدادهایی مانند ItemCommand به برنامه نویس اطلاع می دهد. به این نوع از رویدادها، رویدادهای حبابی می گویند.

رویدادهای Application و Session

برای سازگاری با ASP کلاسیک، در دات نت رویدادهای مربوط به Application و Session وجود دارند، از جمله،ApplicationStart ، ApplicationEnd، SessionStartو .SessionEnd این رویدادها برای یک صفحه خاص اتفاق نمی افتند بلکه برای کل پروژه هستند.

کار کردن با کنترل DataGrid[1]

آشنائی با کنترل DataGrid جهت نمایش اطلاعات استخراج شده از بانک اطلاعاتی

لینک دریافت کد: www.iranasp.net/download/pourshahid01.zip

این کنترل به ما امکان نمایش اطلاعات استخراج شده از بانک اطلاعاتی به صورت جدولی را می دهد و همچنین امکاناتی نظیر انتخاب، مرتب سازی، تصحیح و صفحه بندی اطلاعات را می دهد.

در حالت پیش فرض زمانی که خاصیت AutoGenerateColumns=True برقرار باشد DataGrid برای هر فیلد موجود در بانک اطلاعاتی ما یک BoundColumn می سازد در واقع به هر فیلد یک ستون اختصاص داده می شود و اطلاعات به ترتیب موجود در با نک اطلاعاتی نمایش داده می شوند. نام هر کدام از فیلدها به عنوان یک سر ستون قرار داده می شود و بقیه اطلاعات هم در زیر آن قرار می گیرند. مثال زیر نحوه استفاده از یک کنترل DataGridرا نشان می دهد.

در این مثال ابتدا فضانام مربوطه یعنی System.Data را فراخوانی نموده، سپس نوع زبان مورد نظر خود برای این صفحه را انتخاب می کنیم (در ASP.NET می توان با زبانهای مختلف از جمله VB.NET ، C#و JS.NET برنامه نویسی کرد).

در ادامه جدولی از اطلاعات را بصورت دستی با تابع CreateDataSource می سازیم که شامل پنج ستون با مقادیری از پنج نوع مختلف می باشد. پس از ساخت ستونها به ساخت اطلاعات می پردازیم و نه سطر از آن را ایجاد می کنیم (توسط حلقه موجود) و پس از ایجاد هر سطر آن را به جدول اضافه می کنیم و در زیربرنامه Page_Load با

[1] علیرضا پورشاهید

استفاده از دستورات موجود مقدار جدول تولید شده را به کنترل DataGrid خود اختصاص می دهیم و سپس وارد قسمت HTML صفحه می شویم در این قسمت نیز به بیان مشخصات کنترل گر خود می پردازیم.

```
<%@ Import Namespace ="System.Data" %>

<html>
<script language="VB" runat="server">

Function CreateDataSource() As ICollection

   Dim dt As DataTable
   Dim dr As DataRow
   Dim i As Integer

   'create a DataTable
   dt = New DataTable
   dt.Columns.Add(New DataColumn("IntegerValue", GetType(Integer)))
   dt.Columns.Add(New DataColumn("StringValue", GetType(String)))
   dt.Columns.Add(New DataColumn("DateTimeValue", GetType(Boolean)))
   dt.Columns.Add(New DataColumn("BoolValue", GetType(Integer)))
   dt.Columns.Add(New DataColumn("CurrencyValue", GetType Double)))

   'Make some rows and put some sample data in
   For i = 1 To 9
      dr = dt.NewRow()
      dr(0) = i
```

```
        dr(1) = "Item " + i.ToString()
        dr(2) = DateTime.Now.ToShortTimeString
        If (i Mod 2 <> 0) Then
            dr(3) = True
        Else
            dr(3) = False
        End If
        dr(4) = 1.23 * (i+1)
        'add the row to the datatable
        dt.Rows.Add(dr)
    Next
    'return a DataView to the DataTable
    CreateDataSource = New DataView(dt)

End Function

Sub Page_Load(sender As Object, e As EventArgs)
    MyDataGrid.DataSource = CreateDataSource
    MyDataGrid.DataBind
End Sub

</script>
<body>

<h3><font face="Verdana">Simple DataGrid Example</font></h3>

<form runat=server>
```

```
<ASP:DataGrid id="MyDataGrid" runat="server"
   BorderColor="black"
   BorderWidth="1"
   GridLines="Both"
   CellPadding="3"
   CellSpacing="0"
   Font-Name="Verdana"
   Font-Size="8pt"
   HeaderStyle-BackColor="#aaaadd"
/>

</form>
</body>
</html>
```

آشنائی با کنترل AdRotator[1]

با استفاده از کنترل AdRotator می توان آگهی های تبلیغاتی را در یک سایت به صورت دلخواه نمایش داد .

یکی از کنترلهای طرف سرور می باشد. این کنترل برای کار نیاز به Rotator Schedule دارد که یک فایل با فرمت XML است. این فایل را از طریق خصوصیت Advertisement File (Properties) به کنترل AdRotatorمعرفی می شود.

```
File :AdRotator.aspx
<%@ Page Language="VB" %>
<html>
<body>
<H1>AdRotator in ASP.NET</H1>

<asp:AdRotator id=MyAdRotator runat=server
AdvertisementFile="Advertisement.xml"
BorderWidth=2 />

</body>
</html>
```

[1] سید امین پوریه

عنصر ریشه (Root Element) در فایل XML مذکور Advertisements می باشد .این عنصر بسته به تعداد آگهی های تبلیغاتی دارای یک یا چند عنصر فرزند (Child Element) به نام Ad است. هر کدام از عناصر <Ad>شامل تعدادی توصیف کننده می باشد که در ذیل به آنها اشاره میگردد.

```
File : Advertisement.xml
<Advertisements>
<Ad>
<ImageUrl>images/banner1.gif</ImageUrl>
<NavigateUrl>http://www.Iranasp.net</NavigateUrl>
<AlternateText>ASP.NET</AlternateText>
<Keyword>ASP</Keyword>
<Impressions>5</Impressions>
</Ad>

<Ad>
<ImageUrl>images/banner2.gif</ImageUrl>
<NavigateUrl>http://www.Pishgaman.com</NavigateUrl>
<AlternateText>YAZDISP</AlternateText>
<Keyword>ISP</Keyword>
<Impressions>5</Impressions>
</Ad>
</Advertisements>
```

URL:Image URL ای است که گرافیکی که مربوط به این تبلیغ است و باید در آنجا یافت شود.

URL Navigate : URL ای که کاربر با کلیک کردن بر روی این تبلیغ به آنها متصل میشود.

Alternate Text : متنی که هنگامی که گرافیکها از دید کاربر پنهان است نمایش داده میشود.

Keyword :کلید واژه مرتبط با هر آگهی تبلیغاتی.

Impressions :مقدار اهمیت یک آگهی تبلیغاتی را نشان می دهد .

کنترل AdRotator با انتخاب یک تبلیغ از فایل XML (بصورت تصادفی) و پردازش آن، نتیجه را به صورت HTMLبه طرف Client می فرستد (لازم به ذکر است میتوان با تغییر مقدار Imperssions برای هر تبلیغ تعداد دفعات نمایش آن تبلیغ را کم یا زیاد نمود).

این کنترل توانایی فیلتر نمودن تبلیغات را نیز دارد بدین صورت که می توان با تنظیم خاصیت Keyword Filterفقط تبلیغاتی را مشاهده نمود که با آن کلید واژه همراه هستند.

به عنوان مثال کد زیر تنها تبلیغ مربوط http://www.iranasp.net را نشان میدهد.

```
File :AdRotator.aspx
<asp:AdRotator id=MyAdRotator runat=server
AdvertisementFile="Advertisement.xml"
KeywordFilter="ASP"
BorderWidth=2 />
```

منابع:

http://www.123aspx.com

http://www.aspnextgen.com

کنترلهای اعتبارسنجی در ASP.NET[1]

در این مقاله با نحوه اعتبارسنجی فرم های ورودی وب و نحوه بکارگیری کنترلهای مربوطه در ASP.NETآشنا می شویم .

در زمان ساخت برنامه های کاربردی مبتنی بر وب، اهمیت دارد که مطمئن شوید وارد کردن داده ها توسط کاربر به درستی انجام می شود. این کار از طریق اعتبار سنجی انجام می شود که یک مقدار را در برابر شرطی خاص بررسی و در صورت عدم تحقق آن شرط اعلام خطا می کند. در صفحات ASP.NET ، اعتبار سنجی از طریق استفاده از کنترلهای اعتبارسنجی سمت سرور تضمین می شود. این کنترلها مقدار یک کنترل ورودی دیگر را برای چند نوع شرط خطا بررسی می کنند و در صورت تحقق شرط، شرحی از مساله را نمایش می دهند.

در ASP.NET می توان کنترلهای اعتبارسنجی را تقریبا به همه کنترلهای ورودی صفحه که کنترلهای سرویس دهنده HTML و یا کنترلهای سرویس دهنده فرم وب هستند اضافه کرد. به یک کنترل ورودی خاص می توان بیش از یک نوع کنترل اعتبارسنجی تخصیص داد تا اعتبارسنجی با معیارهای مختلف امکان پذیر شود. زمانی که صفحه ASP.NET دارای کنترلهای اعتبارسنجی، اجرا می شود مقادیر کنترلهای ورودی تحت تاثیر کنترلهای اعتبارسنجی، همان طور که در کنترلهای اعتبارسنجی تعیین شده است، براساس منطق مربوطه پردازش می شوند. به محض سنجش اعتبار کل منطق، خصوصیات کنترلهای اعتبارسنجی، برحسب نتیجه اعتبارسنجی، Trueیا False می شوند.

بعد از پردازش همه کنترلهای اعتبارسنجی، خود صفحه براساس مقادیر خصوصیات اعتبارسنجی کنترلها یک خصوصیت تعیین می کند. اگر یکی از کنترلها در اعتبارسنجی شکست بخورد، خصوصیت صفحه به طور خودکار خصوصیت اعتبارسنجی را False می کند. بعد از بررسی خصوصیت صفحه و دریافت یک نتیجه اعتبارسنجی ناموفق،

[1] مدیریت سایت

معمولا صفحه به کاربر برگردانده می شود و پیامهای خطای نسبت داده شده به کنترلهای اعتبارسنجی ناموفق نمایش داده می شوند.

معمولا کنترل ورودی روی صفحه برای هر نوع شرط خطائی که قصد بررسی آن را دارید یک کنترل اعتبارسنجی دریافت می کند. مثلا اگر بخواهید مطمئن شوید که یک فیلد مقداری در خود دارد و آن مقدار کمتر از ۱۰ ولی بیشتر از ۵ است، از یک کنترل اعتبارسنجی برای بررسی وارد شدن مقدار و کنترلی دیگر برای بررسی آن مقدار در برابر مجموعه ای از مقادیر قابل قبول استفاده می کنید.

انواع کنترلهای اعتبارسنجی

ASP.NETشش نوع کنترل اعتبارسنجی دارد:

- RequiredFieldValidator
- RequlerExpressionValidator
- CompareValidator
- RangeValidator
- CustomVlidator
- ValidationSummary

همه این کنترلها از مجموعه خصوصیات و متدهای مشترکی برخوردارند .بیشتر آنها از کلاس BaseValidator و کلاس WebControl در فضانام System.UI.WebControls و کلاس Control در فضانام System.Web.UIبه ارث گرفته شده اند. استثنای این قاعده خصوصیت Text است که از کلاس Label به ارث می رسد.

این کنترلها دارای خصوصیات مشترکی می باشند که برخی از پرکاربردترین آنها به شرح زیر است.

ControlToValidate :عبارت است از مشخصه یا ID کنترلی مانند TextBox یا غیره که قرار است مقدار ورودی آن توسط این کنترل، اعتبارسنجی شود. مقداردهی این خصوصیت الزامی است.

Display :رفتار نمایشگر کنترل اعتبارسنجی را مشخص می کند. سه تنظیم آن عبارتند از:

None :در این حالت و در صورت عدم اعتبار مقدار کنترل مورد اعتبارسنجی، هیچ پیغام خطائی از سوی کنترل اعتبارسنجی برروی صفحه نمایش داده نمی شود مگر آنکه این کنترل با یک کنترل اعتبارسنجی خلاصه یا ValidationSummary بکار رفته باشد. نکته حائز اهمیت این است که این کنترل همچنان وظیفه اعتبارسنجی خود را انجام می دهد و تنها پیغام خطای خود را نمایش نمی دهد.

Static :در این حالت کنترل اعتبارسنجی قسمتی از صفحه ای را که در آن قرار دارد را جهت نمایش پیغام خطای خود از ابتدا اشغال می کند. به این ترتیب، زمانی که پیام خطا به نمایش در می آید، محل هیچ یک از عناصر موجود در صفحه تغییر داده نمی شود زیرا کنترل اعتبارسنجی از قبل بخشی از صفحه بوده است.

Dynamic: در این حالت برعکس حالتStatic ، کنترل اعتبارسنجی جائی را اشغال نمی کند و تنها زمانی که پیغام خطا را نمایش می دهد جائی را اشغال می کند. این مساله سبب می شود تا به احتمال زیاد محل برخی عناصر در صفحه تغییرکند.

Enabled :مشخص می کند که آیا کنترل اعتبارسنجی فعال است یا خیر.

EnabledClientScrip :مشخص می کند که آیا اعتبارسنجی سمت کاربر فعال است یا خیر.

ErrorMessage :متن پیام خطا را مشخص می کند.

ForeColor :رنگ متن پیام خطای اعتبارسنجی را مشخص می کند.

IsValid :مشخص می کند که آیا کنترل ارجاع شده اعتبارسنجی را گذرانده است یا خیر.

Visible :مشخص می کند که آیا کنترل اعتبارسنجی باید روی صفحه نشان داده شود.

(برگرفته از کتاب آموزشی ASP.NET از انتشارات کانون نشر علوم)

www.nashreoloom.com/showbooks.aspx@id=110

ViewState و نگهداری مقادیر فرمها - قسمت اول[1]

در ASP کلاسیک بعد از تائید یک فرم اطلاعات ورودی از بین می رود. برای رفع این عیب در ASP.NETاز ViewState استفاده می شود .

دکمه Back را فشار دهید... چه اتفاقی افتاده است؟ بله همه داده های ورودی پاک شده و شما مجبورید دوباره از ابتدا شروع کنید. این صفحه از ViewState پشتیبانی نمی کند!

در ASP.NET وقتی یک فرم تائید (ارسال) می شود اگر دوباره به صفحه حاوی فرم برگردید فرم را با تمام داده های ورودی مشاهده خواهید کرد. چطور!؟ به این دلیل که ASP.NET بطور خودکار و با کمک ViewState مقادیر ورودی شما را نگهداری می کند. درواقع ViewState وضعیت مقادیر موجود در یک صفحه را هنگام تائید و ارسال صفحه به سرویس دهنده (سرور)، نگهداری می کند. طرز کار بسیار ساده است: این عمل با قرار دادن یک فیلد مخفی (Hidden Field) انجام می شود. این فیلد به هر صفحه ای که شامل تگ <Form runat="server"> باشد، بطور خودکار و توسط ASP.NET افزوده خواهد شد .کدی که اضافه می شود شبیه به کد زیر می باشد:

```
<Form name="_ct10" method="post" action="page.aspx" id="_ct10">
<input type="hidden" name="__VIEWSTATE"
value="dDwtNTI0ODU5MDE1Ozs+ZBCF2ryjMpeVgUrY2eTj79HNl4Q" />
some code
</form>
```

نگهداری ViewState یکی از تنظیمات پیش فرض ASP.NET برای WebForm ها می باشد. اگر شما نیازی به ViewState ندارید دستور رهنمون یا Directive زیر را به بالای صفحه aspx خود اضافه کنید:

[1] علی رئیس دانا

```
<% @Page EnableViewState="false" %>
```

همچنین می توانید صفت EnableViewState را برای کنترل مورد نظر خود تنظیم نمائید.

```
[ASPX page]
<asp:label id="myLabel" runat="server" EnableViewState="false"></asp:label>

[codebehind]
myLabel.EnableViewState = false
```

در ادامه مثالی جهت آزمایش ViewState ارائه می شود:

```
<script runat="server">
Sub submit(sender As Object, e As EventArgs)
lbl1.Text="Hello " & txt1.Text & "!"
End Sub
</script>
<html>
<body>
<form runat="server">
Your name: <asp:TextBox id="txt1" runat="server" />
<asp:Button OnClick="submit" Text="Submit" runat="server" />
<p><asp:Label id="lbl1" runat="server" /></p>
</form>
</body>
</html>
```

ViewState و نگهداری مقادیر فرمها - قسمت دوم[1]

در ASP کلاسیک بعد از تائید یک فرم اطلاعات ورودی از بین می رود. برای رفع این عیب در ASP.NETاز ViewState استفاده می شود .

بطورکلی هر تایپی که اصطلاحا serializable بوده یا یک TypeConverter برای آن تعریف شده باشد، می تواند در ViewState ذخیره گردد. بهرحال توجه داشته باشید که تایپ هائی که تنها دارای قابلیت serializable باشند نسبت به آنهائی که دارای TypeConverter می باشند کندتر بوده و در ViewState حجم بیشتری از داده را نگهداری می کنند. بطورکلی می توان از کلیه تایپ های اولیه مانند عدد، حرف و غیره به همراه رشته های حرفی یا String و HashTable، ArrayList استفاده نمود.

شما بعنوان یک برنامه نویس باید دقت داشته باشید که کلیه داده های موجود در ViewState در هر بار رفت و برگشت صفحات یا اصطلاحا round-trip همراه صفحات منتقل می شوند و این مساله می تواند از جهت سرعت دریافت صفحات یا امنیت اطلاعات قابل توجه باشد. بعبارت دیگر اگر خواهان صفحات سبکتری هستید نباید اطلاعات زیادی را در ViewState نگهداری کنید یا اینکه به هیچ عنوان اطلاعات محرمانه مانند کلمه رمز، عبارت اتصال به بانک اطلاعاتی (Connection String) و غیره را در ViewState نگهداری نکنید.

جهت انجام این کار لازم است که بدانید علاوه بر یک فرم وب یا صفحه، هر یک از کنترلهای موجود در یک فرم وب نیز دارای یک صفت بنام EnableViewState می باشند که بصورت موردی می توانید تعیین کنید که آیا اطلاعات کنترل مورد نظر در ViewState ذخیره گردد یا خیر. یکی از بهترین کاربردهای این صفت می تواند در مورد کنترلهای Repeater ، DataListو یا DataGrid باشد که همواره حجم زیادی از اطلاعات را در ViewState نگهداری می کنند و در اکثر موارد ممکن است که به این اطلاعات نیازی نداشته باشیم. لذا با false قرار دادن مقدار

[1] مدیریت سایت

صفت EnableViewState برای این کنترلها می توانیم سرعت دریافت صفحات خود را بطور چشمگیری افزایش دهیم.

همچنین شما می توانید اطلاعات مورد نظر خود را بجای ذخیره در Session ، در ViewState قرار دهید. این کار بصورت زیر انجام می گردد:

```
[C#]
ViewState["MyData"] = "This is some data.";

[VB.NET]
ViewState("MyData") = "This is some data."
```

همانطور که می بینید لازم است نام یا کلیدی را برای اطلاعات خود در ViewState تعیین نمائید تا اطلاعات شما تحت آن نام در ViewState ذخیره و قابل بازیابی باشد. این اطلاعات را می توانید پس از رفت و برگشت صفحه بصورت زیر بازیابی نمائید:

```
[C#]
string strTmp = (string) ViewState["MyData"];

[VB.NET]
Dim strTmp As String = CStr(ViewState["MyData"])
```

در مثال زیر یک کلاس فرضی برای کنترل Label را مشاهده می کنید که برای ذخیره متن و اندازه فونت خود از ViewState استفاده نموده است.

```
[VB.NET]
' This control renders values stored in view state for Text and FontSize properties.
```

```
Imports System
Imports System.Web
Imports System.Web.UI

Namespace ViewStateControlSamples

  Public Class LabelVB : Inherits Control

  ' Add property values to view state with set;
  ' retrieve them from view state with get.
  Public Property [Text] As String
    Get
      Return CStr(ViewState("Text"))
    End Get
    Set
      ViewState("Text") = Value
    End Set
  End Property


  Public Property FontSize As Integer
    Get
      Return CInt(ViewState("FontSize"))
    End Get
    Set
      ViewState("FontSize") = Value
```

```
        End Set
    End Property

    Protected Overrides Sub Render(Output As HtmlTextWriter)
        Output.Write("<font size=" & Me.FontSize & ">" & Me.Text & "</font>")
    End Sub

    End Class

End Namespace
```

[C#]

```
// This control renders values stored in view state for Text and FontSize properties.
using System;
using System.Web;
using System.Web.UI;

namespace ViewStateControlSamples {

    public class Label: Control {

    // Add property values to view state with set;
    // retrieve them from view state with get.
    public String Text {
        get {
            return (String) ViewState["Text"];
            }
```

```
    set {
        ViewState["Text"] = value;
        }
    }


    public int FontSize {
        get {
            return (int) ViewState["FontSize"];
            }
        set {
            ViewState["FontSize"] = value;
            }
    }

    protected override void Render(HtmlTextWriter output) {
        output.Write("<font size=" + this.FontSize + ">" + this.Text + "</font>");
        }
    }
}
```

یک کنترل اعتبارسنجی دلخواه برای ListBox[1]

نحوه ساخت یک کنترل اعتبارسنجی نمونه و دلخواه جهت انتخاب تعداد معینی گزینه از یک ListBoxتوسط کاربر .

در پیاده سازی یک برنامه تحت وب با استفاده از ASP.NET نیاز داشتم که کاربر را مجبور کنم که از گزینه های موجود در یکListBox ، حداقل یک مورد را انتخاب کند. بدیهی است که این کار با استفاده از یک کنترل استاندارد و موجود در ASP.NET بنام RequiredFieldValidator براحتی قابل انجام است. کافی است صفت ControlToValidateمربوط به این کنترل اعتبارسنجی را مطابق نام ListBox خود قرار دهید و همین!

اما چند سوال: اگر خواستید کاربر را مجبور به انتخاب دو گزینه یا بیشتر نمائید چطور؟ یا اگر خواستید کنترل بیشتری بر ورودی صفحه تان داشته باشید یا بعبارت دیگر بعنوان یک برنامه نویس حرفه ایASP.NET ، مانور بیشتری در این زمینه داشته باشید؟ یا شاید اصلا نیاز داشتید که کنترل های اعتبارسنجی خاصی را مطابق نیازتان داشته باشید که تاکنون نوشته نشده باشند؟

در این مقاله کوتاه سعی شده است تا جواب همه سوالات فوق با ارائه یک کنترل اعتبارسنجی نمونه و آموزشی داده شود. این کنترل سعی دارد که اندکی بیشتر از رفتار کنترل اعتبارسنجی معروف RequiredFieldValidator را شبیه سازی نماید و شما با یاد گرفتن اصول بکار رفته در آن می توانید آن را توسعه داده یا کنترلهای اعتبارسنجی متنوع دیگری را به کتابخانه کنترلهای خود اضافه نمائید.

قبل از پرداختن به برنامه مذکور دقت داشته باشید که این کنترل دارای محدودیتها و ویژگیهای زیر می باشد:

[1] مدیریت سایت

•این کنترل از کلاس استاندارد BaseValidator مشتق (به ارث) گرفته شده است.

•در این برنامه فقط به اعتبارسنجی سمت کاربر یا اصطلاحا Client Side پرداخته شده است.

•جهت انجام اعتبارسنجی سمت سرور یا Server Side نیاز است که متد مربوطه را خودتان بنویسید و به آن اضافه کنید.

•با استفاده از مقداردهی صفت RequieredItem مربوط به این کنترل می توانید تعداد گزینه هایی که باید توسط کاربر حتما انتخاب شود را تعیین کنید(دقت داشته باشید که برای حالت بیش از یک گزینه لازم است که صفت SelectionMode مربوط به ListBox شما قبلا مطابق مقدار Multiple تنظیم شده باشد).

•می توانید از این کنترل احتمالا با اعمال تغییرات کوچکی برای سایر کنترلها مانند DropDownList ، CheckBoxListیا RadioButtonList نیز استفاده کنید.

خوب فکر کنم حرف زدن کافی است و زودتر برویم سراغ اصل مطلب:

```
[C#]
using System;
using System.Text;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace MyLib
{
   /// <summary>
   /// Summary description for RequieredListBoxValidator.
   /// </summary>
   public class RequieredListBoxValidator : BaseValidator
   {
```

```
protected int _RequieredItem = 1;

public int RequieredItem
{
   get
   {
      return this._RequieredItem;
   }
   set
   {
      this._RequieredItem = value;
   }
}

protected override bool ControlPropertiesValid()
{
   return true;
}

protected override bool EvaluateIsValid()
{
   return this.EvaluateHasItem();
}

protected bool EvaluateHasItem()
{
   ListBox _lstb = ( ( ListBox ) this.FindControl( this.ControlToValidate ) );
```

```
    return (_lstb.Items.Count != 0);
}

protected override void OnPreRender( EventArgs e )
{
    if ( this.EnableClientScript ) { this.ClientScript(); }
    base.OnPreRender( e );
}

protected void ClientScript()
{
    this.Attributes["evaluationfunction"] = "lb_verify";

    StringBuilder sb_Script = new StringBuilder();
    sb_Script.Append( "<script language=\"javascript\">" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "function lb_verify(val) {" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "var val = document.all[document.all[\"" );
    sb_Script.Append( this.ClientID );
    sb_Script.Append( "\"].controltovalidate];" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "if ( val != null ) {" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "if (val.tagName == \"SELECT\") {" );
    sb_Script.Append( "\r" );
```

```
    sb_Script.Append( "var selCount = 0;\r" );
    sb_Script.Append( "var i;\r" );
    sb_Script.Append( "for(i=0;i<val.options.length;i++)\r" );
    sb_Script.Append( "{\r" );
    sb_Script.Append( "if(val.options[i].selected)\r" );
    sb_Script.Append( "selCount++;\r" );
    sb_Script.Append( "}\r" );
    sb_Script.Append( "if ( selCount >= "
       + this._RequieredItem.ToString () + " ) {" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "return true;" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "}" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "}" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "return false;" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "}" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "}" );
    sb_Script.Append( "\r" );
    sb_Script.Append( "</script>" );
    this.Page.RegisterClientScriptBlock( "LSBScript", sb_Script.ToString() );
}
```

```
    }
}
```

کلاس فوق را در قالب یک Assembly (یا همان DLL خودمان اما از نوع دات نت) کامپایل کنید و DLL بدست آمده را در قسمت References برنامه استفاده کننده از این کنترل در VS.NET به برنامه خود الصاق نمائید. حال جهت استفاده از این کنترل مطابق زیر عمل کنید.

```
<%@ Page language="c#" Codebehind="WebForm1.aspx.cs"
AutoEventWireup="false" Inherits="WebApplication1.WebForm1" %>
<%@ Register TagPrefix="mylib" Namespace="MyLib" Assembly="MyLib" %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.0 Transitional//EN" >
<HTML>
  <HEAD>
    <title>WebForm1</title>
    <meta name="GENERATOR" Content="Microsoft Visual Studio 7.0">
    <meta name="CODE_LANGUAGE" Content="C#">
    <meta name="vs_defaultClientScript" content="JavaScript">
    <meta name="vs_targetSchema"
content="http://schemas.microsoft.com/intellisense/ie5">
  </HEAD>
  <body MS_POSITIONING="FlowLayout">
    <form id="WebForm1" method="post" runat="server">
        <asp:ListBox id="ListBox1" runat="server" SelectionMode="Multiple">
        <asp:ListItem Value="1">Item 1</asp:ListItem>
        <asp:ListItem Value="2">Item 2</asp:ListItem>
        <asp:ListItem Value="3">Item 3</asp:ListItem>
      </asp:ListBox>
```

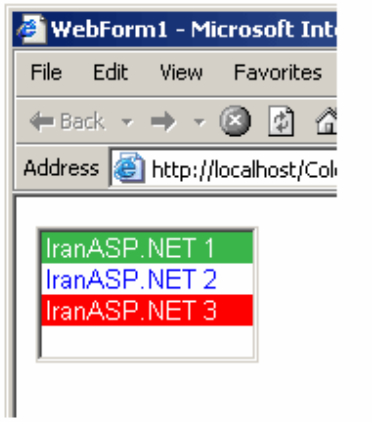
```
        <mylib:RequieredListBoxValidator id="RequieredListBoxValidator1"
runat="server" RequieredItem="2" ErrorMessage="You have to select at least 2 items
from list box!" ControlToValidate="ListBox1"></mylib:RequieredListBoxValidator>
        <br>
        <asp:Button id="Button1" runat="server" Text="Button"></asp:Button>
    </form>
  </body>
</HTML>
```

ListBox با گزینه های رنگی در ASP.NET[1]

در این مقاله نحوه نوشتن یک listbox دلخواه با قابلیت تعیین موارد Style برای گزینه های آن بصورت جداگانه ارائه شده است .

لینک دریافت کد: www.iranasp.net/download/webtech038.zip

در این مقاله آموزشی سعی خواهد شد تا نحوه ساخت و برنامه نویسی کنترلهای دلخواه در ASP.NET نشان داده شود. این کار با در نظر گرفتن طراحی یک ListBox خاص که دارای این قابلیت هست که بتوان رنگ متن و پس زمینه برخی از گزینه های آن را به دلخواه تعیین کرد، انجام می گردد.



کاربرد چنین کنترلی بسته به نیاز و سلیقه شما می تواند متفاوت باشد .مثلا ممکن است در برنامه ای بخواهید گزینه های مرتبط با هم را دسته بندی کرده و هر دسته را با رنگ پس زمینه خاصی نشان دهید. این مساله می تواند در تشخیص یا یافتن سریعتر گزینه های خاصی به کاربر کمک کند.

شاید این سوال برای شما پیش آید که چه احتیاجی به نوشتن یک کنترل ListBox جدید است و مثلا می توان مانند زیر عمل کرد:

[1] مدیریت سایت

```
ListItem item;
for(int i = 1; i<=10; i++)
{
   item = new ListItem ();
   item.Value = i.ToString ();
   item.Text = "Item" + i.ToString ();
   item.Attributes.Add ("Style", "background-color: #0000ff;color: #ffffff");
   myListBox.Items.Add (item);
}
```

اما نکته جالب اینجاست که کد برنامه فوق کار نمی کند! بعبارت دیگر صفت Style برای کلاس ListItem در زمان تولید کد HTML اصطلاحا render نمی شود و به آن ترتیب اثری داده نمی شود. شاید این مساله در نسخه های بعدی ASP.NET رفع شود یا اینکه اصولا این رفتار برای کنترل ListBox براساس طرح آن درست باشد.

بهرحال این مساله به هر شکلی که تعبیر شود برای ما بهانه خوبی است که اصولا نحوه نوشتن کنترلهای دلخواه و پیشرفته تر را بطور کلی یاد بگیریم و توان و مهارت خود را در این زمینه افزایش دهیم. دقت داشته باشید که یک برنامه نویس حرفه ای ASP.NET نمی تواند مصرف کننده صرف کنترلهای استاندارد یا از پیش تهیه شده توسط دیگران باشد، بلکه لازم است که او خود هم گاهی وقتها آستین را بالا زده و به این جمع بپیوندد و کنترلهای خاص مورد نیاز خود یا دیگران را تولید کند.

به جهت اینکه قطعه برنامه فوق کار کند و منظور ما را برآورده سازد لازم است که اندکی در کار ListBox استاندارد ASP.NET دخالت کرده و کد render آن را دستکاری کنیم. خیالتان راحت باشد ما هیچگونه دسترسی به سورس ListBox نداریم! تنها کافی است اندکی ابتکار شئ گرایی خود را به کار انداخته و این کار را با استفاده از مفهوم وراثت یا Inheritance انجام دهیم. (خدا را شکر که دات نت شئ گراست(!

بدین منظور یک کلاس جدید به نام StyledListBox تعریف می کنیم و این کلاس را از کلاس اصلی و استاندارد ListBox به ارث می گیریم. قطعه کد زیر چگونگی این کار را با استفاده از زبان C# به شما نشان می دهد:

```
[C#]
using System;
using System.Collections;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace ColoredListBox
{
    /// <summary>
    /// Summary description for StyledListBox.
    /// </summary>
    public class StyledListBox : System.Web.UI.WebControls.ListBox
    {
    }
}
```

می بینید که این کار چقدر ساده است ColoredListBox .نام فضانام یا نامکده ما است و شما می توانید نام دلخواه خود را استفاده کنید. نام کلاس جدید ما StyledListBox است که بصورت public تعریف شده است تا همگان بتوانند از آن استفاده کنند. پدر کلاس ما اینجا همان ListBox استاندارد ASP.NET یعنی System.Web.UI.WebControls.ListBoxمی باشد و ما با این کار کلیه ویژگیها و متدهای ListBox را مال خود کرده ایم.

بدنه این کلاس خالی است و دارائی های ListBox برای هدف ما کافی نیستند و قابلیتهای دیگری را نیز می خواهیم. بعبارت دیگر در دارائی های ListBox منظور نشده است که گزینه ها را برای ما رنگی کند و خود ما باید این ویژگی را به قدرت و توان ListBox اضافه کنیم.

برای این کار تنها کافیست متد RenderContents مربوط به ListBox را بازنویسی کنیم. در زیر شکل کامل این کلاس را می بینیم:

```
[C#]
using System;
using System.Collections;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace ColoredListBox
{
    /// <summary>
    /// Summary description for StyledListBox.
    /// </summary>
  public class StyledListBox: System.Web.UI.WebControls.ListBox
  {
    override protected void RenderContents(HtmlTextWriter writer)
    {
      for(int c=0;c<Items.Count;c++)
      {
        ListItem i = Items[c];
        writer.WriteBeginTag("option");
```

```
            if(i.Selected)
               writer.WriteAttribute("selected","selected",false);
            writer.WriteAttribute("value",i.Value,true);
            IEnumerator d = Items[c].Attributes.Keys.GetEnumerator();
            while(d.MoveNext())

writer.WriteAttribute(d.Current.ToString(),Items[c].Attributes[d.Current.ToString()]);
            writer.Write('>');
            System.Web.HttpUtility.HtmlEncode(i.Text,writer);
            writer.WriteEndTag("option");
            writer.WriteLine();
         }
      }
   }
}
```

از قیافه کد برنامه فوق نترسید! اگر اندکی با کلاس HtmlTextWriter بیشتر آشنا شوید خواهید دید که این متد چقدر ساده و در عین حال جالب است. عمده کاری که در این متد انجام شده است این است که علاوه بر صفتهای استاندارد گزینه های یک listbox مانند value وselected ، سعی کرده ایم تا کلیه صفات یاAttribute های آن را هم در کد HTML بنویسیم، کاری را که ظاهرا خودِ ListBox نکرده یا نخواسته یا حداقل ما آن را نمی دانیم! این کار با دو خط زیر انجام شده است:

```
[C#]
IEnumerator d = Items[c].Attributes.Keys.GetEnumerator();
while(d.MoveNext())
  writer.WriteAttribute(d.Current.ToString(),Items[c].Attributes[d.Current.ToString()]);
```

حالا وقت آن است که نحوه کاربرد این کلاس را هم بیان کنیم. جهت این کار کافی است همانند ListBox معمولی عمل کرده منتها listbox خود را از نوع ColoredListBox.StyledListBox تعریف کرده و رنگ دلخواه خود را برای گزینه های مختلف تعریف کنیم. به فایل codebehind زیر توجه فرمائید.

```
[C#]
using System;
using System.Collections;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Web;
using System.Web.SessionState;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.HtmlControls;

namespace ColoredListBox
{
    /// <summary>
    /// Summary description for WebForm1.
    /// </summary>
    public class WebForm1 : System.Web.UI.Page
    {
       protected StyledListBox ListBox1;

       private void Page_Load(object sender, System.EventArgs e)
```

```
        {
            // Put user code to initialize the page here
            ListItem item;

            item = new ListItem ("IranASP.NET 1", "1");
            item.Attributes.Add ("Style", "background-color:#39B54A; color:#ffffff");
            this.ListBox1.Items.Add (item);

            item = new ListItem ("IranASP.NET 2", "2");
            item.Attributes.Add ("Style", "color:#0000ff");
            this.ListBox1.Items.Add (item);

            item = new ListItem ("IranASP.NET 3", "3");
            item.Attributes.Add ("Style", "background-color:#ff0000; color:#ffffff");
            this.ListBox1.Items.Add (item);
        }
    }
}
```

تنها یک کار دیگر باقی مانده است و آن تغییر کد فایل aspx در مورد نحوه تعریف کنترل listbox مورد نظر است. ابتدا باید خط زیر را در بالای فایل مذکور قرار دهید تا بدین ترتیب کلاس StyledListBox ما برای این فایل قابل دسترس باشد:

```
<%@ Register TagPrefix="iranasp" Namespace="ColoredListBox"
Assembly="ColoredListBox" %>
```

در اینجا همانند تعریف یک کنترل کاربری یا User Control عمل کرده ایم. حال listbox خود را بصورت زیر تعریف می کنیم:

```
<iranasp:StyledListBox id="ListBox1" runat="server"></iranasp:StyledListBox>
```

و همین!

نکته :اگر دقت کرده باشید علاوه بر رنگ می توان هر نوع صفت موجود در خصیصه Style را برای گزینه های این listbox تعریف نمود.

ASP.NET و JavaScript[1]

این مقاله به بررسی ارتباط میان کدهای Server-Side و Client-Side پرداخته و نحوه برقراری این ارتباط را با ذکر چند مثال کاربردی شرح می دهد .

معمولاً با مشاهده عبارت ASP.NET مفهوم Server-Side (سمت سرور) در ذهن برنامه نویسان وب تداعی می شود. در مقابل، عبارت JavaScript احساس Client-Server (سمت کاربر) را برمی‌انگیزد و هر دو این تداعی کاملا درست است. اما شاید این تلقی ایجاد شود که بدلیل حوزه فعالیت هر یک پس این دو به یکدیگر اصولاً کاری ندارند! و هرکدام راه خود را می روند و ارتباطی بین این دو نیست. شاید می توانستیم بجای JavaScript بگوئیم JScriptاما از آنجا که JavaScript بین اهالی وب رایج تر و شناخته شده تر است از آن جهت JavaScript را برای بحث خود برگزیده ایم .اما بحث ما بطور کلی در مورد کد برنامه سمت سرور و کد برنامه سمت کاربر است.

اما جالب است که بدانید ارتباط بسیار خوبی میان مفهوم Server-Side و Client-Side در ASP.NET وجود دارد. بعبارت دیگر بدون برقراری این پیوند عملا ASP.NET بسیار ناتوان بود. باز هم از نقطه نظر دیگر اتفاقا همین ارتباط قوی سبب افزایش کارآئی، توان و قدرت ASP.NET شده است بگونه ای که ASP.NET در پیاده سازی مفاهیم و امکانات قدرتمند خود بخوبی از کد سمت کاربر بهره برده است. شاید ذکر مختصر یکی دو مثال موضوع را برای شما روشن تر کند.

برای اولین مثال می توان مساله ارسال فرم یا همان Submit را عنوان کرد. همه فرم ها در ASP.NET توسط یک تابع سمت کاربر به زبان JavaScript و بنام __doPostBack ارسال می شوند. همین الان در مرورگر خود کد HTMLهمین صفحه را ببینید و تابع مذکور را در اوایل این کد پیدا کنید. این تابع بصورت runtime هنگامی که این صفحه ابتدا بر روی سرور ایجاد یا اصطلاحا Render می شد به کد HTML این صفحه اضافه شده است.

[1] مدیریت سایت

برای مثال دوم می توان به موضوع اعتبارسنجی در ASP.NET و بهره گیری فراوان آن از کد سمت کاربر اشاره نمود. اگر با مقوله اعتبارسنجی در ASP.NET آشنا نیستید می توانید مقاله کنترلهای اعتبارسنجی در ASP.NET [1] را مطالعه بفرمائید. اگر در سایت خود از کنترلهای اعتبارسنجی استفاده کرده باشید حتما متوجه وجود یک شاخه بنام aspnet_clientشده اید که شما آن را نساخته اید. در شاخه های تودرتوی این شاخه نهایتا به سه عدد فایل محتوی کدهای JScript خواهید رسید که همان کدهای مورد نیاز ASP.NET برای انجام امور مختلف از جمله مساله اعتبارسنجی فرم ها در سمت کاربر می باشند.

شاید با ذکر این دو مثال ساده به اهمیت کدهای سمت کاربر و رابطه خوبی که ASP.NET با آن دارد پی برده باشید. در این مقاله قصد دارم شما را بیشتر با این ارتباط آشنا کنم و به شما نشان دهم که ASP.NET چه امکانات و ابزاری جهت برقراری و توسعه دلخواه این ارتباط دارد و اینکه شما چگونه می توانید کد برنامه سمت کاربر دلخواه خود را به صفحاتتان اضافه نموده و مانور بیشتری بدهید.

متدهای کلاس Page برای کدهای سمت کاربر

کلاس Page که در اصل همه فرم ها و صفحات ما در ASP.NET از آن مشتق می شوند دارای چند متد (Method)بسیار جالب و کاربردی در ارتباط با کدهای اسکریپت سمت کاربر می باشد.

RegisterClientScriptBlock:با استفاده از این متد می توان هر نوع کد JavaScript دلخواهی را جهت اجرا به صفحه مورد نظر افزود. این متد داری دو پارامتر از نوع رشته ای (string) بنام های key و script می باشد. پارامتر script همان کد JavaScript مورد نظر ماست و پارامتر key هم یک نام دلخواه ولی منحصر بفرد است که جهت شناسائی این اسکریپت تعیین می کنیم. اسکریپت مورد نظر شما دقیقاً پس از تگ آغازین <form> در کد

[1] www.iranasp.net/tutorial/article.aspx@articleid=107

HTMLصفحه قرار خواهد گرفت. دقت داشته باشید که اسکریپت مورد نظر شما باید کامل باشد چرا که ASP.NETهیچ پردازشی برروی آن انجام نخواهد داد و چیزی را به آن اضافه نخواهد کرد. بعبارت دیگر لازم است تگ های مورد نیاز یک اسکریپت JavaScrip بهمراه تگ های توضیح (comment) در HTML را مانند مثال زیر قبلا به کد خود افزوده باشید.

```
<script language="JavaScript">
<!--
کد اسکریپت مورد نظر شما
// -->
</script>
```

به مثال زیر توجه فرمائید.

```
[C#]
<html>
<head>
<script language="C#" runat="server">

public void Page_Load(Object sender, EventArgs e) {

   // Form the script that is to be registered at client side.
   String scriptString = "<script language=JavaScript> function DoClick() {";
   scriptString += "myForm.show.value='Welcome to Microsoft .NET'}<";
   scriptString += "/";
   scriptString += "script>";
```

```
   if(!this.IsClientScriptBlockRegistered("clientScript"))
      this.RegisterClientScriptBlock("clientScript", scriptString);
}

</script>
</head>
<body topmargin="20" leftmargin="10">
   <form id="myForm" runat="server">
      <input type="text" id="show" style="width=200"> <input type="button"
value="ClickMe" onclick="DoClick()">
   </form>
</body>
</html>
```

IsClientScriptBlockRegistered:همانگونه که در مثال قبل دید این متد تعیین می کند که آیا اسکریپتی با کلید شناسائی (key) مورد نظر قبلاً توسط متد RegisterClientScriptBlock به صفحه افزوده شده است یا نه. بکارگیری این متد قبل از فراخوانی متد RegisterClientScriptBlock سبب می شود تا از تکرار یک اسکریپت در یک صفحه جلوگیری شده و منابع موجود در سرور به هدر نرود.

RegisterStartupScript:دقیقا همانند RegisterClientScriptBlock می باشد با این تفاوت که اسکریپت شما دقیقا قبل از تگ پایانی </form> قرار خواهد گرفت. به مثال زیر توجه فرمائید.

```
[C#]
<html>
<head>
```

```
<script language="C#" runat="server">
public void Page_Load(Object sender, EventArgs e) {
   // Form the script to be registered at client side.
   String scriptString = "<script language=JavaScript> function DoClick() {";
   scriptString += "showMessage2.innerHTML='<h4>Welcome to Microsoft
.NET!</h4>'}";
   scriptString += "function Page_Load(){ showMessage1.innerHTML=";
   scriptString += "'<h4>RegisterStartupScript Example</h4>'}<";
   scriptString += "/";
   scriptString += "script>";

   if(!this.IsStartupScriptRegistered("Startup"))
      this.RegisterStartupScript("Startup", scriptString);
}


</script>
</head>
<body topmargin="20" leftmargin="10" onload="Page_Load()">
   <form id="myForm" runat="server">
      <span id="showMessage1"></span>
      <br>
      <input type="button" value="ClickMe" onclick="DoClick()">
      <br>
      <span id="showMessage2"></span>
   </form>
```

```
</body>
</html>
```

IsStartupScriptRegistered:همانگونه که در مثال قبل دید این متد تعیین می کند که آیا اسکریپتی با کلید شناسائی (key) مورد نظر قبلاً توسط متد RegisterStartupScript به صفحه افزوده شده است یا نه. بکارگیری این متد قبل از فراخوانی متد RegisterStartupScript سبب می شود تا از تکرار یک اسکریپت در یک صفحه جلوگیری شده و منابع موجود در سرور به هدر نرود.

GetPostBackClientEvent:با فراخوانی این متد می توان به نام تابع سمت کاربری که سبب ارسال (Submit)فرم شده است دست یافت.

GetPostBackEventReference:همانند متد GetPostBackClientEvent با فراخوانی این متد می توان به نام تابع سمت کاربری که سبب ارسال (Submit) فرم شده است دست یافت.

GetPostBackClientHyperlink:با فراخوانی این متد می توان به نام تابع سمت کاربری که سبب ارسال (Submit) فرم شده است و در حالیکه عبارت javascript: به ابتدای آن افزوده شده است، دست یافت.

RegisterOnSubmitStatement:با استفاده از این متد می توان به رویداد OnSubmit مربوط به فرم موجود در صفحه دسترسی داشت و کد مورد نظر خود را به آن افزود. به مثال زیر توجه فرمائید.

```
[C#]
void Page_Load(Object sender, EventArgs e)
{
   String scriptString = "<script language=JavaScript> function doClick() {";
   scriptString += "document.write('<h4>' + myForm.myHiddenField.value+
'</h4>');}<";
```

```
    scriptString += "/" + "script>";

    RegisterHiddenField("myHiddenField", "Welcome to Microsoft .NET!");

    RegisterOnSubmitStatement("submit", "document.write('<h4>Submit button
clicked.</h4>')");

    RegisterStartupScript("startup", scriptString);
}
```

یک کلاس کوچک و ساده ولی کاربردی

در ادامه کلاس نمونه ای ارائه شده است که ممکن است برای برنامه های شما مفید باشد. متدهای موجود در این کلاس کارهای خیلی ساده ای را انجام می دهند اما استفاده از آنها سبب راحتی کار برنامه نویسی، مدیریت آسان برنامه و افزایش قابلیت خواندن کد برنامه می شود .توجه داشته باشید که متدهای این کلاس همگی بصورت static تعریف شده اند و برای استفاده از آنها نیازی به تعریف و ایجاد آبجکت مربوطه ندارید (نیازی به استفاده از عملگر new و تعریف متغیر از این نوع کلاس ندارید). در عوض باید نام نامکده (namespace) و نام کلاس را قبل از نام متد بیاورید (Common.PageUtil.MakeJavaScriptBlock). شما هم می توانید متدهای کاربردی دلخواه خود را به این کلاس نمونه اضافه نمائید.

```
using System;
using System.Web.UI;
using System.Text;
```

```
using System.Web.UI.WebControls;

namespace Common
{
   /// <summary>
   /// Utility class for ASP.NET pages
   ///
   /// Be sure to notice that this code is provided as a technology sample
   /// and 'as is' and no warranties are made by the author.
   ///
   /// </summary>
   public class PageUtil
   {
      // This static methods helps you build your JavaScript blocks easily
      public static string MakeJavaScriptBlock(string strJavascript)
      {
         string sScript;
         sScript = "\n<script language=\"javascript\">\n";
         sScript += "<!--\n";
         sScript += strJavascript;
         sScript += "// -->\n";
         sScript += "</script>\n";
         return sScript;
      }

      // This staic method is used in popup screens
      // to close popup after doing some thing
```

```
    public static void CloseCurrent(System.Web.UI.Page page)
    {
        string sScript = "window.close();";
        sScript = Common.PageUtil.MakeJavaScriptBlock (sScript);
        page.RegisterStartupScript ("CloseCurrent", sScript);
    }

    public static void ShowAlert(string msg, System.Web.UI.Page page)
    {
        if (page.Request.Browser.JavaScript == true)
        {
            string sScript = "alert(\"" + msg + "\");\n";
            sScript = Common.PageUtil.MakeJavaScriptBlock (sScript);
            page.RegisterStartupScript ("ShowAlertScript", sScript);
        }
    }
  }
}
```

آشنایی با کنترل DataList[1]

نمایش داده های یک جدول از بانک اطلاعاتی در یک DataList

در این مقاله کنترل DataList را بررسی خواهیم کرد. پیشنهاد میکنم برای خواندن و اجرای کد برنامه حتما از ویژوال استودیو استفاده کنید.

کنترل DataList کنترلی برای نمایش داده ها با هر فرمتی میباشد. میتوانیم خاصیت DataSource این کنترل را یک Table از DataSet انتخاب کنیم و بعد با متد DataBind داده ها را از روی DataBase نمایش دهیم.

برای فهم بهتر این موضوع نمونه برنامه زیر را اجرا کنید:

```
[Code Behind - VB.NET]

Dim Conn As New OleDbConnection
Dim Comm As New OleDbCommand
Dim ds As New DataSet
Dim adp As New OleDbDataAdapter

Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
   Conn.ConnectionString = "Your ConnectionString"
   Comm.CommandText = "Select Top 10 * from TableName"
   Comm.Connection = Conn
   adp.SelectCommand = Comm
```

[1] میلاد کرباسی زاده

```
    adp.Fill(ds, "Blog")
    Conn.Close()
    DataList1.DataSource = ds.Tables("Blog").DefaultView
    DataList1.DataBind()
End Sub

[HTML Code]

<asp:DataList Width="100%" id="DataList1" runat="server">
    <ItemTemplate>
        <font color="Orange"><%# databinder.Eval(container.dataitem,"RowName")
%></font>
    </ItemTemplate>
</asp:DataList>
```

در کد HTML به جای RowName اسم فیلدی را که مایلید قرار دهید. در تگ <ItemTemplate> میتوانید فرمت نمایش داده ها را مشخص کنید. تگ های <HeaderTemplate> و <FooterTemplate> نیز برای فرمت عنوان و پایان لیست مورد استفاده قرار میگیرند. آرگومان دوم متد Eval نیز نام ستون مورد نظر را میگیرد.

Happy DataListing!

انواع روشهای انتقال مقادیر بین فرمهای وب ASP.NET[1]

در این مقاله انواع روشهای مختلف موجود برای ارسال مقادیر بین فرمهای وب ASP.NET بررسی می‌شوند .

مقدمه

فرمهای وبASP.NET ، مدل برنامه‌نویسی رویدادگرای شگرفی را برای توسعه‌گران فراهم می‌کنند. این موضوع طراحی سرتاسر برنامه کاربردی شما را ساده می‌کند ولی مسائل و مشکلات خاص خود را نشان می‌دهد. برای مثال، در ASP کلاسیک شما به آسانی می‌توانید مقادیر را با استفاده از POST از یک صفحه ASP به صفحه‌ای دیگر ارسال نمایید. اما اگر می‌خواهید در مدل فرمهای وب (یا همان مدل برنامه‌نویسی ASP.NET) برنامه‌نویسی کنید، همان چیز در ASP.NET ممکن نمی‌باشد. اما راههایی برای غلبه بر این وضعیت وجود دارند که می‌توانند مورد استفاده قرار بگیرند .در این مقاله موضوعات زیر را مورد بررسی قرار خواهیم داد:

- چگونگی ارسال مقادیر با استفاده ازQuerystring
- چگونگی استفاده از متغیرهای Session برای ارسال مقادیر
- چگونگی استفاده از متد Server.Transfer برای ارسال مقادیر

استفاده از Querystring

Querystringیک مکانیسم قدیمی برای ارسال مقادیر در بین صفحات است. مزیت اصلی این متد سادگی آن است. اما عیب آن این است که پس از ارسال، مقادیر در نوار آدرس مرورگر قابل مشاهده می‌باشند و نمی‌توان آبجکت‌ها

[1] احسان ولیزاده

را از این طریق ارسال کرد. این متد مناسب‌ترین راه برای ارسال تعداد کمی از مقادیری است که نیازی به محافظت از دید دیگران ندارند. برای اعمال کردن این متد مراحل زیر را انجام دهید:

- یک فرم وب با کنترل‌هایش را درست کنید
- یک کنترل دکمه‌ای Button یا LinkButton برای ارسال فرم به سرور بر روی فرم قرار دهید
- در رویداد کلیک دکمه یک متغیر از نوع String تعریف کنید که URL را برای فرم دیگر که مقادیر قرار است به آنجا ارسال شوند، نگه می‌دارد.
- مقادیر کنترل‌ها را در قالب پارامترهای QueryString در متغیر از نوع String قرار دهید
- از متد Response.Redirect که از متغیر String تعریف شده استفاده می‌کند برای هدایت کاربر به صفحه دیگر استفاده نمایید

قطعه کد زیر چگونگی انجام این مراحل را نشان می‌دهد:

فرم وب منبع

```
private void Button1_Click
(object sender, System.EventArgs e)
{
   string url;
   url="anotherwebform.aspx?name=" +
   TextBox1.Text + "&email=" +
   TextBox2.Text;
```

```
   Response.Redirect(url);
}
```

فرم وب مقصد

```
private void Page_Load
(object sender, System.EventArgs e)
{
   Label1.Text=Request.QueryString["name"];
   Label2.Text=Request.QueryString["email"];
}
```

استفاده از متغیرهای Session

در این روش باید مقادیر کنترل‌ها را در متغیرهای Session ذخیره کنیم و در فرم وب دیگری به آنها دسترسی داشته باشیم. همانطور که می‌دانید ذخیره داده‌های زیاد در Session ممکن است اختلالاتی را در سرور بوجود آورد، بنابراین باید از این متد بدرستی استفاده شود .البته هر وقت که خواستید می‌توانید متغیرهای Session را از بین ببرید. مراحل اصلی برای استفاده از این متد به ترتیب زیر می‌باشد:

- یک فرم وب با کنترل‌هایش را درست کنید
- یک کنترل دکمه‌ای Button یا LinkButton برای ارسال فرم به سرور بر روی فرم قرار دهید
- در رویداد کلیک دکمه، متغیرهای Session را تعریف کرده و مقادیر کنترل‌ها را در آنها قرار دهید
- کاربر را با استفاده از Server.Transfer به صفحه‌ای دیگر هدایت کنید

•در فرم وب دیگر متغیرهای Session را دریافت کرده و پس از دریافت اگر لازم باشد آنها را پاک کنید.

کد زیر این مراحل را در عمل نشان می‌دهد:

فرم وب منبع

```
private void Button1_Click
(object sender, System.EventArgs e)
{
   //textbox1 and textbox2 are webform
   //controls
   Session["name"]=TextBox1.Text;
   Session["email"]=TextBox2.Text;
   Server.Transfer("anotherwebform.aspx");
}
```

فرم وب مقصد

```
private void Page_Load
(object sender, System.EventArgs e)
{
   Label1.Text=Session["name"].ToString();
   Label2.Text=Session["email"].ToString();
   Session.Remove("name");
   Session.Remove("email");
}
```

استفاده از Server.Transfer

این روش متدی پیچیده ولی روش ماهرانه‌ای برای ارسال مقادیر بین صفحات است. در اینجا مقادیری را که می‌خواهید در صفحات دیگر به آنها دسترسی داشته باشید به عنوان خصوصیات کلاس صفحه بیان می‌کنید. در کل این متد واضحتر و شئ‌گراتر از متدهای قبلی است. مراحل زیر را برای استفاده از این متد بترتیب دنبال کنید.

- یک فرم وب با کنترل‌هایش را درست کنید
- رویدادهای خصوصیت Get که مقادیر کنترل‌ها را برخواهند گرداند را تعریف کنید
- یک کنترل دکمه‌ای Button یا LinkButton برای ارسال فرم به سرور بر روی فرم قرار دهید
- در رویداد کلیک دکمه متد Server.Transfer که اجرای برنامه را به فرم تعیین شده انتقال می‌دهد فراخوانی کنید
- در فرم دوم شما می‌توانید با استفاده از خصوصیت Context.Handler به یک نمونه از فرم اول دسترسی داشته باشید. سپس می‌توانید از خصوصیات Get که برای دسترسی به مقادیر کنترل‌ها ایجاد کرده‌ایم استفاده کنید.

کد زیر برای اجرای یک نمونه از مراحل بالا تدارک دیده شده است:

فرم وب منبع

خصوصیات زیر را به فرم وب اضافه کنید:

```
public string Name
{
   get
   {
      return TextBox1.Text;
```

```
    }
}

public string EMail
{
    get
    {
        return TextBox2.Text;
    }
}
```

حال Server.Transfer را فراخوانی کنید.

```
private void Button1_Click
(object sender, System.EventArgs e)
{
    Server.Transfer("anotherwebform.aspx");
}
```

فرم وب مقصد

```
private void Page_Load
(object sender, System.EventArgs e)
{
    //create instance of source web form
    WebForm1 wf1;
    //get reference to current handler instance
    wf1=(WebForm1)Context.Handler;
```

```
    Label1.Text=wf1.Name;
    Label2.Text=wf1.EMail;
}
```

یک مقاله مرتبط با این مقاله را می‌توانید در آدرس زیر مشاهده فرمایید:

http://www.iranasp.net/Articles/ShowArticle.aspx?articleid=102

نمایش تصاویر در DataGrid[1]

آشنایی با کنترل DataGrid و قابلیتهای آن با استفاده از مثالی جهت نمایش تصاویر در آن

لینک دریافت برنامه:www.iranasp.net/download/evalizadeh003.zip

مقدمه

کنترل DataGrid پیچیده ترین کنترلی است که در .NET Framework وجود دارد. با استفاده از این کنترل امکان فرمت دهی و نمایش، مرتب‌سازی و صفحه بندی رکوردهای جداول بانک اطلاعاتی فراهم می‌شود. در اینجا می‌خواهیم با استفاده از قابلیتهای این کنترل روشی را برای نشان دادن تصاویر در آن بررسی کنیم .هر کدام از این تصاویر قرار است در رکورد مربوط به خود در کنترل DataGrid نمایش داده شوند. هر رکورد داده‌های دیگری را نیز شامل می‌شود که همه آنها در جدول products واقع در پایگاه داده products.mdb قرار دارند. هر رکورد شامل ستونی به نام ImageName (از نوع داده Text) می‌باشد که نام تصویر مربوط به رکورد را شامل می‌شود.

تعریف ستون برای کنترلDataGrid

وقتی که یک منبع داده مانند یک DataSet را به کنترل DataGrid مقید (Bind) می‌کنیم، خود کنترل همه کارها را انجام می‌دهد و همه رکوردها را به طور خودکار نمایش می‌دهد. ولی اگر بخواهیم ستونهای کنترل DataGridرا تحت کنترل خود درآورده و فقط ستونهای دلخواه را نمایش دهیم باید خودمان برای کنترل DataGridستون تعریف کنیم. قبل از اینکه بتوانیم ستونی را برای DataGrid تعریف کنیم، لازم است که خصوصیت AutoGenerateColumns آنرا برابر False قرار دهیم. پس از انجام این کار می‌توانیم شروع به تعریف ستون برای کنترل DataGrid نماییم. کنترل DataGrid پنج نوع ستون را می‌شناسد که عبارتند از:

[1] احسان ولیزاده

BoundColumn :نوع پیش فرض ستونها که رکوردها را نمایش می‌دهد.

HyperLinkColumn :که رکوردها را به صورت یک Hyperlink نمایش می‌دهد.

TemplateColumn :که رکوردها را با استفاده از یک الگوی مشخص نمایش می‌دهد.

ButtonColumn :که کنترلهایی از نوع Button را نمایش می‌دهد.

EditCommandButton :که فرامین ویرایش همانندEdit ، Updateو Cancel را نمایش می‌هد.

در نمونه کدی که همراه مقاله میتوانید آنرا دریافت کنید، کنترل DataGrid را به صورت زیر تعریف کرده ایم:

```
<asp:Datagrid id="dg" runat="server" AutoGenerateColumns="False"
BorderColor="Black" Font-Size="10" Font-Name="Arial" BackColor="#C6C3FF"
Headerstyle-Font-Size="12" Headerstyle-Font-Bold="True" Headerstyle-Font-
Name="Arial" Headerstyle-Forecolor="#0F0F0F" Headerstyle-BackColor="#8482C6"
cellspacing="0" cellpadding="0" GridLines="Both">
<columns>
  <asp:BoundColumn HeaderStyle-HorizontalAlign="Center" DataField="ProdName"
HeaderText="ProdName"></asp:BoundColumn>
  <asp:TemplateColumn HeaderStyle-HorizontalAlign="Center">
    <HeaderTemplate>
      Images
    </HeaderTemplate>
    <ItemTemplate>
      <div align="center">
        <img src='/prodimages/<%# Container.DataItem("ImageName") %>'
```

```
border="0" />
        </div>
      </ItemTemplate>
   </asp:TemplateColumn>
   <asp:BoundColumn DataField="Comments"
HeaderText="Comments"></asp:BoundColumn>
</columns>
</asp:Datagrid>
```

در کدهای مربوط به کنترل DataGrid سه ستون برای آن تعریف شده است .ستون اول از نوع BoundColumn، ستون دوم از نوع TemplateColumn و ستون سوم هم دوباره از نوع BoundColumn تعریف شده‌اند. در ستونهایی که از نوع BoundColumn می‌باشند، انتساب نام فیلد جدول بانک اطلاعاتی به خصوصیت DataField ستون، برای نمایش دادن داده‌های آن فیلد در ستون DataGrid کافی می‌باشد. کدهای زیر مربوط به تعریف ستون اول این DataGrid می‌باشد.

```
<asp:BoundColumn HeaderStyle-HorizontalAlign="Center"
DataField="ProdName" HeaderText="ProdName"></asp:BoundColumn>
```

تعریف ستون TemplateColumn برای کنترل DataGrid

وقتی که می‌خواهیم اختیارات بیشتری را در تعیین فرمت یک ستون داشته باشیم ستونی از نوع TemplateColumnرا به تعریف می‌کنیم. ستونی که از نوع TemplateColumn برای کنترل DataGrid تعریف می‌شود، می‌تواند الگوهای زیر را داشته باشد:

HeaderTemplate :تعیین کنندهٔ فرمت متنی است که در بالای ستون نمایش داده می‌شود .
ItemTemplate :تعیین کنندهٔ فرمت آیتم‌های اصلی ستون می‌باشد.

•تعیین کنندهٔ فرمت آیتمی است که به قصد ویرایش انتخاب شده است. :EditItemTemplate
•تعیین کنندهٔ فرمت آیتمی است که پایین ستون نمایش داده می‌شود. :FooterTemplate

در کد مربوط به تعریف کنترل DataGrid از دو الگوی HeaderTemplate و ItemTemplate استفاده شده است. در داخل HeaderTemplate کلمه Images نوشته شده است که موقع اجرا در بالای ستون نمایش داده می‌شود. کار اصلی برای نمایش تصاویر در داخل الگوی ItemTemplate انجام شده است. در اینجا با کدهای HTMLی که در زیر دوباره آنها را مشاهده می‌کنید توانسته‌ایم تصاویر مربوط به رکوردها را نمایش دهیم.

```
<div align="center">
<img src='/prodimages/<%# Container.DataItem("ImageName") %>' border="0" />
</div>
```

در اینجا با استفاده از تگ <img> تصاویر را از داخل پوشه prodimages نمایش داده‌ایم. نام تصاویر با استفاده از عبارت <%# Container.DataItem("ImageName") %> تعیین می‌شود که این عبارت مقدار ستون ImageName برای هر رکورد را باز می‌گرداند.

تمام کد مربوط به صفحه و همچنین پایگاه داده products.mdb را می‌توانید در بالای همین بخش از ***لینک کد*** دریافت نمایید.

افزودن یک ستون انتخاب به DataGrid[1]

حتما دیده‌اید که مثلا صندوق های پستی Hotmail و Yahoo به کاربران اجازه میدهند تا در لیست ایمیل های خود یک یا چند ایمیل را انتخاب کرده و عمل واحدی را روی آن انجام دهند. انتخاب هر ایمیل با تیک زدن یک checkbox کنار آن انجام می شود. شاید بخواهید چنین کاری را در ASP.NET هم انجام دهید. ولی DataGrid موجود در ASP.NET این امکان را بطور خودکار به شما نمی دهد و عمده‌ی کار را خود شما باید انجام دهید. در این مقاله مراحل لازم برای افزودن ستون انتخاب به DataGridتوضیح داده شده است .

لینک دریافت برنامه: www.iranasp.net/download/szhooshmand001.zip

صورت مساله

فرض کنید Datagrid ای داریم که اطلاعات دانشجویان یک کلاس را در خود دارد (دقیقاً نام، نام خانوادگی و IDآنها). میخواهیم یک ستون انتخاب به Datagrid خود اضافه کنیم، در داخل این ستون یک checkbox قرار دارد و با انتخاب آن، سطر انتخاب میشود .

راه حل

برای ایجاد لیست دانشجویان تابع زیر را به کد خود اضافه کنید. این تابع یک DataTable ساخته که دو ستون firstnameو lastname از نوع String و یک ستون ID از نوع Integer دارد که شماره شناسایی منحصر بفرد دانشجوست DataTable .حاوی اطلاعات پنج دانشجو می باشد.

[1] سلمان هوشمند

```
Private Function generateDataSource() As DataTable
    Dim dt As New DataTable
    dt.Columns.Add(New DataColumn("FirstName", Type.GetType("System.String")))
    dt.Columns.Add(New DataColumn("LastName", Type.GetType("System.String")))
    dt.Columns.Add(New DataColumn("ID", Type.GetType("System.Int32")))
    Dim dr As DataRow
    dr = dt.NewRow
    dr.Item(0) = "Salman"
    dr.Item(1) = "Zoroofi"
    dr.Item(2) = "1"
    dt.Rows.Add(dr)
    dr = dt.NewRow
    dr.Item(0) = "Ali"
    dr.Item(1) = "Mohammadi"
    dr.Item(2) = "2"
    dt.Rows.Add(dr)
    dr = dt.NewRow
    dr.Item(0) = "Reza"
    dr.Item(1) = "Maghsoodi"
    dr.Item(2) = "3"
    dt.Rows.Add(dr)
    dr = dt.NewRow
    dr.Item(0) = "Maryam"
    dr.Item(1) = "Moosavi"
    dr.Item(2) = "4"
    dt.Rows.Add(dr)
    dr = dt.NewRow
```

```
    dr.Item(0) = "Ehsan"
    dr.Item(1) = "Mojtahedi"
    dr.Item(2) = "5"
    dt.Rows.Add(dr)
    Return dt
End function
```

برای ارجاع به اینِ DataTable یک متغیر عمومی به نام dt از نوع DataTable تعریف کنید (یعنی آنرا داخل بدنه‌ی کلاس و خارج از بدنه‌ی هر متد تعریف کنید مثلاً درست قبل ازPage_load).

```
Private dt As DataTable
```

متد Page_Load خود را به صورت زیر تغییر دهید. این متد به dt مقدار اولیه داده و بعضی از خصوصیات Datagridرا تنظیم میکند.

```
Private Sub Page_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    DataGrid1.AllowPaging = True
    DataGrid1.PageSize = 3
    If Not IsPostBack Then
        dt = generateDataSource()
        viewstate.Add("dt", dt)
        DataGrid1.DataSource = dt
        DataGrid1.DataBind()
    Else
        dt = CType(viewstate("dt"), DataTable)
        DataGrid1.DataSource = dt
```

```
    End If
End Sub
```

افزودن یک ستون انتخاب به Datagrid

یک Datagrid روی فرم قرار دهید، روی آن کلیک راست کرده و Property Builder را انتخاب کنید. در قسمت Columns زیر عنوانAvailable Columns ، گزینه‌ی Template Column را انتخاب کرده و روی دکمه‌ی > کلیک کنید تا این ستون به قسمت Selected Columns اضافه شود. در قسمت Header text نام آنرا به Select تغییر دهید. پنجره را OK کرده و به فرم برگردید. حال باید به این ستون یک checkbox اضافه کنیم. روی Datagrid کلیک راست کرده و Edit Template و Select - Column [0] را انتخاب کنید. در صفحه‌ی جدیدی که ظاهر میشود یک checkbox به داخل قسمت Item Template بیاندازید. برای برگشت به صفحه‌ی اصلی روی این قسمت کلیک راست کرده و End template editing را انتخاب کنید .

مدیریت لیست سطر های انتخاب شده

نکته‌ی مهم در مورد DataGrid اینجاست که DataGrid با هر بار فراخوانی متدDataBind ، خود را مجدداً از روی Datasource میسازد . و چون ستون انتخاب ما جزئی از DataSource نیست پس تمام سطرهای انتخاب شده با فراخوانی متد Databind به حالت انتخاب نشده میروند.(حالت پیش فرض)راه حل این مشکل این است که قبل ازDatabind ، سطر هایی که انتخاب شده اند را در جایی ذخیره کرده و درست بعد از فراخوانی DataBind آنها را به داخل Datagrid مجددا بارگذاری کنیم .

در اینجا لازم نیست تمام اطلاعات سطرهای انتخاب شده را ذخیره کنیم بلکه کافیست مقدار ستون ID آنها را ذخیره کنیم. زیرا این ID منحصر بفرد بوده و بوسیله‌ی آن میتوان به بقیه اطلاعات فرد هم دسترسی پیدا کرد. این ID ها در یک arraylist به نام arrselected که به صورت عمومی تعریف میشود قرار میگیرند.

```
Public Shared arrselected As New ArrayList
```

دقت کنید که تعریف یک متغیر یه صورت shared باعث میشود که مقدار آن در رفت و برگشت های (postback)متوالی به سرور حفظ شود.

سپس باید برای دو رویداد DataBinding و PreRender شی Datagrid کد نوشت: اولی قبل از انجام DataBind(برای عمل ذخیره) و دومی بعد از DataBind (برای عمل load) رویداد PreRender دقیقاً قبل از تولید کد HTML مربوط به Datagrid روی میدهد. در رویداد DataBinding ، IDسطرهای انتخاب شده را به arrselectedاضافه کرده (یا اگر قبلاً انتخاب شده بود و هم اکنون انتخاب نشده بود از arrselected حذف میکنیم).

```
rivate Sub DataGrid1_DataBinding(ByVal sender As Object, ByVal e As
System.EventArgs) Handles DataGrid1.DataBinding
   For Each i As DataGridItem In DataGrid1.Items
      'add
      If CType(i.FindControl("checkbox1"), CheckBox).Checked AndAlso Not
arrselected.Contains(i.Cells(3).Text) Then
         arrselected.Add(i.Cells(3).Text)
      End If
      'remove
      If Not CType(i.FindControl("checkbox1"), CheckBox).Checked Then
         arrselected.Remove(i.Cells(3).Text)
      End If
   Next
End sub
```

و در رویداد prerender ، ID سطر های Datagrid را باID های موجود در arrayselected مقایسه کرده اگر این ID قبلاً انتخاب شده بود checkbox مربوط به آنرا تیک میزنیم.

```
Private Sub DataGrid1_PreRender(ByVal sender As Object, ByVal e As
System.EventArgs) Handles DataGrid1.PreRender
   For Each i As DataGridItem In DataGrid1.Items
      If arrselected.Contains(i.Cells(3).Text) Then
         CType(i.FindControl("checkbox1"), CheckBox).Checked = True
      End If
   Next
End Sub
```

خصوصیت مهم این روش در این است که حتی اگر Datagrid را sort کرده یا از صفحه‌ی جاری به صفحه‌ی دیگری رفته و دوباره به این صفحه برگردید تیک ها سر جای خود می مانند. برای افزودن قابلیت تغییر صفحه، کد زیر را به code behind اضافه کنید.

```
Private Sub DataGrid1_PageIndexChanged(ByVal source As Object, ByVal e As
System.Web.UI.WebControls.DataGridPageChangedEventArgs) Handles
DataGrid1.PageIndexChanged
DataGrid1.CurrentPageIndex = e.NewPageIndex
DataGrid1.DataBind()
End Sub
```

برای دیدن برنامه در حال اجرا لینک زیر را کلیک کنید

www12.brinkster.com/salmanzh/newfile.aspx

خلاصه

در این مقاله با نحوه‌ی اضافه کردن یک ستون انتخاب به Datagrid آشنا شدید. ابتدا یک ستون template columnکه یک checkbox در خود دارد به Datagrid اضافه کردید. با دو رویداد DataBinding و PreRenderآشنا شده در اولی لیست سطرهای انتخاب شده را به روز میکنید و در دومی وضعیت انتخاب سطرهای Datagridرا از روی این لیست تعیین میکنیم.

چهار روش برای نمایش محتوای DataSet در فرمهای وب[1]

در ADO.NET دو روش برای کار با داده ها وجود دارد DataSet :و .DataReader هر کدام از این دو روش موارد استفاده خود را دارند. اگر هدف نشان دادن سریع داده هاست DataReader مناسب تر از DataSet می باشد ولی اگر قصد انجام عملیاتی بر روی داده ها دارید و لازم است داده ها در حافظه مقیم شوند باید از DataSet استفاده کنید. در این مقاله چهار روش برای نمایش محتوای یک DataSet در فرمهای وب بیان می شود .

www.iranasp.net/download/evalizadeh006.zip

وقتی از DataSet استفاده می کنید بدلیل انبار شدن داده ها در حافظه، استفاده نادرست از آن می تواند سرعت و کارایی برنامه وب شما را کاهش دهد، البته استفاده نادرست از DataSet در برنامه های ویندوز هم تاثیر منفی در کارایی برنامه دارد ولی مسلما به اندازه برنامه های وب محسوس نیست و این بدلیل آن است که در برنامه های وب امکان انجام درخواستهای همزمان از یک صفحه وجود دارد .

در این مقاله چهار روش برای نمایش محتوای یک DataSet در فرمهای وب بیان می شود:

۱ -استفاده از کنترلDataGrid

۲ -استفاده از کنترلDataList

۳ -استفاده از کنترلRepeater

۴ -استفاده از یک متغیر

[1] احسان ولیزاده

هر DataSet می تواند یک یا چندین DataTable را شامل شود. برای وارد کردن داده ها به داخل DataSet نیاز به استفاده از یک DataAdapter داریم. انجام مراحل زیر برای وارد کردن داده ها در DataSet اساسی هستند:

•ایجاد یک Connection به پایگاه داده:

```
Dim strConn as string = “initial catalog=Northwind;integrated
security=SSPI;persist security info=False;workstation id=DOTNET"
Dim MyConn as New SQLConnection(strConn)
```

•ایجاد:DataSet

```
Dim ds as DataSet=New DataSet()
```

•ایجاد:DataAdapter

```
Dim MySQL as string = "Select Employees.FirstName, Employees.LastName,
Employees.Title from Employees"
Dim da as New SQLDataAdapter(MySQL,MyConn)
```

•بعد از انجام این مراحل متد Fill از شئ DataAdapter را فراخوانی می کنیم:

```
da.Fill(ds,"Employees")
```

پارامتر اول در این متد نام DataSet و پارامتر دوم نام DataTable جدیدی است که می خواهیم تعریف کنیم .

اکنون DataSet شامل داده هایی می باشد که از DataAdapter به داخل آن انتقال یافته اند. برای نشان دادن داده های داخل DataSet از یکی از روشهای زیر استفاده می کنیم:

۱ -کنترلDataGrid

با استفاده از کد زیر می توان جدول Employees را (که در داخل DataSet قرار دارد) در DataGrid نشان داد:

```
MyDataGrid.Datasource=ds.Tables("Employees").DefaultView
MyDataGrid.DataBind()
```

همانطور که مشاهده می کنید ابتدا با مشخص نمودن خصوصیت Datasource کنترل - DataGrid که جدول Employees واقع در DataSet می باشد - داده ها به داخل DataGrid انتقال می یابند و سپس با فراخوانی متد DataBind داده ها بر روی صفحه نشان داده می شوند.

۲ -کنترلDataList

برای مقید کردن داده ها بر روی کنترلهای DataList و Repeater نیز همانند کنترل DataGrid می توانید کدهای زیر را بکار ببرید:

```
MyDataList.Datasource=ds.Tables("Employees").DefaultView
MyDataList.DataBind()
```

۳ -کنترل Repeater

```
MyRepeater.datasource = ds.Tables("Employees").DefaultView
MyRepeater.DataBind()
```

کنترلهای DataGrid ، DataListو Repeater کنترلهای DataBound نام دارند. کنترلهای DataBoundامکان تنظیم منابع داده را به صورت Programmatically فراهم می کنند. این کنترلها بعد از تنظیم منبع داده ای بر روی آن تکرار شده و بر اساس فرمت تعیین شده برای کنترل، داده ها را نمایش می دهند. فرمت فقط برای یک سطر مشخص می شود و همه سطرها بر اساس آن نشان داده می شوند.

این که کدام کنترل را برای نمایش داده ها انتخاب کنید به این بستگی دارد که می خواهید چه مقدار کنترل بر روی فرمت نمایش داده ها داشته باشید. کنترل DataGrid آسانترین و ساده ترین راه برای به نمایش درآوردن داده ها در قالب یک جدول است Visual Studio .NET. هم قالب های آماده ای را از طریق گزینه Auto Format برای تغییر ظاهری کنترل DataGrid (و همچنین DataList) در اختیار ما قرار می دهد DataList .کمی پیچیده تر از DataGrid است ولی امکان کنترل زیادی را در رابطه با فرمت بندی از طریق استفاده از الگوها فراهم می کند. کنترل Repeater نسبت به DataList هم پیچیده تر است و می توان کنترل خیلی بیشتری را هنگام استفاده از الگوها بر روی فرمت نمایش داده ها اعمال نمود.

به دلیل این که کنترلهای DataList و Repeater به طور پیش فرض فرمت داخلی ندارند هنگام تعریف باید آنها را فرمت بندی نماییم (فرمت کنترل DataGrid را هم می توان تغییر داد ولی اگر تغییر ندهیم مشکلی پیش نمی آید و داده ها نمایش داده می شوند). به عنوان نمونه برای کنترل DataList می توان چنین نوشت:

```
<asp:DataList id="MyDataList" runat="server" GridLines="None"
cellpadding="2" cellspacing="2" Headerstyle-BackColor="#8080C0" Headerstyle-Font-
Name="Arial" Headerstyle-Font-Size="8" Font-Name="Arial" Font-Bold="false" Font-
Size="8">
```

```
  <ItemTemplate>
     <%# DataBinder.Eval(Container.DataItem, "firstname") %> <%#
DataBinder.Eval(Container.DataItem, "lastname") %> -
<b><i><%#DataBinder.Eval(Container.DataItem, "Title") %> </i></b>
  </ItemTemplate>
</asp:DataList>
```

DataBinder.Evalمتدی است که برای ارزیابی عبارات data-binding در زمان اجرا از آن استفاده می شود. ساختار اصلی این متد به صورت زیر است:

```
<%# DataBinder.Eval(Container, EvalExpression, FormatExpression) %>
```

Containerعبارتی را بیان می کند که می خواهیم ارزیابی کنیم، برای کنترلهای DataBound همیشه مقدار Container.DataItemرا به جای Container قرار می دهیم .EvalExpression نام خصوصیت یا آیتمی است که باید ارزیابی شود و FormatExpression یک عبارت با فرمت رشته ای است که برای فرمت کردن نتیجه که یک رشته است استفاده می شود.

۴ -استفاده از متغیر برای نشان دادن محتوایDataSet

برای نشان دادن محتوای DataSet می توان تکنیک استفاده از متغیر را بکار برد. برای این کار ابتدا محتوای DataSetرا در یک متغیر از نوع String قرار می دهیم. کد لازم برای انجام این کار به صورت زیر است:

```
Dim sEmps as String
Dim dr As DataRow
For Each dr In ds.Tables("Employees").Rows
```

```
  sEmps += "<b>" & dr("firstname") & " " & dr("lastname") & " </b> - " & dr("Title")
& "<br>"
Next
```

روشی که در اینجا استفاده شده است پیمایش تمام سطرهای جدول مورد نظر با استفاده از یک حلقه For Each ... Next است که همه آنها را در داخل متغیر از نوع String قرار می دهد. سپس خصوصیت text یک کنترل literal را برابر این متغیر قرار می دهیم:

```
litEmps.text=sEmps
```

ایجاد یک کنترل - قسمت اول[1]

کنترل یک عنصر ویژوال می باشد که می توان آنرا در زمان طراحی برروی فرم قرار داد و در صورت لزوم خاصیت های آن را تغییر داد. این مقاله به بررسی چگونگی ساخت یک کنترل برای WinForm می پردازد که با اندکی تغییرات می تواند عینا برای فرم های وب نیز مورد استفاده قرار گیرد .

کنترل یک عنصر ویژوال می باشد که می توان آنرا در زمان طراحی برروی فرم قرار داد و در صورت لزوم خاصیت های آنرا تغییر داد. در این قسمت استانداردهای لازم برای ایجاد یک کنترل و روش های آن شرح داده می شود.

ایجاد یک پروژه

بمنظور ایجاد یک کنترل جدید شما نیازمند ایجاد یک پروژه می باشید .در این حالت شما دارای دو انتخاب می باشید Windows Control Library :و .Web Control Library بسته به نوع کنترلی که می خواهید طراحی کنید یکی از موارد فوق را انتخاب نمایید. پس از ایجاد پروژه می توانید در صورت لزوم فایل کنترل ایجاد شده (UserControl1.cs)را پاک کرده و یک کلاس جدید به پروژه اضافه نمایید.

ارث بری از کنترل موجود

برای ایجاد یک کنترل جدید می توانید در صورت لزوم از کنترلهای موجود ارث بری کرده و آنها را بر حسب نیاز تغییر دهید و یا یک کنترل را از ابتدا ایجاد نمایید.

[1] رضا قلعه

```
public class ControlName : System.Windows.Froms.Panel
{
   ...
```

افزودن خصوصیت (Property)

بمنظور ایجاد خصوصیت جدید شما نیازمند تعریف یک متغییر خصوصی و یک تابع عمومی برای مقداردهی و اخذ مقدار خصوصیت می باشید.

```
private System.Windows.Forms.Border3DSide borderSide;

public System.Windows.Forms.Border3DSide BorderSide
{
   get { return this.borderSide; }
   set
   {
      if ( this.borderSide != value )
      {
         this.borderSide = value;
         this.Invalidate(); // force repaint control at design time
      }
   }
}
```

پس از تعریف خصوصیت جدید ترجیحا آنرا در سازنده کلاس مقداردهی اولیه نمایید.

```
public ControlName
{
   this.borderSide = System.Windows.Froms.Boder3DSide.All;
   this.border3DStyle = System.Windows.Froms.Border3DStyle.Etched;
}
```

بازنویسی متدهای ارث برده شده(override)

در صورتیکه کنترل ایجاد شده را از یک کنترل موجود ارث برده باشید، ممکن است نیازمند بازنویسی کد تعدادی از متدهای کنترل اولیه باشید.

```
protected override void OnPaint(System.Windows.From.PaintEventArgs e)
{
   base.OnPaint(e);

   System.Windows.Froms.ControlPaint.DrawBorder3D (
      e.Graphics,
      this.ClientRectangle,
      this.border3DStyle,
      this.borderSide );
}
```

بمنظور پیدا کردن تمامی متدهای موجود کلاس ارث برده شده می توان از Object Browse استفاده نمود و پس از پیدا کردن متد مورد نظر روی آن کلیک راست کرده و Override را از گزینه Add انتخاب نمایید.

افزودن صفات مربوط به خصوصیت ها به منظور نمایش آن در محیط طراحی

بمنظور نمایش صحیح خصوصیت های نوشته شده در پنجره Properties محیط طراحی باید چندین صفت مربوط به خصوصیت را تنظیم نمود.

Bindable:با دادن مقدار true به عنوان پارامتر به این تابع پس از هر بار تغییر این خصوصیت، در محیط طراحی سیگنالی به معنی اینکه مقدار این خصوصیت تغییر کرده صادر می گردد و کلاس designer شما (در ادامه توضیح داده خواهد شد) را آگاه خواهد کرد .

DefualtValue:با دادن مقدار پیش فرض مورد نظر برای این خصوصیت در محیط طراحی مقدار ذکر شده به عنوان پیش فرض نمایان می گردد .

Category:با دادن یک رشته به عنوان پارامتر به این تابع دسته قرار گرفتن خصوصیت در پنجره Propertiesمحیط طراحی مشخص می گردد .

Description:با دادن یک رشته به عنوان پارامتر به این تابع در صورت انتخاب خصوصیت در لیست Properties، توصیف مشخص شده در پایین این پنجره نمایش داده خواهد شد .

ReadOnly:با true قرار دادن پارامتر این متد خصوصیت مورد نظر در محیط طراحی بصورت فقط خواندنی نمایش داده می شود .

Editor:این تابع دو پارامتر می گیرد. اولین پارامتر مشخص کننده نوع Editor و پارامتر دوم مشخص کننده نوع پایه Editor می باشد (در ادامه توضیح داده خواهد شد).

```
[Bindable(true), Category("Border Options"),
 DefaultValue(System.Windows.Forms.Border3DSide.All),
 Description("Specifies the sides of the panel to
   apply a three-dimensional border to.")]
public System.Windows.Forms.Border3DSide BorderSide
{
  ...
```

ایجاد یک کنترل - قسمت دوم[1]

کنترل یک عنصر ویژوال می باشد که می توان آنرا در زمان طراحی برروی فرم قرار داد و در صورت لزوم خاصیت های آن را تغییر داد. این مقاله به بررسی چگونگی ساخت یک کنترل برای WinForm می پردازد که با اندکی تغییرات می تواند عینا برای فرم های وب نیز مورد استفاده قرار گیرد .

ایجاد یک ویرایشگر برای تعیین مقدار خصوصیت در محیط طراحی

بمنظور ایجاد یک ویرایشگر برای یک خصوصیت شما ابتدا باید فرم مورد نظر را طراحی کنید .البته لازم به ذکر می باشد که حتما لازم نیست فرم جدیدی طراحی کنید و می توانید از فرم ها و مدلهای موجود استفاده کنید. در اینجا فرض کنید که یک فرم در شاخه Design کنترل مورد نظر ایجاد کرده اید. پس از طراحی و ایجاد فرم مورد نظر باید یک کلاس ارث برده شده از UITypeEditor ایجاد کرده و فرم مورد نظر را فراخوانی نمایید. در این کلاس باید حداقل دو متد EditValue و GetEditStyle را بازنویسی مجدد نمایید. متد EditValue زمانیکه کاربر برروی دکمه کنار خصوصیت کلیک کند فراخوانی می گردد. در این متد می توانید فرم ایجاد شده را فراخوانی کنید. متد دوم به منظور تعیین حالت نمایش فرم ایجاد شده بکار می رود.

```
public class StringEditor : System.Drawing.Design.UITypeEditor
{
   public override object EditValue(ITypeDescriptorContext context,
   IServiceProvider serviceProvider, object value)
   {
      if ((context != null) && (serviceProvider != null))
      {
```

[1] رضا قلعه

```
        //Set the editor service
        IWindowsFormsEditorService edSvc =
        (IWindowsFormsEditorService)serviceProvider.GetService(
        typeof(IWindowsFormsEditorService));

        if (edSvc != null)
        {
            //Create new designed form
            DesignedForm form = new DesignedForm();
            form.Value = (string)value;
            //Show form as a dialog
            DialogResult result = edSvc.ShowDialog(form);
            if (result == DialogResult.OK)
            {
                value = form.Value;
            }
        }
    }
    return value;
  }

  public override UITypeEditorEditStyle GetEditStyle( ITypeDescriptorContext
context)
  {
    if (context != null) {
        //The editor will be a modal window.
        return UITypeEditorEditStyle.Modal;
```

```
        }
        return base.GetEditStyle(context);
    }
}
```

پس از انجام تعاریف فوق باید صفت Editor خصوصیت را همانطور که در قسمت قبلی ذکر شد مقدار دهی نمایید.

```
[Editor(typeof(StringEditor),
typeof(UITypeEditor))]
public string ServerUrl
{
...
```

در صورتیکه نوع خصوصیت شما از نوع enum باشد در محیط طراحی بصورت خودکار لیستی از مقادیر قابل انتخاب تعریف شده در enum در اختیار کاربر قرار می گیرد. در صورتیکه خواسته باشید خود مقادیر لیست را مشخص کرده و یا نوع خصوصیت از enum نبوده ولی نیازمند لیستی از مقادیر در زمان طراحی باشید می توانید کلاسی از ListboxTypeEditorارث بری کرده و با بازنویسی متد FillInList آن مقادیر مورد نظر خود را در لیست درج نمایید. طریقه استفاده آن در خصوصیت همانند مورد فوق می باشد.

```
public class FormatListboxTypeEditor : ListboxTypeEditor
{
  protected override void FillInList(ITypeDescriptorContext context,
  IServiceProvider provider, ListBox listBox)
  {
    listBox.Items.Add("Item 1");
```

```
        listBox.Items.Add("Item 2");
        listBox.Items.Add("Item 3");
        listBox.BorderStyle = BorderStyle.None;
    }
}
```

افزودن امکان پشتیبانی توسط جعبه ابزار(ToolBox)

افزودن امکان پشتیبانی توسط جعبه ابزار برای یک کنترل بسیار آسان می باشد. بدین منظور نیازمند یک تصویر برای نمایش در جعبه ابزار و چندین تنظیم کوچک برروی کلاس می باشیم.

تصویر مورد استفاده برای کنترل از نوع Bitmap بوده و باید ۱۶ رنگ بوده و اندازه آن ۱۶*۱۶ باشد. خصوصیت Build Actionتصویر نیز باید برابر Embedded Resource قرار گیرد. مرحله نهایی در این قسمت تنظیم صفات ToolboxItemو ToolboxBitmap می باشد.

```
[ToolboxItem(true)]
[ToolboxBitmap(typeof(ImageName))]
public class DividerPanel : System.Windows.Forms.Panel
{
...
```

افزودن کلاس Designer

از این کلاس بمنظور انجام تنظیمات مورد نظر و اعمال آن در محیط طراحی استفاده می شود. بدین منظور یک کلاس ایجاد نموده و مطابق کد زیر آنرا اصلاح نمایید.

```
public class ControlNameDesigner :
System.Windows.Forms.Design.ScrollableControlDesigner
{
}
```

به عنوان مثال در صورتیکه نیاز داشته باشید یک خصوصیت خاص کلاس در محیط طراحی ظاهر نشود می توانید متد PreFilterProperties را باز نویسی مجدد کرده و خصوصیت مورد نظر را در محیط طراحی حذف نمایید.

```
protected override void PreFilterProperties(
System.Collections.IDictionary properties)
{
   properties.Remove("BorderStyle");
}
```

در انتها با مشخص کردن صفت DesignerAttribute کلاس کنترل، کلاس Designer ایجاد شد را به کنترل معرفی نمایید.

```
[DesignerAttribute(typeof(ClassNameDesigner))]
public class ControlName : System.Windows.Forms.Panel
{
...
```

صفات Assembly و امضا

آخرین قدم در راه ایجاد یک کنترل فراهم کردن اطلاعات لازم برای assembly می باشد. این مرحله اجباری نمی باشد ولی بسیار تاکید می شود که این مرحله را نیز انجام دهید. در ابتدا فایل AssemblyInfo.cs را باز کرده و اطلاعات لازم را در آن وارد نمایید.

```
[assembly: AssemblyTitle("Control Title")]
[assembly: AssemblyDescription("Control Description")]
[assembly: AssemblyConfiguration("")]
[assembly: AssemblyCompany("Company")]
[assembly: AssemblyProduct("Product Name")]
[assembly: AssemblyCopyright("Copyright Info")]
[assembly: AssemblyTrademark("")]
[assembly: AssemblyCulture("")]
```

در ادامه چندین صفت که بطور پیش فرض در این فایل موجود نمی باشد را بصورت زیر به این فایل اضافه نمایید.

```
// Flagging your assembly with the CLS Compliant attribute
// lets the Framework know that it will work with all
// CLS Compliant languages, and theoretically with all
// future framework platforms (such as Mono on Linux).
// If possible you should avoid using any non-CLS compliant code
// in your controls such as unsigned integers (uint) and
// calls to non-framework assemblies.
[assembly: System.CLSCompliant(true)]
```

```
// The ComVisible attribute should always be explicitly set for controls.
// Note that in order for your control to be consumed by
// COM callers, you must use parameter-less constructors
// and you cannot use static methods.
[assembly: System.Runtime.InteropServices.ComVisible(true)]
```

آخرین مرحله قبل از استفاده از کنترل امضاء نمودن assembly می باشد. امضاء کردن assembly بمنظور جلوگیری از دستکاری و خراب شدن هنگام بارگذاری و یا خراب شدن در اثر download مؤثر می باشد. در ابتدا شما باید با استفاده از ابزار Strong Name که در محیط Command Prompt ، محیط Visual Studio.NET می وجود می باشد بمنظور ایجاد یک کلید عمومی/خصوصی برای assembly بصورت زیر استفاده نمایید.

```
sn -k [outputfile].snk
```

فایل ایجاد شده را در شاخه پروژه خود کپی کرده و عبارت زیر را در فایل assembly وارد نمایید.

```
[assembly: AssemblyKeyFile("..\\..\\..\\outputfile.snk")]
```

لذت برنامه نویسی تحت وب با ASP.NET 2.0: بکارگیری Master Page ها[1]

ASP.NET 2.0قابلیت های جدیدی را به همراه آورده است. یکی از مهمترین این قابلیتها اصطلاحا Master Pageنام دارد. این نوآوری شما را قادر می سازد تا به راحتی بتوانید یک ظاهر واحد و یکسان برای کلیه صفحات سایت خود بسازید. این مقاله به صورت قدم به قدم شما را با Master Page آشنا می کند .

اکثر قریب به اتفاق وب سایتها دارای یک قالب مشخص می باشند. بطوریکه اگر در آن وب سایت به گشت و گذار بپردازیم و صفحات مختلف آن را مرور کنیم متوجه می شویم که همه صفحات از یک قالب خاص پیروی می کنند. بعنوان مثال لوگوی مربوط به معرفی وب سایت در همه صفحات وجود دارد. یا اینکه قسمتهای سمت راست یا سمت چپ که معمولا" منویی از لینکها می باشند در تمام صفحات شکل ظاهری خود را حفظ می کنند. برنامه نویسان ASP.NETمی دانند که با معرفی User Control ها چه کمک شایانی به آنها جهت ایجاد قالب های ثابت و یا قابل تغییر برای یک وب سایت شد (البته کاربرد User Control ها تنها در ایجاد قالبها نیست). به کمک User Controlها می توانیم تا با یکبار ایجاد یک قالب و برنامه نویسی های مربوطه، آن را در هرصفحه aspx که خواسته باشیم استفاده کنیم .

اکنون با عرضه ASP.NET 2 (نسخه بتا) و معرفی Master Page تهیه قالب برای یک وب سایت از آنچه که فکرش را هم می کردیم ساده تر شده است. در این مقاله قصد داریم تا شما را با نحوه ایجاد و استفاده از Master Pageتحت ASP.NET 2 و در محیط Web Developer 2005 Express Edition آشنا کنیم. برای همقدم شدن در ایجاد و اجرای مثالی که در این مقاله ایجاد می شود به موارد زیر نیاز است.

[1] Shahabfar.com Iranian .NET Community

Web Developer 2005 Express Edition -

The .NET Framework 2 beta -

-چند تصویر برای استفاده در لوگو و قسمتهای دیگر

ایجاد یک وب سایت:

برای ایجاد یک وب سایت با استفاده از Web Developer و بر پایه ASP.NET 2 مراحل زیر را دنبال کنید.

۱ -در Web Dev از منوی فایل گزینه New Web Site را انتخاب کنید که با اینکار صفحه New Webظاهر می شود.Site

۲ -در زیر گزینه Project Types زبان برنامه نویسی مورد نظرتان را انتخاب کنید (در این مثال زبان C# انتخاب می شود). زبانی که در اینجا انتخاب می شود بعنوان زبان پیش فرض در کل پروژه در نظر گرفته می شود ولی این امکان برای شما مهیاست که برای هر صفحه از این پروژه یک زبان جداگانه را انتخاب کنید.

۳ -در زیر عنوان Visual Studio installed templates گزینه ASP.NET Web Site را انتخاب کنید.

۴ -در قسمت Location نام مسیری را که می خواهید صفحات را در آنجا نگه دارید وارد کنید. مثلا " C:\WebSite\WebSite1

اکنون Web Dev یک صفحه پیش فرض بنام Default.aspx برای شما ایجاد می کند. این صفحه را ببندید و به سراغ ایجاد یک Master Page بروید.

ایجاد یک Master Page

Master Pageیک قالب برای صفحات شماست که در واقع قالب بندی و ظاهر صفحات شما را مشخص می کند. در این قسمت شما ابتدا یک Master Page ایجاد می کنید. سپس با استفاده از یکtable ، آن را طوری تقسیم بندی می کنیم که بتوان در قسمت بالایی آن لوگوی سایت، در قسمت چپ منو و در پایین صفحه footer را قرار داد. شما همچنین با کنترل Content Placeholder آشنا خواهید شد که به کمک آن می توانید محتوای صفحات مختلف را در Master Page داشته باشید .

برای ایجاد یک Master Page :

۱ -در Solution Explorer بر روی نام وب سایتی که ایجاد کرده اید راست کلیک کرده و گزینه Add New Itemرا انتخاب کنید.

۲ -در زیر قسمت Visual Studio Installed Templates گزینه Master Page را انتخاب کنید و نام آن را Master1 بگذارید.

۳ -گزینه Place code in separate file را انتخاب کنید چرا که می خواهیم کد نویسی در Code behindانجام شود.

۴ -زبان مورد نظر خود را در قسمت Language انتخاب کنید.

حال Master Page جدید در قسمت Source View ظاهر می شود. همانطور که مشاهده می کنید در قسمت بالای صفحه تعریف بجای وجود دارد. همچنین در قسمتbody ، کنترل asp.contentplaceholder وجود

دارد که محلی در Master Page می باشد که در runtime محتویات صفحات مختلف در آن جایگزین می شود. در مورد این کنترل در ادامه همین مقاله بیشتر می خوانید.

شکل دادن به Master Page

Master Pageمشخص می کند که صفحات aspx چگونه در وب سایت شما به نظر آیند Master Page . می تواند شامل هر ترکیبی از متن های استاتیک و وب کنترلها باشد. همچنین Master Page ها می توانند شامل چندین کنترل ContentPlaceholder باشند. که مشخص کننده محل قرارگرفتن داده های دینامیک در هنگام نمایش صفحه می باشد.

در این مقاله شما از یک table برای نسبت دادن مکانهایی خاص به المانهای موجود در Master Page استفاده خواهید کرد. سپس کنترل ContentPlaceHolder را که از قبل در Master Page موجود بوده است را در خانه ای از table جای گذاری خواهیم کرد.

برای اضافه کردن table و تغییرات مربوطه در Master Page بصورت زیر عمل کنید.

۱ -به قسمت Design View در صفحه Master Page بروید.

۲ -با استفاده از پنجره مشخصات، رنگ پس زمینه Master Page یعنی گزینه BgColor را به دلخواه تغییر دهید .

۳ -از منوی Layout گزینه Insert Table را انتخاب کنید.

۴ -در صفحه Insert Table بر روی Tamplate کلیک کنید. از لیست موجود گزینه Header, footer and sideرا انتخاب کنید.

۵ -پس از اضافه شدن table ، مطابق زیر اندازه خانه های آن را تغییر دهید. عرض ستون سمت چپ برابر ۸۰، ارتفاع ردیف بالایی برابر ۸۰، و ارتفاع ردیف پایینی برابر ۸۰

۶ -رنگ پس زمینه هر سلول از table را به دلخواه تغییر دهید و VAlign مربوط به هر سلول را در حالت top قرار دهید.

برای اضافه کردن محتویات استاتیک به Master Page مطابق زیر عمل کنید.

۱ -سلول سمت راست پایین را انتخاب کرده و عبارتی نظیر " ... Copyright" را در آن بنویسید.

۲ -از گروه Navigation موجود در toolbox یک کنترل Menu به سلول بالایی table اضافه کنید (مشاهده می کنید که در ASP.NET 2 برای ایجاد منو دیگر نیاز به کد نویسی ندارید. کنترل Menu تقریبا" همه امکانات منوهایی که مد نظر دارید را در اختیار شما می گذارد). حال مشخصه Orientation از کنترل Menu را Horizontal انتخاب کنید. از منوی Common Menu Taskکه در کنار کنترل Menu ظاهر می شود، گزینه Edit Menu Items را انتخاب کنید.

۳ -در زیر عنوانItems ، آیکون مربوط به اضافه کردن آیتم به منو را دوبار کلیک کنید تا دو آیتم به منو اضافه شوند. اولین آیتم را انتخاب کرده و Text آن را Home و NavigationUrl آن را

به Home.aspx انتخاب کنید. برای آیتم دوم Text را About و NavigationUrl را About.aspxانتخاب کنید.

۴ -یک تصویر را بعنوان لوگو در سلول بالایی table قرار دهید.

برای اضافه کردن ContentPlaceHolder طریقه زیر را بکار گیرید:

۱ -کنترل ContentPlaceHolder موجود رابه سلول سمت راست از ردیف وسط table انتقال دهید.

۲ -حاصل کار تا اینجا را ذخیره کنید.

ایجاد محتوا برای یک Maste Page

شما با ایجاد یک صفحه ASP.NET که ضمیمه Master Page می شود. می توانید محتوای یک Master Pageرا تعیین کنید. در این مقاله شما دو صفحه aspx که هر دو بعنوان محتوای Master Page تعیین می شوند را به پروژه اضافه خواهید کرد. این دو صفحه با توجه به آیتم هایی که به منو اضافه کرده ایم قطعا" دو صفحه home و aboutخواهند بود.

برای اضافه کردن صفحه home.aspx چنین عمل کنید:

۱ -بر روی نام وب سایت در قسمت Solution Explorer راست کلیک کرده و Add New Item را انتخاب کنید.

۲ -در زیر عنوان Visual Studio Installed Templates بر روی Web Form کلیک کنید.

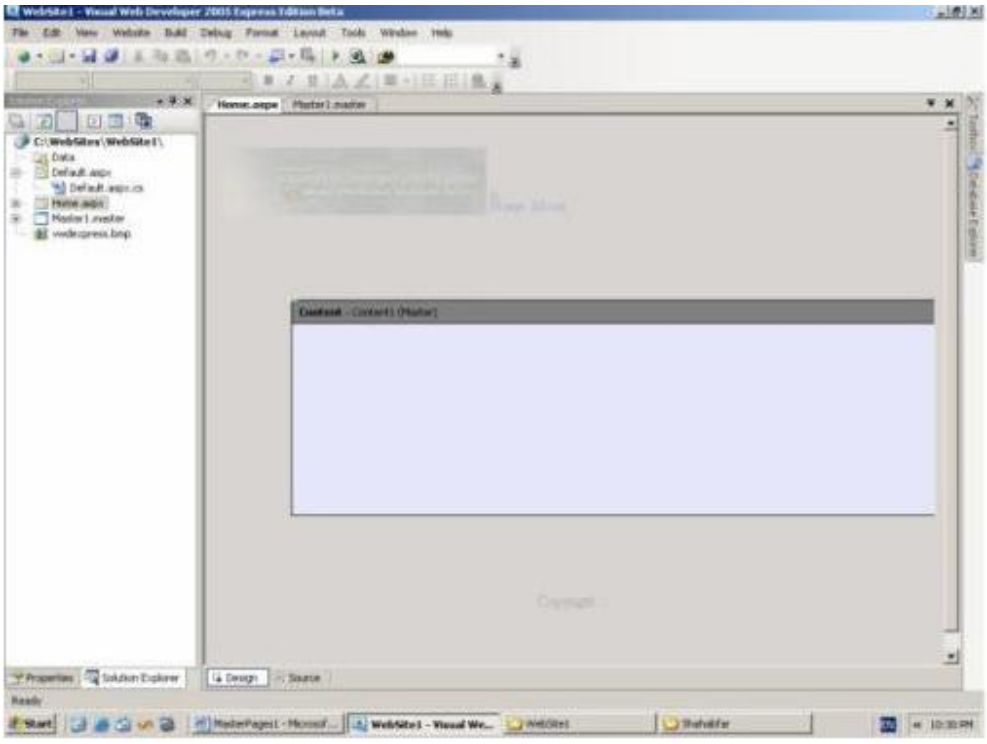
۳ -عنوان صفحه را به Home تغییر دهید.

۴ -زبان مورد نظر خود را در قسمت Language انتخاب کنید.

۵ Select Master Page -را انتخاب کرده و بر روی Add کلیک کنید تا صفحه Select a Master Page ظاهر شود.

۶ -حال Master1.master را انتخاب کرده و بر روی ok کلیک کنید.

صفحه home همانند شکل زیر به پروژه اضافه خواهد شد.

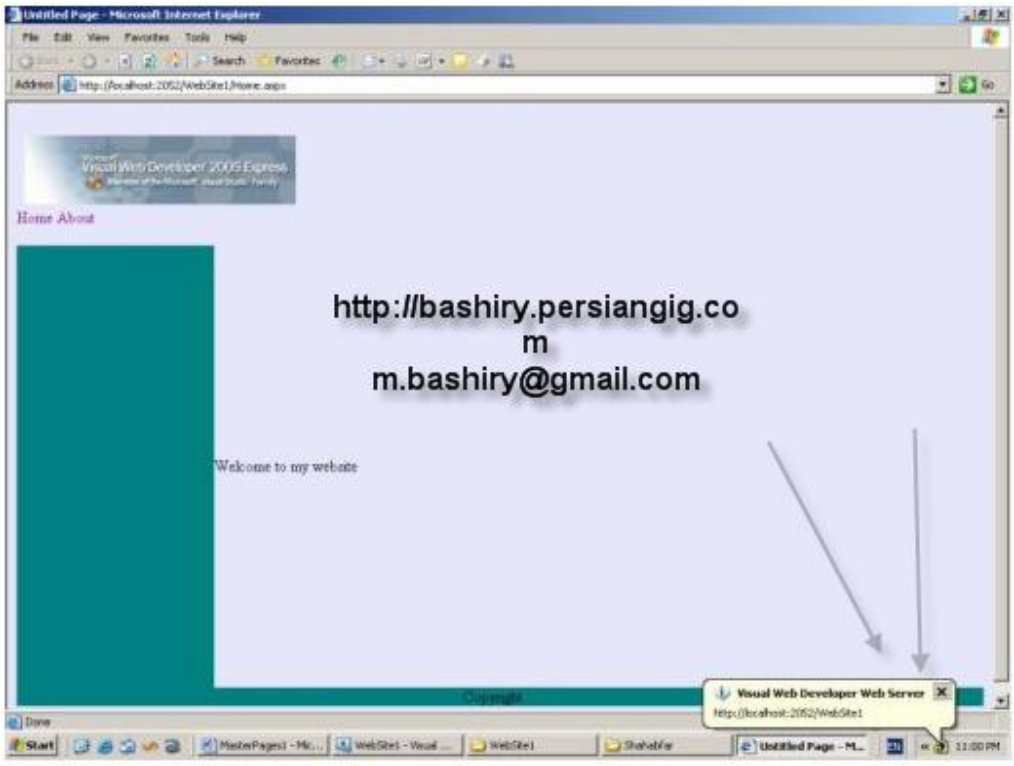


همانطور که مشاهد می کنید قسمتهای استاتیک که در Master Page آن را طراحی کردیم، در اینجا بصورت مات دیده می شود. این بدین معناست که شما در این صفحه نمی توانید آنها را تغییر دهید.

حال به دلخواه خود المانها و کنترلهای مختلفی را به صفحه home.aspx اضافه کنید .برای این منظور بر روی نوار عنوان کنترل ContentPlaceHolder راست کلیک کرده و گزینه CreateCustomContent را انتخاب کنید. حال در کنترل ContentPlaceHolder یک جمله مانند "Welcome to my website" تایپ کرده و حاصل کار را ذخیره کنید.

صفحه about.aspx را نیز بمانند صفحه home به وب سایت اضافه کنید.

حال می توانیدحاصل کار خود را با فشار دادن دکمه های Ctrl+F5 مشاهده کنید .



ASP.NETمحتوای صفحات home و about را با Master Page ترکیب کرده و به نمایش در می آورد.

همان طور که مشاهده می کنید در ASP.NET 2 پروژه ها بر روی شماره پورتهای تصادفی در Localhost اجرا می شوند.

مرجع:MSDN

لذت برنامه نویسی تحت وب با ASP.NET 2.0: تغییر Master Page ها بصورت پویا[1]

در قسمت اول از مجموعه این مقاله ها آموختید که چگونه می توان بوسیله Master Page ها در ASP.NET 2.0به سادگی طرح بندی های مختلفی را برای فرم های وب ایجاد کرد. تحت شرایطی بعضاً ممکن است به تغییر یک Master Page بصورت پویا نیاز پیدا کنیم. مثلاً ممکن است در یک وب سایت برای موضوعات مختلف طرح بندی سایت تغییر کند. این مقاله را بخوانید تا روش انجام این کار را یاد بگیرید .

تحت شرایطی بعضاً ممکن است به تغییر یک Master Page بصورت پویا نیاز پیدا کنیم .مثلا" ممکن است در یک وب سایت برای موضوعات مختلف طرح بندی سایت تغییر کند. بدیهی است که برای این منظور بسته به نیازمان Master Pageهای مختلفی را در ابتدای کار می بایست ایجاد کنیم. در این مقاله می آموزید که چگونه می توان بصورت پویا یک Master Page را با دیگری جایگزین کرد.

برای این منظور Master Page دوم را به وب سایتمان که در قسمت اول مقاله ایجاد کردیم اضافه کرده و یک کنترل button برای سوئیچ کردن بین Master Page ها اضافه می کنیم. از آنجا که Master Page دوم بسیار شبیه اولی می باشد لذا تنها کافیست یک کپی از آن تهیه کرده و برای تمایز، رنگ پس زمینه آن را تغییر دهیم.

برای این منظور مراحل زیر را دنبال کنید:

۱ -در Solution Explorer بر روی Master1.master راست کلیک کرده و گزینه کپی را انتخاب کنید.

[1] Shahabfar.com Iranian .NET Community

۲ -بر روی نام وب سایت راست کلیک کرده و گزینه paste را انتخاب کنید.

۳ -Master Page کپی شده را به Master2.master تغییر نام دهید .

۴ -فایل Master2.master را در حالت Html باز کرده و در قسمت Master1 ، Master@را به Master2 تغییر دهید .

```
Master Language="C#" CompileWith="Master2.master.cs"
ClassName="Master2_master@"
```

۵ -به حالت Design View بروید .

۶ -در پنجره properties از لیست باز شو که در قسمت بالای این پنجره قرار دارد گزینه DOCUMENT را انتخاب کنید .

۷ -مقدار BgColor را به دلخواه تغییر دهید .

۸ -Master Page جدید را در حالت سورس کد باز کنید و نام کلاس را از Master1_master به Master2_masterتغییر دهید .

برای اضافه کردن یک button جهت تغییر پویا Master Page ها مطابق زیر عمل کنید:

۱ -یک کنترل از نوع LinkButton به پایین صفحه Master2.master اضافه کرده و برای آن یک نام انتخاب کنیدمثلا "Change to MP1" "

۲ -بر روی button دو بار کلیک کنید تا تابع رویداد کلیک آن ایجاد شود. سپس کد زیر را به آن اضافه کنید.

```
void LinkButton1_Click(object sender, EventArgs e)

{

Session("masterpage") = "MasterPage1.master";

Response.Redirect(Request.Url.ToString());

}
```

کد فوق موجب بار گزاری نام Master Page اول در Session شده و سپس صفحه جاری دوباره فراخوانی می شود .

۳ -صفحه Master1.master را باز کنید و یک کنترل LinkButton مطابق مراحل فوق به آن اضافه کنید و نوشته روی آن را "Change to MP2" قرار دهید .

۴ -در تابع مربوط به رویداد کلیک آن کد زیر را اضافه کنید .

```
void LinkButton1_Click(object sender, EventArgs e)

{
```

```
Session("masterpage") = "MasterPage2.master";

Response.Redirect(Request.Url.ToString());

}
```

۵ -صفحه About.aspx را در حالت سورس کد باز کنید .

۶ -تابع Page_PreInit را اضافه کرده و کد زیر را در آن بنویسید.

کد فوق باعث جایگزین کردن محتویات Session[“masterpage”] با مقدار مشخصه Me.MasterPageFileمی شود. این کار می بایست حتما" در تابع Page_PreInit صورت گیرد. زیرا در یک WebFormقبل از هرگونه مقدار دهی اولیه می بایست تکلیف Master Page آن مشخص شود. و بهترین گزینه تابع Page_PreInit می باشد. لذا قرار دادن این کد در توابعی مثل Page_Init و Page_Load که بعد از تابع Page_PreInitفراخوانده می شوند، تأثیر گذار نخواهد بود.

حال بعد از اعمال مراحل فوق حاصل کار را مطابق زیر امتحان می کنیم :

۱- CTRL+F5 را فشار دهید تا صفحه home در browser نمایان شود.

۲ -به صفحه About رفته و بر روی Chage to MP2 کلیک کنید. مشاهده می کنید که صفحه About با Master2.masterباز خواهد شد.

۳ -بر روی Change to MP1 کلیک کنید. می بینید که اینبار صفحه About با Master1.master باز می شود.

مراحل فوق یک مثال بسیار ساده برای تغییر Master Page ها بصورت پویا را شامل می شدند. امروزه بعضی از وب سایتها مثل MSN.com این قابلیت را برای کاربران عضو مهیا ساخته اند که بتوانند طرح بندی صفحه مربوط به خود را بصورت پویا براحتی تغییر دهند. مسلماً استفاده از Master Page ها برای این منظور کار را بسیار ساده و سریع خواهد نمود .

دریافت Web Developer 2005 Express Edition (ASP.NET 2) beta از آدرس

www.shahabfar.com/express

مرجع:MSDN

صدا زدن رویدادهای User Control از داخل فرم‌های وب[1]

استفاده از کنترل‌های کاربری یا User Control در میان برنامه نویسان ASP.NET طرفداران زیادی دارد. با استفاده از UC می توان مجموعه ای از کنترلهای سمت سرور یا html را در جاهای مختلف از صفحات aspx یا داخلUC های دیگر استفاده کرده و از نوشتن کدهای تکراری اجتناب نمود. در بسیاری از موارد لازم می‌شود که بتوانیم رویدادی که درون یک UC اتفاق می‌افتد را در وب فرم مادر اصطلاحاً Handle کنیم. این مقاله به نحوه چگونگی انجام این کار می‌پردازد .

لینک دریافت کد: www.iranasp.net/download/shahabfar003.zip

استفاده از کنترل‌های کابریUC ها (User Control) در میان برنامه نویسان ASP.NETطرفداران زیادی دارد. با استفاده از UC می توان مجموعه ای از کنترلهای سمت سرور یا html را در جاهای مختلف از صفحات aspx یا داخلUC های دیگر استفاده کرده و از نوشتن کدهای تکراری اجتناب نمود. همچنین می توان تنها قسمتی از یک Web Formرا با کمکUC ها Cache نمود .(Fragment Caching) یکی دیگر از استفاده‌هایUC ها بکارگیری آنها بعنوان یک کنترل سمت سرور می باشد. که در این صورت می توان برای UC مورد نظرMethod ها و Propertyهای مختلفی را بسته به نیاز تعریف کرده و از آنها استفاده نمود .آن دسته از توابعی که اکثر کنترلهای سمت سرور از آن برخوردار می باشند، توابع رویداد یا همانEvent ها می باشند. مثل تابع رویداد فشردن یک کنترل از نوعButton .

اکنون می خواهیم بدانیم که درUC ها چگونه می توان توابع رویداد تعریف کرده و از آنها استفاده نمود. به دو طریق می توان برای یک UC تابع رویداد تعریف کرد. روش اول تعریف یک تابع رویداد داخلی برای یک UC می باشد. که در این صورت تابع رویداد تنها در داخل UC صدا زده می‌شود. اما روش دوم تعریف توابع رویداد خارجی است

[1] Shahabfar.com Iranian .NET Community

به طوریکه بتوان این توابع را از خارج UC نیز صدا زد. بحث اصلی این مقاله پیرامون همین نوع از توابع رویداد یا Eventمی باشد.

برای درک بهتر مطلب بهتر است که به طرح یک مثال بپردازیم. فرض کنید که می خواهید یک کنترل که عملکرد یک منو را داشته باشد با استفاده از UC ایجاد کنید. این منو دارای آیتم‌های مختلفی است که هر آیتم نیز ایندکس خاص خود را دارد. حال می خواهیم یک Event برای این منو تعریف کنیم بطوریکه بتوانیم آن را از همان جایی که UC را استفاده می کنیم، صدا بزنیم .برای ایجاد این کنترل قبل از هر چیز بهتر است که از یک سرور کنترل تکرار شونده و داده پذیر (Data Bound) مثل Repeater ، DataListو یا DataGrid استفاده کنیم. در این مقاله ما از کنترل DataList استفاده می کنیم.

فرض را بر آن می گیریم که خوانندگان این مقاله با نحوه استفاده از کنترل DataList آشنایی دارند. لذا بیشتر به توضیح کدهای مربوط به اضافه کردن تابع رویداد خارجی می پردازیم.

بسیار خوب، در ابتدا در کلاس مربوط به UC منو، یک کلاس مثلاً بنام SelectionChangedEventArgs تعریف می کنیم. این کلاس می بایست حتماً از کلاس System.EventArgs ارث بری داشته باشد. فعلاً برای این کلاس یک متغیر که نشان دهنده ایندکس انتخاب شده از منو می باشد را تعریف می کنیم. بعداً شما می‌توانید متغیرها و متدهای مورد نظر خود را به این کلاس اضافه کنید. پس تا اینجای کار کلاسی همانند زیر در داخل کلاس مربوط به UCمنو داریم.

```
public class SelectionChangedEventArgs : EventArgs
{
   public int SelectedIndex;
}
```

برای تعریف تابع رویداد، نیار به یک تابع delegate داریم. این تابع delegate را می توان با استفاده از کلاسی که در بالا تعریف کردیم بسازیم. کد مربوط به تعریف تابع delegate و تابع رویداد به شکل زیر می باشد.

```
public delegate void SelectionChangedEventHandler(object sender,
SelectionChangedEventArgs e);
public event SelectionChangedEventHandler SelectionChanged;
```

برای تحقق یافتن این خواسته‌مان نیاز به یک تابع رویداد داخلی نیز داریم. از آنجا که هدف ما مشخص کردن انتخاب آیتم های کنترل منو می باشد، پس می بایست که تابع رویداد مربوط به تغییر ایندکس انتخاب شده را برای کنترل DataList موجود در UC را اضافه کنیم. پس از انجام این کار نوبت به تعریف تابع اصلی مان می‌رسد. یعنی تابعی که در آن مقدمات مربوط به صدا زدن Event خارجی صورت گیرد. این تابع را بصورت زیر تعریف می کنیم.

```
public void SelectMenuItem(int index)
{
   dlMenu.DataBind();
   SelectionChangedEventArgs ev = new SelectionChangedEventArgs();
   ev.SelectedIndex = index;
   if(SelectionChanged != null)
      SelectionChanged(this, ev);
}
```

همانطور که می‌بینید در این تابع ابتدا کنترل DataList تغذیه می‌شود. سپس یک نمونه از کلاسی که چندی قبل تعریف کردیم ایجاد می‌شود و تنها عضو این کلاس که مشخص کننده ایندکس انتخاب شده می باشد توسط تنها پارامتر این تابع مقداردهی می‌شود. پس از آن در صورت null نبودن تابع رویداد SelectionChanged که مربوط به

تابع SelectionChangedEventHandler که خود یک delegate می‌باشد صدا زده می‌شود. این کار باعث می‌شود تا تابع رویدادی که برای این کنترل منو در هرکجا تعریف کرده‌ایم، صدا زده شود.

مثالی که به همراه این مقاله قابل دانلود می باشد شما را با جزئیات کار بیشتر آشنا می کند.

کنترل‌های پویا[1]

چگونگی اضافه کردن یک کنترل کاربری (User Control) در مواقع لازم با استفاده از متد LoadControl

شما با استفاده از کنترل های ASP.net اجزاء مورد نظر برنامه وب خود را در یک صفحه وارد می کنید. این صفحات خود ممکن است تنها HTML و یا در برگیرنده کدهای .net باشند. کنترل ها بصورت ascx. ذخیره می شوند و برای استفاده از آنها بصورت زیر با صفحه aspx شما پیوند می خورند و بصورت یک tag ساده می توان آنها را به هر قسمت از صفحه اضافه کرد .

```
<%@ Register TagPrefix="uc1" TagName="HeaderUserControl"
Src="HeaderUserControl.ascx" %>
...

<uc1:HeaderUserControl id="Header1" runat="server"></uc1:HeaderUserControl>
```

و اما شاید شما بخواهید فقط در مواقعی خاص این کنترل ها load شوند. مثلا هنگام اجرای متد Page_Load و یا Button_Click ، در صورت درست بودن نام کاربری و یا به هر شکل دیگری. چاره چیست؟! خوشبختانه کتابخانهnet. با متد LoadControl خود میتواند پاسخ گوی شما باشد (در ضمن این را هم فراموش نکنید که هر tagساده html و یا کنترل سمت سرور ASP. net را با زبان برنامه نویسی net. هم می توان ایجاد کرد). فرض کنید ما کنترلی به اسم FeaturedProduct.ascx داریم. بدین صورت ابتدا بارگذاری و سپس اضافه خواهد شد:

[1] سعد شمسایی

```
Control FeaturedProductUserControl = LoadControl("FeaturedProduct.ascx");

Controls.Add(FeaturedProductUserControl);
```

البته اشکال این تکنیک این است کنترل دقیقا محلی که شما می خواهید اضافه شود نمایش داده نمی‌شود که خب، این هم با استفاده از place holder's controls collection قابل حل هست. بدین شکل:

```
PlaceHolderLeftMenu.Controls.Add(FeaturedProductUserControl);
```

مطمئناً این تکنیک برای کسانی که برنامه های کاربردی تحت وب را پیاده می کنند جهت مدیریت بهتر، استفاده از صفحات کمتر، حالت های مختلف درخواست‌ها و کلاً قالب بندیCMS ها بسیار مفید خواهد بود، چنانچه که پروژه ای مثل IBuySpy نیز از این تکنیک بهره می‌برد.

منبع:ASPAlliance

در انتها امیدورام که مطالب بیان شده برای دوستان مثمر ثمر واقع شود.

در صورت داشتن هرگونه نظر، پیشنهاد و انتقاد سازنده‌ای می توانید از طریق آدرسهای ایمیل زیر با من در ارتباط باشید.

m.bashiry@gmail.com

mohamad_bashiry@gmail.com

m_bashiry@walla.com

آدرس مقالات و کتب الکترونیکی دیگر:

http://bashiry.persiangig.com/Ebook