عنوان: **شبیه سازی ارسال ایمیل در ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۴ ۱۵:۰۸:۰۶
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

فرض کنید مشغول به کار بر روی کامپیوتری هستید که دسترسی به هیچ شبکه‌ای ندارد و همچنین نیاز است تا قسمت اطلاع رسانی برنامه ASP.Net خود را که از طریق ایمیل کار می‌کند، تست کنید. برای مثال حداقل یکبار شکل و شمایل و محتوای ایمیل واقعی ارسالی آنرا در آوت لوک مشاهده کنید. برای حل این مساله چه باید کرد؟

برای تحقق این منظور باید کمی فایل web.config سایت را ویرایش کرد و سطرهای زیر را به آن افزود (پس از بسته شدن تگ system.web):

```
<system.net>
  <mailSettings>
    <smtp deliveryMethod="SpecifiedPickupDirectory">
      <specifiedPickupDirectory pickupDirectoryLocation="c:\mail"/>
    </smtp>
  </mailSettings>
</system.net>
```

و همچنین در اینجا باید دقت داشت که هنگام کد نویسی دیگر نیازی به ذکر smtp server نخواهد بود و new SmtpClient().Send تنظیمات خودش را از فایل کانفیگ خواهد خواند.

اکنون با هر بار ارسال ایمیل، نتیجه حاصل (مطابق تصاویر زیر) در مسیر c:\mail ذخیره خواهد شد و فرمت حاصل با استفاده از outlook قابل مشاهده است.

شایان ذکر است که این روش با برنامه‌های غیر ASP.Net نیز کار می‌کند و تنها کافی است یک فایل app.config به برنامه اضافه کرده و تنظیمات فوق را به آن اعمال نمائید.

نظرات خوانندگان

نویسنده: salarblog
تاریخ: ۱۳۸۷/۰۹/۰۴ ۱۹:۱۶:۰۰

نکته جالب بود.
بسیار ممنون

عنوان: **تبدیل پلاگین‌های jQuery به کنترل‌های ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۸ ۱۵:۵۸:۱۸
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

امروز داشتم یک سری از پلاگین‌های jQuery را مرور می‌کردم، مورد زیر به نظرم واقعا حرفه‌ای اومد و کمبود آن هم در بین کنترل‌های استاندارد ASP.Net محسوس است:

Masked Input Plugin

استفاده از آن به صورت معمولی بسیار ساده است. فقط کافی است اسکریپت‌های jQuery و سپس این افزونه به هدر صفحه اضافه شوند و بعد هم مطابق صفحه usage آن عمل کرد.

خیلی هم عالی! ولی این شیوه‌ی متداول کار در ASP.Net نیست. آیا بهتر نیست این مجموعه را تبدیل به یک کنترل کنیم و از این پس به سادگی با استفاده از Toolbox ویژوال استودیو آن‌را به صفحات اضافه کرده و بدون درگیر شدن با دستکاری سورس html صفحه، از آن استفاده کنیم؟ به‌عبارتی دیگر یکبار باید با جزئیات درگیر شد، آنرا بسته بندی کرد و سپس بارها از آن استفاده نمود. (مفاهیم شیءگرایی)

برای این‌کار، یک پروژه جدید ایجاد ASP.Net server control را آغاز نمائید (به نام MaskedEditCtrl).

به صورت پیش فرض یک قالب استاندارد ایجاد خواهد شد که کمی نیاز به اصلاح دارد. نام کلاس را به MaskedEdit تغییر خواهیم داد و همچنین در قسمت ToolboxData نیز نام کنترل را به MaskedEdit ویرایش می‌کنیم.

برای اینکه مجبور نشویم یک کنترل کاملا جدید را از صفر ایجاد کنیم، خواص و توانایی‌های اصلی این کنترل را از TextBox استاندارد به ارث خواهیم برد. بنابراین تا اینجای کار داریم:

```
namespace MaskedEditCtrl
{
   [DefaultProperty("MaskFormula")]
   [ToolboxData("<{0}:MaskedEdit runat=server></{0}:MaskedEdit>")]
   [Description("MaskedEdit Control")]
   public class MaskedEdit : TextBox

   {
```

سپس باید رویداد OnPreRender را تحریف (override) کرده و در آن همان اعمالی را که هنگام افزودن اسکریپت‌ها به صورت دستی انجام می‌دادیم با برنامه نویسی پیاده سازی کنیم. چند نکته ریز در اینجا وجود دارد که در ادامه به آن‌ها اشاره خواهد شد.

از ASP.Net 2.0 به بعد، امکان قرار دادن فایل‌های اسکریپت و یا تصاویر همراه یک کنترل، درون فایل dll آن بدون نیاز به توزیع مجزای آنها به صورت WebResource مهیا شده است. برای این منظور اسکریپت‌های jQuery و افزونه mask edit را به پروژه اضافه

نمائید. سپس به قسمت خواص آنها (هر دو اسکریپت) مراجعه کرده و build action آنها را به Embedded Resource تغییر دهید (شکل زیر):

از این پس با کامپایل پروژه، این فایل‌ها به صورت resource به dll ما اضافه خواهند شد. برای تست این مورد می‌توان به برنامه reflector مراجعه کرد (تصویر زیر):

پس از افزودن مقدماتی اسکریپت‌ها و تعریف آنها به صورت resource ، باید آنها را در فایل AssemblyInfo.cs پروژه نیز تعریف کرد (به صورت زیر).

```
[assembly: WebResource("MaskedEditCtrl.jquery.min.js", "text/javascript")]
[assembly: WebResource("MaskedEditCtrl.jquery.maskedinput-1.1.4.pack.js", "text/javascript")]
```

نکته مهم: همانطور که ملاحظه می‌کنید نام فضای نام پروژه (namespace) باید به ابتدای اسکریپت‌های معرفی شده اضافه شود.

پس از آن نوبت به افزودن این اسکریپت‌ها به صورت خودکار در هنگام نمایش کنترل است. برای این منظور داریم:

```
protected override void OnPreRender(EventArgs e)
        {
            base.OnPreRender(e);

            //adding .js files
            if (!Page.ClientScript.IsClientScriptIncludeRegistered("jquery_base"))
            {
                string scriptUrl = Page.ClientScript.GetWebResourceUrl(this.GetType(),
                    "MaskedEditCtrl.jquery.min.js");
                Page.ClientScript.RegisterClientScriptInclude("jquery_base", scriptUrl);
            }

            if (!Page.ClientScript.IsClientScriptIncludeRegistered("edit_ctrl"))
            {
                string scriptUrl = Page.ClientScript.GetWebResourceUrl(this.GetType(),
                    "MaskedEditCtrl.jquery.maskedinput-1.1.4.pack.js");
                Page.ClientScript.RegisterClientScriptInclude("edit_ctrl", scriptUrl);
            }

            if (!Page.ClientScript.IsStartupScriptRegistered("MaskStartup" + this.ID))
            {
                // Form the script to be registered at client side.
                StringBuilder sbStartupScript = new StringBuilder();
                sbStartupScript.AppendLine("jQuery(function($){");
                sbStartupScript.AppendLine("$(\"#" + this.ClientID + "\").mask(\"" + MaskFormula +
"\");");
                sbStartupScript.AppendLine("});");
                Page.ClientScript.RegisterStartupScript(typeof(Page),
                    "MaskStartup" + this.ID, sbStartupScript.ToString(), true);
            }

        }
```

همانطور که ملاحظه می‌کنید، ابتدا WebResource دریافت شده و سپس به صفحه اضافه می‌شود. در آخر مطابق راهنمای افزونه mask edit عمل شد. یعنی اسکریپت مورد نظر را ساخته و به صفحه اضافه کردیم.

نکته جاوا اسکریپتی: علت استفاده از this.ClientID جهت معرفی نام کنترل جاری این است که هنگامیکه کنترل توسط یک master page رندر شود، ID آن توسط موتور ASP.Net کمی تغییر خواهد کرد. برای مثال myTextBox به ctl00_ContentPlaceHolder1_myTextBox تبدیل خواهد شد و اگر صرفا this.ID ذکر شده باشد دیگر دسترسی به آن توسط کدهای جاوا اسکریپت مقدور نخواهد بود. بنابراین از ClientID جهت دریافت ID نهایی رندر شده توسط ASP.Net کمک می‌گیریم.

در اینجا MaskFormula مقداری است که هنگام افزودن کنترل به صفحه می‌توان تعریف کرد.

```
[Description("MaskedEdit Formula such as 99/99/9999")]
        [Bindable(true), Category("MaskedEdit"), DefaultValue(0)]
        public string MaskFormula
        {
            get
            {
                if (ViewState["MaskFormula"] == null) ViewState["MaskFormula"] = "99/99/9999";
                return (string)ViewState["MaskFormula"];
            }
            set { ViewState["MaskFormula"] = value; }

        }
```

این خاصیت public هنگام نمایش در Visual studio به شکل زیر درخواهد آمد:

نکته مهم: در اینجا حتما باید از view state جهت نگهداری مقدار این خاصیت استفاده کرد تا در حین post back ها مقادیر

انتساب داده شده حفظ شوند.

اکنون پروژه را کامپایل کنید. برای افزودن کنترل ایجاد شده به toolbox می‌توان مطابق تصویر عمل کرد:

نکته: برای افزودن آیکون به کنترل (جهت نمایش در نوار ابزار) باید: الف) تصویر مورد نظر از نوع bmp باشد با اندازه 16 در16 pixel . ب) باید آنرا به پروژه افزود و build action آن را به Embedded Resource تغییر داد. سپس آنرا در فایل AssemblyInfo.cs پروژه نیز تعریف کرد (به صورت زیر).

```
[assembly: System.Web.UI.WebResource("MaskedEditCtrl.MaskedEdit.bmp", "img/bmp")]
```

کنترل ما پس از افزوده شدن، شکل زیر را خواهد داشت:

جهت دریافت سورس کامل و فایل باینری این کنترل، اینجا کلیک کنید.

نظرات خوانندگان

نویسنده: شاهین کایست
تاریخ: ۱۳۸۹/۱۰/۲۳ ۲۲:۳۹:۳۰

سلام.
از کنترلی که طراحی کردید درون یک JQuery UI دیالوگ استفاده کردم (درون محتویات Dialog در یک Update Panel هست).
اما پس از قرار دادن کنترل Dialog از کار افتاد.
نکته : Dialog را از سمت Server پس از Postback اجرا کردم.
به نظرتون مشکل از کجا هست؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۳ ۲۲:۵۷:۴۱

سلام،
چند مورد هست:
- یکی اینکه بهتر است نسخه‌ی جدید jQuery را به این سورس اضافه و کامپایل کنید.
- مورد دیگر آشنایی با jQuery Live است : (+) ، که پس از postback ، نیاز به تزریق یا بایند مجدد یک سری اطلاعات می‌باشد و همچنین: (+)

عنوان: آشنایی با انواع Control ID ها در ASP.Net
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۹ ۱۱:۳۴:۳۸
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

اگر مطلب تبدیل پلاگین‌های جی‌کوئری به کنترل‌های ASP.Net را مطالعه کرده باشید، در مورد ClientID بحث شد. با مراجعه به اصول HTML درخواهیم یافت که هر کنترل یا شیء مربوطه می‌تواند شامل ID و name باشد. عموما در کدهای جاوا اسکریپتی برای دسترسی به یک شیء در صفحه از ID آن شیء به صورت document.getElementById استفاده شده و از name برای پردازش‌های سمت سرور استفاده می‌شود. برای مثال اگر به دوران ASP کلاسیک برگردیم، از شیء Request برای دریافت مقادیر ارسال شده به سرور با استفاده از name شیء مورد نظر استفاده می‌شد/می‌شود.

در حالت فرم‌های معمولی ، ID و name عموما یکسان هستند. اما اگر از master page ها استفاده شود یا از یک user control برای ترکیب چندین کنترل کمک گرفته شود، دیگر ID و name در فرم رندر شده نهایی ASP.Net یکسان نخواهند بود. برای مثال:

```
<input name="ctl00$ContentPlaceHolder1$txtID" type="text" id="ctl00_ContentPlaceHolder1_txtID" />
```

اگر به سورس دات نت فریم ورک مراجعه کنیم علت وجود _ بجای $ را در مقدار id می‌توان مشاهده کرد:

```
public virtual string ClientID
{
   get
   {
       this.EnsureID();
       string uniqueID = this.UniqueID;
       if ((uniqueID != null) && (uniqueID.IndexOf(this.IdSeparator) >= 0))
       {
           return uniqueID.Replace(this.IdSeparator, '_');
       }
       return uniqueID;
   }

}
```

و جهت تولید name به صورت زیر عمل می‌شود (یک الگوریتم بازگشتی است):

```
<Container>$<Container>$<Container>$<Container>$...$<ID>
```

زمانیکه یک کنترل توسط ASP.Net رندر می‌شود، خاصیت UniqueID معادل name آن و ClientID معادل ID آن کنترل در سمت کلاینت خواهد بود. اگر از IE استفاده کنید به هر نحوی سعی در پردازش اسکریپت شما خواهد کرد و document.getElementById بالاخره جواب خواهد داد. اما اگر از سایر مرورگرها استفاده کنید، در صورتیکه بجای ID از name استفاده شده باشد، اسکریپت شما با پیغامی مبنی بر یافت نشدن شیء مورد نظر متوقف خواهد شد (چون مقدار این‌دو همانطور که ملاحظه کردید الزاما یکسان نخواهد بود).

این نکته در طراحی کنترل‌های ASP.Net که از کدهای جاوا اسکریپتی استفاده می‌کنند بسیار مهم است. در تست اول و با یک صفحه ساده کنترل شما خوب کار خواهد کرد. اما اگر همین کنترل را بر روی یک صفحه مشتق شده از یک master page قرار دهید، دیگر ID آن یافت نشده و کدهای جاوا اسکریپتی شما کار نخواهند کرد. به همین جهت اگر قرار است قسمت جاوا اسکریپتی کنترل شما توسط کنترل به صورت خودکار ایجاد شود و در این کد ارجاعی به شیء جاری وجود دارد، این شیء جاری با استفاده از ClientID آن در سمت کلاینت قابل دسترسی خواهد بود و نه با استفاده از ID سمت سرور و یا UniqueID آن.

نظرات خوانندگان

نویسنده: hajloo
تاریخ: ۱۳۸۷/۰۹/۰۹ ۱۴:۲۴:۰۰

از query که برای جمع ساعات برای من نوشتید و ارسال کردید واقعا ممنونم . منظور من دقیقا همان کاری بود که شما انجام دادید .
صمیمانه از وقتی که برای نوشتن کوئری صرف کردید سپاسگذارم . پاینده و سربلند باشید ناصر حاجلو.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۰۹ ۱۴:۲۷:۰۰

خواهش می‌کنم. خوشحالم که مفید واقع شد.
موفق و مؤید باشید.

عنوان: **بررسی نحوه برنامه نویسی سایت نستعلیق آنلاین**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۱۰ ۱۳:۰۳:۰۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

سایت نستعلیق آنلاین با استفاده از ASP.Net و فونت ایران نستعلیق ایجاد شده است. شاید این سؤال پیش بیاید که چگونه اینکار را انجام داده‌اند؟ چگونه متن را به تصویر تبدیل کرده‌اند یا از همه مهم‌تر چگونه فونت را به صورت پویا بارگذاری می‌کنند (چون عموما هاست‌ها فونتی را برای شما نصب نخواهند کرد)؟
برای انجام اینکار از کلاس PrivateFontCollection فضای نام System.Drawing.Text می‌توان استفاده کرد. نحوه انجام این‌کار را در یکی از پروژه‌های سایت codeproject می‌توان ملاحظه نمود.
کمی این پروژه را اصلاح کردم به همراه افزودن و تنظیم خواص تولید تصویر با کیفیت بالا. پروژه نهایی را به همراه قلم‌های مربوطه، از اینجا می‌توانید دریافت کنید.

احتمالا در سایت نستعلیق آنلاین از روش موجود در مقاله بالا استفاده شده است که مجبور شده‌اند تصویر نهایی را در یک صفحه دیگر نمایش دهند (تخلیه بافر در مرورگر). در پروژه‌ای که ضمیمه شد، تصویر در یک پوشه ذخیره شده و سپس نمایش داده می‌شود. به این صورت تصویر را می‌توان در هر جایی از صفحه بدون ارجاع کاربر به صفحه دوم نمایش داد. البته باید دقت داشت که یوزر asp.net باید دسترسی write را بر روی این فولدر که در اینجا files نام دارد، داشته باشد.
راه دیگر انجام اینکار استفاده از http handlers است. نحوه پیاده سازی این تکنیک را در این مقاله می‌توانید مشاهده نمائید.

نظرات خوانندگان

نویسنده: hajloo
تاریخ: ۱۳۸۷/۰۹/۱۰ ۱۳:۵۵:۰۰

خیلی استفاده کردم . واقعا مفید بود. اما نم فکر نمی کنم که این سایت فونتی رو روی سیستم کلاینتها نصب کنه. تنها اتفاقی که می افته اینه که یک متن رو به عنوان ورودی می گیره و بعد (با استفاده از فونتی که روی سرور نصب کرده) یک عکس تولید میکنه .
اما در هر صورت مطلب شما خیلی خوب بود .

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۱۰ ۱۴:۰۸:۰۰

بله. روی کلاینت چیزی نصب نمی‌کنند. همین بحث بارگذاری پویای فونت را (در سمت سرور) احتمالا پیاده سازی کرده‌اند که روش آن بحث شد.
یا اگر ساده‌تر عمل کرده باشند می‌شود همان روش نصب فونت روی سرور که در این روش بالا شما انعطاف بیشتری خواهید داشت و محدود به فونت‌های نصب شده نخواهید بود. هر فونتی را می‌شود بارگذاری کرد. یا حتی تولید captcha با یک سری فونت مخصوص هم به این روش ساده‌تر خواهد شد (بدون نیاز به هماهنگی با هاست برای نصب فونت).

نویسنده: Alex's Blog
تاریخ: ۱۳۸۷/۰۹/۱۱ ۰۷:۲۰:۰۰

ممنون

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۴/۲۰ ۱۱:۵۵:۲۶

با سلام و تشکر از مقاله مفیدتان .
لطفا این پروژه را طوری اصلاح کنید که عکسها runtime ایجاد و نمایش داده شوند و هیچ عکسی موقع تولید ، ذخیره نشود ! بلکه فقط در حین ایجاد ، به کلاینت فرستاده شود .
ممنون .

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۲۰ ۱۳:۰۱:۰۸

سلام،
نکته آن‌را اینجا گفته‌ام:
http://www.dotnettips.info/2009/05/blog-post_27.html

عنوان: **غیرفعال سازی کش مرورگر در مورد تصاویر ایستا**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۱۲ ۱۶:۳۶:۵۱
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

امروز هنگام نمایش تصاویر در یک گرید با مشکل زیر مواجه شده بودم. (گرفتار شده بودم!!)
فرض کنید یک فرم درست کرده‌اید که یک تصویر به همراه توضیحاتی توسط آن ثبت می‌شود. جهت اطمینان خاطر ثبت کننده، تصاویر ثبت شده در یک گرید پائین فرم نمایش داده خواهند شد. نام این تصاویر همواره ثابت است (برای مثال نام تصویر تشکیل شده است از یک شماره پروژه بعلاوه پسوند فایل).
مشکلی در ثبت تصاویر یا توضیحات آنها وجود نداشت. مشکل در این بود که پس از ویرایش رکورد و انتخاب تصویری جدید و ثبت آن، همان تصویر قبلی در مرورگر نمایش داده می‌شد. این مورد را می‌شد با فشردن دکمه‌های ctrl+f5 حل کرد (به روز کردن کش مرورگر)، اما این راه حل اصولی حل این مشکل نیست. (تصورش را بکنید که به کاربر گفته شود پس از هر بار فشردن دکمه ثبت، یکبار هم دکمه‌های ctrl+f5 را فشار دهید!)
راه حل‌های استاندارد غیرفعال کردن کش در ASP.Net هم هیچکدام افاقه نکردند. ترکیبی از موارد زیر در page_load صفحه تست شدند:

```
//do not cache...
           Response.CacheControl = "no-cache";
           Response.AddHeader("Pragma", "no-cache");
           Response.Expires = -1;
           Response.Cache.SetNoStore();
           Response.Cache.SetCacheability(HttpCacheability.NoCache);
```

چون نام تصاویر تغییری نمی‌کرد، برای مثال 1234 همیشه همان 1234 باقی می‌ماند (صرفنظر از محتوای جدید آن)، مرورگر این تصویر غیرپویا را کش می‌کرد و فقط با ctrl+f5 این کش به روز می‌شد. (روش فوق در مورد غیرفعال کردن کش کردن یک صفحه پویای ASP.Net مؤثر است (برای مثال توصیه می‌شود که کش کردن صفحات لاگین را حتما به این صورت غیرفعال کنید)، اما در مورد اشیاء غیرپویای صفحه مطابق آزمایش من اثری نداشت)

این مشکل به صورت زیر حل شد: (یک ستون GridView است)

```
<asp:TemplateField HeaderText="لوگو">
 <ItemTemplate>
     <asp:Image ID="Image2" runat="server"
         ImageUrl='<%# "pics/"+Eval("filename")+"?uid="+Guid.NewGuid().ToString("N") %>' />
 </ItemTemplate>
 <ItemStyle HorizontalAlign="Center" />
</asp:TemplateField>
```

یک عبارت منحصر بفرد به صورت کوئری استرینگ به انتهای نام تصویر اضافه شد. مشکلی هم در نمایش تصویر ایجاد نمی‌کند، مرورگر آن‌را یک تصویر جدید به حساب آورده و دوباره همان تصویر قبلی موجود در کش را نمایش نخواهد داد. (با IE و فایرفاکس تست شد و اینجا دیگر مهم نیست که وضعیت تنظیمات به روز رسانی کش مرورگرهای تک تک کاربران به چه صورتی است و آیا باید نگران همه‌ی آنها بود یا خیر یا این‌که اصلا ارزش آن‌را دارد که برای یک صفحه کلا کش مرورگر را غیرفعال کرد؟)

زمانیکه GridView رندر شود، تصویر ما به صورت زیر خواهد بود:

```
<img id="GridView1__ctl2_Image2" src="pics/123456.PNG?uid=2a2c4247ae264ef49be46a2436ae03c9" border="0"
/>
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۱۳ ۱۸:۲۶:۰۰

آقای راد هم به نظر قبلا با این مشکل مواجه شده بودند، البته از نوعی دیگر.
http://brad.barnamenevis.org/?p=120

عنوان: **تهیه قالب برای ایمیل‌های ارسالی یک برنامه ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۱۸ ۱۳:۵۰:۰۴
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

فرض کنید ایمیل اطلاع رسانی برنامه ASP.Net شما قرار است ایمیل زیر را پس از تکمیل یک فرم ارسال کند.

با سلام
احتراما آقاي/خانم [lblName] درخواست چاپ [lblDesc] داراي
[lblNumber] صفحه را داده‌اند. لطفا جهت تائید درخواست ایشان به
برنامه مراجعه بفرمائید.

با تشکر

برای ارسال این قالب که مطابق تصویر هر بار باید سه برچسب آن تغییر کند چه راهی را پیشنهاد می‌دهید؟

راه اول: (راه متداول)
این فرم را در یک html editor درست کرده و جای سه برچسب را خالی می‌گذاریم. سپس html مورد نظر را در تابع ارسال ایمیل خود به صورت یک رشته تعریف نموده و جاهای خالی را پر خواهیم کرد. مثلا:

```
string Name = "علی";
           string Desc = "منابع مورد نیاز";
           int Number = 10;
           string content =
           "<div dir=\"rtl\" style=\"text-align: right; font-family:Tahoma; font-size:9pt\">" +
           "با سلام<br />" +
           "احتراما آقای/خانم" +
           Name +
           " درخواست چاپ" +
           Desc +
           " دارای" +
           Number +
           " صفحه را داده‌اند. لطفا جهت تائید درخواست ایشان به برنامه مراجعه بفرمائید.<br />" +
           "<br />" +
           "با تشکر</div>";
```

ایرادات:
الف) امکان مشاهده شکل نهایی تا زمانیکه ایمیل مورد نظر را دریافت نکرده باشیم، وجود ندارد.
ب) اعمال تغییرات جدید به این فرمت رشته‌ای مشکل است. همیشه استفاده از ابزارهای بصری برای بهبود کار کمک بزرگی هستند که در این حالت از آن‌ها محروم خواهیم شد.
ج)اگر تغییر رسیده جدید، درخواست اضافه کردن لیست پرینت‌های قبلی این شخص بود چه باید کرد؟ آیا جدول مورد نظر را باید به صورت دستی ایجاد و باز هم به صورت یک رشته به این مجموعه اضافه کرد؟ در این حالت از کنترل‌های استانداردی مانند GridView و امثال آن محروم خواهیم شد.
د) هر بار تغییر، نیاز به recompile برنامه دارد.
بنابراین همانطور که مشاهده می‌کنید، نگهداری این روش مشکل است.

راه دوم: استفاده از قالب‌ها

خوشبختانه در ASP.Net امکان رندر کردن کنترل‌ها به صورت یک string نیز موجود است . در مثال ما نیاز است تا چندین کنترل در کنار هم قرار گیرند تا شکل نهایی را ایجاد کنند. بنابراین می‌توان تمام آنها را در یک یوزر کنترل قرار داد. سپس باید کل یوزر کنترل را به صورت یک رشته، رندر کرد که در ادامه به آن خواهیم پرداخت.

اگر قالب فوق را بخواهیم در یک یوزر کنترل طراحی کنیم، سورس صفحه html یوزر کنترل به صورت زیر خواهد بود (فایل WebUserControl1.ascx) :

```
<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="WebUserControl1.ascx.cs"
   Inherits="testWebForms87.WebUserControl1" %>
<div dir="rtl" style="text-align: right; font-family:Tahoma; font-size:9pt">
   با سلام<br />
   احتراما آقای/خانم
   <asp:Label ID="lblName" runat="server"></asp:Label>
    درخواست چاپ
   <asp:Label ID="lblDesc" runat="server"></asp:Label>
    دارای
   <asp:Label ID="lblNumber" runat="server"></asp:Label>
    صفحه را داده‌اند. لطفا جهت تائید درخواست ایشان به برنامه مراجعه بفرمائید.<br />
   <br />
   با تشکر</div>
```

همچنین می‌توان مقادیر برچسب‌ها را از طریق خواصی که برای یوزر کنترل تعریف خواهیم کرد، در تابع ارسال ایمیل خود مقدار دهی نمائیم. در این حالت در سورس صفحه یوزر کنترل داریم (فایل WebUserControl1.ascx.cs) :

```
public string Name { get; set; }
       public int Number { get; set; }
       public string Desc { get; set; }

       protected void Page_Load(object sender, EventArgs e)
       {
           lblNumber.Text = Number.ToString();
           lblName.Text = Name;
           lblDesc.Text = Desc;
       }
```

تا اینجا طراحی اولیه محتوای ایمیل به پایان می‌رسد. دیگر از آن رشته کذایی خبری نیست و همچنین می‌توان از designer ویژوال استودیو برای طراحی بصری قالب مورد نظر استفاده کرد که این مزیت بزرگی است و در آینده اگر نیاز به تغییر متن ایمیل ارسالی وجود داشت، تنها کافی است فایل ascx ما ویرایش شود (بدون نیاز به کامپایل مجدد پروژه). یا در اینجا به سادگی برای مثال می‌توان یک GridView را تعریف، طراحی و bind کرد.

مرحله بعد، رندر کردن خودکار این یوزر کنترل و سپس تبدیل محتوای حاصل به یک رشته است. برای این منظور از تابع زیر می‌توان کمک گرفت (برای مثال تعریف شده در کلاس دلخواه CLoadUC) :

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Reflection;
using System.Web;
using System.Web.UI;

      /// <summary>
      /// تبدیل یک یوزر کنترل به معادل اچ تی ام ال آن
      /// </summary>
      /// <param name="path">مسیر یوزر کنترل</param>
      /// <param name="properties">لیست خواص به همراه مقادیر مورد نظر</param>
      /// <returns></returns>
      /// <exception cref="NotImplementedException"><c>NotImplementedException</c>.</exception>
       public static string RenderUserControl(string path,
             List<KeyValuePair<string,object>> properties)
       {
```

```
            Page pageHolder = new Page();

            UserControl viewControl =
               (UserControl)pageHolder.LoadControl(path);

            Type viewControlType = viewControl.GetType();

            foreach (var pair in properties)
            {
                PropertyInfo property =
                   viewControlType.GetProperty(pair.Key);

                if (property != null)
                {
                    property.SetValue(viewControl, pair.Value, null);
                }
                else
                {
                    throw new NotImplementedException(string.Format(
                       "UserControl: {0} does not have a public {1} property.",
                       path, pair.Key));
                }
            }

            pageHolder.Controls.Add(viewControl);
            StringWriter output = new StringWriter();
            HttpContext.Current.Server.Execute(pageHolder, output, false);
            return output.ToString();
        }
```

ماخذ اصلی تابع فوق این آدرس است.

که البته تابع نهایی آنرا کمی اصلاح کردم تا بتوان لیستی از خواص پابلیک یک یوزر کنترل را به آن پاس کرد و محدود به یک خاصیت نبود.

اکنون استفاده از یوزر کنترلی که تاکنون طراحی کرده‌ایم به سادگی زیر است:

```
List<KeyValuePair<string, object>> lst =
               new List<KeyValuePair<string, object>>
                     {
                         new KeyValuePair<string, object>("Name", "علی"),
                         new KeyValuePair<string, object>("Number", 10),
                         new KeyValuePair<string, object>("Desc", "منابع مورد نیاز")
                     };

            string content = CLoadUC.RenderUserControl("WebUserControl1.ascx", lst);
```

ابتدا لیستی از خواص پابلیک یوزر کنترل تهیه شده و مقدار دهی می‌شوند . سپس مسیر فایل یوزر کنترل به همراه این لیست به تابع رندر کردن نهایی پاس خواهند شد. حاصل، رشته html محتوای ایمیل ما را تشکیل خواهد داد.

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۸/۰۵/۱۴ ۱۳:۱۸:۵۹

توی برنامه های تحت ویندوز چطور؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۴ ۱۳:۳۸:۰۳

- یک فایل html درست کنید و جاهای مورد نظر را replace کنید. (قالب درست کنید)
- حالت پیشرفته‌تر استفاده از xsl transformation است که خود مایکروسافت در محصولاتش زیاد استفاده می‌کنه.

عنوان: **مطلع شدن از خطاهای مدیریت نشده یک برنامه ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۲۴ ۱۸:۳۶:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

راه‌های زیادی برای لاگ کردن خطاهای حاصل در یک برنامه ASP.Net وجود دارند. از روش‌های exception handling معمول تا افزودن یک فایل global.asax به برنامه و دریافت و لاگ کردن خطاهای مدیریت نشده توسط روال رخ داد گردان Application_Error آن.
بررسی این خطاها فوق العاده مهم است ، حداقل به دو دلیل : الف) قبل از این که کاربران به شما بگویند برنامه مشکل پیدا کرده، از طریق ایمیل دریافتی مطلع خواهید شد. (فرض کنید علاوه بر ثبت وقایع ، آنها را ایمیل هم می‌زنید) این مورد در جهت بالا بردن کیفیت کار تمام شده واقعا مؤثر است. ب) رفتارهای مخرب را هم بهتر می‌توانید تحت نظر داشته باشید.

تمام این موارد مستلزم کد نویسی است. دریافت خطا در روال Application_Error و سپس کد نویسی برای ارسال ایمیل. از ASP.Net 2.0 به بعد این کار را بدون کد نویسی و با استفاده از امکانات ASP.NET health monitoring نیز می‌توان به سادگی و دقت هرچه تمامتر انجام داد.

کار زیادی را قرار نیست انجام دهید! فایل وب کانفیگ سایت را باز کنید و چند سطر زیر را به آن اضافه کنید (قسمت healthMonitoring و همچنین قسمت mailSettings):

```
<?xml version="1.0"?>
<configuration>
 <appSettings/>
 <connectionStrings/>
 <system.web>
   <compilation debug="true">
   </compilation>
   <authentication mode="Windows"/>

   <healthMonitoring enabled="true">
     <providers>
       <add name="EmailProvider"
            type="System.Web.Management.SimpleMailWebEventProvider"
            from="you@domain.com"
            to="you@domain.com"
            subjectPrefix="Error: "
            buffer="true"
            bufferMode="Notification"/>
     </providers>
     <rules>
       <add
         provider="EmailProvider"
         name="All App Events"
         eventName="All Errors"/>
     </rules>
   </healthMonitoring>

 </system.web>

 <system.net>
   <mailSettings>
     <smtp deliveryMethod="SpecifiedPickupDirectory">
       <specifiedPickupDirectory pickupDirectoryLocation="C:\emails"/>
     </smtp>
   </mailSettings>
 </system.net>

</configuration>
```

در این مثال قسمت mailSettings طوری تنظیم شده که ایمیل ارسالی در مسیر c:\emails جهت مرور نحوه عملکرد این سیستم، ذخیره شود.

در حالت اجرا بر روی یک سرور ، این قسمت را می‌توان به صورت زیر تنظیم نمود و آدرس smtp server را توسط آن مشخص کرد تا به صورت خودکار مورد استفاده قرار گیرد:

```
<mailSettings>
 <smtp from="you@domain.com">
     <network host="smtp.domain.com" />
 </smtp>
</mailSettings>
```

سیستم ارسال ایمیل آن نیز هوشمند است و میل باکس شما را به یکباره پر نخواهد کرد. اگر در طی یک دقیقه (مقدار پیش فرض) خطاهای حاصل شده مانند هم باشند، تنها یک ایمیل حاوی تمامی آنها را ارسال می‌کند.

شایان ذکر است از ASP.Net 2.0 به بعد امکان ثبت وقایع در event log ویندوز محدود شده است و اگر نیاز به انجام این کار باشد باید دسترسی بیشتری را به یوزر asp.net اعطاء کرد. اما با استفاده از روش فوق، جزئیات خطای حاصل به صورت خودکار به event log ویندوز نیز اضافه می‌شود.

اگر علاقمند باشید که خطاهای حاصل را در یک دیتابیس نیز لاگ کنید، به این مقاله می‌توان رجوع کرد.

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۷/۰۹/۲۴ ۲۳:۵۱:۰۰

سلام استاد نصیری
نیما هستم همون مزاحم همیشگی.مقاله جالبی بود. من هم زمانی راجع به این موضوع تحقیق کردم. این روشی که فرمودین یه مشکلی داره و اون هم اینه که اگر میل سرور ما رو مجبور به استفاده از SSLکنه این روش جواب نمیده و باید یه پروایدر برای ارسال میل نوشت.
من نتیجه تحقیقاتم رو اینجا گذاشتم.http://www.p2p.dotnetsource.com/Default.aspx?g=posts&t=7080
افتخار میدین اگر نگاهی بندازین بهش و نظرتون رو بگین.موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۰۹/۲۵ ۰۱:۱۹:۰۰

سلام
ممنون. خیلی عالی و جامع بود.

درسته، چیزی شبیه به این مورد برای gmail
http://forums.asp.net/t/1124257.aspx

نویسنده: بهروز
تاریخ: ۱۳۸۷/۰۹/۲۵ ۱۲:۲۲:۰۰

تا اومدم همون چیزی رو که نیما گفته بگم، دیدم که نیما گفته!
ضمن اینکه برخی هاست ها ارسال ایمیل با کلاس های دات نت رو فراهم نمی کنن و در نتیجه باید یک پرووایدر سفارشی ساخت.

عنوان: **بدست آوردن اندازه ViewState یک صفحه ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۱۱ ۲۱:۴۲:۰۱
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

یکی از روش‌های معمولی که برای بدست آوردن اندازه ViewState یک صفحه در ASP.Net بکار می‌رود، نمایش صفحه، مراجعه به سورس آن و سپس ذخیره مقدار فیلد مخفی ViewState به صورت یک فایل متنی و مراجعه به اندازه آن فایل است!
راه بهتری هم وجود دارد که به شرح زیر است:
زمانیکه قرار است ViewState به صفحه اضافه شود، روال رخداد گردان SavePageStateToPersistenceMedium اجرا خواهد شد.
نحوه تحریف آن برای بدست آوردن اندازه ViewState به صورت زیر است (چند سطر زیر را به وب فرم خود اضافه کنید):

```
using System;
using System.Diagnostics;
using System.IO;
using System.Web.UI;

       protected override void SavePageStateToPersistenceMedium(object state)
       {
           if (Debugger.IsAttached)
           {
               using (MemoryStream stream = new MemoryStream())
               {
                   LosFormatter formatter = new LosFormatter();
                   formatter.Serialize(stream, state);
                   int size = stream.Capacity / 1024;
                   if (size > 32)
                   {
                       throw new Exception("لطفا به وضعیت ویوو استیت این صفحه رسیدگی نمائید");
                   }
               }
           }
           base.SavePageStateToPersistenceMedium(state);
       }
```

در اینجا اگر در حال دیباگ برنامه باشید و همچنین حجم ViewState از 32 کیلوبایت بیشتر شد، استثنایی جهت گوشزد گردن این مطلب به شما، تولید می‌شود.
اگر علاقمند بودید که این روال را به تمامی صفحات اضافه کنید می‌شود یک کلاس جدید ایجاد کرد و این کلاس را از کلاس استاندارد Page به ارث برد (برای نمونه public class MyPage : Page) . سپس همان چند سطر فوق را به آن اضافه نمائید.
اکنون تغییری که در کدها باید صورت گیرد به صورت زیر است، برای مثال تبدیل:

```
public partial class WebForm4 : System.Web.UI.Page
```

به :

```
public partial class WebForm4 : MyPage
```

نظرات خوانندگان

نویسنده: مهرداد
تاریخ: ۱۳۸۷/۱۰/۱۱ ۲۳:۱۱:۰۰

میشه یک توضیح درباره رویداد SavePageStateToPersistenceMedium اینجا بنویسید .

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۱۱ ۲۳:۱۹:۰۰

هدف اصلی از معرفی این روال رویدادگردان این است که شما بتونید یک ViewState manager سفارشی برای خودتون تهیه کنید.
یعنی چی؟
در روال SavePageStateToPersistenceMedium کل ViewState ایی که قرار است به صفحه به صورت یک فیلد مخفی اضافه شود دراختیار شما خواهد بود. برای مثال آن‌را در دیتابیس ذخیره کنید. دیگر هیچ مقداری به فیلد مخفی ViewState اضافه نمی‌شود.
حالا در روال رویدادگردان LoadPageStateFromPersistenceMedium این ViewState ذخیره شده در دیتابیس را بازیابی کنید و مقدار آنرا بازگشت دهید.
به این صورت صفحه سبک می‌شود. ViewState هم در سمت سرور قابل ذخیره شدن خواهد بود.
برای مثال مقاله زیر از سشن برای حذف ViewState استفاده کرده :
http://www.dotnetjunkies.com/WebLog/whoiskb/archive/2005/06/23/128346.aspx

نویسنده: Anonymous
تاریخ: ۱۳۸۷/۱۰/۱۲ ۰۰:۲۹:۰۰

اینو می دونیم که هنگام پستبک یک صفحه باید ویو استیت به سرور آپلود شود که این مستلزم زمان و هزینه است.
آیا امکان فشرده سازی استیت، برای کاهش حجم ارسالی آن وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۱۲ ۰۱:۱۶:۰۰

یک نمونه فشرده سازی ViewState سازگار با ASP.Net Ajax
http://blog.sb2.fr/post/2008/11/18/Ajax-Enabled-ViewState-Compression-for-ASPNET.aspx

نویسنده: مهدی
تاریخ: ۱۳۸۷/۱۰/۱۴ ۲۳:۳۵:۰۰

مرسی از جوابتون البته مقاله french بود که زحمت ترجمه رو دادم به سرویس translate گوگل.
راستی آقای نصیری به نظرم جای Ajax تو برجسب ها خالیه. فکر کنم بقیه دوستان هم خوشحال می شن اگه راجب ajax از شما بشنوند!
مگه نه بچه ها؟

عنوان: **نمایش حداکثر اندازه مجاز فایل قابل آپلود به کاربر، در ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۳ ۱۲:۰۴:۰۶
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

گاهی از اوقات قبل از درگیر شدن با کاربران (!)، بهتر است حداکثر اندازه مجاز فایل قابل ارسال به سرور را به آن‌ها نمایش داد. درغیراینصورت باید پاسخگوی این باشید که چرا فایل 100 مگابایتی که من ارسال کردم، ذخیره نشده و برنامه کار نمی‌کنه!
خطای دریافتی این خواهد بود: Maximum request length exceeded
در ASP.Net اگر هیچ تنظیم خاصی صورت نگرفته باشد، حداکثر اندازه فایل قابل ارسال به سرور، 4 مگابایت است. این مورد را در machine.config و یا در web.config می‌توان تغییر داد .
برای مثال، جهت بالا بردن اندازه فایل قابل ارسال به سرور در وب کانفیگ برنامه به 39 مگابایت، می‌توان سطر زیر را به قسمت system.web اضافه کرد.

```
<httpRuntime executionTimeout="1200" maxRequestLength="39936" />
```

البته در این حالت بهتر است executionTimeout را نیز تنظیم نمود (بر اساس ثانیه) تا یک فایل حجیم را بتوانند آپلود کنند و در این حین مشکل timeout رخ ندهد (در اینجا به 20 دقیقه تنظیم شده است).

اما یک نکته را هم باید درنظر داشت. اگر هاست مورد استفاده شما فایل machine.config را قفل کرده باشد (که از لحاظ امنیتی توصیه می‌شود)، سطر فوق در web.config هیچ تاثیری نخواهد داشت.

به همین منظور کلاس زیر را تهیه کرده‌ام که تمامی این موارد را لحاظ می‌کند.
ابتدا مقدار پیش فرض 4 مگابایت درنظر گرفته خواهد شد.
سپس سعی می‌شود که مقدار مجاز MaxRequestLength از فایل machine.config خوانده شود. همچنین وضعیت قفل بودن آن نیز دریافت می‌شود.
اگر این قسمت قابل خواندن بود و همچنین قفل نشده بود، مقدار تنظیم شده maxRequestLength در وب کانفیگ، دریافت و استفاده خواهد شد.
و در آخر، اندازه دریافتی، که بر اساس KB است به شکلی قابل خواندن بازگشت داده می‌شود.

```
using System;
using System.Configuration;
using System.Web.Configuration;

/// <summary>
/// کلاسی جهت نمایش اندازه مجاز فایل قابل ارسال به سرور
/// </summary>
public class CMaxLimit
{
  /// <summary>
  /// اندازه مجاز فایل قابل ارسال به سرور
  /// </summary>
  /// <returns></returns>
  public static string MaxFileUploadSizeLimit()
  {
      //مقدار پیش فرض
      int resultKB = 4096;

      //machine.config
      Configuration mConfig =
          WebConfigurationManager.OpenMachineConfiguration();
      bool mConfigIsLocked = false;
      HttpRuntimeSection section =
          mConfig.GetSection("system.web/httpRuntime") as HttpRuntimeSection;
      if (section != null)
      {
```

```
            resultKB = section.MaxRequestLength;
            mConfigIsLocked = section.ElementInformation.IsLocked;
        }

        //web.config
        if (!mConfigIsLocked)
        {
            HttpRuntimeSection httpRuntimeSection =
                WebConfigurationManager.GetSection("system.web/httpRuntime") as HttpRuntimeSection;
            if (httpRuntimeSection != null)
            {
                resultKB = httpRuntimeSection.MaxRequestLength;
            }
        }

        return
             SizeToString(resultKB * 1024);
    }

    /// <summary>
    /// نمایش اندازه یک فایل به صورتی قابل درک
    /// </summary>
    /// <param name="len">اندازه فایل</param>
    /// <returns></returns>
    public static string SizeToString(long len)
    {
        int order = 0;
        string[] sizes = new[] { "B", "KB", "MB", "GB" };
        while (len >= 1024 && order + 1 < sizes.Length)
        {
            order++;
            len = len / 1024;
        }
        return String.Format("{0:0.##} {1}", len, sizes[order]);
    }
}
```

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۷/۱۰/۲۳ ۱۴:۰۱:۰۰

سلام استاد نصیری
در این موردی که فرمودین اگر machin.config قفل باشه اون خط تاثیر گذار نخواهد بود آیا وقتی که قبلش به تگ clear استفاده کنیم هم باز تاثیرگذار نیست؟چون تگ clear تنظیمات والد رو بی تاثیر میکنه.
ممنون از مطلب مفیدتون
موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۳ ۱۵:۲۰:۰۰

سلام،
بله، بی اثر خواهد کرد اگر allowOverride که برای قفل کردن ذکر شد ، به false تنظیم نشده باشد.
http://msdn.microsoft.com/en-us/library/ms178693.aspx
و همانطور که در راهنمای فوق صریحا ذکر شده، هر تلاشی برای تغییر یک section قفل شده، منجر به بروز یک خطا خواهد شد.

کلا قفل کردن ماشین کانفیگ برای هاست‌ها طراحی شده تا از یک سری موارد بتوانند جلوگیری کنند. مثلا شما نتونید با impersonation ، بجای یک یوزر ویندوز NT لوکال عمل خاصی را روی هاست اجرا کنید.
http://www.devx.com/vb2themax/Tip/18801

نویسنده: مسعود
تاریخ: ۱۳۸۷/۱۰/۲۶ ۲۰:۴۵:۰۰

جالب بود استاد.

فقط یه چیزی به نظرت با این کد میشه کل بخش آپلود رو کنترل کرد.

به نظرم مقطعی کار کنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۰/۲۶ ۲۱:۱۷:۰۰

سلام،
نمی‌دونم دقیقا منظور شما چیست؟
این کد فقط عبارات درج شده در فایل‌های کانفیگ رو استخراج و نمایش میده. کار خاص دیگری نمیکنه.

عنوان: **بررسی وجود نام کاربر با استفاده از jQuery Ajax در ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۲۹ ۱۳:۱۸:۰۱
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

شاید بعضی از سایت‌ها را دیده باشید که در حین ثبت نام، پس از وارد کردن یک نام کاربری و سپس مشغول شدن به پر کردن فیلد کلمه‌ی عبور، در قسمت نام کاربری شروع به جستجو در مورد آزاد بودن نام کاربری درخواستی می‌کنند یا نمونه‌ای دیگر، فرم پرداخت الکترونیکی بانک سامان. پس از اینکه شماره قبض را وارد کردید، بلافاصله بدون ریفرش صفحه به شما پیغام می‌دهد که این شماره معتبر است یا خیر. امروز قصد داریم این قابلیت را با استفاده از کتابخانه‌ی Ajax مجموعه jQuery در ASP.Net پیاده سازی کنیم (بدون استفاده از ASP.Net Ajax مایکروسافت).

ابتدا سورس کامل را ملاحظه نمائید:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="AjaxTest.aspx.cs"
Inherits="testWebForms87.AjaxTest" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
<title>jQuery Ajax Text</title>

<script src="jquery.min.js" type="text/javascript"></script>
<script type="text/javascript">
    $(document).ready(function() {
        $("#<%= TextBox1.ClientID %>").blur(function(event) {
            $.ajax({
                type: "POST",
                url: "AjaxTest.aspx/IsUserAvailable",
                data: "{'username': '" + $('#<%= TextBox1.ClientID %>').val() + "'}",
                contentType: "application/json; charset=utf-8",
                dataType: "json",
                success: function(msg) {
                    $('#valid').html("<img src='ajaxImages/waiting.gif' alt='لطفا کمی تامل کنید'>");
                    var delay = function() {
                        AjaxSucceeded(msg);
                    };

                    setTimeout(delay, 2000); //remove this
                },
                error: AjaxFailed
            });
        });
    });
    function AjaxSucceeded(result) {
        if (result.d == true)
            $('#msg').html("<img src='ajaxImages/available.gif' alt='نام کاربری درخواستی موجود
است'>");
        else
            $('#msg').html("<img src='ajaxImages/taken.gif' alt='متاسفانه نام کاربری مورد نظر پیشتر
دریافت شده‌است'>");
    }
    function AjaxFailed(result) {
        alert(result.status + ' ' + result.statusText);
    }
</script>

</head>
<body>
<form id="form1" runat="server">
<div>
    user name:
    <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
    <span id="msg"></span>
    <br />
    pass:
    <asp:TextBox ID="TextBox2" TextMode="Password" runat="server"></asp:TextBox>
</div>
<!-- preload -->
<div style="display: none">
    <img src="ajaxImages/available.gif" alt="available" />
    <img src="ajaxImages/taken.gif" alt="taken" />
```

```
    <img src="ajaxImages/waiting.gif" alt="waiting" />
</div>
</form>
</body>
</html>
```

```
using System;
using System.Web.Services;

namespace testWebForms87
{
public partial class AjaxTest : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
    }

    [WebMethod]
    public static bool IsUserAvailable(string username)
    {
        // این مورد را با خواندن اطلاعات از دیتابیس می‌شود تعویض کرد
        return username != "test";
    }
}
}
```

توضیحات:

همانطور که ملاحظه می‌کنید صفحه‌ی ASP.Net ما بسیار ساده است و از دو تکست باکس استاندارد تشکیل می‌شود، به همراه تصاویر مربوط به Ajax که یک سری تصاویر ساده چرخان معروف منتظر بمانید ، یافت شد یا موجود نیست می‌باشند. این تصاویر در یک div مخفی (display: none) در صفحه قرار گرفته‌اند و در هنگام بارگذاری صفحه، این‌ها نیز بارگذاری شده و حاضر و آماده خواهند بود. بنابراین هنگام استفاده از آن‌ها، کاربر تاخیری را مشاهده نخواهد کرد. همچنین یک span با id مساوی msg را هم پس از تکست باکس اضافه کرده‌ایم تا تصاویر مربوط به رخدادهای Ajax را با استفاده از توانایی‌های jQuery به آن اضافه کنیم.

اسکریپت Ajax ما با دراختیار گرفتن روال رخداد گردان blur شیء textBox1 شروع می‌شود. همانطور که در مقالات پیشین سایت نیز ذکر شد، روش صحیح دریافت ID یک کنترل ASP.Net در کدهای سمت کلاینت جاوا اسکریپتی، بر اساس خاصیت ClientID آن است که در اولین سطر کدهای ما مشخص است (زیرا در ASP.Net نام و ID یک کنترل در هنگام رندر شدن به همراه ID کنترل‌های دربرگیرنده آن نیز خواهد بود، بنابراین بهتر است این مورد را داینامیک کرد).

کار بررسی موجود بودن نام کاربری (یا مثلا یک شماره قبض و امثال آن) توسط WebMethod ایی به نام IsUserAvailable در code behind صفحه انجام می‌شود که پیاده سازی آن‌را ملاحظه می‌کنید. بدیهی است در این مثال ساده، تنها نام کاربری از پیش رزرو شده، کلمه‌ی test است و در یک کد واقعی این مورد با مقایسه‌ی نام کاربری با اطلاعات موجود در دیتابیس باید صورت گیرد (و حملات تزریق اس کیوال را هم فراموش نکنید. برای رهایی از آن‌ها " حتما " باید از پارامترهای ADO.Net استفاده کرد و گرنه کد شما مستعد به این نوع حملات خواهد بود).

سؤال: چرا از web method استفاده شد و همچنین چرا این متد static است؟

زمانیکه یک متد با کلمه کلیدی static مشخص می‌شود حالت state less پیدا می‌کند یعنی مستقل از وهله‌ی کلاس عمل می‌کند. در این حالت نیازی به ارسال ViewState نبوده (بنابراین در کوئری مورد نظر ما بسیار بهینه و سبک عمل می‌کنند) و همچنین نیازی به ایجاد یک وهله‌ای از کلاس صفحه‌ی ما نیز نخواهد بود. برای توضیحات بیشتر به این مقاله مراجعه نمائید. (به صورت خلاصه، دلیل اصلی، کارآیی بالا و بهینه بودن این روش در این مساله ویژه است و در ASP.Net Ajax مایکروسافت به صورت گسترده‌ای در پشت صحنه مورد استفاده قرار می‌گیرد)

استفاده از ویژگی WebMethod عملکرد صفحه‌ی ما را شبیه به یک وب سرویس خواهد کرد و امکان دسترسی به آن در متدهای استاندارد POST به صورت ارسال دیتا به آدرس WebService.asmx/WebMethodName خواهد بود. یک مثال ساده و عملی

بررسی تابع Ajax بکار رفته:
این تابع هنگام فراخوانی رخداد blur تکست‌باکس ما (مطابق کد فوق) فراخوانی می‌شود. ساختار ساده‌ای دارد که به شرح زیر است:

"type: "POST نحوه‌ی ارسال داده را به متد وب سرویس ما مشخص می‌کند.

"url: "AjaxTest.aspx/IsUserAvailable اطلاعات پست شده، به صفحه‌ی AjaxTest.aspx و وب متد IsUserAvailable ارسال خواهد شد

","{'" + ()data: "{'username': '" + $('#<%= TextBox1.ClientID %>').val داده‌ای که به آرگومان username ما ارسال می‌شود، همان مقدار تایپ شده در TextBox1 است (که باز هم دریافت ID آن به صورت داینامیک صورت گرفته تا مشکل زا نشود)

,"contentType: "application/json; charset=utf-8
,"dataType: "json در این دو سطر از نوع داده‌ی json استفاده شده است که فرمت بسیار سبک و بهینه‌ای برای تبادل اطلاعات در وب به‌شمار می‌آید و توسط کتابخانه‌های جاوا اسکریپتی به سادگی پردازش شده و تبدیل به اشیاء مورد نظر خواهند شد. برای مثال اگر خروجی یک وب سرویس در حالت xml به صورت زیر باشد:

```
<xx yy="nn"></xx>
```

معادل json آن به شرح زیر است:

```
{ "xx": {"yy":"nn"} }
```

Success **success: function(msg)** ، پس از موفقیت آمیز بودن عملیات ajax در jQuery فراخوانی می‌شود

error: AjaxFailed و اگر در این بین خطایی رخ داده باشد، قسمت error فراخوانی می‌شود.

در این مثال برای نمایش بهتر عملیات، یک وقفه‌ی 2 ثانیه‌ای توسط setTimeout ایجاد شده و بدیهی است در یک مثال واقعی باید آن‌را حذف نمود.

نکته: با استفاده از افزونه‌ی فایرباگ فایرفاکس، می‌توان جزئیات این عملیات را بهتر مشاهده نمود:

user name: test1
pass:
Inspect Clear Profile
Console HTML CSS Script DOM Net Cookies Reference Pixel Perfect YSlow
POST http://localhost:9884/AjaxTest.aspx/IsUserAvailable 200 OK 42ms
Headers Post Response JSON
{'username': 'test1'}

user name: test1
pass:
Inspect Clear Profile
Console HTML CSS Script DOM Net Cookies Reference Pixel Perfect YSlo
POST http://localhost:9884/AjaxTest.aspx/IsUserAvailable 200 OK 42ms
Headers Post Response JSON
{"d":true}

نظرات خوانندگان

نویسنده: مسعود
تاریخ: ۱۳۸۷/۱۱/۲۹ ۲۲:۳۸:۰۰

سلام استاد.

من برام یه سوال پیش اومده ،

من چند وقته ذهنم درگیر این شده که آیا واقعا vb.net متد های Static رو داره یا نه!

من سر این با خیلی ها بحث کردم.

بعضی ها میگن Modual جایگزینش شده ، بعضی ها متد های Shared رو جایگزین می دونن.

نظر شما چیه؟

راستی شما روی jQuery Ajax کار میکنید؟
من خیلی دوست دارم کار کردن باهاش رو.

دارم یه فریم ورک کوچولو برای Jquery Ajax برای دات نت می سازم.البته پر از ارور :دی

نویسنده: وحید نصیری
تاریخ: ۱۳۸۷/۱۱/۳۰ ۰۱:۰۴:۰۰

سلام
بله. برنامه reflector معروف که مد نظر هست. اسمبلی برنامه فوق را با آن باز کردم و زبان انتخابی هم VB . حاصل به صورت زیر نمایش داده شد:

```
Public Shared Function IsUserAvailable
```

یک سری کار سورس باز دیدم در CodePlex در مورد jQuery Ajax که تلاش‌هایی برای ساده‌تر کردنش کردند. بد نیست برای تکمیل کار یک سری هم به آن‌ها هم بزنید. برای ایده گرفتن خوب است.

نویسنده: میلاد
تاریخ: ۱۳۸۸/۰۵/۰۱ ۱۰:۲۱:۱۳

سلام
اگه بخوایم دو پارامتر ارسال کنیم چی ؟!!

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۰۱ ۱۰:۳۰:۴۶

محدودیتی ندارد. وب متد را اصلاح کنید تا دو پارامتر مورد نظر را قبول کند. همچنین قسمت data در اسکریپت را که با فرمت json باید ارسال شود را نیز اصلاح کنید. مثلا

```
;'{var json = '{"param1":"val1", "param2":"val2", "param3":number3
```

نویسنده: mat21
تاریخ: ۱۳۸۸/۱۰/۰۹ ۱۱:۵۹:۵۷

با سلام و تشکر
من مثال فوق را استفاده کردم اما همیشه قسمت error به اجرا در میاد و error status 200 را بر می گرداند اما زمانی که "datatype :"json رااز کد حذف می کنم قسمت success اجرا می شه اما نتیجه درست نیست.
من از framework 2.0 استفاده می کنم، ممنون می شم راهنمایم کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۰۹ ۱۴:۲۰:۴۴

سلام
در مورد دات نت 2 به آخرین کامنت بنده در این مقاله مراجعه بفرمائید (انتهای صفحه):
http://www.dotnettips.info/2009/07/blog-post_06.html

نویسنده: maryam
تاریخ: ۱۳۸۹/۰۷/۲۰ ۱۲:۳۲:۵۳

من این برنامه رو تست کردم همیشه قسمت error را اجرا می کند و این پیغام می دهد
Internal Server Error 500

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۰ ۱۳:۱۷:۳۸

کمکی نمی‌تونم بهتون بکنم و من عموما پیغام‌هایی مانند پیغام شما رو حذف می‌کنم به این دلایل:
- نه کدی ارائه شده (کل Solution شما نیاز است برای بررسی)
- نه توضیحات کافی که چه مراحلی رو طی کردید
- نه توضیحاتی که از چه ابزاری و چه نگارشی از VS.NET دارید استفاده می‌کنید.

و توقع دارید که من از راه دور کد شما رو در VS.NET ایی که معلوم نیست چه نگارشی است دیباگ کنم، پس از عیب یابی مشخص کنم مشکل چی بوده ...

این‌ها رو هم نوشتم تا کسانی که پیغام‌هاشون حذف شده بدونند علت این امر چی بوده.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۰ ۱۳:۲۰:۴۱

یک مورد کلی را اضافه کنم برای دیباگ پیغام Internal Server Error 500 که عرض کردم از راه دور نمی‌شود آن‌را دیباگ کرد: افزونه‌ی فایرباگ مربوط به فایرفاکس را نصب کنید. سپس آن‌را روی صفحه جاری خودتون فعال کنید. صفحه رو ریفرش کنید و سپس یکبار مراحل رسیدن به خطا رو طی کنید. در قسمت network مربوط به firebug دقیقا اطلاعات رد و بدل شده و همچنین خطای نهایی که سیستم برگردونده لاگ می‌شود. این را بررسی کنید تا مشکل را بشود حل کرد.

نویسنده: شاهین کیاست
تاریخ: ۱۳۸۹/۱۱/۱۱ ۲۱:۴۱:۴۳

سلام آقای نصیری.
من عین کد شما را در Page تست کردم و صحیح کار کرد اما در User Control پیام Forbidden را بر می گرداند. آیا علت خاصی دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۱۱ ۲۳:۲۸:۳۶

بله. در یوزر کنترل‌ها این مورد ساپورت نمی‌شود. در تمام حالات روش صحیح و قابل نگهداری و قابل استفاده در قسمت‌های

مختلف برنامه، استفاده از یک WCF Service است (یا یک درجه پایین‌تر یک وب سرویس معمولی asmx با سرعت کمتر از WCF سرویس).

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۴/۲۱ ۱۴:۲۶

با سلام و خسته نباشید خدمت استاد نصیری
من برنامه ام رو با code first و الگوی UnitOfWork که شما مطرح کردید نوشتم که واقعاً عالی و لذت بخش هست.
مشکلی که هست اینه که اگه بخوام از این امکان jqueryAjax در برنامه خودم استفاده کنم می‌بایست userServiice تعریف شده رو در یک متد استاتیک(با توجه به مطلب بالا) استفاده کنم که اینکار انجام نمیشه.ممنون میشم کمکم کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۱ ۱۶:۴

- در MVC این مشکلات وجود ندارد و سازگاری کاملی با jQuery در آن درنظر گرفته شده. در آنجا لازم نیست چیزی رو استاتیک تعریف کنید. همچنین چون صفحات دارای ViewState هم نیستند، به سادگی می‌شود قسمتی از آن‌را با jQuery Ajax کلا تعویض کرد بدون اینکه پیغام خراب بودن ViewState را در post back به سرور، دریافت کنید. در آنجا مفهومی به نام PartialView تعریف شده که به صورت خودکار یک View را به شکل رشته برای شما رندر می‌کند و در اختیار jQuery Ajax قرار می‌دهد.
- در اینجا اگر نمی‌تونید از وهله سازی خودکار توسط DI Container استفاده کنید، می‌شود آن‌را دستی هم انجام داد. اصطلاحا به آن Service locator گفته می‌شود. نمونه آن‌را در استفاده از ObjectFactory.GetInstance در قسمت 12 دیده‌اید (استفاده از الگوی واحد کار و کلاس‌های سرویس تهیه شده در یک برنامه کنسول ویندوزی).

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۴/۲۱ ۱۶:۳۰

من فراموش کردم که اشاره کنم من از asp.net web form استفاده میکنم.
راه حل برای asp.net web form چیست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۱ ۱۶:۳۳

همان توضیحات بند دوم نظر قبلی بنده (الگوی service locator مخصوص به فناوری خاصی نیست و در صورت لزوم در همه جا می‌توان از آن استفاده کرد. یک ضد الگو است. تا حد امکان نباید از آن استفاده کرد؛ اما اگر امکان استفاده صحیح از تزریق وابستگی‌ها نبود، این روش هم کار می‌کنه).

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۴/۲۱ ۱۶:۵۶

مهندس فقط میتونم بگم " یه دونه ای! "
واقعاً ممنون

کار کرد.البته در تمامی pageها از base Page ارث بری میکنیم.این استفاده از Object Factory با توجه به استفاده از Object Factory در سازنده کلاس base page ، مشکلی ایجاد نمیکند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۱ ۱۷:۱۱

خیر. در پشت صحنه این کتابخانه، Object Factory به صورت خودکار عمل می‌کند. اگر امکان استفاده خودکار از آن نبود، روشی دستی آن (ObjectFactory.GetInstance) همان عملیات را انجام می‌دهد.

نویسنده: مهدی غفاری
تاریخ: ۱۳۹۱/۰۷/۰۸ ۱۵:۴۶

من یک پوشه ایجاد کردم و صفحه AjaxTest.aspx رو تو اون قرار دادم
بعد برای اینکه کسی نتونه با وارد کردن آدرس اون به صفحه دسترسی داشته باشه، توی اون پوشه یه web.config ایجاد کردم و تگ زیر رو به اون اضاف کردم

```
<deny users="*"/>
```

چون میخواستم هیچ کس نتونه اون صفحه رو ببینه. ولی خوب کدهای نوشته شده توسط من باید بتونن به اون صفحه دسترسی داشته باشن (فقط برنامه)
لطفا بفرمائید این رو چطور ادیت کنم تا این کار ممکن بشه؟
ممنونم از شما

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۰۸ ۱۶:۳۱

jQuery یا کلا JavaScript یعنی کدهای سمت کاربر. بنابراین این کدها باید دسترسی رسیدن به مدخل سمت سرور را داشته باشند و اگر عموم از آن منع شده باشند، یک اسکریپت معمولی هم بدیهی است که دسترسی خاصی نخواهد داشت چون سمت سرور اجرا نمی‌شود. فقط نتیجه عملیات آن به سمت سرور Post می‌شود.
یکی از کارهای متداول سمت سرور جهت منع عموم و فقط دسترسی دادن به jQuery مورد زیر است:
« بررسی امنیتی، حین استفاده از jQuery Ajax »

نویسنده: مهدی غفاری
تاریخ: ۱۳۹۱/۰۷/۰۸ ۲۲:۴۶

ممنونم از توجه شما؛
آیا نمیشه از هندلرها استفاده کرد تو همچین مواقعی؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۰۸ ۲۲:۵۲

بله امکان پذیر است. مثل این مثال اسلایدشو می‌تونید JSON بازگشت دهید.

عنوان: **نگارش نهایی ASP.Net MVC ارائه شد**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۰۱ ۱۶:۰۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

دریافت نگارش نهایی

مشاهده‌ی مستندات و راهنماهای مربوطه

کتابی از مدیر پروژه‌ی مربوطه

سایت رسمی

عنوان: آشنایی با کلاس JavaScriptSerializer
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۱/۰۷ ۱۳:۵۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

برای استفاده از jQuery Ajax یکی از روش‌های ارسال دیتا به برنامه، تبدیل داده‌ها به فرمت JSON می‌باشد. برای داده‌های ساده، تشخیص این فرمت ساده است. مثلا اگر امضای تابع وب سرویس اجکس ما به صورت زیر باشد:

```
public static bool IsUserAvailable(string username)
```

اطلاعات جی‌سونی را که قرار است ارسال کنیم، فرمت زیر را باید داشته باشد:

```
{'username':'value'}
```

حال اگر آرگومان‌های ما پیچیده‌تر بودند چطور؟ مثلا بجای یک رشته ساده، یک لیست جنریک داشتیم، فرمت ورودی را چگونه باید تشخیص داد؟

برای این منظور در دات نت 3 و نیم، کلاسی جهت انجام اینگونه تبدیلات پیش بینی شده است که شرح مختصر آن به صورت زیر است:

ابتدا باید ارجاعی را به اسمبلی system.web.extensions به برنامه افزود و سپس جهت سهولت کار می‌توان یک extension method از کلاس JavaScriptSerializer مهیا در فضای نام System.Web.Script.Serialization ایجاد کرد:

```
public static string ToJson(this object data)
      {
          return new JavaScriptSerializer().Serialize(data);
      }
```

اکنون چند مثال زیر را در نظر بگیرید:

```
public static string GetJsonTest0()
      {
          var data = "a1";
          return data.ToJson();
      }

      public static string GetJsonTest1()
      {
          var data = new List<string> { "a1", "a2", "a3" };
          return data.ToJson();
      }

      public static string GetJsonTest2()
      {

          var lst =
             new List<KeyValuePair<string, object>>
                   {
                       new KeyValuePair<string, object>("Name", "علی"),
                       new KeyValuePair<string, object>("Number", 10),
                       new KeyValuePair<string, object>("Desc", "منابع مورد نیاز")
                   };

          return lst.ToJson();

      }
```

خروجی‌های آن‌ها به ترتیب به صورت زیر خواهند بود:

```
"a1"
```

```
["a1","a2","a3"]
[{"Key":"Name","Value":"علی"},{"Key":"Number","Value":10},{"Key":"Desc","Value":"منابع مورد نیاز"}]
```

این کلاس همچنین قابلیت Deserialize و تبدیل داده‌هایی به فرمت JSON به اشیاء مورد نظر ما را نیز دارا است.

عنوان: **کنترل FileUpload داخل ASP.Net Ajax UpdatePanel**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۳ ۲۲:۰۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

کنترل FileUpload زمانیکه داخل Ajax UpdatePanel قرار گیرد کار نمی‌کند و خاصیت Hasfile آن در این حالت همیشه false بر می‌گرداند، زیرا این کنترل برای کار کردن نیاز به post back کامل دارد. برای معرفی این post back داخل update panel ، باید یک تریگر به نام postbacktrigger را افزود و سپس ControlID آن‌را مساوی کنترلی قرار داد که جهت ارسال فایل به سرور مشخص کرده‌ایم.

یک مثال:

```
<asp:UpdatePanel runat="server" id="UpdatePanel1">
        <ContentTemplate>
            <asp:FileUpload runat="server" id="FileUpload1" />
            <asp:Button runat="server" id="btnSubmit" Text="Upload" />
        </ContentTemplate>
        <Triggers>
            <asp:postbacktrigger ControlID="btnSubmit" />
        </Triggers>
    </asp:UpdatePanel>
```

این تریگر عملیات post back را سبب خواهد شد و اکنون عملیات ارسال فایل به سرور صحیح کار خواهد کرد.

نظرات خوانندگان

نویسنده: مهرداد
تاریخ: ۱۳۸۸/۰۲/۰۳ ۲۲:۲۷:۰۰

مثل همیشه بسیار عالی ...
ممنون آقای نصیری .

با تشکر ، مهرداد قاسمی

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۲/۰۷ ۱۹:۰۱:۰۰

آیا امکان اینکه فایل هم به صورت Ajax آپل.د بشه وجود نداره ؟

مثل کاری که Gmail برای Attachment هاش انجام میده ؟

Silverboy

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۷ ۲۰:۰۳:۰۰

باید در نحوه‌ی ارسال فایل به سرور دست برد و اون رو سفارشی کرد. مثلا با استفاده از یک generic handler . نحوه پیاده سازی آن‌را در jQuery می‌تونید اینجا مشاهده کنید:
http://www.ajaxprojects.com/ajax/tutorialdetails.php?itemid=512

توسط ASP.Net ajax هم همین مفهوم به شکل دیگری پیاده سازی می‌شود:
http://msmvps.com/blogs/luisabreu/archive/2006/12/14/uploading-files-without-a-full-postback.aspx

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۲/۲۵ ۱۰:۵۴:۰۰

ممنون خیلی مفید بود

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۴/۱۴ ۱۰:۴۷:۲۱

ممنون خیلی خوب بود. یهreference خوب واسه ajax معرفی می کنید. مثلا همین پنجره captcha رو چجوری با ajax درست می کنن؟؟؟؟

عنوان: کش کردن اطلاعات غیر پویا در ASP.Net - قسمت اول
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۴ ۲۰:۵۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

در مورد افزونه YSlow افزونه Firebug فایرفاکس پیشتر صحبت شد . این افزونه پس از آنالیز یک سایت، پیشنهاداتی را نیز جهت بهبود سرعت، ارائه می‌دهد.

همانطور که در شکل بالا مشخص است، عناصری مانند css و js ، قسمت expires اشان (تاریخ منقضی شدن کش آن‌ها در سمت کلاینت) خالی است و پیشنهاد داده که به هر کدام از این عناصر، هدر مخصوص مشخص سازی مدت زمان کش شدن در سمت کلاینت اضافه شود.

ASP.Net در مورد کش کردن اطلاعات صفحات پویا به اندازه‌ی کافی امکانات در اختیار برنامه نویس قرار می‌دهد اما در مورد اضافه کردن این هدر جهت یک فایل css غیر پویا شاید نتوان مطلب خاصی را یافت.

در IIS7 امکانات ویژه‌ای برای این منظور در نظر گرفته شده که نحوه استفاده از آن در ASP.Net به صورت زیر است:

فایل وب کانفیگ سایت را باز کرده و به قسمت system.webServer چند سطر زیر را اضافه کنید:

```
<staticContent>
<clientCache httpExpires="Sun, 29 Mar 2020 00:00:00 GMT" cacheControlMode="UseExpires" />
</staticContent>
```

این مورد فقط مختص به IIS7 است و بر روی نگارش‌های پایین‌تر کار نمی‌کند.
با این کار، تاریخ منقضی شدن هر آنچه که توسط موتور ASP.net سرو نمی‌شود به سال 2020 تنظیم خواهد شد. (کلیه محتوای غیرپویای سایت، اعم از تصاویر، فایلهای js ، css و غیره)
پس از این تنظیم مجددا YSlow را اجرا کرده و Performance Grade ایی را که نمایش می‌دهد بررسی نمائید.

بدیهی است اگر یکی از فایل‌های css یا js شما تغییر کند، کلاینت، اطلاعات جدیدی را تا سال 2020 دریافت نمی‌کند. برای حل این مشکل یک کوئری استرینگ ساده به انتهای لینک مربوط به css یا js خود اضافه کنید تا URL جدید با URL قبلی آن یکسان نباشد (این کوئری استرینگ تاثیری روی محتوای ایستای ما ندارد). به این صورت این آدرس جدید، مجددا دریافت شده و تا سال 2020 کش خواهد شد.

نکته:
اعمال تنظیم فوق، در IIS7 ویندوز سرور 2008 مجاز است؛ اما در IIS7 ویندوز ویستا قفل شده است و قابل override نیست.
برای تغییر آن، فایل زیر را پیدا کنید:

```
open %systemroot%\System32\inetsrv\config\applicationHost.config
```

و در آن سطر

```
<section name="staticContent" overrideModeDefault="Deny" />
```

را به صورت زیر تغییر دهید تا مجاز به اعمال تغییرات شوید:

```
<section name="staticContent" overrideModeDefault="Allow" />
```

در قسمت بعد در مورد نگارش‌های پایین‌تر IIS توضیح داده خواهد شد.

مآخذ:
YSlow: Add expires header to images in IIS 7
?IIS7: How to set cache control for static content

عنوان: **کش کردن اطلاعات غیر پویا در ASP.Net - قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۶ ۱۹:۲۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

قسمت قبل

به IIS7 اختصاص داشت که شاید برای خیلی‌ها کاربرد نداشته باشد خصوصا اینکه برنامه نویس‌ها ترجیح می‌دهند به روش‌هایی روی بیاورند که کمتر نیاز به دخالت مدیر سرور داشته باشد؛ یا زمانیکه سایت شما بر روی یک هاست اینترنتی قرار گرفته است عملا شاید دسترسی خاصی به تنظیمات IIS نداشته باشید (مگر اینکه یک هاست اختصاصی را تهیه کنید).
برای IIS6 و ماقبل از آن و حتی بعد از آن!، حداقل دو روش برای کش کردن اطلاعات استاتیک وجود دارد:

الف) استفاده از web resources معرفی شده در ASP.Net 2.0 به بعد
در مورد نحوه‌ی تعریف و بکارگیری web resources می‌توان به مقاله " تبدیل پلاگین‌های jQuery به کنترل‌های ASP.Net " رجوع کرد.

Type	URL	Expires	Gzip	RespTime	Size (Ungzip)	ETag
doc	http://localhost/library/login.aspx			104	13.7K	
js	http://localhost/library/WebResource.axd?...	4/26/2010		18	20.7K	
js	http://localhost/library/WebResource.axd?...	4/26/2010		9	21.5K	
js	http://localhost/library/WebResource.axd?...	4/26/2010		10	3.0K	

همانطور که در شکل فوق نیز ملاحظه می‌کنید، هدر مربوط به مدت زمان منقضی شدن کش سمت کلاینت یک web resource توسط موتور ASP.Net به صورت خودکار به سال 2010 تنظیم شده است و این مقدار خالی نیست.

ب) افزودن این هدر به صورت دستی

برای این منظور باید در نحوه‌ی ارائه فایل‌های استاتیک دخالت کنیم و این‌کار را با استفاده از یک generic handler می‌توان انجام داد.

کد این generic handler می‌تواند به صورت زیر باشد:

```
using System;
using System.IO;
using System.Web;
using System.Web.Services;
using System.Reflection;

namespace test1
{
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   public class cache : IHttpHandler
   {

       private static void cacheIt(TimeSpan duration)
       {
           HttpCachePolicy cache = HttpContext.Current.Response.Cache;

           FieldInfo maxAgeField = cache.GetType().GetField("_maxAge", BindingFlags.Instance |
BindingFlags.NonPublic);
           maxAgeField.SetValue(cache, duration);

           cache.SetCacheability(HttpCacheability.Public);
           cache.SetExpires(DateTime.Now.Add(duration));
           cache.SetMaxAge(duration);
           cache.AppendCacheExtension("must-revalidate, proxy-revalidate");
       }

       public void ProcessRequest(HttpContext context)
       {
           string file = context.Request.QueryString["file"];
           if (string.IsNullOrEmpty(file))
           {
               return;
           }

           string contetType = context.Request.QueryString["contetType"];
           if (string.IsNullOrEmpty(contetType))
           {
               return;
           }

           context.Response.Write(File.ReadAllText(context.Server.MapPath(file)));

           //Set the content type
```

```
            context.Response.ContentType = contetType;

            // Cache the resource for 30 Days
            cacheIt(TimeSpan.FromDays(30));
        }

        public bool IsReusable
        {
            get
            {
                return false;
            }
        }
    }
}
```

توضیحات:

این generic handler دو کوئری استرینگ را دریافت می‌کند؛ file جهت دریافت نام فایل و contetType جهت مشخص سازی نوع محتوایی که باید سرو شود؛ مثلا جاوا اسکریپت یا استایل شیت و امثال آن. سپس زمانیکه محتوا را Response.Write می‌کند، هدر مربوط به کش شدن آن‌را نیز به 30 روز تنظیم می‌نماید.

تابع مربوط به کش کردن اطلاعات از مقاله ASP.NET Ajax Under-the-hood Secrets استخراج شد.

روش استفاده در مورد فایل‌های CSS

بجای تعریف یک فایل CSS در صفحه، به صورت استاندارد، اکنون تعریف متداول را به صورت زیر اصلاح کنید:

```
<link type="text/css" href="cache.ashx?v=1&file=site.css&contetType=text/css" rel="Stylesheet" />
```

هر زمانیکه که فایل site.css درخواست می‌شود، باید از فیلتر ما عبور کند و سپس ارائه گردد. در این حین، هدر مربوط به مدت زمان کش شدن سمت کلاینت به آن اضافه می‌شود. از کوئری استرینگ مربوط v هم جهت به روز رسانی‌های بعدی استفاده می‌شود تا اگر تغییری را اعمال کردیم، کلاینت حتما با توجه به آدرس جدید، محتویات جدید را یکبار دیگر دریافت کند. (مرورگر آدرس‌های مشابه را در صورتیکه هدر مربوط به کش شدن آن‌ها تنظیم شده باشد، از کش خواهد خواند و کاری به آخرین تغییرات شما در سرور ندارد)

روش استفاده در مورد فایل‌های JS

```
<script type="text/javascript" src="cache.ashx?v=1&file=js/jquery-
1.3.2.min.js&contetType=application/x-javascript"></script>
```

اکنون اگر سایت را مجددا با افزونه YSlow بررسی کنیم، می‌توان این هدر جدید را مشاهده کرد:

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۲۱:۲۴:۵۴ ۱۳۸۸/۰۴/۲۵

خیلی عالی بود. ممنون

نویسنده: mahdi
تاریخ: ۱۸:۵۸:۵۸ ۱۳۸۸/۱۰/۰۱

با سلام و تشکر
روش کش کردن backgound-image
در css به چه صورت است

نویسنده: وحید نصیری
تاریخ: ۱۹:۳۹:۲۶ ۱۳۸۸/۱۰/۰۱

تصاویر اگر آدرسشان عوض نشود عموما کش می‌شوند اما الزامی هم ندارد.
برای اطمینان حاصل کردن از این موضوع برای تصاویر هم مطابق کدهای فوق یک generic handler با نکات مربوط به کش کردن اطلاعات که ذکر شد ایجاد کنید. به این صورت هر نوع تصویر مورد نظر شما به همراه هدرهای لازم کش شدن ارائه می‌شوند.
generic handler تصویر مثل کدی است که در مقاله زیر آمده:
http://www.dotnettips.info/2009/05/blog-post_27.html
فقط این مقاله چند سطر فوق مربوط به اضافه کردن هدرهای کش شدن اطلاعات را ندارد.

نویسنده: مهدی پایروند
تاریخ: ۱۴:۰۲:۱۷ ۱۳۸۹/۰۲/۱۲

ورژن های پایین در تب HTTP Headers گزینه Enable content expiration وجود دارد آیا این گزینه در صورت استفاده از یک هاست اختصاصی همان امکانات را برای سمت کلاینت انجام میدهد.

نویسنده: وحید نصیری
تاریخ: ۱۴:۲۸:۲۵ ۱۳۸۹/۰۲/۱۲

http://www.dotnettips.info/2009/05/aspnet_17.html

عنوان: **تشخیص نقایص تصاویر صفحات سایت با استفاده از jQuery Ajax**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۱۳ ۲۱:۳۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

این مثال شبیه به مثال بررسی وجود نام کاربر با استفاده از jQuery Ajax است که از ذکر توضیحات مشابه آن، در اینجا خودداری خواهد شد.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TestBrokenImages.aspx.cs"
   Inherits="testWebForms87.TestBrokenImages" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>detecting broken images</title>

   <script src="jquery.min.js" type="text/javascript"></script>

   <script type="text/javascript">
       function errorReplace(arg) {
           //ارسال پیغام خطا
           $.ajax({
               type: "POST",
               url: "TestBrokenImages.aspx/GetErros",
               data: "{'image': '" + arg.src + "','page':'" + location.href + "'}",
               contentType: "application/json; charset=utf-8",
               dataType: "json"
           });
           //نمایش تصویری دلخواه بجای نمونه مفقود
           $(arg).attr('src', 'missing.png');
       }

       //بررسی وضعیت تک تک تصاویر پس از بارگذاری کامل صفحه
       $(document).ready(function() {
           $(window).bind('load', function() {
               $('img').each(function() {
                   if (!this.complete || (!$.browser.msie && (typeof this.naturalWidth == "undefined"
|| this.naturalWidth == 0))) {
                       errorReplace(this);
                   }
               });
           })
       });
   </script>

</head>
<body>
   <form id="form1" runat="server">
   <div>
       <img src="img1.png" />
       <img src="img2.png" />
   </div>
   </form>
</body>
</html>
```

```
using System;
using System.IO;
using System.Web.Services;

namespace testWebForms87
{
   public partial class TestBrokenImages : System.Web.UI.Page
   {
       protected void Page_Load(object sender, EventArgs e)
       {
       }

       [WebMethod]
```

```
        public static void GetErros(string image,string page)
        {
            ارسال ایمیل به مسؤول سایت و یا ذخیره خطاها در دیتابیس//
        }
    }
}
```

در این مثال زمانیکه صفحه کاملا بارگذاری شد، وضعیت تک تک تصاویر بررسی می‌شود، اگر تصویر مفقودی وجود داشت (با اکثر مرورگرها سازگار است)، اطلاعات آن به تابع errorReplace ارسال خواهد شد.
در این تابع با استفاده از jQuery Ajax ، اطلاعات تصویر مفقود و صفحه مربوطه به وب متد GetErros ما ارسال می‌شود. سپس در این متد می‌توان یا آرگومان‌های دریافتی را به صورت یک ایمیل به مسؤول سایت ارسال نمود و یا آن‌ها را جهت بررسی آتی در یک دیتابیس ذخیره کرد.
بدیهی است بجای قرار دادن وب متد فوق در صفحه جاری، می‌توان یک وب سرویس را نیز ایجاد و متد را در آن قرار داد تا نیازی نباشد به ازای هر صفحه سایت یکبار این متد تکرار شود.

اگر موفق به اجرای این مثال نشدید، برای مثال یک break point داخل متد GetErrors قرار دهید و برنامه را در حالت دیباگ در ویژوال استودیو شروع کنید، اگر اتفاق خاصی رخ نداد و به این break point نرسیدید، احتمالا تنظیمات وب کانفیگ شما مناسب نیست. قسمت مربوط به webServices ، system.web.extensions و jsonSerialization باید در وب کانفیگ موجود باشد که VS 2008 این موارد را به صورت خودکار اضافه می‌کند.

عنوان: **کاهش پهنای باند مصرفی یک سایت ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۱ ۲۰:۱۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

مطلبی را دیروز در وبلاگ آقای صحرایی دیدم در مورد بهبود کارآیی برنامه‌ها و سایت‌های ASP.Net ، که یکی از موارد آن "فاصله بین تگ ها را تا حد ممکن از بین ببرید" بود.

برای پیاده سازی آن به صورت زیر می‌توان عمل کرد:

```
using System.Text.RegularExpressions;

        //حذف فاصله‌های خالی
        private static readonly Regex REGEX_BETWEEN_TAGS = new Regex(@">\s+<", RegexOptions.Compiled);
        private static readonly Regex REGEX_LINE_BREAKS = new Regex(@"\n\s+", RegexOptions.Compiled);

        protected override void Render(HtmlTextWriter writer)
        {

            using (HtmlTextWriter htmlwriter = new HtmlTextWriter(new System.IO.StringWriter()))
            {
                base.Render(htmlwriter);
                string html = htmlwriter.InnerWriter.ToString();

                html = REGEX_BETWEEN_TAGS.Replace(html, "> <");
                html = REGEX_LINE_BREAKS.Replace(html, string.Empty);

                writer.Write(html.Trim());
            }
        }
```

این قطعه کد را در مسترپیج سایت خود قرار دهید تا به صورت یکسان به کل سایت اعمال شود.

یا روش دیگر اعمال آن سفارشی ساختن ASP.NET pipeline با کمک Response.Filter آن است. برای مشاهده پیاده سازی آن لطفا به مقاله زیر مراجعه بفرمائید:

Free Bandwidth For All ASP.NET Users

برای بهینه سازی قسمت اعمال regular expressions آن می‌توان به مقاله " چگونه Regex سریعتری داشته باشیم؟ " مراجعه کرد.

عنوان: کش کردن اطلاعات غیر پویا در ASP.Net - قسمت سوم
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۷ ۲۳:۰۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

در دو قسمت قبل در مورد IIS7 و IIS6 صحبت شد (+ و +).
در تکمیل قسمت دوم، یک مورد هم جزو قابلیت‌های ذاتی IIS6 و همچنین IIS5 است که می‌توان آن‌را فعال نمود (اگر دسترسی به سرور دارید) :

تنظیم مدت زمان content expiration ، بدون نیاز به برنامه نویسی خاصی، کار اضافه کردن هدر مربوط به مدت زمان کش شدن سمت کلاینت را به محتویات غیرپویای سایت شما مانند تصاویر ، فایل‌های CSS و غیره انجام می‌دهد. آمارها نشان می‌دهند که این تنظیم، زمان بارگذاری بعدی را بین 50 تا 70 درصد کاهش می‌دهد.
تنظیم این قابلیت را می‌توانید به چک لیست نصب IIS خود اضافه نمائید.

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۲/۲۸ ۱۶:۵۳:۰۰

سلام جناب نصیری
این تنظیمات بیشتر برای سایتهایی که محتوای غیر پویا دارند مناسب نیست؟
چون اگر که اطلاعات ما زود به زود آپدیت بشه ممکنه اطلاعات قدیمی به کاربر نشون داده بشه درسته؟
باتشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۲۸ ۱۷:۲۳:۰۰

روی محتوای پویای سایت شما اثری ندارد. فقط تصاویر ، css و فایل‌های جاوا اسکریپت و امثال آن‌را کش می‌کند (که عموما پویا نیستند). بنابراین برای خیلی از سایت‌ها مفید است.
و همچنین در خیلی از سایت‌ها هم تغییرات css یا تصاویر اصلی سایت، شاید ماهی یکبار باشد. زمانیکه طراحی قالب و یک سری از موارد ثابت سایت تمام شد، این‌ها دیگر هر روز قرار نیست تغییری کنند.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۱۱ ۱۳:۳۰:۳۴

؟؟ بله. این قسمت سوم مربوط است به IIS است. در قسمت‌های دیگر هم شرایط مرتبط دقیقا ذکر شده است.

نویسنده: ehsan
تاریخ: ۱۳۸۹/۰۱/۱۱ ۱۵:۵۷:۲۸

استاد قسمت چهارم چی؟
من کلاس ها رو نوشتم و در web config هم اضافه کردم ولی expire time برای css ها تنظیم نمیشه!
میشه کمک کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۱/۱۱ ۱۶:۳۱:۵۴

برنامه رو دیباگ کنید ببینید اصلا متد context_PreSendRequestHeaders صدا زده می‌شود؟ unhandled exception ها رو لاگ کنید ببینید شاید جایی مشکلی هست.
ضمنا من این رو با IIS7 تست کردم و نتایج قسمت 4 با استفاده از IIS7 منتشر شده.
ولی در کل دیباگ کردن این کدها بهترین روش برای پیدا کردن مشکل است.

عنوان: **معرفی ELMAH**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۰۸ ۱۴:۴۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

عموما کاربران نمی‌توانند گزارش خطای خوبی را ارائه بدهند و البته انتظاری هم از آنان نیست. تنها گزارشی که از یک کاربر دریافت می‌کنید این است: "برنامه کار نمی‌کند!" و همین!
روش‌های متعددی برای لاگ کردن خطاهای یک برنامه ASP.Net موجود است؛ چه خودتان آن‌ها را توسعه دهید و یا از ASP.NET health monitoring استفاده کنید.
روش دیگری که این روزها در وبلاگ‌های متعددی در مورد آن مطلب منتشر می‌شود، استفاده از ELMAH است. (البته ELMAH به تازگی منتشر نشده ولی تا کیفیت محصولی به عموم ثابت شود مدتی زمان می‌برد)

ELMAH یک ماژول رایگان و سورس باز لاگ کردن خطاهای مدیریت نشده برنامه‌های ASP.Net است. برای استفاده از این ماژول نیازی نیست تا تغییری در برنامه خود ایجاد کنید یا حتی آن‌را کامپایل مجدد نمائید. یک فایل dll دارد به همراه کمی تغییر در web.config برنامه جهت معرفی آن و این تمام کاری است که برای برپایی آن لازم است صورت گیرد. این ماژول تمامی خطاهای مدیریت نشده‌ی برنامه شما را لاگ کرده (در حافظه سرور، در یک فایل xml ، در یک دیتابیس اس کیوال سرور یا اوراکل ، در یک دیتابیس اکسس و یا در یک دیتابیس اس کیوال لایت) و برای مرور آن‌ها یک صفحه‌ی وب سفارشی یا فیدی مخصوص را نیز در اختیار شما قرار می‌دهد. همچنین این قابلیت را هم دارد که به محض بروز خطایی یک ایمیل را نیز به شما ارسال نماید.

با توجه به این‌که این ماژول در Google code قرار گرفته احتمالا دسترسی به آن مشکل خواهد بود. سورس و فایل‌های کامپایل شده آن‌را از آدرس‌های زیر نیز می‌توان دریافت نمود:

(+ و + و +)

نحوه استفاده از ELMAH :

برای استفاده از ELMAH دو کار را باید انجام دهید:
الف) کپی کردن فایل Elmah.dll در پوشه bin برنامه
ب) تنظیم وب کانفیگ برنامه

بهترین مرجع برای آشنایی با نحوه بکار گیری این ماژول، مراجعه به فایل web.config موجود در پوشه samples آن است. بر اساس این فایل نمونه:

- ابتدا باید configSections آن را به وب کانفیگ خود اضافه کنید.

- سپس تگ elmah باید اضافه شود. در این تگ موارد زیر مشخص می‌شوند:
الف) آیا خطاها توسط آدرس elmah.axd توسط کاربران راه دور قابل مشاهده شود یا خیر.
ب) خطاها کجا ذخیره شوند؟ موارد زیر پشتیبانی می‌شوند:
دیتابیس‌های اس کیوال سرور ، اوراکل ، حافظه سرور، فایل‌های xml ، دیتابیس SQLite ، دیتابیس اکسس و یا دیتابیسی از نوع VistaDB
ج) آیا خطاها ایمیل هم بشوند؟ اگر بلی، تگ مربوطه را تنظیم کنید.
د) آیا خطاها به اکانت twitter شما نیز ارسال شوند؟

- در ادامه تگ مربوط به معرفی این httpModules باید تنظیم شود.

- سپس httpHandlers ایی به نام elmah.axd که جهت مرور خطاها می‌توان از آن استفاده نمود معرفی می‌گردد.

- از IIS7 استفاده می‌کنید؟ قسمت system.webServer را نیز باید اضافه نمائید.

- و در آخر نحوه‌ی دسترسی به elmah.axd مشخص می‌شود. اگر اجازه دسترسی از راه دور را داده باشید، به این طریق می‌شود دسترسی را فقط به کاربران مجاز و تعیین اعتبار شده، اعطاء کرد و یا به نقشی مشخص مانند ادمین و غیره.

برای نمونه، اگر بخواهید از دیتابیس SQLite جهت ذخیره سازی خطاهای حاصل شده استفاده نمائید و نیز از ارسال ایمیل صرفنظر کنید، وب کانفیگ برنامه شما باید به شکل زیر تغییر یابد:

```
<?xml version="1.0"?>

<configuration>
<configSections>
  <sectionGroup name="elmah">
    <section name="security" requirePermission="false" type="Elmah.SecuritySectionHandler, Elmah"/>
    <section name="errorLog" requirePermission="false" type="Elmah.ErrorLogSectionHandler, Elmah" />
    <section name="errorMail" requirePermission="false" type="Elmah.ErrorMailSectionHandler, Elmah" />
    <section name="errorFilter" requirePermission="false" type="Elmah.ErrorFilterSectionHandler,
Elmah"/>
    <section name="errorTweet" requirePermission="false" type="Elmah.ErrorTweetSectionHandler, Elmah"/>
  </sectionGroup>
</configSections>

<elmah>
  <security allowRemoteAccess="1" />
  <errorLog type="Elmah.SQLiteErrorLog, Elmah" connectionStringName="cn1" />
</elmah>


<appSettings/>

<connectionStrings>
  <add name="cn1" connectionString="data source=~/ErrorsLog/Errors.db" />
</connectionStrings>

<system.web>
  <httpModules>
    <add name="ErrorLog" type="Elmah.ErrorLogModule, Elmah"/>
  </httpModules>

  <httpHandlers>
    <add verb="POST,GET,HEAD" path="elmah.axd" type="Elmah.ErrorLogPageFactory, Elmah" />
  </httpHandlers>

  <compilation debug="true"/>

  <authentication mode="Windows" />
</system.web>

</configuration>
```

در اینجا یک پوشه جدید به نام ErrorsLog را باید به ریشه سایت خود اضافه کنید (یا هر نام دلخواه دیگری که در قسمت connectionStrings باید تنظیم شود). فایل Errors.db به صورت خودکار ایجاد خواهد شد. بدیهی است کاربر ASP.net باید دسترسی write بر روی این پوشه داشته باشد تا عملیات ثبت خطاها با موفقیت صورت گیرد. همچنین فایل System.Data.SQLite.DLL نیز باید در پوشه bin برنامه شما کپی شود.
ساده‌ترین تنظیم این ماژول استفاده از حالت xml است که به ازای هر خطا یک فایل xml را تولید کرده و نیاز به اسمبلی دیگری بجز ماژول مربوطه نخواهد داشت.

تذکر:
از لحاظ امنیتی مثال فوق توصیه نمی‌شود زیرا allowRemoteAccess آن 1 است و قسمت authorization ذکر نشده است. این مثال فقط جهت راه اندازی و آزمایش اولیه ارائه گردیده است. (همچنین بهتر است این نام پیش فرض را به نامی دیگر مثلا myloggermdl.axd تغییر داده و در قسمت httpHandlers تنظیم نمائید. سپس این نام را به تگ location نیز اضافه کنید)

اکنون برای مشاهده خروجی این ماژول به انتهای آدرس سایت خود، elmah.axd را اضافه کرده و سپس enter کنید:

همانطور که در تصویر مشخص است، تمامی خطاهای لاگ شده گزارش داده شده‌اند. همچنین دو نوع فید به همراه امکان دریافت خطاها به صورت CSV نیز موجود است. با کلیک بر روی لینک details ، صفحه‌ی بسیار ارزنده‌ای ارائه می‌شود که تقریبا نحوه‌ی وقوع ماجرا را بازسازی می‌کند.

نکته‌ی مهمی که در صفحه‌ی جزئیات ارائه می‌شود (علاوه بر stack trace و مشخصات کاربر)، مقادیر تمامی فیلدهای یک صفحه هنگام بروز خطا است (قسمت Raw/Source data in XML or in JSON در این صفحه) :

به این صورت دیگر نیازی نیست از کاربر بپرسید چه چیزی را وارد کرده بودید که خطا حاصل شد. دقیقا مقادیر فیلدهای همان صفحه‌ی زمان بروز خطا نیز برای شما لاگ می‌گردد.

نکته:
ماژول SQLite ایی که به همراه مجموعه ELMAH ارائه می‌شود 32 بیتی کامپایل شده (64 بیتی آن نیز موجود است که باید از آن در صورت لزوم استفاده شود). بنابراین برای اینکه در یک سرور 64 بیتی به مشکل برنخورید و خطای BadImageFormat را دریافت نکنید نیاز است تا به این نکته دقت داشت.

برای مطالعه بیشتر:

Error Logging Modules And Handlers
Sending ELMAH Errors Via GMail
Exception-Driven Development
Using HTTP Modules and Handlers to Create Pluggable ASP.NET Components

نظرات خوانندگان

نویسنده: افشار محبی
تاریخ: ۱۳۸۸/۰۳/۰۹ ۱۰:۱۳:۵۱

جناب نصیری

شما واقعا ابزارهای جالب و مفیدی را می‌شناسید. خیلی از آنها کار بنده را راه انداخته است!

نویسنده: Mohsen
تاریخ: ۱۳۸۹/۰۵/۲۲ ۱۲:۵۰:۰۵

ممنون
جالب بود

نویسنده: mmshahshahani
تاریخ: ۱۳۸۹/۰۵/۲۴ ۱۱:۲۶:۱۸

خیلی حال داد ، ممنون

نویسنده: maysam.sa
تاریخ: ۱۳۸۹/۰۵/۲۵ ۲۳:۳۰:۲۵

سلام
با تشکر از مطالب جالب و آموزنده شما
میشه دوباره یه لینک برا دانلود بذارید؟
لینک قبلی خراب شده

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۵/۲۵ ۲۳:۴۵:۱۹

http://rapidshare.com/files/412297366/ELMAH-1.1-bin.zip.html

نویسنده: Majid
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۴:۲۰:۰۴

با سلام ودرود خدمت آقای نصیری:
elmah معرفی شده یکی از پرکاربردترین و دوست داشتنی ترین مواردی که توسط استاد بنده (آقای مهدی کرامتی) معرفی (ارجاع به وبلاگ شما) و از اینجا مورد استفاده قرار گرفت.
به نظر در پیاده سازی و عملکرد elmah در پروژه هیچ مشکلی وجود ندارد ...
اما مشکلی در مورد ارسال ایمیل وجود دارد که خدمتتان عرض می کنم.
در حالت localایمیل به زیبایی هر چه تمام تر ارسال می شود ...
اما پس از آپلود سایت در ارسال ایمیل مشکل وجود دارد !
به نظر شما چگونه باید این مشکل را حل نمایم ؟
این مشکل به سرور مربوط می شود یا به web.config

من قسمت نظرات رو هر روز بررسی می کنم تا بلکه پاسخی از طرف شما در یافت نمایم و یا اینکه پاسخ را بدون زحمت به آدرس زیر ایمیل بزنید:
majid.darab@gmail.com

با تشکر فراوان از وب لاگ تخصصی حضرتعالی

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۵:۴۱:۰۸

سلام
تمام هاست‌ها برای ارسال ایمیل نیاز به smtp authentication دارند. برای این منظور باید یوزر نیم و پسورد داده شده به شما را در web.config برنامه اضافه کنید که در این مقاله توضیح داده شده است:
http://www.4guysfromrolla.com/articles/072606-1.aspx

نویسنده: Majid
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۶:۱۷:۵۹

با تشکر از پاسخگویی شما...
یعنی authentication زیر که مربوط به تنظیمات خود elmah بود کفایت نمی کند و باید در web.config آئوتنتیکیشن دیگری هم تعریف نمایم:

من کمی گیج شدم ...

همان طور که گفتم در local این موضوع به خوبی کار می کنداما پس از آپلود مشکل بوجود آمد.

من از داخل کدبیهایند (بدون دست کاری در web.config) هم سیستم ارسال ایمیل رو دارم و این قسمت پس از آپلود به خوبی کار می کند.
تقریباً مشابه همون عملکردو در مورد elmah تو web.config پیاده سازی کردیم (طبق فرمول بالا)که در local درست و پس از آپلود مشکل دار شد.

اگر لازم است در web.config کدی اضافه نمایم پس چرا ایمیل داخل کدبیهایند داره به خوبی کار می کنه

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۸:۱۷:۴۵

در ASP.NET برای تعریف smtp authentication یا باید از طریق code behind عمل کنید که این حالت نیازی به تعریف همان موارد در web.config ندارد.
یا این تنظیمات رو مطابق مقاله‌ای که ذکر کردم می‌تونید به web.config انتقال بدید (قسمت system.net -- mailSettings -- smtp -- network) و نیازی به تعریف مجدد آن‌ها در code behind نیست و به صورت سراسری به برنامه اعمال می‌شود. یکی از این دو حالت رو انتخاب کنید.
در elmah مطابق مقاله http://www.dotnettips.info/2010/08/elmah-wcf-ria-services.html باید قسمت errorMail اضافه شود که در آن مقاله البته comment شده که باید comment آن برداشته شود.
سپس elmah هم چون از همان کدهای متداول دات نتی استفاده می‌کند تنظیمات smtp authentication را که بحث مجزایی است می‌تواند از web.config هم دریافت کند. راه دیگر هم این است که این یوزر نیم و پسورد را در همان تگ errorMail هم مطابق مثال web.config موجود در پوشه samples آن هم می‌شود تعریف کرد. تمام این‌ها یک اثر را دارند و فقط یکی باید اعمال شود.

ضمنا یک مطلب رو هم به صراحت خدمت شما عرض کنم. تابحال هاستی رو ندیدم که با همان تنظیمات local شما بدون مشکل کار کند. علت هم این است که کسی نتواند از امکانات ارسال ایمیل آن‌ها سوء استفاده کند. بنابراین بر روی هاست حتما نیاز به smtp authentication هست. یا این اطلاعات را باید از هاست بگیرید. یا اگر کنترل پنل ارسال ایمیل هم سایت دارد یک یوزر جدید در آن تعریف کنید و مشخصات آن‌را در برنامه استفاده کنید. در غیر اینصورت آن هاست از لحاظ امنیتی به شدت مشکل دارد و دیر یا زود زیر بار ارسال میلیون‌ها اسپم از پا در خواهد آمد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۸:۳۴:۳۵

برای تکمیل مبحث هم حتما یکبار فایل web.config موجود در پوشه samples آن را مفصل بررسی کنید و یک سری کامنت دارد که باید بخونید.
برای مثال اگر هاست شما از IIS7 به بعد استفاده می‌کند قسمت system.webServer فایل web.config شما باید شامل Elmah.ErrorMailModule شود در غیر اینصورت ایمیلی ارسال نخواهد شد.
این موارد دقیقا در این فایل راهنمای همراه مجموعه موجود است.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۰۸ ۱۸:۴۶:۴۸

خلاصه بحث ارسال ایمیل:
- امکان ندارد هاست شما برای ارسال ایمیل نیاز به smtp authentication نداشته باشد (من ندیدم). بنابراین یا این اطلاعات را باید زنگ بزنید و از هاست بگیرید یا در کنترل پنل ایمیل‌های سایت یک یوزر مخصوص را درست کنید تا بشود در برنامه از آن استفاده کرد.
- سپس تگ errorMail باید دقیقا مقدار دهی شود. آدرس و نام کاربری و غیره. (این موارد مرتبط به ارسال ایمیل است و با نام کاربری مثلا ftp سایت متفاوت می‌باشد)
- در IIS6 : باید در قسمت system.web -- httpModules ، مطابق مثال web.config پوشه sample کتابخانه، قسمت Elmah.ErrorMailModule را از کامنت خارج کنید.
- در IIS7 : باید در قسمت system.webServer فایل web.config شما Elmah.ErrorMailModule نیز مطابق مثال یاد شده ، ذکر شده باشد.

نویسنده: Majid
تاریخ: ۱۳۸۹/۰۶/۰۸ ۲۱:۲۰:۰۵

از اینکه با صبر و حوصله پاسخگوی مشکل مطرح شده بودید از شما به شدت قدر دانی می کنم و فکر کنم فهمیدم مشکل از کجا بود!
شما مورد ایمیل را کاملاً مو شکافی کردید و فکر می کنم حق مطلب در مورد آن ادا شده است.
من تنظیمات الماه را براساس iis سرور 2003 گذاشته بودم (قاعدتاً باید 6 می بود)و الان که ورژن iis رو دیدم ، با تعجب متوجه شدم شده ورژن 76,0.
حالا اینکه این ورژن 6 یا 7 است با تغییر کانفیگ الماه مشخّص خواهد گردید.

درود فراوان بر وب لاگ شما

نویسنده: Majid
تاریخ: ۱۳۸۹/۰۶/۰۹ ۰۱:۳۳:۱۴

سلام آقای نصیری :
نزدیک به 2 ساعت هر جور تغییری که بگید در web.config دادم اما نشد که نشد.

دایماً کامنت ها رو بر می داشتم و دوباره می ذاشتم.

من به سرور دسترسی remote کامل دارم و از این طریق حالت های مختلف web.config رو بررسی کردم.
(team viewer)

من یک ایمیل توسط گوگل ایجاد کردم و از اون برای ارسال ایمیل استفاده می کنم.

تنها تنظیماتی که انجام دادم در خود تگ error mail الماه بود و بس.

وقتی پروژه در local کار می کنه ایمیل ارسال میشه ، اما وقتی پابلیش میشه میره رو سرور نمی دونم چرا کار نمی کنه؟

ویندوز سرور هم 2003 است و ورژن iis اونو زده 76,0

فایروال سرور هم از کار انداختم و یه بار تست کردم ، اما از ایمیل خبری نشد.

آیا منظور از autheticate کردن این است که :
1-
یا در تگ error mail تنظیمات کانکت شدن به سروری که ایمیل مربوط به آن است را انجام دهیم (شامل user name و پسورد) که دراین مورد gmail مورد نظر می باشد!
2-
یا در جای دیگری از وب کافیگ این تعاریف را انجام دهیم که برای کل پروژه عمومیت داشته باشه!
3-
یا در کد بیهایند جداگانه برای هر ایمیل زدن انجام دهیم!

این سوالو برا این پرسیدم چون گفته بودید از هاست خود user name و پسورد بگیرید.
این user name و پسورد مربوط به چیست ؟
از آنجا که سروری که من دارم استفاده می کنم تازه راه اندازی شده لذا مفاهیمی به نام mail server یا کنترل پنل برای آن وجود ندارد و فقط یک وب سرور است با یک دامین ثبت شده که به خوبی کار می کند.

کجای کار می لنگد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۰۹ ۰۹:۲۲:۰۹

GMail نیاز به SSL دارد و پورت آن هم متفاوت است. برای تنظیمات مخصوص ELMAH در این مورد ویژه (که بهتر بود از ابتدا به آن اشاره می‌کردید) به مقاله زیر مراجعه کنید:
http://dotnetslackers.com/ASP_NET/re-205775_Sending_ELMAH_Errors_Via_GMail.aspx

نویسنده: سیدخلیل
تاریخ: ۱۳۹۱/۰۴/۱۰ ۹:۳۵

سلام
آقای نصیری واقعا از مطالب مفیدتون تشکر میکنم
بی زحمت میشه لینک جدید دانلود elmah رو بذارید؟
چون لینکها منقضی شدن!
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۰ ۹:۴۴

نیازی نیست. به سادگی و بدون مشکل بسته بودن سورس اصلی، از NuGet استفاده کنید:
http://nuget.org/packages/elmah

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۴/۱۰ ۲۱:۴۶

خیلی ممنون استاد عالی بود فقط دو سوال :
1- آیا امکان داره ظاهر errorLogی که ایمیل میشود را تغییر داد تا خواناتر شود؟
2-برای امنیت بیشتر امکان Hash کردن UserName و Password مربوط SMTP وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۰ ۲۲:۵

بله. برای اطلاعات بیشتر به مقالات زیر مراجعه کنید:

Customizing ELMAH's Error Emails

Encrypting Web.config sections in ASP.NET

نویسنده: حسین
تاریخ: ۱۳۹۱/۰۶/۲۰ ۱۲:۴۲

سلام
آیا ابزاری مشابه برای winforms هم وجود داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۰ ۱۳:۶

این دو مطلب را با هم ترکیب کنید تا به نمونه مشابهی در Winforms و WPF برسید:
(^) و (^)

نویسنده: ناصر
تاریخ: ۱۳۹۱/۰۷/۲۳ ۱۲:۲۱

آقای نصیری آیا باید این تگ حتما true باشه؟

```
<compilation debug="true"/>
```

این قضیه باعث پایین آمدن راندمان وب سایت ما نمیشه. چون وقتی میزارم روی false دیگه اطلاعات دقیق خطا رو میل نمیکنه برام؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۳ ۱۲:۴۶

خیر. در زمان ارائه نهایی نباید به این ترتیب عمل شود. در مورد دریافت جزئیات بیشتر در حالت release به مقاله لینک داده شده مراجعه کنید.

نویسنده: ناصر
تاریخ: ۱۳۹۱/۰۷/۲۳ ۱۶:۳۱

ممنون؛
به نظر میادروشی که گفتید تو وب فرم جواب نمیده. جون من اون release رو پیدا نکردم. من میخام زمانی که <compilation debug="false"/> هستش شماره خطا و جزئیاتش هم برام میل بشه. آیا امکانش هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۲۴ ۱۲:۲۸

خلاصه مطلب ذکر شده این است: در حالت release، باید فایل pdb متناظر با dll پروژه شما در کنارش باشد تا شماره خط

متناظر با استثنای رخ داده شده، قابل لاگ کردن باشد.

عنوان: **نمایش یک پیغام به کاربر در ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۳/۱۵ ۲۱:۵۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

عموما در برنامه‌های وب مرسوم است که پیغام به کاربر را در همان لابلای html صفحه نمایش می‌دهند. مثلا یک برچسب و سپس تنظیم متن آن در کد برنامه به صورت پویا.
با استفاده از پلاگین‌های jQuery این‌کار را به صورت شکیل‌تری می‌توان انجام داد. برای مثال:

پلاگین کم حجمی برای این منظور موجود است به نام jQuery Notice (یکی از چند ده نمونه موجود)

```
<script type="text/javascript">
$(document).ready(function()
{
jQuery.noticeAdd({
  text: 'پیغامی به کاربر',
  stay: false
 });
});
</script>
```

کمی این پلاگین را اصلاح کردم تا مشکل عدم نمایش آن هنگام اسکرول طولانی صفحه در IE حل شود (به صورت پیش فرض با فایرفاکس مشکلی ندارد). برای مثال این div را در نظر بگیرید:

```
<div id="myElement" style="position: absolute">This stays at the top</div>
```

قصد داریم مکان آن‌را در بالای صفحه ثابت کنیم (حتی با یک اسکرول طولانی مانند تصویر فوق، باز هم همان بالا باقی بماند و قابل مشاهده باشد).

با استفاده از jQuery این‌کار به صورت زیر قابل انجام است:

```
<script type="text/javascript">
$(document).ready(function()
{
$(window).scroll(function() {
    $('#myElement').css('top', $(this).scrollTop() + "px");
});
});
</script>
```

زمانیکه scroll ایی در window جاری صورت گیرد، div ایی با id مساوی myElement یافت شده و سپس مقدار top آن تنظیم شده و در بالای صفحه نمایش داده می‌شود.

ولی این روش جهت نمایش پیغامی پویا به کاربر مشکل دارد.

نیاز است به ازای هر پیغام پویا یکبار به نحوی این اسکریپت به صفحه تزریق شود که روش انجام کار در ASP.Net به صورت زیر می‌تواند باشد:

```
using System;
using System.Web.UI;
using System.Web;

 public class CAddJqueryNotice
 {
     /// <summary>
     /// نمایش یک پیغام بر اساس پلاگین نوتیس
     /// </summary>
     /// <param name="title">عنوان</param>
     /// <param name="msg">پیغام</param>
     /// <param name="rtl">راست به چپ؟</param>
     /// <param name="duration">مدت زمان نمایش</param>
     /// <param name="autoHide">به صورت خودکار بسته شود؟</param>
     public static void Show(string title, string msg, bool rtl, int duration, bool autoHide)
     {
         string scriptBlock
             = string.Format(@"<script type=""text/javascript"">
                                 $(document).ready(function() {{
                                     jQuery.noticeAdd({{
                                         text: '<b>{0}</b><br/><div align=left dir={1}>{2}</div>',
                                         stay: {3},
                                         stayTime: {4}
                                     }});
                                 }});
                             </script>",
                             title,
                             (rtl ? "rtl" : "ltr"),
                             msg,
                             (autoHide ? "false" : "true"),
                             duration);

         if (HttpContext.Current == null || HttpContext.Current.Handler == null) return;
         Page page = HttpContext.Current.Handler as Page;
         if (page != null)
             page.ClientScript.RegisterStartupScript(
                 page.GetType(),
                 "script" + new Guid().ToString("N"),
                 scriptBlock,
                 false);
     }

 }
```

از آنجائیکه در یک کلاس دیگر خارج از صفحه اصلی مشغول به کار هستیم، دسترسی مستقیم به شیء Page و سپس متد

ClientScript.RegisterStartupScript آن جهت تزریق اسکریپت خود به صفحه نداریم. اما با استفاده از HttpContext.Current.Handler می‌توان به این مقصود رسید و مشکل حل می‌شود.

برای آزمایش آن یک دکمه را در صفحه قرار داده و در روال رخداد گردان کلیک آن کد زیر را اضافه کنید:

```
CAddJqueryNotice.Show( "مشکلی رخ داده است" ,"لطفا دوباره سعی کنید", true, 2000, true);
```

بدیهی است قبل از استفاده از کد فوق، باید چند سطر زیر را به هدر master page سایت خود اضافه کنید:

```
<script src="jquery-1.3.2.js" type="text/javascript"></script>
<link href="jquery.notice.css" type="text/css" media="screen" rel="stylesheet" />
<script src="jquery.notice.js" type="text/javascript"></script>
```

نظرات خوانندگان

نویسنده: amin
تاریخ: ۰۲:۱۷:۲۶ ۱۳۸۸/۰۵/۰۸

سلام . واقعا مطالب عال هستند .
اگر امکان دارد یک مثال کامل در این مورد با asp.net بگزارید .
موفق باشید .

نویسنده: مصطفی دیندار
تاریخ: ۱۶:۳۱:۵۲ ۱۳۸۹/۰۴/۲۳

سلام ,

متاسفانه در UpdatePanel کار نمیکند

نویسنده: وحید نصیری
تاریخ: ۱۷:۳۲:۳۲ ۱۳۸۹/۰۴/۲۳

تکمیل مبحث در اینجا:
Post/279/نمایش-پیغامی-به-کاربر-در-هنگام-استفاده-از-MS-Ajax

عنوان: **اسکریپت‌های خود را یکی کنید**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۱۳ ۲۰:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

به ScriptManager ارائه شده در دات نت فریم ورک 3 و نیم، سرویس پک یک، امکان ترکیب و یکی کردن اسکریپت‌های یک سایت نیز اضافه شده است. به مثال زیر دقت بفرمائید:

```
<asp:ScriptManager runat="server" ID="scMan1">
          <CompositeScript>
              <Scripts>
                  <asp:ScriptReference Path="~/flash_detect.js" />
                  <asp:ScriptReference Path="~/jquery.min.js"  />
              </Scripts>
          </CompositeScript>
      </asp:ScriptManager>
```

به صورت خودکار تمامی ScriptReference های قرار گرفته درون تگ CompositeScript ، یکی شده و توسط یک ScriptResource به صفحه اضافه خواهند شد. برای مثال:

```
<script src="/ScriptResource.axd?d=cAKulKR-axYxbFwMRvm-IlAnHOZjn3-BvtfYzmKItFijWImCOZdjuDVGIFvqZLFX0"
type="text/javascript"></script>
```

مزایا:
- کم شدن رفت و برگشت‌ها به سرور. به این صورت مرورگر تنها یکبار درخواست دریافت اسکریپت فوق را به سرور ارسال می‌کند و نه چندین بار به ازای هر یک از اسکریپت‌های سایت.
- کش شدن خودکار اسکریپت حاصل. (هدرهای لازم اضافه می‌شوند)
- اگر مرورگر HTTP compression را پشتیبانی کند، نتیجه حاصل GZip شده ارائه می‌گردد.

و نهایتا استفاده از این گزینه به بالا رفتن سرعت بارگذاری سایت، کمک شایانی را خواهد نمود.

ماخذ

نظرات خوانندگان

نویسنده: پیمان
تاریخ: ۱۳۸۸/۰۴/۱۳ ۱۹:۳۷:۲۴

امکان واقعا جالبی هست اما با توجه به اینکه اکثر ماها در ایران هاست هایمان .net 2 هست درنتیجه خودمون باید این merg رو انجام بدیم ... مطالب این سری جالب بود لطفاروی مقالات jquery و asp.net بیشتر فوکوس کنید با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۱۳ ۲۰:۲۵:۲۲

Redistributable Package های دات نت فریم ورک رایگان هستند. اگر با هاست مکاتبه کنید مشکل لایسنس برای نصب آن نخواهند داشت.

عنوان: **بارگذاری یک یوزرکنترل با استفاده از جی‌کوئری**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۱۵ ۲۰:۰۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

مزیت استفاده از یوزر کنترل‌ها، ماژولار کردن برنامه است. برای مثال اگر صفحه جاری شما قرار است از چهار قسمت اخبار، منوی پویا ، سخن روز و آمار کاربران تشکیل شود، می‌توان هر کدام را توسط یک یوزر کنترل پیاده سازی کرده و سپس صفحه اصلی را از کنار هم قرار دادن این یوزر کنترل‌ها تهیه نمود.

با این توضیحات اکنون می‌خواهیم یک یوزکنترل ASP.Net را توسط jQuery Ajax بارگذاری کرده و نمایش دهیم. حداقل دو مورد کاربرد را می‌توان برای آن متصور شد:

الف) در اولین باری که یک صفحه در حال بارگذاری است، قسمت‌های مختلف آن‌را بتوان از یوزر کنترل‌های مختلف خواند و تا زمان بارگذاری کامل هر کدام، یک عبارت لطفا منتظر بمانید را نمایش داد. نمونه‌ی آن‌را شاید در بعضی از CMS های جدید دیده باشید. صفحه به سرعت بارگذاری می‌شود. در حالیکه مشغول مرور صفحه جاری هستید، قسمت‌های مختلف صفحه پدیدار می‌شوند.

ب) بارگذاری یک قسمت دلخواه صفحه بر اساس درخواست کاربر. مثلا کلیک بر روی یک دکمه و امثال آن.

روش کلی کار:

1) تهیه یک متد وب سرویس که یوزر کنترل را بر روی سرور اجرا کرده و حاصل را تبدیل به یک رشته کند.

2) استفاده از متد Ajax جی‌کوئری برای فراخوانی این متد وب سرویس و افزودن رشته دریافت شده به صفحه.

بدیهی است زمانیکه متد Ajax فراخوانی می‌شود می‌توان عبارت یا تصویر منتظر بمانید را نمایش داد و پس از پایان کار این متد، عبارت (یا تصویر) را مخفی نمود.

پیاده سازی:

قسمت تبدیل یک یوزر کنترل به رشته را قبلا در مقاله " تهیه قالب برای ایمیل‌های ارسالی یک برنامه ASP.Net " مشاهده کرده‌اید. در این‌جا برای استفاده از این متد در یک وب سرویس نیاز به کمی تغییر وجود داشت (KeyValuePair ها درست سریالایز نمی‌شوند) که نتیجه نهایی به صورت زیر است. یک فایل Ajax.asmx را به برنامه اضافه کرده و سپس در صفحه Ajax.asmx.cs کد آن به صورت زیر می‌تواند باشد:

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Reflection;
using System.Text.RegularExpressions;
using System.Web;
using System.Web.Script.Services;
using System.Web.Services;
using System.Web.UI;
using System.Web.UI.HtmlControls;

namespace AjaxTest
{
   public class KeyVal
   {
       public string Key { set; get; }
       public object Value { set; get; }
   }

   /// <summary>
   /// Summary description for Ajax
   /// </summary>
   [ScriptService]
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.ComponentModel.ToolboxItem(false)]
   public class Ajax : WebService
   {
       /// <summary>
       /// Removes Form tags using Regular Expression
```

```
        /// </summary>
        private static string cleanHtml(string html)
        {
            return Regex.Replace(html, @"<[/]?(form)[^>]*?>", string.Empty, RegexOptions.IgnoreCase);
        }

        /// <summary>
        /// تبدیل یک یوزر کنترل به معادل اچ تی ام ال آن
        /// </summary>
        /// <param name="path">مسیر یوزر کنترل</param>
        /// <param name="properties">لیست خواص به همراه مقادیر مورد نظر</param>
        /// <returns></returns>
        /// <exception cref="NotImplementedException"><c>NotImplementedException</c>.</exception>
        [WebMethod(EnableSession = true)]
        [ScriptMethod(ResponseFormat = ResponseFormat.Json)]
        public string RenderUserControl(string path,
              List<KeyVal> properties)
        {
            Page pageHolder = new Page();

            UserControl viewControl =
               (UserControl)pageHolder.LoadControl(path);

            viewControl.EnableViewState = false;

            Type viewControlType = viewControl.GetType();

            if (properties != null)
                foreach (var pair in properties)
                {
                    if (pair.Key != null)
                    {
                        PropertyInfo property =
                            viewControlType.GetProperty(pair.Key);

                        if (property != null)
                        {
                            if (pair.Value != null) property.SetValue(viewControl, pair.Value, null);
                        }
                        else
                        {
                            throw new NotImplementedException(string.Format(
                                                                    "UserControl: {0} does not have a
public {1} property.",
                                                                    path, pair.Key));
                        }
                    }
                }

            //Form control is mandatory on page control to process User Controls
            HtmlForm form = new HtmlForm();

            //Add user control to the form
            form.Controls.Add(viewControl);

            //Add form to the page
            pageHolder.Controls.Add(form);

            //Write the control Html to text writer
            StringWriter textWriter = new StringWriter();

            //execute page on server
            HttpContext.Current.Server.Execute(pageHolder, textWriter, false);

            // Clean up code and return html
            return cleanHtml(textWriter.ToString());
        }
    }
}
```

تا این‌جا متد وب سرویسی را داریم که می‌تواند مسیر یک یوزر کنترل را به همراه خواص عمومی آن‌را دریافت کرده و سپس یوزر کنترل را رندر نموده و حاصل را به صورت HTML به شما تحویل دهد. با استفاده از reflection خواص عمومی یوزر کنترل یافت شده و مقادیر لازم به آن‌ها پاس می‌شوند.

چند نکته:

الف) وب کانفیگ برنامه ASP.Net شما اگر با VS 2008 ایجاد شده باشد مداخل لازم را برای استفاده از این وب سرویس توسط jQuery Ajax دارد در غیر اینصورت موفق به استفاده از آن نخواهید شد.
ب) هنگام بازگرداندن این اطلاعات با فرمت json = ResponseFormat.Json جهت استفاده در jQuery Ajax ، گاهی از اوقات بسته به حجم بازگردانده شده ممکن است خطایی حاصل شده و عملیات متوقف شد. این طول پیش فرض را (maxJsonLength) در وب کانفیگ به صورت زیر تنظیم کنید تا مشکل حل شود:

```
<system.web.extensions>
   <scripting>
     <webServices>
       <jsonSerialization maxJsonLength="10000000"></jsonSerialization>
     </webServices>
   </scripting>
 </system.web.extensions>
```

برای پیاده سازی قسمت Ajax آن برای اینکه کار کمی تمیزتر و با قابلیت استفاده مجدد شود یک پلاگین تهیه شده (فایلی با نام jquery.advloaduc.js) که سورس آن به صورت زیر است:

```
$.fn.advloaduc = function(options) {
   var defaults = {
       webServiceName: 'Ajax.asmx', //نام فایل وب سرویس ما
       renderUCMethod: 'RenderUserControl', //متد وب سرویس
       ucMethodJsonParams: '{path:\'\'}',//پارامترهایی که قرار است پاس شوند
       completeHandler: null //پس از پایان کار وب سرویس این متد جاوا اسکریپتی فراخوانی می‌شود
   };
   var options = $.extend(defaults, options);

   return this.each(function() {
       var obj = $(this);
       obj.prepend("<div align='center'> لطفا اندکی تامل بفرمائید... <img
src=\"images/loading.gif\"/></div>");

       $.ajax({
           type: "POST",
           url: options.webServiceName + "/" + options.renderUCMethod,
           data: options.ucMethodJsonParams,
           contentType: "application/json; charset=utf-8",
           dataType: "json",
           success:
                   function(msg) {
                       obj.html(msg.d);

                       // if specified make callback and pass element
                       if (options.completeHandler)
                           options.completeHandler(this);
                   },
           error:
                   function(XMLHttpRequest, textStatus, errorThrown) {
                       obj.html("امکان اتصال به سرور در این لحظه مقدور نیست. لطفا مجددا سعی کنید.");
                   }
       });
   });
};
```

برای اینکه با کلیات این روش آشنا شوید می‌توان به مقاله " بررسی وجود نام کاربر با استفاده از jQuery Ajax در ASP.Net " مراجعه نمود که از ذکر مجدد آن‌ها خودداری می‌شود. همچنین در مورد نوشتن یک پلاگین جی‌کوئری در مقاله " افزونه جملات قصار jQuery " توضیحاتی داده شده است.
عمده کاری که در این پلاگین صورت می‌گیرد فراخوانی متد Ajax جی‌کوئری است. سپس به متد وب سرویس ما (که در اینجا نام آن به صورت پارامتر نیز قابل دریافت است)، پارامترهای لازم پاس شده و سپس نتیجه حاصل به یک شیء در صفحه اضافه می‌شود.
completeHandler آن اختیاری است و پس از پایان کار متد اجکس فراخوانی می‌شود. در صورتیکه به آن نیازی نداشتید یا مقدار آن را null قرار دهید یا اصلا آن‌را ذکر نکنید.

مثالی در مورد استفاده از این وب سرویس و همچنین پلاگین جی‌کوئری نوشته شده:

الف) یوزر کنترل ساده زیر را به پروژه اضافه کنید:

```
<%@ Control Language="C#" AutoEventWireup="true" CodeBehind="part1.ascx.cs"
Inherits="TestJQueryAjax.part1" %>
<asp:Label runat="server" ID="lblData" ></asp:Label>
```

بدیهی است یک یوزر کنترل می‌تواند به اندازه یک صفحه کامل پیچیده باشد به همراه انواع و اقسام ارتباطات با دیتابیس و غیره.

سپس کد آن‌را به صورت زیر تغییر دهید:

```
using System;
using System.Threading;

namespace TestJQueryAjax
{
   public partial class part1 : System.Web.UI.UserControl
   {
       public string Text1 { set; get; }
       public string Text2 { set; get; }

       protected void Page_Load(object sender, EventArgs e)
       {
           Thread.Sleep(3000);
           if (!string.IsNullOrEmpty(Text1) && !string.IsNullOrEmpty(Text2))
               lblData.Text = Text1 + "<br/>" + Text2;
       }
   }
}
```

این یوزر کنترل دو خاصیت عمومی دارد که توسط وب سرویس مقدار دهی خواهد شد و نهایتا حاصل نهایی را در یک لیبل در دو سطر نمایش می‌دهد.
عمدا یک sleep سه ثانیه‌ای در اینجا در نظر گرفته شده تا اثر آن‌را بهتر بتوان مشاهده کرد.

ب) اکنون کد مربوط به صفحه‌ای که قرار است این یوزر کنترل را به صورت غیرهمزمان بارگذاری کند به صورت زیر خواهد بود (مهم‌ترین قسمت آن نحوه تشکیل پارامترها و مقدار دهی خواص یوزر کنترل است):

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="TestJQueryAjax._Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>

   <script src="js/jquery.js" type="text/javascript"></script>
   <script src="js/jquery.advloaduc.js" type="text/javascript"></script>
   <script src="js/json2.js" type="text/javascript"></script>

   <script type="text/javascript">
       function showAlert() {
           alert('finished!');
       }

       //تشکیل پارامترهای متد وب سرویس جهت ارسال به آن
       var fileName = 'part1.ascx';
       var props = [{ 'Key': 'Text1', 'Value': 'سطر یک' }, { 'Key': 'Text2', 'Value': 'سطر 2'}];
       var jsonText = JSON.stringify({ path: fileName, properties: props });

       $(document).ready(function() {
           $("#loadMyUc").advloaduc({
               webServiceName: 'Ajax.asmx',
               renderUCMethod: 'RenderUserControl',
               ucMethodJsonParams: jsonText,
               completeHandler: showAlert
           });
       });
```

```
   </script>

</head>
<body>
   <form id="form1" runat="server">
   <div id="loadMyUc">
   </div>
   </form>
</body>

</html>
```

نکته:

برای ارسال صحیح و امن اطلاعات json به سرور، از اسکریپت استاندارد [json2.js](#) استفاده شد.

ابزار دیباگ:

بهترین ابزار برای دیباگ این نوع اسکریپت‌ها استفاده از افزونه فایرباگ فایرفاکس است. برای مثال مطابق تصویر زیر، یوزر کنترلی فراخوانی شده است که در سرور وجود ندارد:

[دریافت مثال فوق](#)

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۴/۱۵ ۲۲:۱۳:۴۲

سلام استاد نصیری
بسیار مقاله مفید و Trick جالبی بود.به تازگی افزونه ای برای فایر باگ عرضه شده به اسم Fire Query که مواردی را برای کار و دیباگ jQuery فراهم میکنه.
یه سوال از حضورتون داشتم چرا خروجی Execute در وب سرویس شامل تگ های html و body نیست؟آیا این همیشه ثابته؟ و اینکه مواردی مثل \r و \n رو هم از خروجی حذف کنیم بهتر نیست؟
باسپاس از شما استاد عزیز

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۴/۱۵ ۲۲:۳۷:۳۶

سلام آقای نصیری
امکانش هست در مورد متد stringify و عملی که در اینجا انجام میده کمی توضیح بدهید؟ من اون لینک صفحه JSON رو خوندم ولی متوجه نشدم چرا شما اینجا از این متد استفاده کردید.
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۱۵ ۲۳:۱۰:۱۶

@نیما
بله. یک trim را هم می‌توانید اضافه کنید بعلاوه نکته حذف فاصله خالی بین تگ‌ها که در کاهش حجم نهایی مؤثر است.

@ناشناس
می‌شد این آرگومان‌های مورد نظر رو به شکل یک رشته یک سطری هم در آورد (فرقی نمی‌کرد). اما با استفاده از این تابع شما می‌تونید یک شیء جاوا اسکریپتی را تبدیل به json کنید که از این لحاظ خوب یک قدم پیشرفت است و در خطایابی کدها کمک بزرگی است (رشته قابل خطایابی نیست).

نویسنده: مهدی
تاریخ: ۱۳۸۸/۰۵/۰۹ ۱۷:۴۵:۰۹

ایده خیلی جالبی بود. اما چیزی که الان فکر من رو مشغول کرده، اینه که چطور ممکنه که از postbackهای این کنترل استفاده کرد.
اگر خودتون در این مورد کار کردین هم یک توضیحی بدین. اگر هم کاری نکردین که من خودم اگر وقت کردم یک تستی میکنم و اگر جواب داد به شما هم خبر میدم. نباید کار سختی باشه ولی بازم تا شروع نکنم نمیتونم قطعی نظر بدم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۰۹ ۱۹:۰۸:۲۳

اگر خطای viewstate is corrupted را دریافت می‌کنید به این علت است که الان در صفحه دو فیلد مخفی viewstate داریم.
برای حذف این مورد دوم سطر زیر را به تابع cleanHtmlاضافه کنید:

```
;(html = Regex.Replace(html, "<input.*name=\"__VIEWSTATE\"[^>]*>", string.Empty, RegexOptions.IgnoreCase
```

+ در این حالت چون اطلاعات کنترل‌ها به ViewState اضافه نمی‌شوند، ASP.Net Postback هم کار نمی‌کنه.
در این حالت برای ارسال دیتا و غیره امکان استفاده از وب سرویس و پاس کردن مقدار به آن هست (با استفاده از جی‌کوئری اجکس).

کمی بیشتر کار میبره نسبت به MS Ajax.

نویسنده: sirasad
تاریخ: ۱۳۸۸/۰۵/۱۴ ۱۲:۳۲:۰۴

آقای نصیری عزیز من خواستم این کد رو در .net 2.0 تست بکنم , مقادیر اون تکست ها منتقل نمی شود ولی showalert رو فایرآپ میکنه .
یه مسئله دیگه اینکه این ajax شما بدون پارامتر ورودی کار میکنه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۴ ۱۳:۳۳:۳۷

بله. اگر پارامتر ورودی نداشت یک '{}' قرار دهید کفایت می‌کنه.
برای دات نت 2 کمی کار شما بیشتر می‌شود چون در VS2008 خیلی از موارد را خودکار اضافه می‌کند.
در این حالت یا باید ASP.NET AJAX Extensions روی سرور نصب باشد یا dll های آن‌را به صورت دستی به دایرکتوری bin برنامه اضافه کنید و یک ارجاع از آن‌ها را نیز به برنامه اضافه کنید.
همچنین وب کانفیگ را نیز باید ویرایش کنید تا این تغییر در آن لحاظ شود. (در غیر اینصورت کار نمی‌کند)
یک نکته دیگر هم اینکه بجای msg.d شما فقط msg را در asp.net 2 دارید و این d از 3.5 به بعد اضافه شده (در قسمت success و هنگام دریافت پاسخ از سرور توسط jQuery).

نویسنده: javad.mjt
تاریخ: ۱۳۸۹/۱۰/۲۱ ۱۵:۴۵:۳۸

سلام آقای نصیری
زمانی که یک دکمه یا کنترلی که postback ایجاد می کند را استفاده می کنم به خطای زیر بر می خورم
The state information is invalid for this page and might be corrupted.
Description: An unhandled exception occurred during the execution of the current web request. Please review the stack trace for more information about the error and where it originated in the code.

Exception Details: System.Web.HttpException: The state information is invalid for this page and might be corrupted.
لطفا راهنمایی کنید با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۱ ۱۷:۱۵:۰۲

لطفا کامنت‌ها را مطالعه کنید در بالا توضیح دادم (جولای ۳۱, ۲۰۰۹، ساعت ۲۰:۰۸).
1- باید ViewState مربوط به این یوزر کنترل را خاموش کنید.
2- خاموش کردن ViewState سبب حذف فیلد مخفی آن به صورت کامل نمی‌شود، علاوه بر آن باید فیلد مخفی ViewState موجود در رشته را با استفاده از Regex ایی که در بالا توضیح دادم کلا حذف کنید تا با ViewState اصلی صفحه تداخل نکند.

عنوان: **فشرده سازی خروجی یک وب سرویس**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۴/۲۲ ۲۱:۵۱:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

جهت بهینه سازی روش ارائه شده در مقاله " بارگذاری یک یوزرکنترل با استفاده از جی‌کوئری " ، می‌توان مبحث فشرده سازی را نیز به آن افزود.

برای این منظور نیاز است تا بتوان response حاصل را کاملا کنترل کرد و این مورد از طریق یک http module به خوبی قابل انجام است. مبحث http compression و پیاده سازی آن‌را احتمالا بارها در سایت‌های مختلف نیز دیده‌اید:

```
using System;
using System.IO;
using System.IO.Compression;
using System.Globalization;
using System.Web;


public class JsonCompressionModule : IHttpModule
{
   public JsonCompressionModule()
   {
   }

   public void Dispose()
   {
   }

   public void Init(HttpApplication app)
   {
       app.PreRequestHandlerExecute += new EventHandler(Compress);
   }

   private void Compress(object sender, EventArgs e)
   {
       HttpApplication app = (HttpApplication)sender;
       HttpRequest request = app.Request;
       HttpResponse response = app.Response;

       if (request.ContentType.ToLower(CultureInfo.InvariantCulture).StartsWith("application/json"))
       {
           if (!((request.Browser.IsBrowser("IE")) && (request.Browser.MajorVersion <= 6)))
           {
               string acceptEncoding = request.Headers["Accept-Encoding"];

               if (!string.IsNullOrEmpty(acceptEncoding))
               {
                   acceptEncoding = acceptEncoding.ToLower(CultureInfo.InvariantCulture);

                   if (acceptEncoding.Contains("gzip"))
                   {
                       response.Filter = new GZipStream(response.Filter, CompressionMode.Compress);
                       response.AddHeader("Content-encoding", "gzip");
                   }
                   else if (acceptEncoding.Contains("deflate"))
                   {
                       response.Filter = new DeflateStream(response.Filter, CompressionMode.Compress);
                       response.AddHeader("Content-encoding", "deflate");
                   }
               }
           }
       }
   }
}
```

در این ماژول تنها درخواست‌هایی از نوع application/json بررسی خواهند شد. هر چند این فشرده سازی را بر روی خروجی هر نوع WebMethod ایی نیز می‌توان اعمال کرد. در این حالت، سطر بررسی json را حذف کرده و آن‌را به صورت زیر تغییر دهید:

```
if ( !request.Url.PathAndQuery.ToLower().Contains( ".asmx" ) )
```

```
    return;
```

مرورگر IE6 و پایین‌تر نیز از این فشرده سازی معاف شده‌اند (چون یا پشتیبانی کاملی را ارائه نمی‌دهند یا باید منتظر کرش مرورگر بود).

جهت اعمال این ماژول به برنامه ASP.Net خود، کافی است سطر زیر را به قسمت httpModules وب کانفیگ افزود:

```
<httpModules>
   <add name="JsonCompressionModule" type="JsonCompressionModule"/>
</httpModules>
```

روش آزمایش ماژول تهیه شده:

متاسفانه افزونه‌ی فایرباگ فایرفاکس اندازه‌ی نهایی response را نمایش می‌دهد و در گزارش آن حتی خبری از Content-encoding اضافه شده نیز نخواهد بود. بنابراین برای بررسی این روش مناسب نیست.
ابزار دیگری که اساسا برای این نوع آزمایشات طراحی شده است، برنامه معروف فیدلر می‌باشد (که توسط مدیر پروژه تیم IE برنامه نویسی شده است).
برای استفاده از فیدلر جهت دیباگ درخواست‌های local باید یک نکته‌ی کوچک را رعایت کرد:

/http://localhost . :25413

همانطور که در URL فوق مشاهده می‌کنید یک نقطه پس از localhost اضافه شده است تا خروجی محلی مربوطه قابل بررسی شود.

مطابق تصویر فوق، هم content-encoding اضافه شده مشخص است و هم حجم پاسخ دریافتی از 40 کیلوبایت (بر اساس یک تست معمولی روی صفحه‌ای مشخص) به نزدیک یک کیلوبایت و اندی کاهش یافته است.

عنوان: **نمایش پیغامی به کاربر در هنگام استفاده از MS Ajax**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۰۹ ۲۱:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

مطلب " نمایش یک پیغام به کاربر در ASP.Net " را احتمالا به خاطر دارید. این روش در حالت postback کامل و یا در حالت page_load بدون مشکل کار می‌کند. اما اگر در حال استفاده از آپدیت پنل MS Ajax باشیم، دیگر کار نخواهد کرد چون تنها قسمتی از صفحه به روز رسانی می‌شود و یک postback کامل رخ نمی‌دهد تا از شیء صفحه جهت تزریق اسکریپت خود بتوانیم استفاده نمائیم. خوشبختانه طراحان MS Ajax به این مورد هم توجه داشته‌اند و متد RegisterStartupScript را جزئی از کلاس ScriptManager قرار داده‌اند. برای مثال اگر نیاز باشد در پایان عملیات یک متد که از آپدیت پنل کمک می‌گیرد، یک alert نمایش داده شود می‌توان از متد زیر استفاده کرد:

```
void jsCall(string alert)
{
 string script = @"alert('" + alert + "');";
 ScriptManager.RegisterStartupScript(this, this.GetType(), "jsCall", script, true);
}
```

با توجه به این نکته، مطلب "نمایش یک پیغام به کاربر در ASP.Net" را به صورت زیر نیز می‌توان با MS Ajax سازگار کرد:

```
if (page != null)
{
    ScriptManager sMgr = ScriptManager.GetCurrent(page);
     if (sMgr != null)
     {
        ScriptManager.RegisterStartupScript(page, page.GetType(),
            "script" + new Guid().ToString("N"),
             scriptBlock, false);
      }
      else
        //همان روش قبلی
...
```

عنوان: **استفاده از jQuery Ajax جهت تعیین اعتبار یک فرم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۰ ۲۰:۵۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

فرض کنید تعیین اعتبار یکی از فیلدهای فرم نیاز به انجام محاسباتی در سمت سرور دارد و این‌کار را می‌خواهیم با استفاده از jQuery Ajax انجام دهیم. مشکلی که در اینجا وجود دارد، این است که A در Ajax به معنای asynchronous است. یعنی زمانیکه کاربر دکمه submit را فشرد، دیگر برنامه منتظر این نخواهد شد که پاسخ کامل دریافت شود ، سایر پردازش‌ها صورت گیرد و سپس فرم را به سرور ارسال نماید (شبیه به ایجاد یک ترد جدید در برنامه‌های ویندوزی). مثال زیر را در نظر بگیرید:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TestCustomValidation.aspx.cs"
 Inherits="TestJQueryAjax.TestCustomValodation" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
 <title></title>

 <script src="js/jquery.js" type="text/javascript"></script>

 <script type="text/javascript">
     function validate() {
         var number1 = $("#<%=txtNumber1.ClientID %>").val();
         var number2 = $("#<%=txtNumber2.ClientID %>").val();
         var result = false;
         $.ajax({
             type: "POST",
             url: 'AjaxSrv.asmx/ValidateIt',
             data: '{"number1":' + number1 + ',"number2":' + number2 + '}',
             contentType: "application/json; charset=utf-8",
             dataType: "json",
             success:
                 function(msg) {
                     if (msg.d) {
                         result = true;
                         alert('بسیار خوب');
                     }
                     else {
                         result = false;
                         alert('دوباره سعی کنید');
                     }
                 },
             error:
                 function(XMLHttpRequest, textStatus, errorThrown) {
                     result = false;
                     alert("خطایی رخ داده است");
                 }
         });
         //debugger;
         return result;
     }
 </script>

</head>
<body>
 <form id="form1" runat="server">
 <div>
     number 1 :
     <asp:TextBox runat="server" ID="txtNumber1" />
     <br />
     number 2 :
     <asp:TextBox runat="server" ID="txtNumber2" />
     <br />
     <asp:Button ID="btnSubmit" Text="Submit" UseSubmitBehavior="false" runat="server"
         OnClientClick="if(!validate()){ return false;}" OnClick="btnSubmitClick" />
 </div>
 </form>
</body>
</html>
```

این مثال یک نوع اعتبار سنجی سفارشی را در حین submit با استفاده از وب سرویس زیر انجام می‌دهد (حاصلضرب دو عدد دریافتی را بررسی می‌کند که باید مساوی 10 باشند. البته هدف از این مثال ساده، آشنایی با نحوه‌ی انجام این نوع عملیات است که می‌تواند شامل کار با دیتابیس و غیره هم باشد. و گرنه بدیهی است این بررسی را با دو سطر کد جاوا اسکریپتی نیز می‌شد انجام داد):

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Web.Script.Services;

namespace TestJQueryAjax
{
 /// <summary>
 /// Summary description for AjaxSrv
 /// </summary>
 [WebService(Namespace = "http://tempuri.org/")]
 [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
 [System.ComponentModel.ToolboxItem(false)]
 [ScriptService]
 public class AjaxSrv : System.Web.Services.WebService
 {
     [WebMethod(EnableSession = true)]
     [ScriptMethod(ResponseFormat = ResponseFormat.Json)]
     public bool ValidateIt(int number1, int number2)
     {
         return number1 * number2 == 10;
     }
 }
}
```

در این مثال هنگام submit ، عملیات اعتبار سنجی با توجه به وضعیت asynchronous عملیات Ajax ، تمام و کمال رخ نداده ، یا اعتبار سنجی انجام نمی‌شود و یا پیغام خطایی را دریافت خواهیم کرد.

راه حل چیست؟

راه حل‌های فضایی بسیاری را در وب در این مورد می‌توان پیدا کرد؛ اما راه حل استاندارد آن در این حالت ویژه، استفاده از Ajax در حالت غیرهمزمان است. یعنی این فریم ورک Ajax را وادار کنیم که تا پایان عملیات مورد نظر، منتظر بماند و سپس فرم را ارسال کند. برای این منظور تنها کافی است یک سطر زیر را پیش از فراخوانی تابع Ajax ، اضافه و فراخوانی نمائیم:

```
$.ajaxSetup({async: false}) ;
```

نکته:

UseSubmitBehavior دکمه ما را به شکل زیر رندر می‌کند (دکمه به یک button معمولی (بجای حالت submit) تبدیل شده و سپس یک doPostBack را اضافه خواهد کرد):

```
<input id="btnSubmit" type="button" onclick="if(!validate()){ return
false;};__doPostBack('btnSubmit','')" value="Submit" name="btnSubmit"/>
```

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۱۳۸۸/۰۵/۱۲ ۰۹:۳۲:۴۸

سلام و خسته نباشید.

جناب نصیری من کلا هدف شما از این کار رو متوجه نشدم.

یعنی شما قصد اینو دارید که تا اعتبار سنجی انجان نشده و جواب فرستاده نشده فرم منتظر بمونه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۲ ۱۰:۰۴:۵۴

سلام،
دو هدف در این مثال مد نظر بود:
الف) تا پایان عملیات اعتبار سنجی توسط عملیات Ajax ایی صبر شود و فرم به یکباره به سرور ارسال نگردد.
ب) اگر فرم اعتبار سنجی نشد، به سرور submit نشود.

عنوان: **تعیین اعتبار یک checkBoxList با کمک jQuery**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۱۹ ۱۶:۰۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

checkBoxList جزو کنترل‌هایی در ASP.Net است که نمی‌توان RequiredFieldValidator استاندارد را بر آن اعمال کرد. به عبارتی اگر نیاز بود حداقل یک آیتم چک باکس لیست حتما توسط کاربر انتخاب شود، راه حل آماده‌ای برای آن وجود ندارد. پیاده سازی این‌کار با استفاده از jQuery به سادگی میسر است که در ادامه آن‌را مرور خواهیم کرد.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="CheckBoxListValidator._Default" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
   <script src="jquery.min.js" type="text/javascript"></script>
   <script type="text/javascript">
       //<![CDATA[
       function CheckItems(sender, args) {
           //Get the total nuumber of selected CheckBoxes
           var num = jQuery("table#<%=CheckBoxList1.ClientID%> input:checked").length;
           args.IsValid = num > 0;
       }
       //]]>
   </script>
</head>
<body>
   <form id="form1" runat="server">
   <div>
       <asp:CheckBoxList ID="CheckBoxList1" runat="server">
           <asp:ListItem>item1</asp:ListItem>
           <asp:ListItem>item2</asp:ListItem>
       </asp:CheckBoxList>
       <asp:CustomValidator ClientValidationFunction="CheckItems" ID="ValidateIt"
           runat="server" ErrorMessage="(*)"> </asp:CustomValidator>
       <asp:Button ID="Button1" runat="server" />
   </div>
   </form>
</body>
</html>
```

توضیحات:
یک CustomValidator استاندارد را به فرم اضافه کرده‌ایم تا توسط تابعی که به ClientValidationFunction آن معرفی می‌شود، کار اعتبار سنجی سمت کاربر را انجام دهد. این تابع یا همان CheckItems مثال فوق، امضای استاندارد و آشنایی دارد. اگر تعیین اعتبار صورت گرفته باشد، باید args.IsValid در آن به true تنظیم شود یا بر عکس.
اصل قضیه هم، همین یک سطر کد زیر است:

```
var num = jQuery("table#<%=CheckBoxList1.ClientID%> input:checked").length;
```

کار این سطر که از جی‌کوئری استفاده می‌کند، پیدا کردن جدولی است که ID آن مساوی آی دی سمت کلاینت چک باکس لیست ما است (ASP.Net یک چک باکس لیست را به صورت یک جدول حاوی چک باکس‌ها رندر می‌کند). سپس در همان ناحیه مشغول به جستجوی چک باکس‌هایی می‌شود که تیک خورده‌اند. نهایتا تعداد آن‌ها را بر می‌گرداند.

نظرات خوانندگان

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۰۵/۱۹ ۲۰:۳۰:۰۶

با سلام خدمت استاد نصیری
می دونم که این مطلب به پست شما ربطی نداره ولی خواهش می کنم راهنمایی کنید.

آقای نصیری مطلبی رو پیدا کردم که توسط اون میشه از توابع نوشته شده در C# در Crystal report استفاده کرد:
اما هر کاری می کنم که اون رو در بخش فرمول نویسی در بخش Additional function ببینم اصلا گزینه مربوطه (همون فایل dll)نمایش داده نمی شه.
روی چند سیستم هم این کارو انجام دادم ولی گزینه مورد نظر وجود نداشته!!!

چیکار کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۲۰ ۰۰:۱۶:۳۰

سلام،
این dll ها (یا اسمبلی‌های دات نتی) باید ComVisible شوند تا در Crystal report دیده شوند. بعلاوه در تنظیمات پروژه باید تیک مربوط به Register for COM Interop در برگه build گذاشته شود (مهم).
توضیح قدم به قدم آن در این‌جا هست:
http://msdn.microsoft.com/en-us/library/ms227603(VS.80).aspx

نویسنده: Hossein Moradinia
تاریخ: ۱۳۸۸/۰۵/۲۵ ۲۰:۴۸:۲۹

ممنون وحید جان
مشکلم حل شد.
از اینکه جواب دادید واقعا ممنون

نویسنده: مجید مهرتاش
تاریخ: ۱۳۸۸/۰۸/۲۱ ۱۳:۳۸:۰۰

ممنون ، خیلی جالب بود !

عنوان: **آشنایی با jQuery Live**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۵/۲۶ ۱۹:۲۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

در نگارش‌های اخیر کتابخانه jQuery (از نگارش 1.3 به بعد) متدی به نام live به آن اضافه شده است که کاربرد آن‌را در ادامه مرور خواهیم کرد.

ابتدا مثال زیر را در نظر بگیرید:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TestLive.aspx.cs"
Inherits="TestJQueryAjax.TestLive" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
 <title></title>

 <script src="js/jquery.js" type="text/javascript"></script>

 <script type="text/javascript">
     $(document).ready(function() {
         $('a.mylink').click(function(e) {
             var $a = $(this);
             alert($a.attr('id'));
         });

         $('a#lnkLoad').click(function(e) {
             $('div#dynContent').load('live.ashx');
         });
     });
 </script>

</head>
<body>
 <form id="form1" runat="server">
 <a href='#' id='lnk1' class='mylink'>link1</a>
 <br />
 <a href='#' id='lnkLoad'>load .ashx</a>
 <div id='dynContent'>
 </div>
 </form>
</body>

</html>
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace TestJQueryAjax
{
 /// <summary>
 /// Summary description for $codebehindclassname$
 /// </summary>
 [WebService(Namespace = "http://tempuri.org/")]
 [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
 public class live : IHttpHandler
 {

     public void ProcessRequest(HttpContext context)
     {
         context.Response.ContentType = "text/plain";
         context.Response.Write("<a href='#' id='lnk2' class='mylink'>link2</a>");
     }

     public bool IsReusable
     {
```

```
        get
        {
            return false;
        }
    }
 }
}
```

در این مثال ساده با کلیک بر روی لینک‌هایی با کلاس mylink ، یک alert حاوی id آن لینک نمایش داده خواهد شد.
همچنین اگر بر روی لینکی با id مساوی lnkLoad کلیک شود، محتوایی پویا از یک generic handler به نام live.ashx دریافت شده و به div ایی با id مساوی dynContent اضافه می‌گردد.
این محتوای دریافتی از generic handler ما نیز کلاسی مساوی mylink دارد، اما این‌بار هر چقدر بر روی لینک اضافه شده به صفحه کلیک کنیم کار نمی‌کند. چرا؟ چون در هنگام فراخوانی document.ready ، این لینک وجود نداشته و روال رخدادگردانی به آن bind نشده است.
به صورت خلاصه می‌خواهیم روال کلیک بر روی لینک‌هایی با کلاس mylink همیشه کار کند. (چه در مورد عناصری در صفحه که از قبل وجود داشته‌اند و چه عناصری که توسط عملیاتی Ajax ایی بعدا اضافه خواهند شد)
این مشکل با معرفی متد live حل شده است. برای این منظور تنها کافی است کد ما به صورت زیر تغییر کند:

```
<script type="text/javascript">
    $(document).ready(function() {
        $('a.mylink').live("click", function() {
            var $a = $(this);
            alert($a.attr('id'));
        });
.
.
.
```

اکنون jQuery کلیه لینک‌هایی با کلاس مساوی mylink را که از این پس اضافه خواهند شد، به صورت live و زنده تحت نظر قرار می‌دهد و عکس العمل نشان خواهد داد.

نظرات خوانندگان

نویسنده: مجتبی صحرائی
تاریخ: ۲۳:۱۵:۱۹ ۱۳۸۸/۰۵/۲۶

ممنون؛ نکته بسیار خوبی بود

نویسنده: احمد احمدی
تاریخ: ۰:۲۴ ۱۳۹۱/۰۶/۱۹

بعضی وقتا که از متد live استفاده می‌کنیم ، رویداد یک المنت چندبار اجرا میشه .
برای شما هم پیش اومده ؟
راه حل چیست ؟

نویسنده: وحید نصیری
تاریخ: ۰:۴۱ ۱۳۹۱/۰۶/۱۹

نه این مشکل رو نداشتم ولی راه حلش اضافه کردن return false به انتهای روال رویدادگردان است.

نویسنده: احمد احمدی
تاریخ: ۱۳:۲۶ ۱۳۹۱/۰۶/۲۴

ظاهرا این مشکل در سایت جاری هم وجود دارد :

نویسنده: وحید نصیری
تاریخ: ۱۳:۳۳ ۱۳۹۱/۰۶/۲۴

چند جایی رو تست کردم چنین چیزی مشاهده نشد. در کدام صفحه دقیقا؟ (تمام متدهای live سایت return false دارند)
آدرس Post یکی است؛ مقادیر Post ارسالی متفاوت (بسته به کلیک بر روی المانی خاص).

نویسنده: احمد احمدی
تاریخ: ۱۴:۱۸ ۱۳۹۱/۰۶/۲۴

هنگام امتیاز دهی به یک مطلب . مثلا همین الان 2 درخواست برای امتیازدهی همین پست ارسال شد .
در کنسول فایرباگ این کد رو اجرا کنید :

```
$(".postRating[sectiontype=BlogPost] .rating[value=5]").click();
```

نتیجه 2 درخواست بود برای من .

دفعات بعد هم شد 3 درخواست .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۴ ۱۸:۵۴

ممنون. مشکل برطرف شد.

عنوان: **ELMAH 1.1**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۲ ۱۹:۱۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

نگارش جدید ELMAH ارائه شده است. این نگارش کاملا با نگارش قبلی سازگار بوده و تنها بازنویسی فایل dll آن با نگارش جدید کفایت می‌کند.

لیست موارد فیکس شده

```
- System.Data.SQLite.dll causes image problems on 64-bit IIS/ASP.NET
- Incorrect jscript filter sample
```

لیست تازه‌های آن

```
- Ability to set mail priority in errorMail config section
```

```
- Allow configuration-based filtering rules to be diffrent for e-mail and log
```

```
- Adjust behavior of smtpPort attribute in
```

دریافت از گوگل کد و یا دریافت از رپید شیر (سورس + باینری)

عنوان: **تعیین اعتبار ورودی Int64 یا بزرگتر در ASP.Net**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۹ ۲۱:۳۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

کنترل range validator در ASP.net امکان کنترل ورودی از نوع double ، Int32 و غیره را فراهم می‌کند . اما اگر کاربر حتما باید عددی صحیح را وارد کرده و این عدد از بازه‌ی اعداد Int32 خارج بود (مثلا اعداد Int64 یا حتی بزرگتر) این کنترل کاربرد خود را از دست می‌دهد.

در این حالت بجای استفاده از کنترل range validaotr می‌شود از RegularExpressionValidator معروف به صورت زیر استفاده کرد:

```
<asp:RegularExpressionValidator ID="RegularExpressionValidator1" runat="server"
           ControlToValidate="TextBox1" ErrorMessage="RegularExpressionValidator"
           SetFocusOnError="True" ValidationExpression="\d+"></asp:RegularExpressionValidator>
```

در این‌جا ValidationExpression آن عبارت باقاعده‌ای است که ورودی‌هایی را متشکل از رشته‌ای حاوی یک یا چند عدد، مجاز اعلام خواهد کرد (و در این حالت مشکلی برای تعیین اعتبار اعداد صحیح بزرگ وجود نخواهد داشت).

عنوان: **استفاده از jQuery یا MS Ajax control toolkit**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۱۱ ۲۲:۱۳:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

به نظر من jQuery به چندین دلیل از کتابخانه MS Ajax و ملحقات آن مهم‌تر است و باید به آن پرداخته شود:

1- دانش شما قابل انتقال است. اگر روزی به PHP یا JSP یا موارد مشابه دیگری مهاجرت کردید، دانش jQuery شما باز هم قابل استفاده خواهد بود.
2- این کتابخانه بسیار سبک‌تر و کم حجم تر از MS Ajax است. (حجم آن کمتر از نصف است(+))
3- تقریبا از اکثر فریم ورک‌های جاوا اسکریپتی موجود سریع‌تر است. (+)
4- پشتیبانی آن از مرورگرهای مختلف بی‌نظیر است. لازم نیست عمر گرانمایه را صرف این کنید که چرا این اسکریپت در فایرفاکس کار می‌کنه اما در IE خیر.
5- کوهی از افزونه‌های مختلف برای آن موجود است. (jQuery plugin را در گوگل جستجو کنید)
6- استفاده از آن، میزان کد جاوا اسکریپتی را که باید نوشت تا حد بسیار قابل ملاحظه‌ای کاهش می‌دهد، که در نهایت سبب ایجاد کدی خواناتر در مدت زمانی کمتر می‌گردد.
7- کدنویسی با jQuery از کد نویسی JavaScript ایی خالص بسیار ساده‌تر است. افرادی که تا دیروز حتی طرف جاوا اسکریپت هم نمی‌رفتند، امروزه پلاگین‌های jQuery می‌نویسند.
8- مستندات کاملی داشته به همراه مثال‌هایی بسیار کاربردی.
9- به دلیل استفاده شدن از آن در پلتفرم‌های مختلف، مثال‌های کاربردی بیشتری داشته، همچنین منابع آموزشی پر محتواتری را نیز می‌توان در این رابطه یافت. (می‌توان tutorial های مربوط به php را مشاهده و آن‌ها را به ASP.Net تبدیل کرد و صرفا منحصر به یک عده و یک پلتفرم نیست)
10- خلاقیت شما را نکشته و از شما فردی که هیچگونه درکی نسبت به عملیات اجکسی رخ داده ندارد، نخواهد ساخت. (MS Ajax برای ASP.Net یک شاهکار است. چند عدد محصور کننده را استفاده کنید، برنامه نویسی شما با روش سابق "هیچ" تفاوتی نخواهد داشت. مسلما این توانمندی تیم مایکروسافت را نمایش می‌دهد، نه توانمندی من و شما را! اما اگر نیاز به کمی ابتکار وجود داشت اینجا است که گیر خواهید افتاد و موارد 6 و 7 و 9 ذکر شده، jQuery را بر این کتابخانه مقدم خواهد ساخت)
11- با آمدن ASP.Net MVC تمایل به استفاده از jQuery به شدت افزایش یافته که نمود آن‌را در وبلاگ‌های مطرح در این زمینه می‌توان مشاهده کرد (اکثر مطالب ارسالی این روزهای بلاگ‌های ASP.Net حول و حوش ASP.Net MVC و jQuery است).

نظرات خوانندگان

نویسنده: Anonymous
تاریخ: ۲۱:۵۰:۰۲ ۱۳۸۸/۰۶/۱۱

قبلا یک بحثی در سایت برنامه نویس در این باره شده بود من کاملا با حرفتون موافقم asp.net ajax خیلی سنگینه

نویسنده: مهدی یوسفی
تاریخ: ۰۷:۱۰:۴۹ ۱۳۸۸/۰۶/۱۵

سلام
من مطلب جالبی از نویسنده کتاب Asp.net MVC خواندم که طبق تحقیقات انجام شده توسط ماکروسافت است:
اونها به Asp.net عبارت Bloated را دادند و این به این معنی است:

:The term bloated is often thrown at frameworks or technologies that either

People don’t understand or aren’t completely aware of and don’t want to take the time to learn

Come from a provider people don’t respect or like

People have heard bad things about

Are actually bloated

و این نظریه که اغلب مردم نسبت به Asp.net Ajax دارند.

نویسنده: وحید نصیری
تاریخ: ۰۸:۳۲:۴۸ ۱۳۸۸/۰۶/۱۵

Bloated رو در مورد هر چیزی بکار می‌برند که زیاد شلوغش کرده باشه یا سنگین باشه و منابع زیادی را مصرف کند.

عنوان: **گزینه "مرا به خاطر بسپار" درست کار نمی‌کند**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۱۷ ۱۸:۱۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

حالت forms authentication در ASP.Net ، امکان تعریف کوکی‌هایی ماندگار را نیز جهت ورود خودکار کاربران در دفعات بعدی بازدید آن‌ها فراهم می‌کند. اما زمان منقضی شدن این کوکی‌های ماندگار در ASP.Net 1.1 و ASP.Net 2.0 به بعد کاملا با هم متفاوت بوده و اگر برنامه نویس از این تغییر حاصل شده مطلع نباشد ممکن است بارها و بارها برنامه را آزمایش کند اما نتیجه‌ای نگیرد.
مدت زمان منقضی شدن کوکی‌های ماندگار forms authentication در ASP.Net 1.1 به صورت زیر است (بود):
زمان منقضی شدن کوکی سشن = زمان جاری + زمان منقضی شدن سشن کاربر
زمان منقضی شدن کوکی ماندگار (persistent) = زمان جاری + 50 سال

این عدد 50 سال بسیار غیرمنطقی بوده و در ASP.Net 2.0 به بعد به صورت زیر اصلاح شده است:
زمان منقضی شدن کوکی سشن = زمان جاری + زمان منقضی شدن سشن کاربر
زمان منقضی شدن کوکی ماندگار (persistent) = زمان جاری + زمان منقضی شدن سشن کاربر (پیش فرض آن 30 دقیقه است)

و جهت بیشتر کردن طول عمر این ماندگاری باید زمان مورد نظر را در وب کانفیگ به صورت زیر اعمال کرد:

```
<authentication mode="Forms">
     <forms
       …
        name="MyCookieName"
       slidingExpiration="true"
       timeout="43200"
       />
   </authentication>
```

مطابق زمان تنظیم شده فوق که بر حسب دقیقه است، این کوکی به مدت یک ماه ماندگاری پیدا می‌کند در غیر اینصورت اعمال گزینه "مرا به خاطر بسپار" مطابق نظر شما کار نخواهد کرد و همانند ASP.Net 1.1 به صورت پیش فرض به 50 سال تنظیم نمی‌شود.

همچنین نکته‌ی مهم دیگری را که باید رعایت کرد، name ایی است که در این فایل config عنوان می‌کنید (در قسمت تنظیمات form authentication). اگر بر روی یک وب سرور، چندین برنامه وب ASP.Net را در حال اجرا دارید، باید برای هر کدام از این‌ها نامی جداگانه و منحصربفرد انتخاب کنید، در غیراینصورت تداخل رخ داده و باز هم گزینه مرا به خاطر بسپار شما کار نخواهد کرد.

کار slidingExpiration که در اینجا تنظیم شده است نیز به صورت زیر می‌باشد:
اگر لاگین موفقیت آمیزی ساعت 5 عصر صورت گیرد و timeout شما به عدد 10 تنظیم شده باشد، این لاگین به صورت خودکار در 5:10 منقضی خواهد شد. اما اگر در این حین در ساعت 5:05 ، کاربر یکی از صفحات سایت شما را مرور کند، زمان منقضی شدن کوکی ذکر شده به 5:15 تنظیم خواهد شد. (مفهوم تنظیم slidingExpiration)
لازم به ذکر است که اگر کاربر پیش از نصف زمان منقضی شدن کوکی (مثلا در 5:04)، یکی از صفحات را مرور کند، تغییری در این زمان نهایی منقضی شدن رخ نخواهد داد.

عنوان: **استفاده‌ی همزمان از آپدیت پنل ASP.Net و پلاگین‌های جی‌کوئری**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۲۰ ۱۸:۴۷:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

مشکل: زمانیکه یک AsyncPostback در آپدیت پنلAjax ASP.Net رخ دهد، پس از پایان کار، پلاگین جی‌کوئری که در حال استفاده از آن بودید و در هنگام بارگذاری اولیه صفحه بسیار خوب کار می‌کرد، اکنون از کار افتاده است و دیگر جواب نمی‌دهد.

قبل از شروع، نیاز به یک سری پیش زمینه هست (شاید بر اساس روش استفاده شما از آن پلاگین جی‌کوئری، مشکل را حل کنند):
الف) رفع تداخل جی‌کوئری با سایر کتابخانه‌های مشابه.
ب) آشنایی با jQuery Live جهت بایند رخدادها به عناصری که بعدا به صفحه اضافه خواهند شد.
ج) تزریق اسکریپت به صفحه در حین یک AsyncPostback رخ داده در آپدیت پنل

علت بروز مشکل:
علت رخدادن این مشکل (علاوه بر قسمت الف ذکر شده)، عدم فراخوانی document.ready تعریف شده، جهت بایند مجدد پلاگین jQuery مورد استفاده شما پس از هر AsyncPostback رخ داده در آپدیت پنل ASP.Net Ajax است. راه حل استاندارد جی‌کوئری هم همان مورد (ب) فوق می‌باشد، اما ممکن است جهت استفاده از آن نیاز به بازنویسی یک پلاگین موجود خاص وجود داشته باشد، که آنچنان مقرون به صرفه نیست.

مثالی جهت مشاهده‌ی این مشکل در عمل:
می‌خواهیم افزونه‌ی Colorize - jQuery Table را به یک گرید ویوو ASP.Net قرار گرفته درون یک آپدیت پنل اعمال کنیم.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="UpdatePanelTest.aspx.cs"
   Inherits="TestJQueryAjax.UpdatePanelTest" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body>
   <form id="form1" runat="server">
   <asp:ScriptManager ID="sm" runat="server">
       <Scripts>
           <asp:ScriptReference Path="~/js/jquery.js" />
           <asp:ScriptReference Path="~/js/jquery.colorize-1.6.0.js" />
       </Scripts>
   </asp:ScriptManager>
   <asp:UpdatePanel ID="uppnl" runat="server">
       <ContentTemplate>
           <asp:GridView ID="GridView1" runat="server" AllowPaging="True"
            OnPageIndexChanging="GridView1_PageIndexChanging">
           </asp:GridView>
       </ContentTemplate>
   </asp:UpdatePanel>
   </form>

   <script type="text/javascript">
       $(document).ready(function() {
           $('#<%=GridView1.ClientID %>').colorize();
       });
   </script>
</body>
</html>
```

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
```

```
using System.Web.UI.WebControls;

namespace TestJQueryAjax
{
   public partial class UpdatePanelTest : System.Web.UI.Page
   {
       void BindTo()
       {
           List<string> rows = new List<string>();
           for (int i = 0; i < 1000; i++)
               rows.Add(string.Format("row{0}", i));

           GridView1.DataSource = rows;
           GridView1.DataBind();
       }

       protected void Page_Load(object sender, EventArgs e)
       {
           if (!Page.IsPostBack)
           {
               BindTo();
           }
       }

       protected void GridView1_PageIndexChanging(object sender, GridViewPageEventArgs e)
       {
           GridView1.PageIndex = e.NewPageIndex;
           BindTo();
       }
   }
}
```

مثال بسیار ساده‌ای است جهت اعمال این افرونه به یک گریدویو و مشاهده کار کردن این افزونه در هنگام بارگذاری اولیه صفحه و سپس از کار افتادن آن پس از مشاهده صفحه دوم گرید. در این مثال از نکته " اسکریپت‌های خود را یکی کنید " استفاده شده است.

راه حل:

از ویژگی‌های ذاتی ASP.Net Ajax باید کمک گرفت برای مثال:

```
<script type="text/javascript">
       $(document).ready(function() {
           $('#<%=GridView1.ClientID %>').colorize();
       });

       function pageLoad(sender, args) {
           if (args.get_isPartialLoad()) {
               $('#<%=GridView1.ClientID %>').colorize();
           }
       }
   </script>
```

متد استاندارد pageLoad به صورت خودکار پس از هر AsyncPostback رخ داده در آپدیت پنل ASP.Net Ajax فراخوانی می‌شود (و همچنین پس از پایان پردازش و بارگذاری اولیه DOM صفحه). در این متد بررسی می‌کنیم که آیا یک partial postback رخ داده است؟ اگر بله، مجددا عملیات بایند افزونه به گرید را انجام می‌دهیم و مشکل برطرف خواهد شد.

برای مطالعه بیشتر

نظرات خوانندگان

نویسنده: نیما
تاریخ: ۱۳۸۸/۰۶/۲۱ ۰۱:۲۸:۴۲

سلام استاد عزیز
ممنون از مطلب بسیار مفیدتون.

عنوان: **غیرفعال کردن کش مرورگر هنگام استفاده از jQuery Ajax**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۷/۱۴ ۲۰:۲۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

برای استفاده از قابلیت‌های Ajax کتابخانه jQuery ، شش متد زیر در اختیار برنامه نویس‌ها است:

```
$.ajax(), load(), $.get(), $.getJSON(), $.getScript(), and $.post()
```

که در حقیقت 5 مورد آخر ذکر شده صرفا بیان اولین متد ajax فوق به نحوی دیگر می‌باشند و محصور کننده توانایی‌های آن هستند.

برای مثال کد زیر زمان جاری را از سرور دریافت کرده و نتیجه را در سه تکست باکس قرار داده شده در صفحه نمایش می‌دهد.
ابتدا وب سرویس ساده زیر را در نظر بگیرید که زمان شمسی جاری را بر می‌گرداند:

```
using System;
using System.Globalization;
using System.Web.Script.Services;
using System.Web.Services;

namespace TestJQueryAjax
{
 [WebService(Namespace = "http://tempuri.org/")]
 [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
 [System.ComponentModel.ToolboxItem(false)]
 [ScriptService]
 public class AjaxSrv : WebService
 {
     public class TimeInfo
     {
         public string Date { set; get; }
         public string Hr { set; get; }
         public string Min { set; get; }
     }

     [WebMethod(EnableSession = true)]
     [ScriptMethod(ResponseFormat = ResponseFormat.Json)]
     public TimeInfo GetTime()
     {
         DateTime now = new DateTime(DateTime.Now.Year, DateTime.Now.Month, DateTime.Now.Day);
         PersianCalendar pc = new PersianCalendar();
         string today = string.Format("{0}/{1}/{2}",
                                      pc.GetYear(now),
                                      pc.GetMonth(now).ToString("00"),
                                      pc.GetDayOfMonth(now).ToString("00"));
         TimeInfo ti = new TimeInfo
         {
             Date = today,
             Hr = DateTime.Now.Hour.ToString("00"),
             Min = DateTime.Now.Minute.ToString("00")
         };
         return ti;
     }
 }
}
```

سپس اگر از تابع Ajax کتابخانه jQuery جهت دریافت زمان جاری از وب سرویس استفاده نمائیم، کد صفحه ما به صورت زیر خواهد بود:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="TestFillCtrls.aspx.cs"
 Inherits="TestJQueryAjax.TestFillCtrls" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
 <title></title>

 <script src="js/jquery.js" type="text/javascript"></script>

 <script type="text/javascript">
     function validate() {
         $.ajax({
             type: "POST",
             url: 'AjaxSrv.asmx/GetTime',
             data: '{}',
             contentType: "application/json; charset=utf-8",
             dataType: "json",
             success:
                 function(msg) {
                     $("#<%=txtDate.ClientID %>").val(msg.d.Date);
                     $("#<%=txtHr.ClientID %>").val(msg.d.Hr);
                     $("#<%=txtMin.ClientID %>").val(msg.d.Min);
                 },
             error:
                 function(XMLHttpRequest, textStatus, errorThrown) {
                     alert("خطایی رخ داده است");
                 }
         });
         //debugger;
     }
 </script>

</head>
<body>
 <form id="form1" runat="server">
 <div>
     <asp:TextBox ID="txtDate" runat="server" />
     <br />
     <asp:TextBox ID="txtHr" runat="server" />
     <br />
     <asp:TextBox ID="txtMin" runat="server" />
     <br />
     <asp:Button ID="btnGetTime" runat="server" Text="Click here!" UseSubmitBehavior="false"
             OnClientClick="validate();return false;" />
 </div>
 </form>
</body>
</html>
```

تنها نکته‌ی جدید این اسکریپت، نحوه‌ی استفاده از خروجی JSON وب متد ما است که از نوع TimeInfo تعریف شده است.
خروجی نمونه این وب متد به صورت زیر می‌تواند باشد:

```
{"d":{"__type":"TestJQueryAjax.AjaxSrv+TimeInfo","Date":"1388/07/14","Hr":"12","Min":"59"}}
```

که نحوه‌ی دسترسی به اجزای آن‌را در متد validate ملاحظه می‌نمائید.

باید به خاطر داشت که برای هر 6 متد Ajax ایی jQuery ، عملیات کش شدن اطلاعات در مرورگر کاربر به صورت پیش فرض فعال است. اما این نکته تنها زمانیکه dataType مورد استفاده از نوع script یا jsonp باشد، صادق نبوده و کش شدن به صورت خودکار غیرفعال می‌گردد.
روش سنتی غیرفعال کردن کش در حین عملیات اجکسی، استفاده از یک کوئری استرینگ متغیر در پایان url درخواستی است. به این صورت مرورگر درخواست صادره را جدید فرض کرده و از کش خود استفاده نمی‌نماید (همین مورد در حالت کش شدن تصاویر هم صادق است).
jQuery نیز همین عملیات را در پشت صحنه انجام داده اما تنظیم آن‌را به نحوی مطلوب‌تری ارائه می‌دهد. یا پارامتر cache را در تعریف متد ajax خود اضافه نموده و مقدار آن را مساوی false قرار دهید و یا جهت تاثیر گذاری بر روی کلیه متدهای مورد استفاده، پیش از استفاده از آن‌ها این تنظیم را مشخص سازید:

```
$.ajaxSetup({cache: false});
```

نظرات خوانندگان

نویسنده: شاهین کیاست
تاریخ: ۱۳۸۹/۱۱/۱۱ ۲۱:۵۶:۴۰

سلام.
علت Return False که آخر این دستور گزاشتید چیست؟
;OnClientClick="validate();return false
ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۱۱ ۲۳:۲۰:۰۰

سلام، از UseSubmitBehavior هم استفاده شده. علت غیرفعال کردن postback است تا بشود نمایی از عملیات Ajax را مشاهده کرد.

عنوان: **jQuery Ajax و نحوه صحیح ارسال مقادیر به یک وب سرویس**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۱۰ ۱۹:۴۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

در مورد کار با jQuery Ajax و نحوه فراخوانی یک متد وب سرویس توسط آن، چند مطلب پیشتر ارائه شدند:
بررسی وجود نام کاربر با استفاده از jQuery Ajax در ASP.Net
و ...

تمام این مقالات یک ایراد مهم دارند که امروز با آن مواجه شدم و خلاصه آن به شرح زیر است:
پارامتر data متد Ajax جی‌کوئری را به صورت زیر در نظر بگیرید:

```
data: "{'username': '" + $('#<%= TextBox1.ClientID %>').val() + "'}",
```

این روش شاید با بسیاری از ورودی‌ها کار کند اما یک سری از کاراکترها زمانیکه با فرمت JSON ارسال می‌شوند باید اصطلاحا escape شوند که لیست آن‌ها به صورت زیر است:

```
\b  Backspace (ascii code 08)
\f  Form feed (ascii code 0C)
\n  New line
\r  Carriage return
\t  Tab
\v  Vertical tab
'  Apostrophe or single quote
"  Double quote
\  Backslash caracter
```

برای حل این مشکل می‌توان از اسکریپت استاندارد json2.js که در سایت json.org معرفی شده، استفاده کرد.

ابتدا ارجاعی از این اسکریپت باید به صفحه اضافه شود:

```
<script src="js/json2.js" type="text/javascript"></script>
```

سپس مقدار پارامتر data را باید به شکل زیر اصلاح کرد:

```
var jsonText = JSON.stringify({ username: $('#<%= TextBox1.ClientID %>').val() });
...
data: jsonText,
```

متد stringify تمام ملاحظات لازم را در مورد escape کاراکترهای یاد شده و بسیاری از موارد دیگر به صورت خودکار اعمال می‌کند. همچنین از این لحاظ که یک ساختار داده‌ای جاوا اسکریپتی را نیز می‌تواند به معادل متنی آن تبدیل کند، بسیار جالب توجه است.
به عنوان مثالی دیگر، اگر متد وب سرویس ما دو پارامتر داشت، jsonText به شکل زیر در خواهد آمد:

```
var jsonText = JSON.stringify({ param1: val1, param2: val2 });
```

لازم به ذکر است که پشتیبانی از این متد در نگارش‌های آخر اکثر مرورگرهای امروزی در جهت سازگاری با آخرین استانداردهای مصوب جاوا اسکریپت به صورت بومی وجود دارد (از IE8 به بعد و فایرفاکس 3 و یک به بعد)، اما از آنجائیکه کاربران ممکن است هنوز از نگارش‌های قدیمی آن‌ها استفاده کنند، کمک گرفتن از فایل json2.js ضروری به نظر می‌رسد.

نظرات خوانندگان

نویسنده: peyman naji
تاریخ: ۲۱:۳۰:۴۹ ۱۳۸۸/۰۸/۱۰

بسیار عالی بود . یه سوالی داشتم و اینکه برای کار با jquery ajax باید در visualstudio پروژه Enabale Ajax ایجاد نمود یا اینکه با پروژه معمولی هم کار میکنه چرا که هنگام پاس دادن اطلاعات بصورت json فایر باگ میگفت فرمتت صحیح نیست و وقتی که با پروژه Enable ajax کار کردم مشکل حل شد در این مورد یه توضیحی میدید . با سپاس فراوان/

نویسنده: DotNetCoders
تاریخ: ۲۱:۳۵:۴۸ ۱۳۸۸/۰۸/۱۰

سلام !

خسته نباشید.

فکر کنم 2 بار دیگه این مطلب رو گفته بودید.

/

(; happy new weblog name

نویسنده: وحید نصیری
تاریخ: ۲۱:۵۷:۳۳ ۱۳۸۸/۰۸/۱۰

@peyman naji
هنگامیکه پروژه Ajax enabled شود یک سری مدخل به فایل web.config در مورد تبادلات اجکسی اضافه می‌شود که حتما لازم است (در مورد فعال سازی scripting و webServices است). یک نمونه آن‌را در اینجا توضیح داده‌ام:
http://www.dotnettips.info/2009/07/blog-post_06.html

@DotNetCoders
- یکبار بود، البته نه به این صورت و واضحی.
- ممنون. اسم قبلی کمی زیاده روی بود :)

عنوان: **چه زمان‌هایی ممکن است Page_Load یک صفحه ASP.Net دوبار اجرا شود؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۹/۱۴ ۲۰:۳۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

سؤال مربوط به حالت نخ نمای Page.IsPostBack نیست. مربوط به حالتی است که دقیقا در اولین بار مشاهده‌ی عادی یک صفحه، Page_Load دوبار یا بیشتر (!) اجرا می‌شود.

الف) برنامه‌ی ASP.Net 1.x خود را به نگارش‌های 2+ ارتقاء داده‌اید.
در این حالت هر چند VS.Net پیغام تبدیل با موفقیت یک پروژه‌ی قدیمی را به شما ارائه خواهد داد اما یک سری موارد را پس از تبدیل، باید اصلاح کرد.
پروژه‌های قدیمی ASP.Net در روال InitializeComponent خود سطر زیر را همانند یک پروژه WinForm و امثال آن برای معرفی روال رخداد گردان Page_Load دارند.

```
this.Load += new System.EventHandler(this.Page_Load);
```

اما در پروژه‌های ASP.Net 2.0 به بعد دیگر از این روال خبری نیست. پس در این پروژه‌ها کامپایلر چگونه متوجه خواهد شد که Page_Load واقعا یک روال رخ داد گردان است و نه یک روال معمولی؟ در پروژه‌های جدید VS.Net ، خاصیت AutoEventWireup صفحه به true تنظیم شده و دیگر نیازی به معرفی صریح روال‌های رخ داد گردانی مانند Page_Load و یا Page_Init نبوده و تشخیص آن‌ها به صورت خودکار انجام می‌شود.

```
<% @Page AutoEventWireup="true" %>
```

بر این اساس با انتقال یک پروژه قدیمی به VS.Net جدید، هم AutoEventWireup=true را خواهیم داشت و هم سطرهای موجود در متد قدیمی InitializeComponent از پروژه حذف نشده‌اند. بنابراین متد Page_Load در دو نقطه، یکبار به صورت خودکار (متد Page_Load بر اساس نام آن با توجه به AutoEventWireup=true تشخیص داده شده و اجرا می‌شود) و یکبار هم بر اساس تعریف دستی موجود فراخوانی می‌شود.
برای حل این مشکل و سازگاری بهتر با نگارش‌های جدیدتر، سطر تعریف دستی روال رخ داد گردان متد Page_Load را حذف کنید.
لازم به ذکر است که این سیم کشی خودکار تنها برای متدهای زیر انجام خواهد شد و نسبت به حذف سایر موارد موجود اقدام نکنید!

```
Page_PreInit
Page_Init
Page_InitComplete
Page_PreLoad
Page_Load
Page_LoadComplete
Page_DataBind
Page_SaveStateComplete
Page_PreRender
Page_PreRenderComplete
Page_Unload
Page_Error
Page_AbortTransaction
Page_CommitTransaction
```

ب) وجود هر نوع دکمه‌ی تصویری یا کلا تصویری با ویژگی src مقدار دهی نشده در صفحه
مرورگر IE با این مساله مشکلی ندارد. اما فایرفاکس‌های جدید اگر به src مقدار دهی نشده‌ی تصویری برخورد کنند دقیقا آدرس

جاری صفحه را بجای مقدار src قرار داده و مجددا صفحه را درخواست می‌کنند (و البته این مورد ارتباط مستقیمی به ASP.Net یا PHP و امثال آن ندارد و یک مساله‌ی عمومی است). این مورد سبب خواهد شد که Page_Load صفحه، نه فقط دوبار بلکه به تعداد بار src خالی تصاویری که در صفحه وجود دارند، بر اساس درخواست‌های مجدد فایرفاکس از سرور اجرا شود. (مرورگر IE بجای فراخوانی آدرس صفحه جاری، یک null/ را به انتهای آدرس جاری اضافه کرده و آن‌را فراخوانی می‌کند. بنابراین سبب اجرای مجدد هیچ روالی نخواهد شد.)

مآخذ:

Inside AutoEventWireup

How Firefox Handles Empty SRC tags

نظرات خوانندگان

نویسنده: افشار محبی
تاریخ: ۱۳۸۸/۰۹/۱۸ ۱۰:۵۵:۴۵

در حالتی که از UpdatePanel استفاده می‌شود هم به نظر می‌رسد که Page_Load دو بار اجرا می‌شود. البته واقعیت این است که هر Partial Postback هم خود یک Postback مستقل محسوب شده و Page_Load را فراخوانی می‌کند در نتیجه خیلی‌ها (به اشتباه) فکر می‌کنند که Postback دو بار (یا بیشتر) اجرا می‌شود...

عنوان: **کش کردن اطلاعات غیر پویا در ASP.Net - قسمت چهارم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۱۰/۲۱ ۱۲:۳۹:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

قسمت‌های اول تا سوم این مقاله: + و + و +

در قسمت چهارم قصد داریم هدر مربوط به Content Expiration Date را توسط یک Http module به محتوای غیرپویای سایت مانند تصاویر ، فایل‌های CSS و غیره اعمال کنیم. این روش از روش قسمت دوم ساده‌تر است و جامع‌تر.
ابتدا یک پروژه‌ی Class library جدید را به نام StaticContentCacheModule ایجاد کرده و سپس ارجاعی را به اسمبلی استاندارد System.Web.dll به آن خواهیم افزود. سپس کدهای مرتبط با این ماژول به شرح زیر هستند:

```
//StaticCache .cs
using System;
using System.Web;

namespace StaticContentCacheModule
{
   public class StaticCache : IHttpModule
   {
       public void Init(HttpApplication context)
       {
           context.PreSendRequestHeaders += context_PreSendRequestHeaders;
       }

       static void context_PreSendRequestHeaders(object sender, EventArgs e)
       {
           //capture the current Response
           var currentResponse = ((HttpApplication)sender).Response;

           if (CacheManager.ShouldCache(currentResponse.ContentType))
           {
               currentResponse.AddHeader("cache-control", "public");
               currentResponse.AddHeader("Expires",
DateTime.Now.Add(TimeSpan.FromDays(30)).ToString());
           }
       }

       public void Dispose() { }
   }
}
```

در اینجا ContentType تک تک عناصری که توسط وب سرور ارائه خواهند شد، بررسی می‌شود. اگر نیازی به کش شدن آن‌ها وجود داشت (توسط کلاس CacheManager این امر مشخص می‌گردد)، هدر مربوطه اضافه می‌گردد.

```
//CacheManager.cs
using System;

namespace StaticContentCacheModule
{
   class CacheManager
   {
       public static bool ShouldCache(string contentType)
       {
           contentType = contentType.ToLower();
           string[] parts =
               contentType.Split(
                   new[] { ';' },
                   StringSplitOptions.RemoveEmptyEntries
               );

           if (parts.Length > 0)
               contentType = parts[0];

           bool cache = false;
```

```
            switch (contentType)
            {
                case "text/css":
                    cache = true; break;
                case "text/javascript":
                case "text/jscript":
                    cache = true; break;
                case "image/jpeg":
                    cache = true; break;
                case "image/gif":
                    cache = true; break;
                case "application/octet-stream":
                    cache = true; break;
                default:
                    {
                        if (contentType.Contains("javascript"))
                            cache = true;
                        if (contentType.Contains("css"))
                            cache = true;
                        if (contentType.Contains("image"))
                            cache = true;
                        if (contentType.Contains("application"))
                            cache = true;
                    }
                    break;
            }

            return cache;
        }
    }
}
```

در این کلاس، contentType دریافتی بررسی می‌شود. اگر نوع محتوای قابل ارائه از نوع JavaScript ، CSS ، تصویر و یا Application بود، یک مقدار true بازگشت داده خواهد شد.

نهایتا برای استفاده از این Http module جدید در IIS6 به قبل در وب کانفیگ برنامه خواهیم داشت:

```
<httpModules>
     <add name="StaticContentCacheModule" type="StaticContentCacheModule.StaticCache,
StaticContentCacheModule"/>
</httpModules>
```

و یا در IIS7 این تغییرات به صورت زیر می‌تواند باشد:

```
<system.webServer>
<modules>
  <add name="StaticContentCacheModule" type="StaticContentCacheModule.StaticCache,
StaticContentCacheModule"/>
</modules>
```

اکنون اگر یک پروژه‌ی آزمایشی جدید ASP.Net را گشوده و فایل css ساده‌ای را به آن اضافه کنیم، بررسی هدر نهایی توسط افزونه‌ی YSlow به صورت زیر خواهد بود:

URL	EXPIRES (Y/M/D)
http://localhost/TestStaticContentCacheModule/	no expires
http://localhost/TestStaticContentCacheModule/Stylesheet1.css	2010/2/10

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۰ ۰۹:۴۴:۱۴

نکته: اگر کلاس‌های فوق را به برنامه‌ی خود اضافه کرده‌اید و ماژول‌ها کار نمی‌کنند باید به این مطلب دقت داشته باشید: (+)

نویسنده: mohammad
تاریخ: ۱۳۸۹/۱۰/۲۰ ۱۹:۲۵:۱۰

سلام آقا بازم نمی شه من کد رو به vb تبدیل کردم ودر وب کانفیگ هم به صورت زیر گذاشتم :

```
add name="StaticContentCacheModule" type="StaticContentCacheModule.StaticCache, App_Code.7pqudgp-,
"Version=0.0.0.0, Culture=neutral, PublicKeyToken=null
```

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۰ ۲۲:۱۳:۳۳

سورس کامل مرتبط با این مطلب در همان زمان: (+)

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۰ ۲۳:۴۳:۲۳

اگر روش فوق برای شما کار نمی‌کند، روش ذکر شده در این مطلب: (+) حتما کار خواهد کرد و حداقل برای اسکریپت‌های سایت می‌توانید از آن استفاده کنید.

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۹/۱۰/۲۱ ۱۰:۴۰:۲۵

بنظرم،مشکل با عبارت This operation requires IIS integrated pipeline mode ارتباط داره که هم در IIS6 و IIS7 وجود داره

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۱ ۱۱:۳۲:۳۱

فقط IIS های 7 به بعد integrated pipeline mode را دارند. به همین جهت بجای currentResponse.Headers[something] = something از currentResponse.AddHeader استفاده شده.

نویسنده: مهدی پایروند
تاریخ: ۱۳۸۹/۱۰/۲۲ ۱۳:۳۷:۳۴

من مشکل رو در iis7 در ویندوز 7 میبینم و نکته دیگه اینکه در ii6 و ویندوز سرور 2003 هم کار نکرده و با اینکه این چند روز خیلی دنبال این مسئله تو اینترنت بودم حدود 70 مقاله که اکثرا شبیه به هم بودند رو بررسی کردم حتی iis7 رو دوباره نصب کردم ولی مشکل همچنان باقیست.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۲۲ ۱۳:۵۵:۳۰

اگر دسترسی به IIS دارید که نیازی به این مقاله نیست. باید به اینجا مراجعه کنید: (+)
در مورد IIS-7 هم اطلاعات بیشتر اینجا است: (+)

در IIS7 این مورد توسط انتخاب application pool مناسب تنظیم می‌شود. برای مثال Classic mode آن با IIS6 سازگار است.

عنوان: کدامیک از محصولات مهم تجاری 2010 مایکروسافت از ASP.Net MVC استفاده می‌کنند؟
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۲/۰۸ ۱۳:۳۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

پاسخ : هیچکدام!
برای نمونه دو مورد از محصولات مهم تجاری و پر درآمد مایکروسافت در مقیاس سازمانی SharePoint و Exchange server هستند (البته اینجا منظور برنامه web access مربوط به Exchange server است). جالب اینجا است که هر دو محصول، مبتنی بر دات نت فریم ورک سه و نیم بوده و از ASP.Net WebForms استفاده می‌کنند. تفاوت مهم آن‌ها با نگارش سال 2007 هر کدام، استفاده از ASP.Net Ajax مایکروسافت در این محصولات است و همچنین استفاده‌ی وسیع از توانمندی‌های پاورشل 2 خصوصا امکان مدیریت از راه دور پاور شل 2 که برای مثال در برنامه web access مربوط به exchange server 2010 ، امکان مدیریت خود exchange server را نیز فراهم آورده است یا در SharePoint 2010 جایگزین stsadm شده است (هر چند stsadm هنوز موجود است اما منسوخ شده در نظر گرفته می‌شود).
به علاوه هر دو محصول فقط با ویندوزهای سرور 2008 به بعد، آن هم نسخه‌ی 64 بیتی کار می‌کنند. (البته از آنجائیکه هسته‌ی ویندوز 7 با هسته‌ی ویندوز سرور 2008 نگارش R2 یکی است (یا حداقل بر مبنای یک code base هستند)، SharePoint 2010 را بر روی ویندوز 7 شصت و چهار بیتی هم می‌توان جهت آزمایش و توسعه نصب کرد)
یک دوره‌ی مدیریتی SharePoint 2010 را می‌توانید در آدرس زیر مشاهده نمائید:
Microsoft SharePoint 2010 Administration

جهت اثبات این مدعا (استفاده از WebForms و نه MVC) دو تصویر ذیل به اندازه‌ی کافی گویا هستند:

شیرپوینت 2010

Web Access در Exchange server 2010

EditAdminRoleGroup.aspx - Microsoft Visual Studio

File Edit View Project Debug Tools Window Help

Style Application: Target Rule:

EditAdminRoleGroup.aspx

Server Objects & Events (No Events)

```
<%@ Page Language="C#" AutoEventWireup="false" Inherits="Microsoft.Exchange.Management
    MasterPageFile="~/CommonMaster.Master" Title="<%$ Strings:EditRoleGroup %>"
    HelpId="EditRoleGroup" LocStringsResource="EditRoleGroup" EnableViewState="false"

<asp:Content ID="MainContent" ContentPlaceHolderID="ResultPanePlaceHolder" runat="serve
    <ecp:properties id="AdminRoleGroupProperties" runat="server" ServiceUrl="~/UsersGr
        <sections>
            <ecp:Section ID="GroupInformationSection" runat="server" Title="<%$ String
                <Content>
                    <div class="PropertyDiv">
                        <div>
                            <asp:Label runat="server" ID="displayName_label" Text="<%$
                            <asp:Label runat="server" ID="displayNameContent_label" Da
                        </div>
                    </div>
```

Design Split Source

Ready Ln 1 Col 1 Ch 1 INS

نظرات خوانندگان

نویسنده: ramin
تاریخ: ۱۳۸۹/۰۲/۰۸ ۱۴:۱۶:۲۴

ظاهرن microsoft نمی تونه از webform دست بکشه و asp.net mvc نقش فرزند ناخوانده رو براش داره!

نویسنده: Sadegh
تاریخ: ۱۳۸۹/۰۲/۰۹ ۲۰:۱۳:۵۶

ناخوانده یا ناخواسته؟

عنوان: **باگ امنیتی در ASP.NET و نحوه‌ی رفع آن**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۶/۲۷ ۱۷:۳۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

بعد از مدت‌ها بالاخره یک باگ امنیتی در ASP.NET کشف شده! البته این "بعد از مدت‌هایی" که عنوان شد بر اساس آمار است که در سایت بی‌طرف Secunia قابل بررسی و مشاهده است (و اگر از یک سری کامپوننت و برنامه‌های جانبی سایر برنامه نویس‌ها صرفنظر کنیم، انصافا ضریب امنیتی بالایی را ارائه داده):

http://secunia.com/advisories/search/?search=asp.net

البته این مورد جدید هم مرتبط با خود ASP.NET نیست؛ بلکه مرتبط است با نحوه‌ی پیاده سازی الگوریتم AES در دات نت فریم ورک (Padding Oracle / AES cookie encryption vulnerability).

توضیحات بیشتر در سایت Scottgu

Important: ASP.NET Security Vulnerability

خلاصه این توضیحات:

همانطور که هزاران بار به برنامه نویسان ASP.NET (و SharePoint) گوشزد شده است، لطفا صفحه‌ی نمایش خطاهای سفارشی سایت را (customErrors) در web.config برنامه on کنید. این خطای امنیتی موجود در پیاده سازی الگوریتم AES نیاز به سعی و خطاهای زیاد روی سایت و بررسی پاسخ‌های داده شده یا ریزخطاهای منتشر شده دارد. اگر صفحه‌ی خطای سفارشی سایت شما تنظیم شده و روشن است، خطری سایت شما را تهدید نمی‌کند؛ زیرا شخص مهاجم هیچ خروجی خطایی را جهت بررسی مشاهده نخواهد کرد.

عنوان: **پایان پروژه ASP.NET Ajax Control Toolkit !**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۶ ۱۷:۵۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

بله! همانطور که حدس زده می‌شد بالاخره مایکروسافت تکلیف خودش را با کتابخانه‌های Ajax ایی تولید شده در طی این چند سال مشخص کرد و از این پس انتخاب اصلی جهت تولید برنامه‌های ASP.NET مبتنی بر Ajax ، تنها jQuery است.
اصل مطلب رو می‌تونید اینجا مطالعه کنید:

Javascript Libraries and ASP.NET: A Guide to jQuery, AJAX and Microsoft

The Murky Future of ASP.NET AJAX

خلاصه آن:
ASP.NET AJAX در آینده نیز کاملا پشتیبانی می‌شود، اما شهروند درجه یک محسوب نخواهد شد؛ مانند استفاده از ScriptManager و UpdatePanel
Ajax Control Toolkit که اکنون به صورت سورس باز در سایت CodePlex نیز قابل دریافت است، منسوخ شده در نظر گرفته شده و تنها در آینده شاید هر از چندگاهی به رفع باگ‌های گزارش شده در آن پرداخته شود (اما دیگر به صورت فعال توسعه داده نخواهد شد).
Microsoft Ajax Library / ASP.NET Ajax Library هر چند تشابه اسمی با ASP.NET Ajax دارند اما جزئی از کتابخانه‌ی AJAX Control Toolkit بوده و این مورد نیز از این پس منسوخ شده در نظر گرفته خواهد شد (مانند استفاده از DataView یا Sys.require).

نظرات خوانندگان

نویسنده: farbod
تاریخ: ۱۳۸۹/۰۷/۲۶ ۲۲:۵۱:۴۴

سلام آقای نصیری. من، هم با این کتابخانه و updatepanel کار کردم و هم jquery . درسته که خروجی کار با jquery بسیار با کیفیت تر و انعطاف پذیری هم بیشتره،ولی حقیقتش هر موقع از این روش استفاده می کنم احساس خوبی ندارم! به نظرم کدها بسیار نا مرتب و به قول معروف اسپاگتی هستن. من تمام تلاشم رو میکنم که در داخل فایل جاوااسکریپت بسیار با قاعده و طبقه بندی شده بنویسم،ولی بازهم .. به خصوص وقتی لازم میشه html رو داخل جاوااسکریپت و به صورت رشته ای تولید کنم.
می خواستم بدونم واقعا به همین صورته یا من از روش های نادرست استفاده میکنم.

نویسنده: محمد امین شریفی
تاریخ: ۱۳۸۹/۰۷/۲۶ ۲۳:۱۱:۳۰

سلام.

درست مثل linq to sql، که اولش تبلیغ می کرد از ADO.NET بیاین سمت Linq to SQL و بعدش گفت از Linq to SQL بیاین به سمت ADO.net Entity
MFC هم همینطور محجور مانده،ASP کلاسیک

بدی محصولات ماکروسافت این هست که opensource نیست که توسط third party همیشه پشتیبانی و به مرور بهنگام بشه و خوبیش هم این هست که یک فناوری خوش دست و راحتی میده.

مهندس نصیری، نظرت درباره جاوا چی هست؟، پستی را برای مقایسه این دو می نویسید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۷ ۰۰:۱۰:۲۳

@farbod
برای کلا جاوا اسکریپت هم کتابخانه‌های Unit test وجود دارد تا از این دلچرکین بودن خارج شوید:
QUnit

@محمد امین شریفی
اینبار انتخاب مایکروسافت درست بوده:
1- هزینه توسعه رو کاهش دادند. برای مثال هدف از Ajax Control Toolkit چیزی بوده شبیه به پلاگین‌های jQUery اما با هزینه بیشتر. اینبار شما هزاران پلاگین جی کوئری در اختیارتون هست و نه صرفا مجموعه‌ی Ajax Control Toolkit .
2- موردی رو که در مقاله بالا بهش اشاره شده و دلیل اصلی کنار گذاشتن ASP.NET Ajax بوده پایین بودن کارآیی ASP.NET Ajax است. هم حجم بالایی دارد و هم تبادلات شبکه‌ای حجیمی را به همراه دارد که در هر دو مورد جی کوئری بهینه‌تر است.
و یک سری موارد دیگر که قبلا بهش اشاره کرده بودم: (+)
3- انتخاب اصلی کتابخانه Ajax در ASP.NET MVC الان همین جی‌کوئری است و در مقاله فوق به یک سری افزونه و کتابخانه کمکی برای آن اشاره شده (که در حال توسعه و تکمیل است).

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۷ ۰۰:۲۸:۳۵

ضمنا یک مورد رو در باره‌ی LINQ to SQL و ASP کلاسیک اضافه کنم. جایگزین شدن entity framework بجای L2S یا ASP.NET به جای ASP کلاسیک، یک روند سالم و سلامت توسعه است. LINQ to SQL فقط محدود است به SQL Server اما الان اکثر بانک‌های اطلاعاتی موجود پروایدر EF دارند و مدل توسعه‌ی آن بسته نیست. ASP کلاسیک رو نمی‌دونم باهاش کار کرده بودید یا نه؟ رسما

یک فاجعه بود! مخلوطی از کدهای برنامه داخل کدهای HTML و وابستگی آن به اشیاء COM و غیره (اگر می‌خواستید مثلا رمزنگاری را به آن اضافه کنید باید Active-X می‌نوشتید و در سرور رجیستر می‌کردید!). این مورد اصلا قابل قیاس نیست با ASP.NET و امکانات دات نت فریم ورک.

نویسنده: shahin
تاریخ: ۱۳۸۹/۰۷/۲۷ ۱۷:۵۹:۳۷

سلام جناب نصیری یکی از مشکلاتی که برنامه نویسان تازه کار را اذیت می کند اجرای Jquery از سمت سرور هست.
برای حل این ضعف چه توصیه ای دارید
مثلا Ajax پارامتر ها را سمت سرور ست می کردیم و استفاده می کردیم.
اما Jquery باید سمت HTML همه کار را انجام داد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۷ ۱۸:۳۴:۵۱

من در مورد jQuery در این سایت مطلب زیاد منتشر کردم. به عمد هم تمامشون با دید ASP.NET بوده.
به تگ‌های JavaScript و یا ASP.NET در کنار صفحه مراجعه کنید یا کلا فایل CHM خلاصه سایت رو جهت سهولت مطالعه دریافت کنید.
و بله. مایکروسافت اومده تمام این‌ها رو برای شما در ASP.NET Ajax کپسوله و از دید شما مخفی کرده. همین کپسوله سازی در پلاگین‌های jQuery هم با کیفیت بالا موجود است مثلا : (+)

نویسنده: Mehran Zand
تاریخ: ۱۳۸۹/۰۷/۲۷ ۲۱:۴۲:۱۵

با سلام،
تمام این موارد که مبفرمایید درست، من خودم از طرفداران استقاده از jQuery هستم اما یک مسئله برای من هنوز وجود دارد اون هم RAD است. در یکی از کتابخانه های آژاکسی برای asp.net مثل Anthem.Net
این مسئله خیلی خوب رعایت شده و کار توسعه نرم افزار مبتنی بر استفاده از آژاکس به سرعت انجام میشه. این برای من هنوز سواله با jQuery چطور می توان به این سرعت نرم افزار های Full-Ajax را توسعه داد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۷/۲۷ ۲۲:۴۷:۵۶

مایکروسافت ASP.NET Ajax را با اون عظمت و زحمتی که براش کشیده بودند، در مقابل jQuery بازنده اعلام کرده الان شما صحبت از Anthem.Net می‌کنید که فقط منحصر است به ASP.NET web forms آن هم نگارش‌های قبل از سه آن + پشتیبانی از مرورگرهای مختلف هم در آن ضعیف است.
هدف مایکروسافت از اینکار مدیریت هر دو پروژه MVC و Web forms است آن هم با هزینه کم، کیفیت بالا و سازگار با تمام مرورگرها.
ضمنا این رو در نظر باشید که یکی از توانمندی‌های jQuery، Ajax است (از کار با DOM گرفته تا دستکاری CSS نمایش داده شده، تا Animation ، مدیریت ساده‌تر رخدادها، اعمال قالب به سایت و غیره) و این مورد مزیت مهمی است نسبت به تمام کتابخانه‌هایی که فقط برای یک کار و آن هم سهولت تولید برنامه‌های مبتنی بر Ajax ایجاد شده‌اند و از چند مشکل مهم رنج می‌برند:
- تک کاره‌اند. فقط Ajax .
- مشکل سازگاری با مرورگرهای مختلف را دارند.
- به صورت فعال توسعه داده نمی‌شوند؛ رفع باگ نمی‌شوند و غیره. برای مثال فواصل به روز رسانی همان Anthem.Net را بررسی کنید.
- توسعه پذیر نیستند. برای مثال آیا می‌توان برای Anthem.Net افزونه نوشت؟
- حجم بالایی دارند.
- سرعت پایینی دارند.

jQuery در مورد تمام موارد عنوان شده حرف برای گفتن دارد از حجم کم تا سرعت بالاتر نسبت به اکثر کتابخانه‌های جاوا اسکریپتی دیگر تا توسعه‌ی منظم، سازگاری عالی با مرورگرها، توسعه پذیری و صدها و هزاران افزونه‌ی مهیا برای آن و غیره.

و RAD هزینه بر است. یعنی چی؟ یعنی حجم بالای کدهای اسکریپتی که باید به برنامه‌ی شما مثلا توسط ASP.NET Ajax تزریق شود که مبادا شما بخواهید دست خودتان را به نوشتن چند سطر کد جاوا اسکریپتی آلوده کنید. همچنین حجم تبادل اطلاعات ASP.NET Ajax را هم که مبتنی است بر RAD را هم با حجم اطلاعات مبادله شده توسط jQuery مقایسه کنید (با پلاگین فایرباگ مربوط به فایرفاکس). این حجم واقعا زیاد است و قابل مقایسه نیست. (تمام این‌ها هزینه‌های RAD است)
ضمنا وجود 100 ها افزونه و پلاگین نوشته شده برای jQuery کار شما را بسیار بسیار ساده می‌کنند، مانند پاسخ قبلی من در این مطلب.
فقط باید کمی وقت بگذارید و چیزی را بیاموزید که واقعا ارزش دارد. گیرم فردا نخواستید با ASP.NET کار کنید. تمام این اطلاعات در PHP هم به درد شما می‌خورد، چون قسمت سمت کلاینت آنچنان تفاوتی نمی‌کند و jQuery یک کتابخانه‌ی سمت کلاینت است.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۲/۱۹ ۰۱:۳۷:۵۳

در ادامه ... (+)

عنوان: **انجمن سایت LLBLGEN سورس باز شد**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۰ ۰۸:۴۰:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

LLBLGEN

یکی از ORM های تجاری بسیار با کیفیت دات نت است و علاوه بر اینکه هویت مستقلی دارد، امکان تولید کدهای مخصوص Entity framework و NHibernate را هم دارا است.

اخیرا این شرکت تصمیم گرفته است که سیستم پشتیبانی مشتریان خودش را که نمونه‌ای از آن‌را در اینجا می‌توانید ملاحظه کنید، سورس باز کند.

حداقل پیشنیازهای نصب این انجمن به شرح زیر است:

```
- ASP.NET 2.0+
- SQL Server 2000 or higher / CE Desktop 3.5
- NET 3.5+
```

لطفا جهت دریافت آن به این آدرس و جهت ملاحظه‌ی قابلیت‌های آن به این آدرس مراجعه نمائید.

نظرات خوانندگان

نویسنده: سامان نام نیک
تاریخ: ۱۳۸۹/۰۹/۱۱ ۱۲:۰۲:۲۶

با سلام
آقای نصیری به نظر شما آیا LLBLGEN مناسب تر و بهتر از orm های دیگر مثل Entity framework و NHibernate هستش
ممنون می شم یک مقایسه کلی کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۱ ۱۴:۱۷:۰۶

سلام،
تقریبا بله!
- طول عمر آن از EF بیشتر است.
- رابط گرافیکی تولید نگاشت‌های آن از EF و نمونه‌ی موجود در VS.NET بسیار بهتر و پایدارتر است.
- پشتیبانی بهتری از انواع بانک‌های اطلاعاتی دارد.
- code generator دارد.
- پشتیبانی کاملی از LINQ دارد.
و ...

ولی باز هم من ترجیح می‌دم سراغ راه‌حل‌های تجاری غیرمایکروسافتی نروم. حداقل می‌دونم مایکروسافت تا 10 سال دیگر هست. چندتا کتاب در مورد هر کدام از فریم ورک‌هاش موجود است، انواع و اقسام بلاگ‌ها، فوروم‌ها و غیره برای آن هست که قابل چشم پوشی نیست.

برای مطالعه بیشتر :
(+) , (+)

عنوان: **متغیرهای استاتیک و برنامه‌های ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۸ ۱۹:۳۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

هر متغیر استاتیک تنها دارای یک مقدار، در یک AppDomain مشخص است (مگر اینکه با ویژگی ThreadStatic مزین شود). هر برنامه‌ی ASP.NET هم AppDomain جداگانه و منحصر به خود را دارا است. بنابراین تعریف یک متغیر استاتیک در یک برنامه‌ی ASP.NET به معنای به اشتراک گذاری آن در بین تمامی درخواست‌های رسیده به سرور است. بنابراین عموما استفاده از متغیرهای استاتیک در برنامه‌های چند کاربره ASP.NET یک اشتباه بزرگ است و در صورت استفاده از آن باید منتظر تخریب اطلاعات یا دریافت نتایج غیرمنتظره‌ای باشید (مگر اینکه واقعا می‌دانید دارید چکار می‌کنید، برای مثال کش کردن نگاشت‌های NHibernate به این صورت و استفاده از الگوی singleton یا روش‌های مشابه که باید بین تمام کاربران به یک صورت و یک شکل به اشتراک گذاشه شود و در حین اجرای برنامه تغییری در آن حاصل نمی‌شود). برای مثال اگر کاربر یک، در صفحه‌ی یک، متغیر استاتیکی را مقدار دهی کند، کاربر 2 نیز با مقدار به روز شده‌ی کاربر یک کار خواهد کرد که به طور قطع این مورد مد نظر شما نیست (چون به احتمال زیاد طراحی شما بر اساس کار کاربر در یک Session است و نه یک مقدار برای تمام سشن‌های موجود در سایت) و همچنین باید دقت داشت که امنیت سیستم نیز در این حالت زیر سؤال است (زیرا در این حالت تمامی کاربران، صرفنظر از سطوح دسترسی تعریف شده برای آن‌ها، دسترسی به اطلاعاتی خواهند داشت که نباید داشته باشند).
نکته‌ی دیگری را هم که باید در مورد ASP.NET به خاطر داشت این است که ویژگی ThreadStatic نیز در اینجا کمکی نمی‌کند؛ زیرا مطابق طراحی آن از تردها استفاده‌ی مجدد می‌گردد.به عبارت دیگر در ASP.NET الزامی ندارد که آغاز یک درخواست جدید حتما به همراه ایجاد یک ترد جدید باشد.
طول عمر این نوع متغیرها هم تا زمانی است که وب سرور یا برنامه ری استارت شوند. فقط در این حالت است که نمونه‌ی موجود تخریب شده و سپس با اجرای مجدد برنامه، بازسازی خواهند شد.
بنابراین متغیرهای استاتیک در ASP.NET همانند شیء Application عمل می‌کنند و از آن سریع‌تر هستند زیرا زمانیکه به آن‌ها ارجاع می‌شود نیازی به جستجو در یک جدول و یافتن آن‌ها نیست (برخلاف شیء Application) و همچنین در اینجا نیازی هم به عملیات تبدیل نوع داده‌ای وجود ندارد (برخلاف نوع شیء Application که به صورت Object تعریف شده است). وجود اشیاء Application در ASP.NET فقط به جهت حفظ سازگاری آن با ASP کلاسیک است و توصیه شده است در ASP.NET به دلایلی که ذکر شد، اگر و تنها اگر نیاز به اشیایی در سطح برنامه داشتید از متغیرهای استاتیک استفاده کنید. شیء Cache نیز در ASP.NET همین کاربرد را دارد با این تفاوت که می‌توان برای آن مدت زمان منقضی شدن تعریف کرد یا اینکه وب سرور بسته به حق تقدم و اهمیتی که برای آن تعریف شده است، مجاز به حذف کردن آن در زمانی است که با کمبود منابع مواجه می‌شود. همچنین باید دقت داشت که تنها مکان ذخیره سازی متغیرهای استاتیک حافظه است اما امکان ذخیره سازی کش بر روی فایل سیستم تا بانک اطلاعاتی و غیره نیز مهیا است.

سؤال: آیا تعریف SqlConnection به صورت استاتیک جزو مواردی است که "مگر واقعا می‌دانید دارید چکار می‌کنید؟" ؟
پاسخ: خیر. در اینجا هم واقعا این شخص نمی‌داند که دارد چکار می‌کند! یعنی در مورد سازوکار درونی ADO.NET اطلاعاتی ندارد. باز کردن یک کانکشن در ADO.NET به معنای مراجعه به استخر (pool) کانکشن‌ها و بازکردن یکی از آن‌ها و در مقابل، بستن یک کانکشن هم به معنای علامتگذاری یک کانکشن به صورت غیرفعال است و آماده سازی آن برای استفاده در درخواست بعدی. به معنای دیگر این عملیات سربار آنچنانی ندارد که بخواهید آن‌را استاتیک تعریف کنید.
همچنین مورد دیگری را هم که این برنامه نویس نمی‌داند این است که متغیرهای استاتیک thread safe نیستند. به عبارتی حین استفاده از آن‌ها در یک برنامه‌ی چندکاربره‌ی ASP.NET حتما باید مکانیزم‌های قفل‌گذاری بر روی این نوع متغیرها و اشیاء اعمال شود (که این هم خود یک سربار اضافی است در مقیاس چند 10 یا چند 100 کاربر همزمان). این مشکلات همزمانی به چه معنا است؟
فرض کنید کاربر یک، شیء استاتیک SqlConnection ایی را باز کرده است و با آن مشغول کوئری گرفتن است. کاربر 2 نیز همزمان شروع به استفاده از این کانکشن باز در حال استفاده می‌کند (SqlConnection استاتیک یعنی استفاده‌ی تمام کاربران فقط و فقط از یک کانکشن باز شده)، نتیجه این خواهد بود که برای مثال پیغام خطایی را دریافت می‌کند مانند: فیلد مورد نظر در جدول موجود نیست! چرا؟ چون روی شیء استاتیک SqlConnection تعریف شده قفل گذاری صورت نگرفته است و در حین استفاده از آن هر کاربری در سایت نیز همان را استفاده خواهد کرد یا از آن بدتر ممکن است یک کاربر زودتر از کاربر دیگری آن‌را ببندد!

کاربر سوم در وسط کار با پیغام غیرمعتبر بودن کانکشن مواجه می‌شود، یا اینکه به صورت پیش فرض یک datareader را بیشتر نمی‌توان بر روی یک کانکشن باز شده اعمال کرد. کاربر 4 مشغول خواندن اطلاعات است، کاربر 5 ، پیغام غیرمعتبر بودن کوئری را دریافت می‌کند.

نظرات خوانندگان

نویسنده: Farhad
تاریخ: ۱۳۸۹/۰۹/۱۸ ۲۱:۰۶:۱۷

مطلب بسیار جالبی بود.

نویسنده: Mehran Zand
تاریخ: ۱۳۸۹/۰۹/۱۸ ۲۲:۱۸:۵۷

در لایه بیزینس تعریف مند ها استاتیک مشکلی ایجاد میکند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۸ ۲۳:۲۹:۰۰

لطفا جهت تکمیل بحث به این مطلب مراجعه کنید: (+)

نویسنده: A
تاریخ: ۱۳۸۹/۰۹/۱۹ ۰۰:۵۲:۵۴

خیلی ممنون.

من برای معماری دسترسی به داده‌ام از روشی استفاده کرده بودم که لازم داشتم متن داده‌ام برای هر ریسمان یکتا باشد.
این کار را با استفاده از متدهای
Thread.AllocateDataSlot
Thread.GetData
Thread.SetData
انجام داده بودم.

اکنون استفاده از یک متغییر Static دارای ویژگی ThreadStatic را تست کردم و به خوبی جواب گرفتم.

فقط یک نکته کوچک وجود دارد. من یک متغییر bool نیز دارم که چون ValueType است در اولین بار که مورد دسترسی قرار می‌گیرد مقدار Defaultاش که همان false است را خواهد داشت. درحالی که باید از دسترسی برای اولین بار به این متغییر آگاهی پیدا کنم. البته یک روش ساده که فعلا به ذهنم رسید استفاده از bool? است.

باز هم ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۹ ۰۱:۳۳:۰۲

ببینید محور اصلی بحث این تاپیک ASP.NET است و اگر از ویژگی ThreadStatic استفاده کرده‌اید، اشتباه است چون بحث استفاده مجدد از یک ترد موجود در ThreadPool در اینجا مطرح است (در بالا ذکر کردم).
در ASP.NET اگر می‌خواهید اطلاعاتی را صرفا برای استفاده در طول عمر یک درخواست ذخیره کرده و به اشتراک بگذارید از HttpContext.Current.Items استفاده کنید که برای این منظور طراحی شده و استاندارد است : (+)

نویسنده: A
تاریخ: ۱۳۸۹/۰۹/۱۹ ۰۲:۰۷:۵۲

بله کاملا متوجه این موضوع بودم. در اینجا محیطی که برای آن طراحی می‌کنم نیز ASP.NET است. اما من حتماً اسرار دارم که به

ازاء هر Thread یک شی DataContext در حافظه داشته باشم و نه به ازاء هر درخواست. حال اگر ASP.NET از Thread استفاده مجدد می‌کند چه بهتر، از Dispose و New اضافی خودداری می‌شود. و البته علاوه بر این در نهایت باعث خواهد شد که طراحی این لایه از لایه Presentation بیشتر جدا شود و حتی برای Win نیز کاربرد داشته باشد.

البته! در ASP.NET اگر امکان داشت که دو Client همزمان از یک Thread دو خروجی متفاوت بگیرند با مشکل مواجه می‌شدم؛ که فکر می‌کنم چنین چیزی ممکن نیست.

در کل تنها مشکلی که داشتم این بود که وقتی فکر این اگوی طراحی به ذهنم خطور کرد از ThreadStaticAttribute اطلاع نداشتم و با آن متدها که گفتم کار کردم. که توسط شما از این موضوع مطلع شدم.

باز هم ممنون برای نکات و ارجاء.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۹ ۰۸:۴۷:۱۱

در ASP.NET این استفاده‌ی مجدد از یک Thread منحصر به یک سشن نیست. همچنین از یک ترد مشخص الزاما برای درخواست بعدی استفاده نمی‌شود.
به علاوه DataContext نیز thread safe نیست و مباحثی را که در بالا ذکر شد در نظر داشته باشید. ممکن است وسط کار توسط یک کاربر دیگر استفاده شود.
+ این کار کردن با یک مرورگر و load یک کاربر ... روش صحیحی برای آزمودن نیست.
اگر خیلی علاقمند به انجام اینکار هستید باید از روش یک DataContext به ازای هر درخواست (per request) استفاده کنید، یعنی همان روش استفاده از Current.Items ذکر شده. اصطلاحا به این الگو، الگوی UnitOfWork گفته می‌شود. یک پیاده سازی خوب در این مورد اینجا هست: (+) و (+)

نویسنده: A
تاریخ: ۱۳۸۹/۰۹/۱۹ ۱۱:۲۹:۱۹

آیا این امکان وجود دارد که یک متغییر که دارای ویژگی ThreadStatic است توسط دو Thread همزمان مورد استفاده قرار بگیرد؟ جواب Microsoft خیر است. پس اینچنین متغییری نیاز به ThreadSafe بودن و حتی lock (فارق از محیط اجرا) ندارد. مگر اینکه راه دیگری برای اجرای چند کد به صورت موازی در یک AppDomain علاوه بر Thread وجود داشته باشد. به عبارت دیگر من Thread را Atomic ترین شی موازی‌سازی می‌دانم. اشتباه می‌کنم؟

البته اینکه یک Thread لزوماً برای درخواستی بعدی استفاده نشود نیز مشکلی ایجاد نمی‌کند.

لینک‌ها را مطالعه کردم. ولی بازهم فکر می‌کنم مشکلی در استفاده از این روش نیست! با توجه به جمله‌ای که در بالا گفتم نمی‌دانم چه مشکلی می‌تواند از نظر Concurrency اتفاق بیفتد؟

در مورد تست، من هم با شما موافقم و تست کردن فقط با یک کاربر در یک مرورگر را مطمئناً تست کاملی نمی‌دانم. درواقع فقط با بررسی نقاط بحرانی و حالت‌های بحرانی در ذهنم موارد Concurrent را چک کرده‌ام. که البته حتماً سعی می‌کنم با استفاده از یک نرم‌افزار خوب یک محیط چند کاربره را شبیه‌سازی کنم.

راستی یک نکته خیلی مهم. در این روش هر شیی که DataContext را برای اولین بار به وجود آورده باید بعد از اتمام کارش خودش آنرا Dispose کند. تمامی عملیات کار با داده در یک بلاک using صورت می‌گیرد اما شیی که در using استفاده شده مانند یک شی Repository است که مدیریت DataContext را برعهده دارد اما در داخل تفاوت‌هایی برای بالابردن کارایی دارد.

البته اسرار من بر این روش به جای استفاده از HttpContext این است که: لایه دسترسی به داده به صورت BlackBox کاملا از لایه Presentaion جدا باشد. درواقع بدون پیاده‌سازی یک Interface یا چیزی شبیه آن بتوان از یک لایه داده «همزمان» توسط

چند لایه دیگر (در یک پروژه) استفاده شود و البته برنامه‌نویس به دردسر نیفتد.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۹/۱۹ ۱۱:۵۲:۴۱

- Atomic Operation داریم ولی شیء Atomic ؟
- بحث استفاده مجدد از یک ترد در ASP.NET به این معنا است:
ترد یک در سشن کاربر یک، یک DataContext استاتیک را ایجاد کرده. حتی آن‌را با ویژگی ThreadStatic هم مزین کرده است. اکنون به نظر در پایان درخواست کارش به پایان رسیده است. در این لحظه ASP.NET این ترد یک را در اختیار سشن کاربر 2 قرار می‌دهد. این DataContext استاتیک شما که با توجه به ویژگی ThreadStatic بودن آن در این ترد زنده است و جهت Tracking بسیاری از موجودیت‌ها از آن استفاده شده، اطلاعات خود را در اختیار کاربر 2 قرار داده است. این DataContext نه لزوما می‌تواند معتبر باشد (شاید dispose شده) و یا شاید حاوی اطلاعات حساس و غیرضروری. هر دو مورد در یک برنامه چند کاربره مشکل ساز است.
- زمانیکه از ORM استفاده می‌کنید، لایه دسترسی به داده همان ORM است و از دید لایه‌های دیگر مخفی است. شما یک لایه دیگر به نام BLL برای جداسازی اعمال انجام شده توسط آن از لایه نمایش بایدایجاد کنید.

پ.ن.
- اسرار با اصرار کمی متفاوت است.

نویسنده: A
تاریخ: ۱۳۸۹/۰۹/۱۹ ۱۳:۴۶:۵۷

منظورم از Atomic این است که بیشتر از این نمی‌توانیم یک موجودیت را به اجزاء ریزتر تقسیم کنیم. مثل یک نقش(Role)اتمیک.

صحبت شما درست است ولی در این مورد این مشکلاتی که گفتید ایجاد نمی‌شود. عرض کردم که قبل از اینکه Thread بخواهد به کاربر 2 سپرده شود DataContext نابود شده است. معماری من کمی متفاوت است. فکر میکنم اگر بخواهیم بحث کنیم باید بر سر معماری بحث کنیم.
پاسخ اینکه آیا این معماری درست کار میکند یا نه در این جواب است که آیا امکان دارد دو Thread همزمان به یک شی که دارای ویژگی ThreadStatic است دسترسی پیدا کنند؟ اگر جواب خیر است پس همه چیز مرتب است (التبه کمی توضیح دارد).

و همچنین بله صحبت شما در مورد BLL درست است اما گاهی برای پروژه‌های کوچک بهتر است لایه‌ها با یک شبه BLL که داخل DAL قرار دارد صحبت کنند. البته فکر می‌کنم نمی‌توانم به درستی بیان کنم.

اما در کل شما به یک نکته خیلی خوب اشاره کردید و آن هم امنیت است. چک نکردم اما فکر می‌کنم در صورتی که «برنامه‌نویس خراب کاری کند» ممکن است این سیستم امنیت خوبی نداشته باشد. سعی میکنم برای این موضوع فکری کنم. اگر این مورد جدی باشد و اشتباهات برنامه‌نویس بتواند موجب مشکل شود حتماً از این روش صرف نظر کرده و از همان HttpContext.Current استفاده خواهم کرد.

دیکته من هرگز خوب نبوده و نیست. منظورم از اسرار، رازها نبوده منظورم پافشاری بوده (اصرار) :)

ممنون.

نویسنده: Nima
تاریخ: ۱۳۸۹/۰۹/۱۹ ۱۸:۱۴:۴۲

بسیار عالی، استفاده کردم و دقیقا همچین اشتباهی رو داشتم. میرم که بیشتر تحقیق کنم
ممنون

نویسنده: میثم نوایی
تاریخ: ۱۳۸۹/۱۰/۱۴ ۱۷:۱۵:۴۱

من یکامپوننت چت آنلاین نوشتم وچون کاربران هر چهارثانیه یکبار لیست پیغام های دریافتی خود را چک میکند برای اینکه سرعت برنامه بالا باشه مجبور شدم کلیه پیغام هارو در یک لیست استاتیک قرار بدم.آیا کسی پیشنهاد بهتری نسبت به لیست استاتیک برای اینکار سراغ داره؟
بازم سپاسگزارم.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۴ ۱۷:۳۱:۴۴

این مطلب جاری به معنای نفی استفاده از متغیرهای استاتیک نبود. اگر بد است چرا اصلا در زبان قرار داده شده؟ بنابراین با دید صرفا منفی به این قضیه نگاه نکنید.
در کار شما آیا این لیست برای تمام کاربران یکسان است؟ آیا سطح دسترسی در کار نیست؟ آیا همه موارد مشابهی را مشاهده می‌کنند؟ اگر بله مشکلی ندارد، فقط در نظر داشته باشید که متغیرهای استاتیک thread safe نیستند. برای این موارد کلاس Cache قرار گرفته در فضای نام System.Web.Caching ، مطابق مستندات آن Thread safe است و read/write آن در یک محیط چند کاربره مشکل زا نیست: (+)

ضمنا در مورد طراحی سیستم چت خوب در ASP.NET در مورد COMET تحقیق کنید. این روش سربار کمی دارد چون سرور به کلاینت پیغام ارسال می‌کند نه اینکه کلاینت متناوبا سرور را چک کند که آیا پیغام جدیدی هست یا نه: (+)

نویسنده: میثم نوایی
تاریخ: ۱۳۸۹/۱۰/۲۶ ۰۹:۵۸:۲۵

سپاسگزارم.راهنمایی های شما مخصوصا معرفی این کامپوننت آخری خیلی بدرد بخور بود.احتمالا بهترین پیشنهاد برای پیاده سازی سیستم چت استفاده از همین کامپوننت می باشد.

عنوان: **نمایش حجم ViewState برنامه‌های ASP.NET WebForms**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۹ ۱۰:۰۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

امکان اندازه گیری دقیق حجم ViewState در برنامه‌های ASP.NET WebForms وجود دارد (+) ، اما خوب، این روش یک ایراد مهم هم دارد. چند نفر حاضرند تمام صفحات خود را ویرایش کرده و ارث بری ذکر شده را پیاده سازی کنند؟
یک روش دیگر اعمال آن به تمام صفحات، استفاده از پوشه‌ی استاندارد App_Browsers و سپس ایجاد فایلی مانند ViewStateManager.browser می‌باشد:

```
<!--Applies to all pages-->
<browsers>
<browser refID="Default">
  <controlAdapters>
    <adapter controlType="System.Web.UI.Page" adapterType="ViewStateManagerCore.SomeClass" />
  </controlAdapters>
</browser>
</browsers>
```

به این صورت ارث بری مورد نظر به صورت خودکار بر روی تمام صفحات اعمال خواهد شد.
علاوه بر این دو روش (ارث بری دستی و ارث بری خودکار)، افزونه‌ای هم برای فایرفاکس جهت نمایش حجم ViewState صفحات طراحی شده است که از آدرس زیر قابل دریافت می‌باشد :

Viewstate Size

این موضوع چه اهمیتی دارد؟
ممکن است کاربران سایت شما گاهی از اوقات در بعضی از صفحات با خطای "Validation of viewstate MAC failed" مواجه و متوقف شوند. عموما حجم بالای ViewState این مشکل را درست می‌کند. حجم ViewState بالا است (چند صد کیلوبایت ...)، صفحه دیر رندر می‌شود یا هنوز کامل نشده، شخص صفحه را متوقف می‌کند. ASP.NET در این حالت اجازه‌ی ارسال اطلاعات از این صفحه‌ی ناقص را به دلایل امنیتی نمی‌دهد که خوب است (شاید جعل شده باشد؟).

پ.ن.
راه حل پیشنهادی مایکروسافت جهت حل این مشکل (شروع شده از سال 2007)، ارتقاء برنامه‌های شما به ASP.NET MVC می‌باشد :)

نظرات خوانندگان

نویسنده: shahin
تاریخ: ۱۳۸۹/۱۰/۱۹ ۱۲:۵۲:۰۷

سلام.
جناب نصیری حجم Viewstate تا چقدر باشد ایرادی ندارد؟
مثلا حجم 7 کیلو بایت زیاد هست ؟
شنیدم راههایی جهت فشرده سازی آن است.
آقای نصیری آیا ما برنامه نویسان Web Form اگر به ASP.NET MVC مهاجرت نکنیم ضرر می کنیم؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۰/۱۹ ۱۳:۰۰:۵۷

سلام،
- بستگی دارد. اگر شبکه داخلی است. کلا بحث اندازه گیری آن‌را فراموش کنید. چون سرعت دریافت بالا است. اگر برای اینترنت برنامه می‌نویسید مهم است و بالای 30 کیلوبایت به ازای هر صفحه را باید زیاد دانست.
- همان لینک داده شده در ابتدای مقاله را دنبال کنید. یک سری لینک در کامنت‌های آن برای فشرده سازی هست به همراه پیاده سازی کامل.
- هم...، باید منتظر نگارش بعدی ASP.NET بود یعنی نگارش 5 . اگر در نگارش 5 هم تغییرات آنچنانی حاصل نشده بود (مثل نگارش 4 فعلی که فقط یک سری موارد جزئی در آن تغییر کرده، مثلا فشرده سازی سشن، مشخص کردن ID سمت کلاینت و از این نوع بحث‌هایی که خیلی وقت است خود برنامه نویس‌ها برای آن راه حل پیدا کرده‌اند و ... به نظر من جدید نیستند)، آنگاه ... MVC تنها راه حل "زنده" و در حال پیگیری از طرف مایکروسافت تلقی خواهد شد.

عنوان: **ثبت استثناهای مدیریت شده توسط ELMAH**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۰۹ ۱۳:۰۷:۰۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

در مورد ELMAH پیشتر مطالبی در این سایت منتشر شده است:

معرفی ELMAH

ELMAH 1.1

یکپارچه کردن ELMAH با WCF RIA Services

از آن می‌توان جهت ثبت استثناهای مدیریت نشده در انواع و اقسام برنامه‌های وب دات نتی استفاده کرد. از برنامه‌های ASP.NET MVC ، ASP.NET Webforms تا سرویس‌های WCF RIA ، WCF و غیره. سپس این Http module ویژه امکان مرور خطاهای ثبت شده را از راه دور میسر می‌سازد، به همراه فید RSS از خطاها، امکان مشاهده‌ی مقادیر فیلدها در لحظه‌ی بروز خطا، ارسال ایمیل خطاها و امکان ذخیره سازی آن‌ها در فایل‌های XML تا انواع بانک‌های اطلاعاتی.
برای نمونه، سایت Stackoverflow از یک نمونه‌ی سفارشی شده‌ی آن استفاده می‌کند (+).

همانطور که در این معرفی آمده است : ثبت استثناهای "مدیریت نشده". اما آیا امکان ثبت استثناهای مدیریت شده هم توسط آن وجود دارد؟
پاسخ: بله. به صورت زیر:

```
...
}
catch (Exception ex)
{
   Elmah.ErrorSignal.FromCurrentContext().Raise(ex);
...
```

فقط کافی است ارجاعی به اسمبلی ELMAH اضافه شود و از سطر فوق استفاده گردد. به این صورت خروجی شبیه به همان خطاهای مدیریت نشده در لیست خطاهای ثبت شده توسط ELMAH ظاهر خواهد شد.

نظرات خوانندگان

نویسنده: Iman
تاریخ: ۱۳۸۹/۱۱/۱۰ ۲۳:۴۲:۵۰

آیا سایت stackoverflow سورس بازه
سورسشو از کجا می شه دانلود کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۱۱ ۰۰:۴۵:۴۷

نه. یا نه هنوز : (+)
ولی دیتابیس سایتشون رو میشه دریافت کرد و هر ماه یک نسخه جدید ارائه می‌دن: (+)

البته نسخه‌های مشابه نوشته شده : (+) و (+)
و یک سایت ایرانی نمونه: (+)

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۱۶ ۱۰:۲۶:۱۹

تکمیل این بحث: در یک سری از متدهای فایل global.asax عموما Context ایی وجود ندارد (دقیقا مانند برنامه‌های دسکتاپ). در این حالت از (Elmah.ErrorLog.GetDefault(null).Log(new Error(ex)) استفاده کنید.

نویسنده: احسان
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۳:۹

با سلام.
دستور فوق رو گذاشتم اما به ErrorLog_Logged یا Application_Error منتقل نشد. منظور از ارجاع به اسمبلی رو متوجه نشدم.
میشه یک نمونه مثال بزنید.
ممنون...

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۳:۳۱

- بله؛ منتقل نمی‌شود به روال‌های رویدادگردان یاد شده. سطر یاد شده در متن فوق، صرفا خطای مدنظر را به لیست لاگ‌های قابل گزارشگیری ELMAH اضافه می‌کند.
- Elmah یک فایل DLL دارد به نام Elmah.dll. به این فایل‌های DLL در دات نت، اصطلاحا اسمبلی گفته می‌شود. افزودن ارجاعی به اسمبلی، یعنی مراجعه به منوی پروژه در VS.NET و سپس انتخاب گزینه‌ی Add reference در آنجا و انتخاب DLL یاد شده.
- در قسمت «مطالب مرتبط » ذیل هر بحث، یا حتی مقدمه‌ی بحث جاری، مثال‌های بیشتری ارائه شده‌اند.

عنوان: **چه زمان‌هایی یک برنامه‌ی ASP.NET ری استارت می‌شود؟**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۱۴ ۱۳:۳۳:۰۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

برای ری استارت کردن یک برنامه‌ی ASP.NET حتما نیازی نیست تا IIS را متوقف و سپس راه اندازی کرد یا تنظیمات App pool برنامه را در IIS تغییر داد. روش‌های دیگری نیز وجود دارند که عدم آگاهی از آن‌ها می‌تواند سبب بروز مشکلات عدیده‌ای گردد و گاها خطایابی آن‌ها بسیار مشکل است؛ زیرا ری استارت شدن برنامه = از دست رفتن آنی تمام سشن‌های InProc تمام کاربران سایت؛ پاک شدن کش برنامه در IIS؛ از دست رفتن تمام متغیرهای استاتیک، Application State و مواردی از این دست:

- نوشتن در پوشه bin برنامه (ایجاد فایل یا ایجاد پوشه یا تغییر نام پوشه‌ها و مواردی از این دست). (بنابراین قرار دادن بانک اطلاعاتی برنامه در این پوشه، اشتباه بوده و به همین جهت پوشه‌ی استاندارد App_Data طراحی شده است)
- نوشتن در فایل web.config برنامه (به صورت دستی، حتی در حد اضافه کردن یک فاصله یا توسط خود برنامه و یا استفاده از متد File.SetLastWriteTime). در این حالت ASP.NET FileSystemWatcher بلافاصله وارد عمل شده و برنامه را ری استارت می‌کند. (هدف اصلی وجودی این فایل، برخورد فقط خواندنی با آن است نه ذخیره سازی تنظیمات پویای برنامه در آن. برای ذخیره سازی تنظیمات پویا، از بانک اطلاعاتی و یا یک فایل XML مجزای از وب کانفیگ استفاده کنید یا مواردی مشابه)
- هر گونه تغییری در فایل‌ها و یا پوشه‌های machine.config ، Global.asax ، App_Code ، App_WebReferences ، App_GlobalResources و App_LocalResources نیز سبب ری استارت برنامه خواهند شد.
- با ایجاد، حذف یا تغییر نام یکی از ساب دایرکتوری‌های واقع شده در ریشه برنامه. بنابراین اگر برنامه‌ی شما به صورت پویا پوشه‌هایی را ایجاد یا حذف می‌کند باید منتظر ری استارت‌های پی در پی باشید (البته این مورد با از کار انداختن FileChangesMonitor مربوط به HttpRuntime قابل حل می‌باشد (+)، ولی همانطور که عنوان شد به صورت پیش فرض همواره فعال است)
- فراخوانی متد System.Web.HttpRuntime.UnloadAppDomain شبیه به همان Application.Exit در برنامه‌های دسکتاپ است و بلافاصله سبب خاتمه‌ی برنامه می‌شود. قرار دادن فایل App_Offline.htm در پوشه اصلی برنامه نیز چنین رفتاری را سبب خواهد شد. علاوه بر آن تگ httpRuntime در وب کانفیگ نیز دارای گزینه‌ی enable است و تنظیم آن به false ، سبب خاتمه‌ی سرویس دهی برنامه خواهد شد.
- رسیدن به عدد numRecompilesBeforeAppRestart تعریف شده در فایل machine.config که عموما به عدد 15 تنظیم شده است. اگر تغییرات زیادی را در فایل‌های (مرتبط با ASP.NET مانند asmx ، aspx و غیره) برنامه داده باشید (بیشتر از 15 مورد) و نیازی به ری کامپایل اساسی وجود داشته باشد، ASP.NET FileSystemWatcher به صورت خودکار برنامه را ری استارت خواهد کرد.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۹

چند نکته‌ی تکمیلی

- اگر هاست شما تمام سایت‌ها را با یک Application pool مدیریت می‌کند، کرش یکی از چند ده سایت دیگر می‌تواند سبب ری‌استارت شدن سایت شما هم بشود؛ چون برنامه‌ها از همه ایزوله نشده‌اند. راه حل آن ایجاد یک Application pool مجزا به ازای هر سایت هست (توسط هاست‌دار).

زمانیکه تمام سایت‌ها با یک Application pool واحد مدیریت می‌شوند، تمام آن‌ها توسط یک وهله از w3wp.exe اجرا خواهند شد. با تعریف Application poolهای مجزا، هر سایت، یک وهله‌ی مجزا از w3wp.exe را به خود اختصاص خواهد داد. یعنی اگر Task manager سرور را بررسی کنید، به ازای هر سایت، یک w3wp.exe با pid مجزا قابل مشاهده است. به این ترتیب اگر pid=1234 کرش کرد، تاثیری روی pid=4321 نخواهد داشت.

- یک برنامه‌ی ASP.NET پس از مدتی بیکاری (قابل تنظیم در Application pool برنامه)، به صورت خودکار توسط IIS از حافظه خارج می‌شود. با درخواست بعدی که به آن برنامه می‌رسد (مثلا گشودن یک صفحه‌ی آن توسط یک کاربر)، مجددا از صفر اجرا خواهد شد. این مورد نیز به معنای ری‌استارت کامل برنامه است.

- در تنظیمات Application pool موارد زیادی را می‌توان تنظیم کرد که سبب ری‌استارت شدن برنامه می‌شوند. برای مثال اگر مصرف CPU و یا حافظه‌ی برنامه به حد مشخصی رسید، برنامه ری‌استارت شود و امثال آن.

عنوان: **استفاده از افزونه‌ی jQuery Autocomplete در ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۲۳ ۱۰:۱۶:۰۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

با استفاده از AutoComplete TextBoxes می‌توان گوشه‌ای از زندگی روزمره‌ی کاربران یک برنامه را ساده‌تر کرد. مشکل مهم dropDownList ها دریک برنامه‌ی وب، عدم امکان تایپ قسمتی از متن مورد نظر و سپس نمایان شدن آیتم‌های متناظر با آن در اسرع وقت می‌باشد. همچنین با تعداد بالای آیتم‌ها هم حجم صفحه و زمان بارگذاری را افزایش می‌دهند. راه حل‌های بسیار زیادی برای حل این مشکل وجود دارند و یکی از آن‌ها ایجاد AutoComplete TextBoxes است. پلاگین‌های متعددی هم جهت پیاده سازی این قابلیت نوشته شده‌اند منجمله jQuery Autocomplete . این پلاگین دیگر توسط نویسنده‌ی اصلی آن نگهداری نمی‌شود اما توسط برنامه نویسی دیگر در github ادامه یافته است. در ادامه نحوه‌ی استفاده از این افزونه را در ASP.NET Webforms بررسی خواهیم کرد.

الف) دریافت افزونه

لطفا به آدرس GitHub ذکر شده مراجعه نمائید.

سپس برای مثال پوشه‌ی js را به پروژه افزوده و فایل‌های jquery.autocomplete.js ، jquery-1.5.min.js
jquery.autocomplete.css و indicator.gif را در آن کپی کنید. فایل indicator.gif به همراه مجموعه‌ی دریافتی ارائه نمی‌شود و یک آیکن loading معروف می‌تواند باشد.
علاوه بر آن یک فایل جدید custom.js را نیز جهت تعاریف سفارشی خودمان اضافه خواهیم کرد.

ب) افزودن تعاریف افزونه به صفحه

در ذیل نحوه‌ی افزودن فایل‌های فوق به یک master page نمایش داده شده است.
در اینجا از قابلیت‌های جدید ScriptManager (موجود در سرویس پک یک دات نت سه و نیم و یا دات نت چهار) جهت یکی کردن اسکریپت‌ها کمک گرفته شده است. به این صورت تعداد رفت و برگشت‌ها به سرور به‌جای سه مورد (تعداد فایل‌های اسکریپت مورد استفاده)، یک مورد (نهایی یکی شده) خواهد بود و همچنین حاصل نهایی به صورت خودکار به شکلی فشرده شده به مرورگر تحویل داده شده، سرآیندهای کش شدن اطلاعات به آن اضافه می‌گردد (که در سایر حالات متداول اینگونه نیست)؛ به علاوه Url نهایی آن هم بر اساس hash فایل‌ها تولید می‌شود. یعنی اگر محتوای یکی از این فایل‌ها تغییر کرد، چون Url نهایی تغییر می‌کند، دیگر لازم نیست نگران کش شدن و به روز نشدن اسکریپت‌ها در سمت کاربر باشیم.

```
<%@ Master Language="C#" AutoEventWireup="true" CodeBehind="Site.master.cs"
Inherits="AspNetjQueryAutocompleteTest.Site" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
   <asp:PlaceHolder Runat="server">
       <link href="<%= ResolveClientUrl("~/js/jquery.autocomplete.css")%>" rel="stylesheet"
type="text/css" />
   </asp:PlaceHolder>
   <asp:ContentPlaceHolder ID="head" runat="server">
   </asp:ContentPlaceHolder>
</head>
<body>
   <form id="form1" runat="server">
   <asp:ScriptManager ID="ScriptManager1" runat="server">
       <CompositeScript>
           <Scripts>
               <asp:ScriptReference Path="~/js/jquery-1.5.min.js" />
               <asp:ScriptReference Path="~/js/jquery.autocomplete.js" />
```

```
                <asp:ScriptReference Path="~/js/custom.js" />
            </Scripts>
        </CompositeScript>
    </asp:ScriptManager>
    <div>
        <asp:ContentPlaceHolder ID="ContentPlaceHolder1" runat="server">
        </asp:ContentPlaceHolder>
    </div>
    </form>
</body>
</html>
```

علت استفاده از ResolveClientUrl در حین تعریف فایل css در اینجا به عدم مجاز بودن استفاده از ~ جهت مسیر دهی فایل‌های css در header صفحه بر می‌گردد.

ج) افزودن یک صفحه‌ی ساده به برنامه

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master" AutoEventWireup="true"
   CodeBehind="default.aspx.cs" Inherits="AspNetjQueryAutocompleteTest._default" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
   <asp:TextBox ID="txtShenas" runat="server" />
</asp:Content>
```

فرض کنید می‌خواهیم افزونه‌ی ذکر شده را به TextBox استاندارد فوق اعمال کنیم. ID این TextBox در نهایت به شکل ContentPlaceHolder1_txtShenas رندر خواهد شد. البته در ASP.NET 4.0 با تنظیم ClientIDMode=Static می‌توان ID انتخابی خود را به جای این ID خودکار درنظر گرفت و اعمال کرد. اهمیت این مساله در قسمت (ه) مشخص می‌گردد.

د) فراهم آوردن اطلاعات مورد استفاده توسط افزونه‌ی AutoComplete به صورت پویا

مهم‌ترین قسمت استفاده از این افزونه، تهیه‌ی اطلاعاتی است که باید نمایش دهد. این اطلاعات باید به صورت فایلی که هر سطر آن حاوی یکی از آیتم‌های مورد نظر است، تهیه گردد. برای این منظور می‌توان از فایل‌های ASHX یا همان Generic handlers استفاده کرد:

```
using System;
using System.Data.SqlClient;
using System.Text;
using System.Web;

namespace AspNetjQueryAutocompleteTest
{
   public class AutoComplete : IHttpHandler
   {
       public void ProcessRequest(HttpContext context)
       {
           string prefixText = context.Request.QueryString["q"];
           var sb = new StringBuilder();

           using (var conn = new SqlConnection())
           {
               //todo: این مورد باید از فایل کانفیگ خوانده شود
               conn.ConnectionString = "Data Source=(local);Initial Catalog=MyDB;Integrated Security =
true";
               using (var cmd = new SqlCommand())
               {
                   cmd.CommandText = @" select Field1 ,Field2 from tblData where  Field1 like
@SearchText + '%' ";
                   cmd.Parameters.AddWithValue("@SearchText", prefixText);
                   cmd.Connection = conn;
                   conn.Open();
                   using (var sdr = cmd.ExecuteReader())
                   {
                       if (sdr != null)
```

```
                            while (sdr.Read())
                            {
                                string field1 = sdr.GetValue(0) == DBNull.Value ? string.Empty :
sdr.GetValue(0).ToString().Trim();
                                string field2 = sdr.GetValue(1) == DBNull.Value ? string.Empty :
sdr.GetValue(1).ToString().Trim();
                                sb.AppendLine(field1 + "|" + field2);
                            }
                        }
                    }
                }

                context.Response.Write(sb.ToString());
            }

            public bool IsReusable
            {
                get
                {
                    return false;
                }
            }
        }
}
```

در این مثال از ADO.NET کلاسیک استفاده شده است تا به عمد نحوه‌ی تعریف پارامترها یکبار دیگر مرور گردند. اگر از LINQ to SQL یا Entity framework یا NHibernate و موارد مشابه استفاده می‌کنید، جای نگرانی نیست؛ زیرا کوئری‌های SQL تولیدی توسط این ORMs به صورت پیش فرض از نوع پارامتری هستند (+).

در این مثال اطلاعات دو فیلد یک و دوی فرضی از جدولی با توجه به استفاده از like تعریف شده دریافت می‌گردد. به عبارتی همان متد StartsWith معروف LINQ بکارگرفته شده است.

به صورت خلاصه افزونه، کوئری استرینگ q را به این فایل ashx ارسال می‌کند. سپس کلیه آیتم‌های شروع شده با مقدار دریافتی، از بانک اطلاعاتی دریافت شده و هر کدام قرارگرفته در یک سطر جدید بازگشت داده می‌شوند.

اگر دقت کرده باشید در قسمت sb.AppendLine ، با استفاده از "|" دو مقدار دریافتی از هم جدا شده‌اند. عموما یک مقدار کفایت می‌کند (در 98 درصد موارد) ولی اگر نیاز بود تا توضیحاتی نیز نمایش داده شود از این روش نیز می‌توان استفاده کرد. برای مثال یک مقدار خاص به همراه توضیحات آن به عنوان یک آیتم نمایش داده شده مد نظر است.

ه) اعمال نهایی افزونه به TextBox

در ادامه پیاده سازی فایل custom.js برای استفاده از امکانات فراهم شده در قسمت‌های قبل ارائه گردیده است:

```
function formatItem(row) {
   return row[0] + "<br/><span style='text-align:justify;' dir='rtl'>" + row[1] + "</span>";
}

$(document).ready(function () {
   $("#ContentPlaceHolder1_txtShenas").autocomplete('AutoComplete.ashx', {
       //Minimum number of characters a user has to type before the autocompleter activates
       minChars: 0,
       delay: 5,
       //Only suggested values are valid
       mustMatch: true,
       //The number of items in the select box
       max: 20,
       //Fill the input while still selecting a value
       autoFill: false,
       //The comparison doesn't looks inside
       matchContains: false,
       formatItem: formatItem
   });
});
```

پس از این مقدمات، اعمال افزونه‌ی autocomplete به textBox ایی با id مساوی ContentPlaceHolder1_txtShenas ساده

است. اطلاعات از فایل AutoComplete.ashx دریافت می‌گردد و تعدادی از خواص پیش فرض این افزونه در اینجا مقدار دهی شده‌اند. لیست کامل آن‌ها را در فایل jquery.autocomplete.js می‌توان مشاهده کرد.

تنها نکته‌ی مهم آن استفاده از پارامتر اختیاری formatItem است. اگر در حین تهیه‌ی AutoComplete.ashx خود تنها یک آیتم را در هر سطر نمایش می‌دهید و از "|" استفاده نکرده‌اید، نیازی به ذکر آن نیست. در این مثال ویژه، فیلد یک در یک سطر و فیلد دو در سطر دوم یک آیتم نمایش داده می‌شوند:

نظرات خوانندگان

نویسنده: شاهین کیاست
تاریخ: ۱۳۸۹/۱۱/۲۳ ۲۰:۱۵:۳۴

سلام آقای نصیری.
بسیار ممنونم :)
چند تا سوال دارم.
1) این روش چه برتری بر پلاگین jQuery UI دارد؟
2) می شود از Web Service هم استفاده کرد. آیا Generic handlers سرعت و کارایی بهتری می دهد؟
چگونه می توان ID رکوردی که بازگشته را در جایی نگه داشت تا جهت انجام عملیات سمت سرور از آن استفاده کرد.
با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۲۳ ۲۱:۲۲:۵۱

سلام،
- برتری خاصی ندارد. فقط اگر کسی نمی‌خواهد از jQuery UI و فایل‌های آن استفاده کند، این روش هم هست و سبک‌تر است. در این مورد باز هم بگردید افزونه هست. البته بعدش باید بگردید به چه صورت باید دیتا به آن‌ها پاس کرد و روش‌ها یکی نیست.
- بله. حتی می‌شود از یک صفحه معمولی aspx هم استفاده کرد. در page_load آن کافی است این Response.Write ذکر شده قرار گیرد. در این مثال فوق از یکی از چندین روش ممکن، استفاده شده با سربار حداقل ممکن.
- سوال خوبیه. به عبارتی اگر قرار است بجای یک dropdown که قبلا text هر ردیف آن متن خاصی بود و value آن id یا همان primary key رکورد جاری و از دید کاربر مخفی بود، الان باید چکار کرد تا همان id را بتوان به سرور ارسال کرد. چون به ظاهر فقط text دریافت می‌شود.
شما در اینجا از همان روش row_0 و row_1 که ذکر شد استفاده کنید. row_1 را با id در سمت سرور پر کنید field1+"|"+id .
سپس این پلاگین متد result هم دارد (مقدار انتخاب شده توسط کاربر را بازگشت می‌دهد). به این صورت فرصت خواهد بود تا یک فیلد مخفی را جهت ارسال به سرور مقدار دهی کرد:

```
(result(function (evt, row, formatted.
}
;([hiddenIDbox").val(row[1#")$
{
```

لزومی هم ندارد تا این id یا همان row_1 را در تابع formatItem نمایش داد. برای نمایش همان row_0 کافی است.

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۰:۲۵

پیشنهادات امنیتی تون برای استفاده از فایل‌های ASHX یا Generic handlers چیست ؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۰:۲۸

در مثال جاری با توجه به اینکه ورودی دریافتی از کاربر به کمک SQL پارامتری توسط ADO.NET پردازش می‌شود، مقاومت در برابر تزریق اس‌کیوال آن افزایش یافته است.

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۰:۴۶

منظورم در حالت کلی است.

یک موردش : "چنانچه نیاز به استفاده از Session در فایل ASHX است، واسط IReadOnlySessionState به کار گرفته شود. "

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۰:۵۱

در حالت کلی تفاوت خاصی وجود ندارد با سایر قسمت‌ها و مسایل دیگر امنیتی ASP.NET. بیشتر، ورودی‌های دریافتی توسط آن‌را باید صحیح بکار بگیرید و مراقب آن‌ها باشید.

عنوان: **نکات نصب برنامه‌های ASP.NET 4.0 بر روی IIS 6**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۱۱/۲۵ ۰۸:۴۱:۰۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

سه نکته مهم حین توزیع برنامه‌های ASP.NET 4.0 بر روی IIS 6.0 نسبت به سایر نگارش‌های قبلی وجود دارند که باید در نظر گرفته شوند:

الف) پس از اتقای برنامه از نگارش‌های قبلی به دات نت 4 (با فرض اینکه دات نت 4 بر روی سرور نصب است)، پیغام 404 یا به عبارتی فایل مورد نظر بر روی سرور یافت نشد را دریافت می‌کنید (با تمام فایل‌های موجود):
در کنسول IIS ، ذیل قسمت Web Services Extensions ، باید دو مورد از حالت prohibited خارج شوند:
All unknown ISAPI extensions
ASP.NET 4.0

ب) پس از اجرای برنامه پیغام غیر معتبر بودن تگ‌های جدید فایل Web.Config را ملاحظه می‌کنید:
- در برگه‌ی خواص سایت در IIS 6.0 ، اکنون امکان انتخاب ASP.NET 4.0 هم میسر است که حتما باید این مورد انتخاب گردد (تا دات نت سه و نیم این نام تنها ASP.NET 2.0 بود). در غیر اینصورت تگ‌های جدید فایل Web.Config شناخته نخواهند شد.

ج) بلافاصله پس از اجرای برنامه، پیغام Server Application Unavailable قابل مشاهده است:
نکته‌ی مهم دیگری که به همراه برنامه‌های دات نت 4 باید به آن توجه داشت، ضرورت اجرای آن‌ها در یک پروسه جدید است. پروسه جدید در IIS 6.0 به معنای یک Application pool جدید است. به عبارتی اگر هم اکنون بر روی IIS 6.0 شما برای مثال 2 برنامه‌ی دات نت سه و دات نت 4 قصد استفاده از یک Application pool را داشته باشند، پیغام Server Application Unavailable ظاهر خواهد شد، زیرا نمی‌توان این دو نگارش را تحت یک پروسه اجرا کرد. برای حل این مشکل باید یک Application pool جدید و اختصاصی را جهت برنامه‌ی ASP.Net 4.0 خود تعریف و انتساب دهید.

نظرات خوانندگان

نویسنده: حمید
تاریخ: ۱۳۹۲/۱۰/۰۲ ۱۱:۶

خیلی ممنون از آموزش‌های مفید و گرانقدرتون.
در مورد Application Pool ممکنه بیشتر توضیح بدین.اینکه فرمودین به ازای هر برنامه باید یک Application Pool ایجاد کرد ، ازدیاد آنها معایبی را در بر نداره ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۰۲ ۱۱:۵۵

خیر. «باید» به ازای هر برنامه یک application pool مجزا درست کنید. برای مطالعه بیشتر:
- برای مدیریت نگارش‌های مختلف دات نت مورد استفاده (مطلب جاری)
- جهت مدیریت بهتر میزان مصرف حافظه یک برنامه: مروری بر تاریخچه محدودیت حافظه مصرفی برنامه‌های ASP.NET در IIS
- برای جلوگیری از ری‌استارت شدن بی‌جهت سایر برنامه‌ها: چه زمان‌هایی یک برنامه‌ی ASP.NET ری استارت می‌شود؟
- جهت کنترل میزان مصرف CPU یک برنامه: بررسی علت CPU Usage بالای برنامه در حال اجرا

عنوان: **شایعاتی در مورد نسخه‌ی بعدی ASP.NET Webforms**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۱/۱۷ ۰۰:۱۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

مدتی قبل مطلبی تحت عنوان " What’s coming in the next version of ASP.NET Webforms " منتشر شد (که نویسنده آن دقیقا مشخص نیست این اطلاعات را از کجا آورده و همچنین تکذیبیه‌ای هم جایی در مورد آن صادر نشد ...)؛ بنابراین خلاصه‌ای از آن‌را با هم مرور خواهیم کرد:

اخیرا تمام توجه تیم ASP.NET معطوف نسخه‌ی MVC آن شده است؛ هر چند هنوز تعداد قابل توجهی از پروژه‌های ASP.NET بر اساس Webforms تهیه شده‌اند یا می‌شوند. همچنین برخلاف مطالب منتشره در انجمن‌ها یا بلاگ‌های مرتبط، تیم ASP.NET ، نگارش Webforms را فراموش نکرده و حتی نگارش 4 آن نیز تعدادی از قابلیت‌های MVC مانند URL Routing، حجم کمتر ViewState و کنترل بیشتر بر روی HTML نهایی را به همراه داشته است.
به روز رسانی‌های متوالی MVC (که اکنون به نگارش 3 رسیده است)، شاید این تصور را پیش آورده باشد که دیگر Webforms مرده است! اما مهترین دلیل به روز رسانی‌های دیر هنگام نسخه‌ی Webforms ، یکی بودن اسمبلی‌های آن با مجموعه‌ی اصلی دات نت فریم ورک است (برخلاف نسخه‌ی MVC که به صورت افزونه‌ای برای این مجموعه ارائه شده است).

نسخه‌ی بعدی Webforms (حداقل) شامل تازه‌ها و پیشرفت‌های زیر خواهد بود:

MVC ModelBinders
در نسخه‌ی MVC مفهومی به نام Model binders وجود دارد. کار آن مقدار دهی مدل برنامه به صورت خودکار بر اساس اطلاعات وارد شده توسط کاربر در رابط کاربری برنامه است. برای مثال در Webforms داریم employee.Name = txtName.Text . به این معنا که مقدار Text یک جعبه‌ی متنی به نام txtName را به خاصیت Name شیء employee نسبت بده. اینکار (انقیاد اطلاعات رابط کاربر به مدل برنامه) با وجود Model binders در نسخه‌ی MVC به صورت خودکار انجام می‌شود. این مورد دو مزیت عمده را به همراه خواهد داشت: الف) سادگی و حجم کمتر کد ب) امکان تهیه ساده‌تر unit test جهت قسمت‌های مختلف برنامه (چون دیگر به txtName گره نخواهد خورد).
امکانات Model binders ، گفته شده (مطابق مرجع فوق!) که قرار است جزئی از نگارش بعدی Webforms باشد ... (امیدوارم!)

بهبودهای حاصل شده در اعتبار سنجی
نسخه‌ی بعدی Webforms شامل پیشرفت‌های اعتبارسنجی نسخه‌ی MVC نیز خواهد بود. به این معنا که امکان کنارگذاشتن کنترل‌های اعتبار سنجی Webforms و استفاده یکپارچه از امکانات jQuery فراهم خواهد شد (به این صورت دیگر شما محدود به یک سری کنترل از پیش تعیین شده نخواهید بود و امکان دسترسی به کوهی از افزونه‌های اعتبار سنجی jQuery را خواهید داشت).

CSS Sprites
CSS Sprites که در نگارش بعدی Webforms پشتیبانی خواهد شد (+)، تکنیکی است جهت کاهش تعداد رفت و برگشت‌های به سرور با ارائه‌ی یک فایل حاوی تمام تصاویر قرار گرفته شده در یک شبکه یا گرید. به این صورت بجای دها یا صدها رفت و برگشت به سرور جهت دریافت تصاویر یک صفحه، تنها یک رفت و برگشت انجام خواهد شد.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۰۱/۲۷ ۱۷:۱۶:۰۹

Web Forms: Reports of my Death are Greatly Exaggerated

عنوان: **حساسیت سیستم فیلترینگ به عبارات کوکی‌های سایت‌ها**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۱/۳۰ ۰۹:۱۶:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

از دیروز نمی‌تونستم به یک سایت آزمایشی ASP.NET وصل بشم. مشکل هم با فایرفاکس بود. با IE تست کردم و سایت وارد شد. بنابراین سایت فیلتر نشده بود. مشکل از کجا بود؟

بله ... سیستم فیلترینگ به هدرها هم حساس شده. کوکی را دستی پاک کردم. مجددا پس از لاگین، با ساخته شدن کوکی جدید، مشکل برطرف شد.
این حروف هم به صورت اتفاقی تولید می‌شوند؛ بنابراین ممکن است با این نوع مسایل زیاد مواجه شوید! اگر یک سایت را با دو مرورگر متفاوت، یکی فیلتر شده و دیگری بدون مشکل باز کردید، یعنی یکبار باید کوکی‌های سایت را در مرورگری که با آن مشکل وجود دارد، پاک کنید!

خطایی را هم که مشاهده خواهید کرد این است:

```
IMB2-6
http://10.10.34.34?type=Invalid Pattern&policy=MainPolicy
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۱/۳۰ ۱۳:۰۵:۳۸

من امروز کارم شده پاک کردن کوکی‌های ASPXROLES . از این افزونه هم استفاده می‌کنم تا سریعتر باشد: (^)

نویسنده: Salar
تاریخ: ۱۳۹۰/۱۱/۳۰ ۲۰:۲۸:۳۱

این مشکل تو کوکیهای یاهو هم هست. پس یه خاطر خودشون هم شده درستش می کنن!

نویسنده: Mohsen Ara
تاریخ: ۱۳۹۰/۱۲/۰۲ ۰۸:۴۸:۰۱

جالب بود چند روز پیش ولنتاین رو تو یاهور سرچ کردم فیلتر بود ولی x /e/ s رو سرچ کردم فیلتر نبود !!
معلوم نیست واقعا چی به چیه !!!

نویسنده: nile
تاریخ: ۱۳۹۰/۱۲/۰۲ ۱۲:۲۵:۴۳

:)) چقدر باحال!!!!!!!!
چطوریه با هربار لاگین یک کوکی جدید ایحاد میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۰/۱۲/۰۲ ۱۳:۰۶:۱۰

به این مبحث مراجعه کنید:

Role Cookies & Cryptographically Random Keys

عنوان: **SignalR**
نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۳/۲۸ ۲۲:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, SignalR

چند وقتی هست که در کنار بدنه اصلی دات‌نت فریم‌ورک چندین کتابخونه به صورت متن‌باز در حال توسعه هستند. این مورد در ASP.NET بیشتر فعاله و مثلا دو کتابخونه **SignalR** و **WebApi** توسط خود مایکروسافت توسعه داده میشه.
SignalR همونطور که در سایت بسیار خلاصه و مفید یک صفحه‌ای! خودش توضیح داده شده (^) یک کتابخونه برای توسعه برنامه‌های وب «زمان واقعی»! (**real-time web**) است:

```
Async library for .NET to help build real-time, multi-user interactive web applications.
```

برنامه‌های زمان واقعی به صورت خلاصه و ساده به‌صورت زیر تعریف میشن (^):

```
The real-time web is a set of technologies and practices that enable users to receive information as
soon as it is published by its authors, rather than requiring that they or their software check a
source periodically for updates.
```

یعنی کاربر سیستم ما بدون نیاز به ارسال درخواستی صریح! برای دریافت آخرین اطلاعات به روز شده در سرور، در برنامه کلاینتش از این تغییرات آگاه بشه. مثلا برنامه‌هایی که برای نمایش نمودارهای آماری داده‌ها استفاده میشه (بورس، قیمت ارز و طلا و ...) و یا مهمترین مثالش میتونه برنامه «چت» باشه. متاسفانه پروتوکل HTTP مورد استفاده در وب محدودیت‌هایی برای پیاده‌سازی این گونه برنامه‌ها داره. روش‌های گوناگونی برای پیاده‌سازی برنامه‌های زمان واقعی در وب وجود داره که کتابخونه SignalR فعلا از موارد زیر استفاده میکنه:
تکنولوژی جدید **WebSocket** (^) که خوشبختانه پشتیبانی کاملی از اون در دات نت 4.5 (چهار نقطه پنج! نه چهار و نیم!) وجود داره. اما تمام مرورگرها و تمام وب سرورها از این تکنولوژی پشتیبانی نمیکنند و تنها برخی نسخه‌های جدید قابلیت استفاده از آخرین ورژن WebSocket رو دارند که میشه به کروم 16 به بالا و فایرفاکس 11 به بالا و اینترنت اکسپلورر 10 اشاره کرد (برای استفاده از این تکنولوژی در ویندوز نیاز به **IIS 8.0** است که متاسفانه فقط در ویندوز 8.0 موجوده):

Chrome 16, Firefox 11 and Internet Explorer 10 are currently the only browsers supporting the latest specification (RFC 6455).

یه روش دیگه **Server-sent Events** نام داره که داده‌های جدید رو به فرم رویدادهای DOM به سمت کلاینت میفرسته(^).

روش دیگه‌ای که موجوده به **Forever Frame** معروفه که در این روش یک iframe مخفی درون کد html مسئول تبادل داده‌هاست. این iframe مخفی به‌صورت یک بلاک Chunked (^) به سمت کلاینت فرستاده میشه. این iframe که مسئول رندر داده‌های جدید در سمت کلاینت هست ارتباط خودش رو با سرور تا ابد! (برای همین بهش forever میگن) حفظ میکنه. هر وقت رویدادی سمت سرور رخ میده با استفاده از این روش داده‌ها به‌صورت تگ‌های script به این فریم مخفی فرستاده می‌شوند و چون مرورگرها محتوای html رو به صورت افزایشی (incrementally) رندر میکنن بنابراین این اسکریپتها به‌ترتیب زمان دریافت اجرا می‌شوند. (البته ظاهرا عبارت forever frame در صنعت عکاسی! معروف‌تره بنابراین در جستجو در زمینه این روش ممکنه کمی مشکل داشته باشین) (^).
روش آخر که در کتابخونه SignalR ازش استفاده میشه **long-polling** نام داره. در روش polling معمولی پس از ارسال درخواست توسط کلاینت، سرور بلافاصله نتیجه حاصله رو به سمت کلاینت میفرسته و ارتباط قطع میشه . بنابراین برای داده‌های جدید درخواست جدیدی باید به سمت سرور فرستاده بشه که تکرار این روش باعث افزایش شدید بار بر روی سرور و کاهش کارآمدی اون می‌شه. اما در روش long-polling پس از برقراری ارتباط کلاینت با سرور این ارتباط تا مدت زمان معینی (که توسط یه مقدار تایم اوت مشخص میشه و مقدار پیش‌فرضش 2 دقیقه است) برقرار میمونه . بنابراین کلاینت میتونه بدون ایجاد مشکلی در کارایی، داده‌های جدید رو از سرور دریافت کنه. به این روش در برنامه‌نویسی وب اصطلاحا برنامه‌نویسی کامت (**Comet Programming**) میگن (^ ^).

(البته روش‌های دیگری هم برای پیاده‌سازی برنامه‌های زمان اجرا وجود داره مثل کتابخونه **node.js** که جستجوی بیشتر به خوانندگان واگذار میشه)

SignalR برای برقراری ارتباط ابتدا بررسی میکنه که آیا هر دو سمت سرور و کلاینت قابلیت پشتیبانی از WebSocket رو دارند. در غیراینصورت سراغ روش Server-sent Events میره. اگر باز هم موفق نشد سعی به برقراری ارتباط با روش forever frame میکنه و اگر باز هم موفق نشد در آخر سراغ long-polling میره.
با استفاده از SignalR شما میتونین از سرور، متدهایی رو در سمت کلاینت فراخونی کنین. یعنی درواقع با استفاده از کدهای سی شارپ میشه متدهای جاوااسکریپت سمت کلاینت رو صدا زد!
بطور خلاصه در این کتابخونه دو کلاس پایه وجود داره:
کلاس سطح پایین **PersistentConnection**

کلاس سطح بالای **Hub**

علت این نامگذاری به این دلیله که کلاس سطح پایین پیاده‌سازی پیچیده‌تر و تنظیمات بیشتری نیاز داره اما امکانات بیشتری هم در اختیار برنامه‌نویس قرار می‌ده.
خوب پس از این مقدمه نسبتا طولانی برای دیدن یک مثال ساده میتونین با استفاده از نوگت (Nuget) مثال زیر رو نصب و اجرا کنین (اگه تا حالا از نوگت استفاده نکردین قویا پیشنهاد میکنم که کار رو با دریافتش از اینجا آغاز کنین) :

```
PM> Install-Package SignalR.Sample
```

پس از کامل شدن نصب این مثال اون رو اجرا کنین. این یک مثال فرضی ساده از برنامه نمایش ارزش آنلاین سهام برخی شرکتهاست. میتونین این برنامه رو همزمان در چند مرورگر اجرا کنین و نتیجه رو مشاهده کنین.
حالا میریم سراغ یک مثال ساده. میخوایم یک برنامه چت ساده بنویسیم. ابتدا یک برنامه وب اپلیکیشن خالی رو ایجاد کرده و با استفاده از دستور زیر در خط فرمان نوگت، کتابخونه SignalR رو نصب کنین:

```
PM> Install-Package SignalR
```

پس از کامل شدن نصب این کتابخونه، ریفرنس‌های زیر به برنامه اضافه میشن:

```
Microsoft.Web.Infrastructure
Newtonsoft.Json
SignalR
SignalR.Hosting.AspNet
SignalR.Hosting.Common
```

برای کسب اطلاعات مختصر و مفید از تمام اجزای این کتابخونه به اینجا مراجعه کنین.
همچنین اسکریپت‌های زیر به پوشه Scripts اضافه میشن (این نسخه‌ها مربوط به زمان نگارش این مطلب است):

```
jquery-1.6.4.js
jquery.signalR-0.5.1.js
```

بعد یک کلاس با نام SimpleChat به برنامه اضافه و محتوای زیر رو در اون وارد کنین:

```
using SignalR.Hubs;
namespace SimpleChatWithSignalR
{
  public class SimpleChat : Hub
  {
    public void SendMessage(string message)
    {
      Clients.reciveMessage(message);
    }
```

```
  }
}
```

دقت کنین که این کلاس از کلاس Hub مشتق شده و همچنین خاصیت **Clients** از نوع **dynamic** است. (در مورد جزئیات این کتابخونه در قسمت‌های بعدی توضیحات مفصل‌تری داده میشه)

سپس یک فرم به برنامه اضافه کرده و محتوای زیر رو در اون اضافه کنین:

```
<input type="text" id="msg" />
<input type="button" value="Send" id="send" /><br />
<textarea id='messages' readonly="true" style="height: 200px; width: 200px;"></textarea>
<script src="Scripts/jquery-1.6.4.min.js" type="text/javascript"></script>
<script src="Scripts/jquery.signalR-0.5.1.min.js" type="text/javascript"></script>
<script src="signalr/hubs" type="text/javascript"></script>
<script type="text/javascript">
  var chat = $.connection.simpleChat;
  chat.reciveMessage = function (msg) {
   $('#messages').val($('#messages').val() + "-" + msg + "\r\n");
  };
  $.connection.hub.start();
  $('#send').click(function () {
    chat.sendMessage($('#msg').val());
  });
</script>
```

همونطور که میبینین برنامه چت ما آماده شد! حالا برنامه رو اجرا کنین و با استفاده از دو مرورگر مختلف نتیجه رو مشاهده کنین.

نکته کلیدی کار SignalR در خط زیر نهفته است:

```
<script src="signalr/hubs" type="text/javascript"></script>
```

اگر محتوای آدرس فوق رو دریافت کنین می‌بینین که موتور این کتابخانه تمامی متدهای موردنیاز در سمت کلاینت رو با استفاده از کدهای جاوااسکریپت تولید کرده. البته در این کد تولیدی از نامگذاری camel Casing استفاده میشه، بنابراین متد SendMessage در سمت سرور به‌صورت sendMessage در سمت کلاینت در دسترسه.

امیدوارم تا اینجا تونسته باشم علاقه شما به استفاده از این کتابخونه رو جلب کرده باشم. در قسمت‌های بعد موارد پیشرفته‌تر این کتابخونه معرفی میشه.

اگه علاقه‌مند باشین میتونین از این ویکی اطلاعات بیشتری بدست بیارین.

به روز رسانی

در دوره‌ای به نام SignalR در سایت ، به روز شده‌ای این مباحث را می‌توانید مطالعه کنید.

نظرات خوانندگان

نویسنده: روح الله
تاریخ: ۱۳۹۱/۰۳/۲۹ ۰:۸

جالب بود. اتفاقا امروز با دوستان صحبت از همچین نیازمندی بود. ممنون از زحمات شما.

نویسنده: افشار محبی
تاریخ: ۱۳۹۱/۰۳/۲۹ ۹:۱۷

به نظر فریمورک جذاب و مفیدی میاد. همینجا به آقای نصیری و همکارانش به خاطر راه اندازی وبلاگ جدید تبریک می‌گویم.

نویسنده: علیرضا جهانشاهلو
تاریخ: ۱۳۹۱/۰۳/۲۹ ۹:۳۹

خیلی عالی بود! اگه بدونید این کتابخونه پایه اساس خیلی از ایده‌های تجاریه!؟ که من در حال حاضر مشغول به کار بر روی یکی از همین ایده‌ها هستم.

نویسنده: احمد
تاریخ: ۱۳۹۱/۰۳/۲۹ ۱۹:۵۶

سلام و تبریک برای سایت جدید
از اینکه مطلب بسیار مفید و خوبی گذاشتید بسیار سپاسگزارم.اما اسکریپت signalr/hubs کجا قرار داره؟

نویسنده: Mona
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۰:۳۵

با سلام خدمت دوستان عزیز
ممنون از مطالب دقیق و کاربردی شما
لطفاً اگر امکان دارد مقایسه ای بین SignalR و Nod.js انجام دهید، زیرا در هرکجای وب که قدم میگذارید بحث از Nod.js است.
ممنون

نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۱:۴۶

این مسیر درواقع آدرس پیش‌فرض signalr برای کلاس پایه Hub هست که البته میتونه عوض هم بشه. این آدرس درواقع به یه هندلر مپ شده و این هندلر به محض دریافت یه درخواست از سمت کلاینت شروع به گشتن کل اسمبلی برای یافتن کلاسهای مشتق شده از Hub میگرده و برای تمام اجزای public اونا جهت در دسترس بودن در سمت کلاینت متدهای جاوااسکریپت متناظر تولید کرده و در متن پاسخ ارسالی مینویسه (البته یکسری کد دیگه هم برای کار با کلاس Hub تولید میکنه).
برای دریافت این اسکریپت تولیدی، شما میتونین این مسیر رو در نوار آدرس مرورگرتون وارد کنید و کد جاوااسکریپت تولیدی رو مشاهده کنین. حتی میتونین این کد تولیدشده رو در یک فایل ذخیره کرده و به جای ریفرنس اصلی استفاده کنین. البته به شرطی که در اجزای پابلیک کلاسهاتون بعدا تغییری ایجاد نشه.

نویسنده: ramin
تاریخ: ۱۳۹۱/۰۳/۳۰ ۱۰:۵۹

سلام
یعنی مسیر و محتوای signalr/hubs در هتگام اجرا تولید میشه؟
من مثال شما رو دنبال کردم ولی خروجی مورد نظر رو نگرفتم

نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۳/۳۰ ۱۱:۱۰

من خودم تخصصی در زمینه node.js ندارم. البته ازش خوشم هم نمیاد :)
اینجوری که خودشون میگن این یه پلتفرمه که از Chrome's JavaScript runtime استفاده میکنه و شونصدتا ویژگی دیگه هم داره!
برای استفاده ازش باید برنامشو رو که بهش node میگن دریافت و نصب کنین (حدود 3 مگابایت نسخه 0.6.19). بعد کد سمت سرور کاملا با استفاده از زبان جاوااسکریپت نوشته میشه (عیب) بنابراین کاملا cross platform هستش (مزیت). یعنی با استفاده از جاوااسکریپت میتونین مثلا یه وب سرور بسیار سبک (مزیت) ایجاد کنین که به یکسری درخواستا پاسخ میده. حالا یکسری اومدن برای این پلتفرم ماژولهایی نوشتن مثل socket.io و nowJS که برقراری ارتباطی راحت و آسان با سرور و تبادل داده‌ها رو به عهده میگیره (مزیت).
در مقابل SignalR یه کتابخونه غیرهمزمان (Async) هستش که برا دات نت نوشته شده (مزیت)، مخصوص وب زمان واقعی چندکاربره و شونصدتا ویژگی دیگه هم نداره (مزیت) :). بنابراین سمت سرور از مزایای بیشماری برخورداره، مثل یک IDE قدرتمند (VS) (مزیت2x) و یک کتابخونه کامل (.NetFx) (مزیت2x)
درضمن یه فریمورک دیگه هم برای ASP.NET وجود داره: PokeIn که نسخه تجاری (پولی) هم داره و از WebSocket استفاده میکنه.
جایی خوندم که کاراییش خوبه در حد 10 تا 50 هزار کلاینت همزمان(^).

برای SignalR یه اپلیکیشن توسط خود نویسندگانش نوشته شده (SignalR-Crank) که برای آزمایش بار روی سرور استفاده میشه. طبق گفته خودشون تا 100k (صد هزار!) کلاینت رو تونستن بدون هیچ مشکلی راه بندازن (البته با مصرف 5 گیگ رم).
نمیدونم این تست چی بوده ولی این عدد خیلی زیاده.

در مورد بحث کارآیی خودم دارم یه تستایی انجام میدم. نمیدونم بتونم آزمایش مشابهی رو بقیه فریمورکها انجام بدم یا نه. اگه نتیجه‌ای حاصل شد اینجا میزارم.

با تشکر

نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۳/۳۰ ۱۱:۱۴

اگه برنامه چت داره درست اجرا میشه، مسیر مورد نظر برای دریافت این کد جاوااسکریپت مثلا رو سیستم من اینه:
http://localhost:16869/signalr/hubs

آره در زمان اجرا تولید میشه و بار کوچیکی روی سرور میزاره. برا همین پیشنهاد میشه تو نسخه ریلیز برنامه‌ها این کد تولیدی تو یه فایل ذخیره بشه و به جای اون مسیر ریفرنس داده بشه.

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۷/۲۹ ۱:۲

سلام
ممنونم بابت مطلب خوب کاربردیتون
می تونم بگم یکی از سئوال‌های ذهن من پاسخ داده شد و این امکان راه حل مشکل چند هفته ای من بود
تشکر

نویسنده: hossein
تاریخ: ۱۳۹۲/۰۳/۰۸ ۱۲:۲۶

با سلام میخواستم بپرسم ایا این روش (signallr) برای لود کردن مثلا 40-50 تا عکس اسلاید شو کارایی بالاتری داره یا روش lazy loading پلاگینی برای jquery اون وجود داره
با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۳/۰۸ ۱۲:۳۳

سؤالت ربطی به بحث نداره. کار SiganlR به زبان ساده فرستادن پیغام از طرف سرور به کلاینت‌ها هست. مثلا 2 تا ایمیل جدید داری. رکورد جدیدی به صفحه اضافه شده. یا برای نمونه درصد پیشرفت عملیات سمت سرور مثلا 50 درصد بوده و ارسالش به تمام کلاینت‌های متصل. هدف تبادل اطلاعات همزمان و بلادرنگ و بدون معطلی هست؛ خصوصا از طرف سرور به کلاینت.

نویسنده: ابوالفضل روشن ضمیر
تاریخ: ۱۳۹۲/۰۵/۳۰ ۱۶:۲۵

سلام

من پکیج Jquery و SignalR را در برنامه نصب کردم ... ولی وقتی روی نام پروژه راست کلیک می‌کنم تا کلاس SignalR Hub Class

را اضافه کنم وجود نداره ....

ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۳۰ ۱۷:۵۹

- آپدیت سوم VS 2012 را نصب کنید .

+ هیچ نیازی به این پیشنیازها نیست. با *VS 2010* کل مطالب SignalR قابل پیاده سازی هستند. فقط باید بتونید با نیوگت کار کنید و بسته‌های لازم رو اضافه کنید. بعد از آن کلاس هاب ، یک کلاس ساده است که امضایی مشخص داره و از کلاسی به نام Hub مشتق می‌شود و همچنین ویژگی HubName هم می‌تواند داشته باشد.

نویسنده: mostafa
تاریخ: ۱۳۹۲/۰۹/۳۰ ۲۳:۱۶

سلام
میشه این رو برای website معمولی visual studio استفاده کرد؟ یعنی غیر از وب اپلیکشن
ممنون

نویسنده: bahman
تاریخ: ۱۳۹۲/۱۰/۰۹ ۱۱:۵۴

سلام.
آیا این امکان وجود داره که SingnalR رو با RestFull سرویس در سمت سرور ترکیب کنم و از سرویس در سمت کلاینت ، داخل یک برنامه WPF استفاده کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۰۹ ۱۳:۴

نگاهی به گزینه‌های مختلف مهیای جهت میزبانی SignalR

نویسنده: عرفان
تاریخ: ۱۳۹۳/۰۳/۲۵ ۱۱:۲۲

(Clients.reciveMessage(message
خطا میده. همچین کلاسی وجود نداره حتی اگه غلط املاییش رو هم درست کنیم!

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۳/۲۵ ۱۱:۳۶

شما که به غلط املاییش دقت کردی، به انتهای بحث که نوشته شده این مباحث به روز شده‌اش در دوره SignalR سایت ارائه شدند، دقت نکردی؟ اون متد dynamic هست؛ یعنی اصلا نیازی نیست وجود خارجی داشته باشه. فقط کمی در نگارش‌های جدید، Refactoring انجام دادن، بعدش باید مشخص کنی به All یا به گروه خاصی این پیام‌ها ارسال بشه. کلیاتش یکی هست. فقط کمی تعاریف اولیه رو Refactor کردن. در بحث معرفی hubs دوره‌ای که نام برده شد این‌ها هست.

عنوان: **پیاده سازی رخداد گردانهای یک شیء در الگوی MVP**
نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۱/۰۳/۲۹ ۰:۱۴
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Design patterns, MVP, Telerik, Grid, RadGrid

یکی از ضروریات دنیای برنامه نویسی امروز، داشتن یک الگوی مناسب می‌باشد. یکی از الگوهای مناسب برای وب فرم‌ها، استفاده از الگوی MVP است.

اگر در خلال پیاده سازی، گاهی اوقات نیاز به handle کردن رخدادها را داشته باشید بدین منظور به روش زیر عمل می‌کنیم:

(توجه: شیء مورد نظر ما در این پست RadGrid از کنترلهای Telerik در نظر گرفته شده است.)

```
// ASPX page
<telerik:RadGrid ID="RadGrid1" runat="server"></telerik:RadGrid>
// Asp.Net Code Behind
protected void Page_Load(object sender, EventArgs e)
{
    GridPresenter presenter = new GridPresenter(this);
}
// view interface
public interface IGridView
{
    Telerik.Web.UI.RadGrid myGrid { get; }
}
// presenter
protected readonly IGridView _view;
public GridPresenter(IGridView view)
{
    _view = view;
    _view.myGrid.UpdateCommand += new Telerik.Web.UI.GridCommandEventHandler(onUpdateCommand);
    _view.myGrid.InsertCommand += new Telerik.Web.UI.GridCommandEventHandler(onInsertCommand);
    _view.myGrid.EditCommand += new Telerik.Web.UI.GridCommandEventHandler(onEditCommand);
}
private void onUpdateCommand(object sender, Telerik.Web.UI.GridCommandEventArgs e)
{
    // Code for updating
}
private void onInsertCommand(object sender, Telerik.Web.UI.GridCommandEventArgs e)
{
    // Code for inserting
}
private void onEditCommand(object sender, Telerik.Web.UI.GridCommandEventArgs e)
{
    // Code for editcommand
}
```

نظرات خوانندگان

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۳/۲۹ ۰:۴۹

خوبه
اما من یه قسمت رو متوجه نشدم.ببینید ما میگیم MVP (Model View Presenter).
خب حالا سوال اینجاست که در بخش Asp.net Code Behind از چه کدی استفاده میکنیم؟
یعنی کدهای View و Presenter چه موقع فراخوانی میشوند؟
از کجا اجرا میشود و به کجا ختم میشود؟
مدل ما کجاست؟
و ...

نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۱/۰۳/۲۹ ۱۱:۱

ابتدا وب فرم مورد نظر اینترفیس IGridView را پیاده سازی می‌کند و سپس در Code behind در Page_Load یک وهله از presenter با پارامتر this ایجاد می‌کنید، در نتیجه کلیه رخدادها در presenter مدیریت می‌شوند.
نکته در این است که چون هر بارگذاری صفحه نیاز به متصل کردن رخدادها به شی هستیم مجبوریم در Page_Load یک وهله از presenter ایجاد کنیم.
در این پست هدف پیاده سازی مدل نبوده است، ولی شما می‌توانید مدل خود را با توجه به الگوی MVP طراحی کنید و از آن در presenter استفاده کنید.

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۳/۲۹ ۱۴:۲۶

مرسی
مشکل قطعه کد زیر بود که جا افتاد بود و باعث شده بود که نحوه استفاده رو متوجه نشم!

```
GridPresenter presenter = new GridPresenter(this);
```

در مورد Model هم شما درست میگید.هدف این پست استفاده از Model نبود.

نویسنده: جلال
تاریخ: ۱۳۹۱/۰۷/۲۲ ۱۲:۴۱

سلام،
با این قضیه ارجاع مستقیم به یک کنترل خاص در IView می‌تونم کنار بیام، ولی با ارجاع به اون در CPresenter نه!
دلیل خاصی داشته؟ آخه معمولا برای رخدادگردانی، EventHandlerها رو در IView معرفی می‌کنن و در CPresenter استفاده می‌کنند.

البته در این مقاله ، کلا منکر استفاده از EventHandlerها در IView میشه و ترجیح میده فقط تابع Presenter رو از Code Behind فراخوانی کنه!

نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۱/۰۸/۱۶ ۱۸:۴۸

یکی از دلایل استفاده از این روش، تمیز نگه داشته داشتن ویو (با توسعه برنامه پارامترهای لازم برای 4عمل اصلی اضافه می‌شد

که نگهداری پارامترهای رخداد گردانها پیچیده و دشوار می‌شد) بود، زمانی که داشتم این پیاده سازی رو در پروژه انجام می‌دادم به این نتیجه رسیده بودم بهتر است کلیه عملیات‌های افزودن، حذف، ویرایش و حذف شی تلریک گرید ویو رو در presenter انجام بدم چون قرار بود یک سری عملیات منطقی بر روی 4عمل اصلی انجام بدم.

عنوان: **مدیریت محل اعمال Google analytics در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۲۹ ۱۹:۵۸
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC

ذخیره سازی اطلاعات بازدیدهای کاربران، در طول زمان حجم بالایی از بانک اطلاعاتی را به خود اختصاص خواهد داد؛ به علاوه کند شدن کوئری‌های مرتبط با آن، به همراه مصرف بالای منابع سیستم. به همین جهت اکثر سایت‌ها از Google analytics برای مدیریت جمع آوری بازدیدهای کاربران خود استفاده می‌کنند و این ابزار واقعا عالی و حرفه‌ای طراحی شده و پیاده سازی همانند آن شاید در حد یک پروژه‌ی چندساله باشد.
اضافه کردن Google analytics به یک سایت، بسیار ساده است. در آن ثبت نام می‌کنید؛ سپس آدرس دومین خود را وارد کرده و یک قطعه کد جاوا اسکریپتی را دریافت خواهید کرد که باید به انتهای تمام صفحات سایت خود اضافه نمائید و ... همین.
اضافه کردن این کد در ASP.NET MVC می‌تواند در فایل layout یا همان master page سایت انجام شود تا به صورت خودکار به تمام صفحات اعمال گردد.

مشکل!
من نمی‌خواهم که صفحات غیرعمومی سایت نیز دارای کدهای Google analytics باشند و بی‌جهت Google به این‌جاها نیز سرکشی زاید کند! چکار باید کرد؟
احتمالا عنوان می‌کنید که باید یک if و else به همراه آرایه‌ای از نام‌ها و آدرس‌های صفحات غیرعمومی سایت تهیه کرد و بر این اساس کدهای Google analytics را در master page درج کرد یا خیر.
بله. این روش کار می‌کنه ولی بهینه نیست و همچنین نگهداری آن در طول زمان مشکل است. سایت توسعه خواهد یافت، صفحات غیرعمومی بیشتر خواهند شد و ممکن است در این بین فراموش شود که کدهای مرتبط به روز شوند.

روش بهتر :
آیا می‌توان در یک View مشخص کرد که فیلتر Authorize در اکشن متد متناظری که آن‌را رندر کرده است بکار گرفته شده است یا خیر؟
صفحات غیرعمومی سایت در ASP.NET MVC با فیلتر Authorize محافظت می‌شوند. این فیلتر را می‌توان به کل یک کنترلر اعمال کرد تا به تمام اکشن متدهای آن اعمال شود؛ یا فقط به یک اکشن متد خاص که Viewایی خاص را رندر می‌کند.

نحوه پیاده سازی تشخیص وجود فیلتر Authorize را در یک View رندر شده، در متد کمکی زیر می‌توان مشاهده کرد:

```
@helper IncludeGoogleAnalytics(WebViewPage page)
    {
        var controller = page.ViewContext.Controller;
        var controllerHasAuthorizeAttribute =
controller.GetType().GetCustomAttributes(typeof(AuthorizeAttribute), true).Any();
        var currentActionName =
page.ViewContext.Controller.ValueProvider.GetValue("action").RawValue.ToString();
        var actionHasAuthorizeAttribute = controller.GetType().GetMethods()
                                                    .Where(x => x.Name == currentActionName &&

x.GetCustomAttributes(typeof(AuthorizeAttribute), true).Any())
                                                    .Any();
        if (!controllerHasAuthorizeAttribute && !actionHasAuthorizeAttribute)
        {
            string trackingId = ConfigurationManager.AppSettings["GoogleAnalyticsID"];
    <script type="text/javascript">
        var _gaq = _gaq || [];
        _gaq.push(['_setAccount', '@trackingId']);
        _gaq.push(['_trackPageview']);
        (function () {
            var ga = document.createElement('script'); ga.type = 'text/javascript'; ga.async = true;
            ga.src = ('https:' == document.location.protocol ? 'https://ssl' : 'http://www') +
'.google-analytics.com/ga.js';
            var s = document.getElementsByTagName('script')[0]; s.parentNode.insertBefore(ga, s);
        })();
    </script>
        }
```

توضیحات:

این متد در فایلی به نام HtmlUtils قرار گرفته در پوشه app_code تعریف شده است و بکارگیری آن در یک فایل master page به نحو زیر خواهد بود:

```
@HtmlUtils.IncludeGoogleAnalytics(this)
```

در این متد به کمک خاصیت page.ViewContext.Controller می‌توان به کنترلری که در حال رندر کردن View جاری است دسترسی یافت. اکنون که به کنترلر دسترسی داریم، به کمک Reflection، ویژگی‌ها یا Attributes آن‌را یافته و بررسی می‌کنیم که آیا دارای AuthorizeAttribute است یا خیر. بر این اساس می‌توان تصمیم گرفت که آیا View در حال نمایش عمومی است یا خصوصی. اگر عمومی بود، کدهای اسکریپتی Google analytics به صورت خودکار به صفحه تزریق می‌شوند.

همچنین در اینجا فرض بر این است که Id منتسب به دومین جاری در کلیدی به نام GoogleAnalyticsID در فایل کانفیگ برنامه در قسمت app settings آن تعریف شده است.

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۱:۴

خیلی خوشحالم که مجددا شروع به نوشتن کردید.
یه ایراد کوچیک درباره سایت جدید میخوام بگیرم. در سایت جدید دیگه نمیشه به راحتی به مطالب قبلی دسترسی داشت. در سایت قبل، گزینه‌های ارسال‌های قبلی و ارسال‌های جدیدتر وجود داشت که میشد به راحتی به ترتیب ارسال پست، سایت رو مرور کرد.
در صورتی که این امکان گذاشته بشه، از مطالب بهتر میشه استفاده کرد

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۱:۱۰

ممنون. در این سایت هم هست. در پایین صفحه اصلی سایت سه لینک هست به نام‌های
آخرین نظرات
آخرین مطالب ارسالی
آخرین اشتراک‌ها

که همین کار را انجام می‌دهند.

نویسنده: Mohammad
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۱:۲۰

ازتون تشکر میکنم.فقط احساس میکنم یه باگی تو این سیستم جدیدتون مربوط به تعداد نظرات هستش.به عنوان مثال همین مطلب تو صفحه اصلی سایت 0 نظر داره ولی داخل خود پست که میشی میبنی 2 تا نظر داره.ولی بعد از چند دقیقه درست میشه.یا شایدم کش میشن.

نویسنده: Ara
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۱:۳۴

سلام

تبریک میگم وحید جان !
خیلی خوشحالم باز میتونیم از مطالب خوبتون استفاده کنیم !

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۲:۳۶

بله. از کش استفاده شده.

نویسنده: مجتبی حسینی
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۲:۵۸

سلام مهندس جان

از اینکه میتوونم اولین کار روزانه خودم را چک کردن سایتت قرار بدم خیلی خوشحالم

ایکاش مانند گذشته فایل chm سایت رو نیز در اختیار مراجعه کننده‌ها برای استفاده آفلاین قرار میدادی

نکته دیگر آنکه تعداد برچسب‌های سایت خیلی کم است و این دسته بندی یافتن مطالب رو سخت میکنه

و دیگر آنکه چند کاربره شدن سایت دامنه مطالب ارسالی رو بازتر کرده و ممکنه اولا سطح مطالب افت کنه و یا زیادی پراکنده بشه اگر برای نویسندگان سایت راهبری هایی انجام بشه شاید مطالب هماهنگ‌تر و مفیدتر و مورد نیازتر بشن متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۳:۳

سلام؛ ممنون. برچسب‌های سایت کم نیست. لیست کامل آن‌ها در اینجا قابل مشاهده است:
http://www.dotnettips.info/search/label

نویسنده: مومن
تاریخ: ۱۳۹۱/۰۳/۳۰ ۸:۵۳

مبارک است
هر روز بهتر از دیروز!

نویسنده: مهدی
تاریخ: ۱۳۹۱/۰۳/۳۱ ۱۲:۳۹

واقعا عالی
عجب انرژیکی آقا وحید شما....

نویسنده: مجتبی حسینی
تاریخ: ۱۳۹۱/۰۳/۳۱ ۲۱:۲

جناب مهندس عزیز
مثل اینکه چند روزی هست که google analytic نیز ایران رو تحریم کرده چاره چیه؟جایگزین داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۳/۳۱ ۲۱:۲۸

مثل بقیه سرویس‌های آن با V.P.N کار می‌کنه. شما فقط نیاز به یک tracking id دارید. ثبت سایت و دریافت آن کار جمعا یک تا دو دقیقه است.

نویسنده: آریا
تاریخ: ۱۳۹۱/۰۴/۰۱ ۱۲:۴

ممنون بابت مطلب خوبتون

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۲/۰۵/۰۱ ۱۸:۱۶

با سلام. امکان دارد در مورد نحوه استخراج اطلاعات از این سرویس و نمایش در سایت کمی راهنمایی کنید؟ باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۰۱ ۱۸:۲۶

Google APIs Client Library for .NET

DotNET Client Library Developer's Guide

نویسنده: ایاک
تاریخ: ۱۳۹۳/۰۲/۲۳ ۱۱:۱۹

با تشکر.

من دقیقاً روش بالا را استفاده کرده ام. در کلاینت خوب جواب می‌دهد ولی وقتی بر روی سرور آپلود میکنم ، نتیجه ای حاصل نمی‌شود.

```
string username = ConfigurationManager.AppSettings["GoogleAnalyticsUsername"];
        string password = ConfigurationManager.AppSettings["GoogleAnalyticsPassword"];
        string profileId = ConfigurationManager.AppSettings["GoogleAnalyticsProfileId"];
        string yesterdayDate = DateTime.Today.AddDays(-1).ToString("yyyy-MM-dd");
var service = new AnalyticsService("WebSiteAnalytics");

                service.setUserCredentials(username, password);
                var dataQuery = new DataQuery()
                {
                    Ids = profileId,
                    Metrics = "ga:pageviews",
                    Sort = "ga:pageviews",
                };
                DataFeed dataFeed = null;

                //--- total visit
                dataQuery.GAStartDate = new DateTime(2012, 3, 1).ToString("yyyy-MM-dd");
                dataQuery.GAEndDate = DateTime.Now.ToString("yyyy-MM-dd");
                dataFeed = service.Query(dataQuery);
                var totalEntry = (dataFeed.Entries == null || dataFeed.Entries.Count == 0) ? "0" :
                    (dataFeed.Entries[0] as DataEntry).Metrics[0].Value;

                //--- yesterday visit
                dataQuery.GAStartDate = DateTime.Now.AddDays(-1).ToString("yyyy-MM-dd");
                dataQuery.GAEndDate = DateTime.Now.AddDays(-1).ToString("yyyy-MM-dd");
                dataFeed = service.Query(dataQuery);
                var yesterdayEntry = (dataFeed.Entries == null || dataFeed.Entries.Count == 0) ? "0" :
                    (dataFeed.Entries[0] as DataEntry).Metrics[0].Value;

                //--- today visit
                dataQuery.GAStartDate = DateTime.Now.ToString("yyyy-MM-dd");
                dataQuery.GAEndDate = DateTime.Now.ToString("yyyy-MM-dd");
                dataFeed = service.Query(dataQuery);
                var todayEntry = (dataFeed.Entries == null || dataFeed.Entries.Count == 0) ? "0" :
                    (dataFeed.Entries[0] as DataEntry).Metrics[0].Value;

                //--- last week visit
                dataQuery.GAStartDate = DateTime.Now.AddDays(-7).ToString("yyyy-MM-dd");
                dataQuery.GAEndDate = DateTime.Now.ToString("yyyy-MM-dd");
                dataFeed = service.Query(dataQuery);
                var lastWeekEntry = (dataFeed.Entries == null || dataFeed.Entries.Count == 0) ? "0" :
                    (dataFeed.Entries[0] as DataEntry).Metrics[0].Value;

                //--- last month visit
                dataQuery.GAStartDate = DateTime.Now.AddDays(-30).ToString("yyyy-MM-dd");
                dataQuery.GAEndDate = DateTime.Now.ToString("yyyy-MM-dd");
                dataFeed = service.Query(dataQuery);
                var lastMonthEntry = (dataFeed.Entries == null || dataFeed.Entries.Count == 0) ? "0" :
                    (dataFeed.Entries[0] as DataEntry).Metrics[0].Value;
```

نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۱:۲۰

- این روش منسوخ شده‌است و مربوط است به پروتکل‌های قدیمی گوگل.
- کار کردن با آن امن نیست چون باید اطلاعات حساس اکانت گوگل خود را در فایل کانفیگ برنامه قرار دهید.
- برای کار با این نوع پروتکل‌ها، در بسیاری از اوقات گوگل نیاز دارد تا شما توسط مرورگر سیستم جاری (در اینجا سرور وب)، یکبار به اکانت خود لاگین کنید.
- ga:pageviews با visits ارتباطی ندارد. فقط به این معنا است که صفحات سایت شما چندبار مشاهده شده‌اند و نه به معنای اینکه چندنفر آن‌هارا مشاهده کرده‌اند.

روش بهتر در اینجا

نویسنده: امیر
تاریخ: ۱۳۹۳/۰۳/۰۷ ۹:۳۱

با سلام و درود؛ یک سوال ؟ برای بدست آوردن کاربران Unique از session استفاده میکنیم و اما گوگل با استفاده از کدهای جاواسکریپت چطور میتونه کاربران unique را بدست بیاره ؟!
و یک سوال دیگه اگه راهنمایی کنید ممنون میشم.
من یک آمارگیر برای سایتم نوشتم که یک مشکل با کاربرانی داره که با موبایل یا تبلت از سایت دیدن میکنند
و با هر بار بازدید چند بار session اجرا میشه ! یعنی با یک بازدید به جای اینکه یک عدد به کاربران unique اضافه بشه چند عدد اضافه میشه ! که مشکل برای کاربرانی که از طریق ویندوز سایت را مشاهده میکنند وجود نداره.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۷ ۹:۳۸

در Google Analytics (نه راه حل‌های شخصی):
تعداد بار مراجعه کاربران: «sessions»
کاربران منحصربفرد مراجعه کننده به سایت: «users»
- تفاوت بین ga:sessions و ga:users در Google Analytics
- آنالیز محتوای کوکی‌ها utma و utmz + بررسی IP کاربر و تهیه یک چکیده از user-agent مرورگر و خیلی از مسایل ریز دیگر او برای تشخیص منحصربفرد بودن کاربر و نه فقط اینکه سشنی شروع شده، بنابراین کاربر جدید است.

عنوان: SignalR - قسمت دوم
نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۳/۲۹ ۲۱:۳۱
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, SignalR

در قسمت اول بحث‌های مقدماتی درباره وب زمان واقعی (real time web) و معرفی کتابخونه SignalR به همراه یک مثال ساده رو با هم دیدیم. در ادامه به جزئیات ریزی از کتابخونه SignalR که توسط آقای David Fowler توسعه داده میشه میپردازم.

همونطور که قبلا هم اشاره شد قلب این کتابخونه در سمت سرور دو کلاس پایه PersistentConnection و Hub هستن که اولی سطحی پایینتر داره یعنی به تنظیمات و کدنویسی (بسیار) بیشتری برای پیاده‌سازی نیاز داره اما در عوض امکانات سطح پایینتری هم در اختیار برنامه نویس قرار میده که در برخی موارد موردنیاز هستن. در مورد بخشهای مختلف این دو کلاس و نحوه پیاده‌سازی هر دو کلاس فوق، تو آدرسی که قبلا اشاره کردم (آدرس پروژه متن باز این کتابخونه در github) راهنمایی‌های نسبتا مفصلی ارائه شده و نیازی به تکرار این مطالب در اینجا نیست. پیشنهاد میکنم که این مطالب رو با دقت مطالعه کنین.

SignalR به صورت توکار از 4 روشی که در قسمت قبل به اون اشاره شد برای برقراری ارتباط استفاده میکنه. WebSocket که به بسترهای جدیدی نیاز داره. Server-sent Events که تنها در مرورگرهایی که پشتیبانی کاملی از html5 دارند قابل استفاده است (بنابراین ie9 نمیتونه از این روش استفاده کنه). forever frame داده‌ها رو به‌صورت chunked (بخشی از استاندارد http 1.1) دریافت میکنه و روش آخر که Long-polling نام داره از هموش روش قدیمی اِیجکس (Ajax) بهره میبره و با استفاده از آبجکت معروف XmlHttpRequest کار ارتباط و تبادل داده‌ها رو انجام میده.

در اینجا برای بررسی این ارتباطات ابتدا برنامه چت ساده قسمت قبل رو در اینترنت اکسپلورر اجرا کنین و با استفاده developer tool (کلید F12) به درخواستهای مختلف ارسال شده به سرور نگاه کنین. پس از اجرای برنامه و قبل ارسال هرگونه داده به سرور در تب Network این ابزار چیزی شبیه به شکل زیر مشاهده خواهید کرد (البته باید قبل از ورود به صفحه برنامه چت روی دکمه Start capturing کلیک کنین):

با توجه به تصویر بالا SignalR در شرایط موجود بهترین روش برای برقراری ارتباط با سرور رو forever frame تشخیص داده و مشاهده میشه که این ارتباط دائمیه و فعلا نتیجه‌ای از سمت سرور دریافت نکرده و ارتباط کاملا زنده است. البته اگر در این ابزار درباره درخواستهای ارسالی به سرور بیشتر جستجو بکنین اطلاعات بیشتری نصیبتون میشه که آوردنش اینجا بحث رو طولانی میکنه.

حالا برنامه رو در یه مرورگر دیگه که از html5 پشتیبانی میکنه اجرا کنین. مثلا نتیجه در گوگل کروم و ابزار توسعه اون به شکل زیره:

index.htm	GET	200 OK	text/html	Other	1.10KB 1.02KB	5ms 5ms
jquery-1.6.4.min.js /Scripts	GET	200 OK	application/x-javascript	index.htm:5 Parser	40.40KB 89.52KB	166ms 166ms
jquery.signalR-0.5.1.min.js /Scripts	GET	200 OK	application/x-javascript	index.htm:5 Parser	5.84KB 12.92KB	81ms 81ms
negotiate /SimpleChat	GET	200 OK	application/json	jquery-1.6.4.min.js:4 Script	557B 147B	12ms 2ms
connect /SimpleChat	GET	200 OK	text/event-stream	jquery.signalR-0.5.1.min.js Script	477B 29B	33.62s 13ms

http://localhost:16869/SimpleChat/connect?transport=serverSentEvents&connectionId=921a3946-b983-4b3a-8d23-96aae5e7f53e

همونطور که میبینین در اینجا روش استفاده شده Server Sent Events هست.

در فایرفاکس هم با استفاده از ابزار محبوب firebug نتیجه مشابه کروم بدست میاد:

Console HTML CSS Script DOM Net

Clear Persist All HTML CSS JS XHR Images Flash Media

URL	Status	Domain	Size	Remote IP	Timeline
Net panel activated. Any requests while the net panel is inactive are not shown.					
GET index.htm	200 OK	localhost:16869	658 B	::1:16869	45ms
GET jquery-1.6.4.min.js	200 OK	localhost:16869	39.9 KB	::1:16869	
GET jquery.signalR-0.5.1.min.js	200 OK	localhost:16869	5.3 KB	::1:16869	
GET negotiate?_=1340033212593	200 OK	localhost:16869	147 B	::1:16869	
http://localhost:16869/SimpleChat/connect?transport=serverSentEvents&connectionId=4d8b307a-a94f-44a2-a471-5685ed7e7eff					
5 requests			46 KB		

البته اگر علاقه زیادی به کندوکاو در جزئیات این درخواستها دارین (مثل خود من) چیزی بهتر از fiddler2 پیدا نمیشه. میتونین پس از ارسال یک متن دوباره این درخواستها رو مورد بررسی قرار بدین و ببینین که چیجوری کانالهای ارتباط پس از ارسال و دریافت دیتا قطع و برقرار میشه.

این نکته رو هم باید یادآور بشم که هرچند که این کتابخونه بهترین روش رو میتونه انتخاب کنه اما به برنامه نویس امکان تعیین صریح روش ارتباط رو هم میده. اگر به راهنماهای این کتابخونه سر بزنین میبینین که امکانات زیادی بهش اضافه شده و امکانات زیادی هم در آینده به اون اضافه میشه. امکاناتی از قبیل ارسال داده‌ها به یک کلاینت خاص و یا به گروهی خاص از کلاینتها، خصوصی‌سازی آدرس سرور و همچنین پشتیبانی از Cross Domain در آخرین نسخه، امکان استفاده از Reactive Extension (بلاگ)، بحث Self Hosting که امکان خیلی جالبیه و میتونه خیلی جاها یه عنوان یک راه‌حل سبک و سریع به کار بیاد، قابلیت فوق العاده در بایندینگ داده‌ها در سمت سرور و مخصوصا کلاینت، امکان تشخیص برقراری یا قطع ارتباط کلاینتها در سمت سرور، استفاده از امکانات این کتابخونه برای برقراری ارتباط با کلاینتها در خارج از فضای کلاسهای مشتق شده از دو کلاس پایه (Hub و PersistentConnection) و چند مورد دیگه تا نسخه جاری اضافه شدند.

درحال حاضر دارم روی یه برنامه چت با امکانات بیشتر کار میکنم که پس از آماده شدن ارائه میدمش. یکی از پروژه‌های متن بازی که با استفاده از این کتابخونه توسعه داده شده jabbr.net است. یه اتاق گفتگوی کامل با امکانات جالبه که میتونین به اون هم یه سری بزنین.

در آخر هم یه لینک جالب برای مطالعه معرفی میکنم: Highest voted Signalr Questions - stackoverflow

عنوان: **استفاده از HttpGet در ASP.NET MVC، آری یا خیر؟!**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۱ ۱۹:۵۷
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC

در ASP.NET MVC به کمک یک سری فیلتر می‌توان مشخص کرد که یک اکشن متد تنها به درخواست‌هایی از نوع Get پاسخ دهد، دیگری به درخواست‌هایی از نوع Post و الی آخر. عادت متداول من هم برای نمایش معمولی صفحات، استفاده از حالت HttpGet است که در شبکه‌های داخلی بدون مشکل کار می‌کند چون Bot ایی در این شبکه‌ها وجود ندارد و اگر باشد احتمالا یک ویروس است!

```
[HttpGet]
public ActionResult Index()
{
    return View();
}
```

اما روی اینترنت وضع فرق می‌کند. برای مثال یک کنترلر Feed ممکن است درخواست‌هایی از نوع Head نیز داشته باشد و یا بعضی از Botها بجای Get جهت دریافت اطلاعات فید، از Post هم استفاده می‌کنند! (مانند فیدبرنر گوگل مطابق لاگ‌های برنامه!) و در این حالت اگر متد رندر کردن یک فید را به HttpGet تنظیم کرده باشیم، مدام با خطای A public action method xyz was not found on controller abc در لاگ‌های خطای برنامه مواجه خواهیم شد.

بنابراین بهتر است که در سایت‌های عمومی، جهت ارائه اطلاعات و صفحات عمومی، از هیچ نوع فیلتری برای محدود کردن یک اکشن متد که کارش صرفا رندر کردن اطلاعات است، استفاده نکنیم. به این ترتیب اگر کسی توسط درخواستی از نوع Head فقط قصد بررسی تغییرات صورت گرفته را داشت و نه دریافت کل فید، بی‌جهت آن‌را محدود نکرده باشیم.

عنوان: **SignalR - قسمت سوم**
نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۴/۰۲ ۱۶:۳۱
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, SignalR

در قسمت قبل درباره روشهای برقراری ارتباط با سرور در کتابخونه SignalR کمی بحث شد. برای ادامه بهتره که به برنامه چت ساده ای که تو این مدت کمی تکمیلش کردم یه نگاهی بندازین: SimpleChat.rar

لطف کنین این برنامه رو دانلود و اجرا کنین تا کمی با جزئیات این کتابخونه بیشتر آشنا بشین. این برنامه قدم به قدم نوشته شده و حاوی نسخه‌های مختلفی از برنامه چت هست که هر کدوم تو یه فایل html استفاده شده. نسخه آخر شامل عملیات لاگین، چت گروهی، چت خصوصی و امکان تغییر گروه است. درضمن این برنامه کمی با عجله نوشته شده پس اگه باگ یا موردی مشاهده کردین و یا پیشنهادی دارین اشاره کنین تا بقیه هم استفاده کنن.
حالا به یه نکته در مورد آغاز برقراری ارتباط کلاینت با سرور اشاره میکنم. قبل از برقراری این ارتباط (که در قسمت قبل توضیحاتی در این مورد داده شده) برنامه کلاینت یک درخواست به سمت سرور ارسال میکنه. به تصویر زیر دقت کنین:

به این درخواست اولیه در کتابخونه SignalR همونطور که مشاهده میشه مذاکره (negotiate) گفته میشه. برقراری ارتباط این درخواست اولیه هم توسط XHR انجام میشه. نتیجه این مذاکره در تب Response Body قابل مشاهده است:

می‌بینین که علاوه بر آی دی ارتباط که یک guid است (تلفظ مرسومش «گوئِد» هستش) امکان برقراری ارتباط از طریق روش WebSocket رو هم از طرف سرور مشخص میکنه که با توجه به استفاده من از ویندوز 7 امکانش وجود نداره. یعنی اگر مثلا شما از ویندوز 8 و IIS 8 استفاده کنین مقدار TryWebSocket برابر true بوده و همچنین پارامتر WebSocketServerUrl نال نخواهد بود. البته این پارامتر تنها مربوط به سروره و برنامه کلاینت مورد استفاده (در اینجا یک مرورگر) هم باید توانایی استفاده از این روش رو داشته باشه. پس از اتمام این مذاکره ارتباط اصلی برقرار میشه.

یکی از قابلیتهای خوب این کتابخونه ارسال خطاهای رخ داده در سمت سرور به کلاینت هست. در تصویر زیر بدنه یک نمونه از

پاسخهای سرور که نمایش دهنده خطای رخ داده در سمت سرور هست رو نشون داده شده:

برای راحتی دوستان در استفاده از راهنماهای این کتابخونه، یه مقدار کار روشون انجام دادم و با خلاصه کردن محتوای اونا (و کاهش حجم 95 درصدی!) برای استفاده آفلاین آماده کردم:

SignarR github docs.rar

اگه فرصتی پیش بیاد و دوستان هم علاقه داشته باشن در قسمت بعدی برنامه چت رو بیشتر با هم بررسی میکنیم.

در ادامه قصد دارم تا روی بازدهی و کارایی این کتابخونه تو بار زیاد یه بررسی هایی انجام بدم (البته اگه وقت کنم چون راه اندازی یه محیط تست برا این جور کتابخونه‌ها چندان آسون نیست) و مطلبی هم در مورد نحوه راه اندازی تست بار ارائه بدم.

نظرات خوانندگان

نویسنده: علی
تاریخ: ۱۳۹۱/۰۴/۰۲ ۱۷:۱۰

مرسی آقای یوسف نژاد, امیدوارم این بحثتون ادامه پیدا کنه

نویسنده: باربد
تاریخ: ۱۳۹۱/۰۴/۰۳ ۸:۱۱

خیلی عالی بود ...
راستش من هم تو فاصله این دو تا مقاله باهاش کار کردم.
چیز خیلی فوق العاده ایه . البته اگر لطف کنید و دو تا ساختار اصلی این مبحث را که Hubs و Persistent Connection را به تفکیک و با جزئیات توضیح بدید ، خیلی خیلی عالی خواهد بود. (فقط یه سوال؟ تو نسخه بعد دات نت ، جزء ساختار اصلی فریم ورک خواهد بود؟)
باز هم متشکرم و موفق باشید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۳ ۹:۱۶

WebSockets Protocol در ASP.NET 4.5 پشتیبانی می‌شود: (^)

نویسنده: باربد
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۰:۳۵

سپاس :)

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۵:۵۹

سلام ... عالی بود ... خیلی استفاده بردیم ... برنامه ای که نوشتینم خیلی مفیده ;) ... به نظرم اگه در مورد برنامه ای که نوشتین تو قسمت‌های بعدی سرمایه گذاری کنین خیلی خوب میشه

نویسنده: بنیامین
تاریخ: ۱۳۹۱/۱۰/۱۱ ۱۸:۸

با سلام و خسته نباشید
واقعا ممنون از مطالبی که گذاشتین. خیلی از مشکلات من رو حل کرد. خواهش می‌کنم بحث رو ادامه بدید.
با تشکر

نویسنده: MSN
تاریخ: ۱۳۹۱/۱۰/۲۲ ۱۷:۲۸

خیلی خوب و کاربردی , واقعا خسته نباشید.

سئوال اینجاست , زمانی که پیامی رو ارسال میکنیم دستور زیر رو اجرا میکنیم که این پیام واسه بقیه اعضا هم ارسال بشه

```
public void SendMessage(string message, string id)
{
      Clients.reciveMessage(message, Users[id]);
}
```

زمانی که مثلا توی یک اتاق 10 نفر در یک زمان پیامی رو ارسال میکنند , واسه همه اعضا 10 تا Response جدا میاد ؟ اگه اینطور هست چطور میشه این مورد رو بهینه کرد ؟

نویسنده: میثم ق
تاریخ: ۱۳۹۲/۰۲/۱۵ ۱:۱۱

سلام

خیلی خوب بود.واقعا دستت درد نکنه

امیدوارم این بحث و ادامه بدبن

خیلی ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۵ ۹:۵۸

جهت اطلاع؛ دوره SignalR سایت این مباحث رو بر اساس نگارش جدیدترش تکمیل کرده.

نویسنده: یوسف نژاد
تاریخ: ۱۳۹۲/۰۲/۱۵ ۱۶:۵۷

با توجه به ریلیز شدن نسخه نهایی این کتابخونه با عنوان ASP.NET SignalR و تغییرات بوجود اومده، در حال تهیه مطلبی جدید و نیز بروزرسانی برنامه چت ارائه شده و تهیه چند برنامه جدید مفید دیگه هستم که امیدوارم به زودی آماده بشن.

عنوان: **OpenID چیست؟**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۴/۰۲ ۱۹:۲۳
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, C#, others, OpenID

چند وقتی میشه که دنبال روش‌های OpenID هستم که ببینم چطوری کار می‌کنند، خودم هم تازه شروع کردم خوب قبل از هر چیزی اول ببینیم مفهوم OpenID چی هست؟ و کم کم جلو میریم و مثال هایی معرفی می‌کنیم.

OpenID به شما اجازه می‌دهد با استفاده از اکانت (نام کاربری) که در یک سایت دارید بتوانید به سایت‌های متفاوتی وارد شوید (لاگین کنید) بدون این که نیاز به ثبت نام دوباره در آن سایت‌ها داشته باشید.

نمونه بارز آن می‌توان به سایت هایی مانند StackOverflow اشاره کرد.

OpenID به شما اجازه می‌دهد شما در یک نام کاربری که برای خود ایجاد کرده اید اطلاعاتی را که دارید با دیگر وبسایت‌ها به اشتراک بگذارید.
با OpenID کلمه عبور شما فقط توسط سرویس دهنده گرفته میشود و سرویس دهنده هویت شما را برای بازید از سایت دیگر تایید می‌کند.از طرف دیگر سرویس دهنده شما تا شما اجازه ندهید هیچ وب سایتی کلمه عبور شما را به هیچ وب سایتی نمی‌بیند. بنابراین نیازی نیست در مورد کلمه عبور خود نگرانی به خود راه دهید.

OpenID به سرعت در حال گسترش بروی وب استف در حال حاضر بیش از یک میلیارد نام کاربری (اکانت) وجود دارد و بیش از 50000 سایت OpenID را پذیرفته و با آن کار می‌کنند. چندین سازمان بزرگ موضوع OpenID را پذیرفته اند، سازمان هایی مانند Google, FaceBook,Yahoo!,Microsoft,AOL,MySpace,Sears, Universal Music Group, France Telecom, Novell, Sun, Telecom Italia
و بسیاری از سازمان‌های دیگر.

در کل می‌توان گفت ما در یک سایت خاص مانند یاهو ، گوگل، و... یک نام کاربری خواهیم داشت سپس برای ارتباط سایت مقصد (مثلا همین سایت) با نام کاربری ما در گوگل به این صورت عمل می‌شود که ابتدا از طریق این سایت ما به سایت گوگل هدایت می‌شویم در آنجا از ما یک تاییدیه جهت استفاده از سرویس OpenID از کاربر میگیرد. در صورت تایید کاربر سایت گوگل از این لحظه جهت احراز هویت کاربر برای ورود به سایت مقصد استفاده می‌کند .
زمانی که کاربر می‌خواهد به این سایت لاگین کند سایت نام کاربری و کلمه عبور او را (در صورتی که قبلا به گوگل لاگین کرده باشد نیازی نیست و وارد سایت می‌شود) به گوگل می‌فرستد و پس از تایید هویت در صورت صحیح بودن اجازه می‌دهد کاربر به آسانی وارد سایت مقصد شود.
تصویر زیر نمایانگر این روش می‌باشد.

در پست‌های آینده مثال‌ها و کامپوننت‌های سورس بازی جهت این کار به شما معرفی خواهیم کرد. جهت مطالعات بیشتر می‌توانید به این لینک‌ها مراجعه کنین (^ و ^ و ^).

نظرات خوانندگان

نویسنده: ashi mashi
تاریخ: ۱۳۹۱/۰۴/۱۸ ۱۱:۳۵

سلام،
این چیزی که شما میگید ، آیا مشابه single sign on هستش ؟
همون کار رو انجام میده ؟
آیا با Active directory قابل اتصال هست یا نه ؟

با تشکر....

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۴/۱۸ ۱۱:۳۸

فکر نمی‌کنم بشه
چون Active Directory با Domain تنظیم میشه
اما این حالت احراز هویت روی فرم تنظیم میشه
برای SSO مطلب توی همین سایت هست در واقع توی اون روش ما با یک اکانت که توی یک سایت داریم از سایت‌های دیگه روی همون سرویس استفاده میکنیم و میشه گفت حالت اختصاصی داره
اما توی این حالت شما با هر سایتی که این قابلیت داشته باشه میتونی کار کنی

نویسنده: Hamid NCH
تاریخ: ۱۳۹۲/۰۹/۱۴ ۱۲:۳۱

نقطه شروع یادگیری من در برنامه نویسی وب، پروژه رایگان شما جناب آقای استاد صابر فتح الهی بود.
بعد از اون هم که با سایت آقای دلشاد آشنا شدم و اینک سایت فوق العاده پربار جاری به همت تمام عزیزان.
و باز هم الان دارم از شما آقای فتح الهی چیز یاد میگیرم.

عنوان: **اثر وجود سشن بر پردازش موازی در ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۰:۱۳
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Parallel Extensions

در حین جستجوی مطلبی در فوروم‌های ASP.NET مطلبی رو از یکی از اعضای تیم ASP.NET دیدم که خلاصه‌اش به این شرح است:

اگر چندین درخواست با یک Session ID به موتور پردازشی ASP.NET وارد شوند و حداقل یکی از آن‌ها به صورت سشنی خواندنی+نوشتنی علامت گذاری شده باشد، اینگونه درخواست‌ها serialized می‌شود. به این معنا که این نوع درخواست‌ها در صف قرار داده شده و به ترتیب رسیدن، اجرا می‌شوند. این کار هم از این جهت لازم است که چندین درخواست یا به عبارتی چندین ترد همزمان، سبب انجام تغییرات ناخواسته‌ای در مقادیر سشن مورد استفاده نشوند.
و اگر چندین درخواست با یک Session ID به موتور پردازشی ASP.NET وارد شوند و تمام آن‌ها سشن «فقط خواندنی» باشند یا در آن‌ها سشن «غیرفعال» شده باشد، این درخواست‌ها به صورت موازی پردازش خواهند شد.

بنابراین اگر در برنامه خود از سشن استفاده نمی‌کنید، سطر زیر را به وب کانفیگ برنامه اضافه نمائید تا حداکثر استفاده را از پردازش موازی خودکار کرده باشید:

```
<httpModules>
   <remove name="Session"/>
</httpModules>
```

```
<sessionState mode="Off" />
```

و یا در ASP.NET MVC می‌توان کنترل ریزتری را به نحو زیر نیز اعمال کرد:

```
[SessionState(SessionStateBehavior.Disabled)]
public class AjaxTestController : Controller
{
   //...
}
```

در اینجا تنها حالت سشن مرتبط با اکشن متدی (و یا تمام اکشن متدهای یک کنترلر) که برای درخواست‌های Ajax طراحی شده، خاموش گردیده است تا درخواست‌های غیرهمزمان Ajaxایی بتوانند حداکثر استفاده را از پردازش موازی خودکار کنند.
نکته دیگر اینکه اگر نیاز به استفاده از سشن در یک اکشن متد وجود داشته باشد می‌توان از فیلتر سشن‌های فقط خواندنی استفاده کرد تا باز هم به همان پردازش موازی پشت صحنه برسیم:

```
[SessionState(SessionStateBehavior.ReadOnly)]
```

نظرات خوانندگان

نویسنده: بهاره قدمی
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۱:۵۰

با سلام و ممنون از سایت خیلی خوب شما
سوالم اینه که آیا اگر بخوایم در یک برنامه asp.net از سشن استفاده کنیم و ان را در کل سایت readOnly اعلام کنیم تا بتونیم پردازش موازی داشته باشیم راهی هست که در وب کانفیگ این تنظیم را انجام دهیم ؟یا حتی هنگام تعریف اولیه سشن پس از مقدار دهی آنرا readOnly کنیم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۲:۳

در وب کانفیگ خیر. تعریف‌های انجام شده در وب کانفیگ سراسری است. به این معنا که اگر می‌شد سشن را در وب کانفیگ readonly کرد، بعد کجا باید در آن مقداری را درج می‌کردیم؟ بنابراین استفاده از سشن غیرممکن می‌شد؛ چون برنامه اجازه‌ی درج هیچ مقداری را در هیچ نوع سشنی نمی‌داد.
جهت تکمیل بحث:
در وب فرم‌ها هم امکان readonly کردن سشن مورد استفاده در یک صفحه وجود دارد :

```
pagesSection.EnableSessionState = PagesEnableSessionState.ReadOnly
```

```
<% @Page EnableSessionState="ReadOnly" %>
```

نویسنده: reza
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۵:۵۳

آقای نصیری آیا استفاده از سشن در زمانی که کاربران زیاد هستند بار زیادی روی سرور ایجاد نمیکنه. تا چه تعداد کاربر رو میشه با سشن ساپورت کرد. مثلا همون اطلاعات یوزری که لاگین کرده؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۳ ۱۶:۱۴

سؤال شما خارج از بحث پردازش موازی است. سشن به صورت پیش فرض در حافظه سرور ذخیره می‌شود بنابراین حد و حدود آن مشخص است. البته سشن را در ASP.NET می‌شود در SQL Server هم ذخیره کرد. کمی کندتر از حافظه است اما مشکل مقیاس پذیری آنچنانی نداره.

نویسنده: omid
تاریخ: ۱۳۹۱/۰۸/۲۳ ۲۰:۴

سلام
منظورتون از چندین درخواست یا یک sessionId یکسان به موتور asp.net برسند رو میشه توضیح بدید ؟
آیا منظورتون از Session همون Session[""] هست ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۲۳ ۲۲:۴۶

توضیحات بیشتر در مورد SessionId : (^)

نویسنده: علی
تاریخ: ۱۳۹۱/۱۰/۲۸ ۲۳:۵

سلام
من UserId کاربری که به سایت لاگین کرده رو در session نگه می‌دارم تا برای کارهای مختلف هربار به دیتابیس کوئری نزنم. آیا روش دیگری هست که بجای سشن بشه استفاده کرد (خصوصاً در MVC)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۲۸ ۲۳:۲۰

بله. از Forms authentication استفاده کنید.

```
FormsAuthentication.SetAuthCookie(user.Id.ToString(CultureInfo.InvariantCulture),
loginInfo.RememberMe);
FormsAuthentication.RedirectFromLoginPage(user.Id.ToString(), loginInfo.RememberMe);
```

بعد userId رو به شکل فوق تنظیم کنید. از این به بعد با مراجعه به شیء User به صورت User.Identity.Name به مقدار UserId خواهید رسید.

نویسنده: علی
تاریخ: ۱۳۹۱/۱۰/۲۸ ۲۳:۳۰

ممنون، آیا اطلاعات دیگری را هم می‌شود با این روش ذخیره کرد، مثلا نام کاربر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۲۸ ۲۳:۳۸

کاری که در اینجا انجام میشه ذخیره اطلاعات به صورت رمزنگاری شده در یک کوکی است. رمزنگاری، ذخیره و مدیریت این کوکی (AuthCookie) خودکار است. در سایر موارد خودتون این مراحل مدیریت کوکی‌ها رو با کدنویسی شبیه سازی کنید.

نویسنده: علی
تاریخ: ۱۳۹۱/۱۰/۲۸ ۲۳:۳۹

ممنون بابت راهنمایی خوبتون

نویسنده: رضا گرمارودی
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۱:۱۰

سلام من واقعا متوجه نحوه استفاده یا عدم استفاده از Sessionها نشدم!
در Asp.net برای نگهداری یکسری از اطلاعات کاربر و یا هرگونه اطلاعات مربوط به یک کاربر خاص از Sessionها استفاده می‌کردیم.
حال در Mvc برای نگهداری هر گونه اطلاعات کاربر Login شده از چی باید استفاده کرد. اگر جواب کوکی هاست این کوکی‌ها در سمت کلاینت ثبت می‌شود و نیاز است که اطلاعات در سمت سرور باشد.
کدم بهتره است : بخشی از حافظه برای اطلاعات کاربر جاری مصرف شود و درعوض در هر درخواستی از آن استفاده شود و یا خیر هربار این اطلاعات واکشی شود (البته بدون درنظر گرفتن سطح دوم کش).
همچنین Sessionها در Mvc با Session در Asp.net چه تفاوت هایی دارد؟
ممنون میشم ابهام بنده را برطرف بفرمایید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۱:۳۰

- در سایت جاری ماژول سشن در وب کانفیگ کلا حذف شده. در هیچ‌جایی از آن از سشن استفاده نشده. (یک نمونه عملی)

- به کوکی سمت کلاینت در سمت سرور دسترسی وجود دارد. مانند سیستم Forms Authentication که اساس آن استفاده از کوکی‌های رمزنگاری شده‌است؛ چه برای نگهداری اطلاعات کاربر لاگین شده به سیستم و چه برای نگهداری اطلاعات نقش‌های منتسب به او.
- تفاوتی ندارند. یک هسته‌ی اصلی ASP.NET وجود دارد که حداقل دو فریم ورک وب فرمز و MVC از آن استفاده می‌کنند.

نویسنده: همراز
تاریخ: ۱۳۹۳/۰۹/۱۱ ۱۱:۰

سلام؛ اگر بخواهیم از سشن استفاده نکنیم، برای نگهداری اطلاعات کسی که لاگین کرده چه راهی وجود داره؟ مثلاً UserId یا LoginId که با اون بشه به UserId رسید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۱۱ ۱۱:۲۴

- در سایر نظرات مطلب جاری پیشتر بحث شده در این مورد.
- هیچکدام از روش‌های مدرن authentication در ASP.NET از سشن استفاده نمی‌کنند. برای مثال از forms authentication و یا asp.net identity استفاده کنید.

عنوان: **استفاده از دکمه‌های CSS توئیتر در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۷ ۱۲:۴۴
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC

یک سری نکته ریز را جهت بهبود ظاهر برنامه‌های وب می‌توان درنظر داشت؛ برای مثال:
مجموعه‌ی Twitter Bootstrap که به عنوان یکی از فریم‌ورک‌های خوب CSS مطرح است، دارای تعدادی دکمه تهیه شده با CSS است : (^)
برای نمایش یک چنین دکمه‌هایی فقط کافی است یک span را به صفحه اضافه کرده و class آن‌را مثلا مساوی btn btn-info قرار دهیم تا دکمه‌ای آبی رنگ نمایش داده شود.

طراحی زیبایی دارد. با مرورگرهای جدید سازگار است و ... در اصل یک span بیشتر نیست و قابلیت post back به سرور را ندارد.
برای اضافه کردن چنین قابلیتی می‌توان از نکته زیر استفاده کرد:

```
<span style=" margin:7px;" onclick="$('#loginForm').submit()">ورود به سیستم</span>
```

jQuery که جزء پیش فرض برنامه‌های ASP.NET MVC است. متد submit را هم می‌توان در اینجا در صورت کلیک بر روی یک span فراخوانی کرد. نکته مهم آن ذکر Id فرم جاری است که مثلا به این شکل قابل تعریف است:

```
@using (Html.BeginForm(actionName: "LogOn", controllerName: "Login",
                       method: FormMethod.Post, htmlAttributes: new { id = "loginForm" }))
{
}
```

نظرات خوانندگان

نویسنده: احمد
تاریخ: ۱۳۹۱/۰۴/۰۸ ۱۱:۱۶

بسیار عالی!
وقتی css رو به صفحه اضافه میکنیم روی تمام کنترلها اثر میذاره. که مطلوب نیست. راه حل چیه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۸ ۱۱:۳۰

ضرورتی به استفاده از تمام پکیج آن نیست. مثلا می‌شود فقط از فایل css مربوط به دکمه‌های آن استفاده کرد تا بر روی سایر کنترل‌ها تاثیر گذار نباشد. مانند این فایل:

buttons.css

عنوان: **نحوه صحیح تولید Url در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۰۸ ۱۰:۱۶
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC

کار متد کمکی Url.Action ایجاد یک Url بر اساس تعاریف مسیریابی برنامه است. البته متد کمکی مشابهی نیز مانند Html.ActionLink در ASP.NET MVC وجود دارد که کار آن تولید یک لینک قابل کلیک است؛ اما ممکن است در حالتی خاص تنها نیاز به خود Url داشته باشیم و نه لینک قابل کلیک آن.

الف) اگر از jQuery Ajax استفاده می‌کنید، حتما باید استفاده از Url.Action را لحاظ کنید

برای نمونه اگر قسمتی از عملیات Ajaxایی برنامه شما به نحو زیر تعریف شده است :

```
$.ajax({
          type: "POST",
           url: "/Home/EmployeeInfo",
...
```

... غلط است!
در این حالت برنامه شما تنها در زمانیکه در ریشه یک دومین قرار گرفته باشد کار خواهد کرد. اگر برنامه شما در مسیری مانند http://www.site.com/myNewApp نصب شود، کلیه فراخوانی‌های Ajax ایی آن دیگر کار نخواهند کرد چون مسیر url فوق به ریشه سایت اشاره می‌کند و نه مسیر جاری برنامه شما (در یک sub domain جدید).
به همین جهت در یک چنین حالتی حتما باید به کمک Url.Action مسیر یک اکشن متد را معرفی کرد تا به صورت خودکار بر اساس محل قرارگیری برنامه و تعاریف مسیریابی آن، Url صحیحی تولید شود.

```
@Url.Action("EmployeeInfo", "Home")
```

ب) دریافت Url مطلق از یک Url.Action

Urlهای تولید شده توسط Url.Action به صورت پیش فرض نسبی هستند (نسبت به محل نصب و قرارگیری برنامه تعریف می‌شوند). اگر نیاز به دریافت یک مسیر مطلق که با http برای مثال شروع می‌شود دارید، باید به نحو زیر عمل کرد:

```
@Url.Action("About", "Home", null, "http")
```

پارامتر چهارم تعریف شده، سبب خواهد شد لینک تولیدی از حالت نسبی خارج شود.

ج) استفاده از Url.Action در یک کنترلر

فرض کنید قصد تولید یک فید RSS را در کنترلری دارید. یکی از آیتم‌هایی که باید ارائه دهید، لینک به مطلب مورد نظر است. این لینک باید مطلق باشد همچنین در یک View هم قرار نداریم که به کمک @ بلافاصله به متد کمکی Url.Action دسترسی پیدا کنیم.
در کنترلرها، وهله جاری کلاس به شیء Url و متد Action آن به نحو زیر دسترسی دارد و خروجی نهایی آن یک رشته است:

```
var url = this.Url.Action(actionName: "Index",
                                  controllerName: "Post",
                                  protocol: "http",
                                  routeValues: new { id = item.Id });
```

بنابراین در کنترلرها نیز Urlها را دستی تولید نکنید و اجازه دهید بر اساس پارامترهای زیادی که در پشت صحنه Url.Action

لحاظ می‌شود، Url صحیحی برای شما تولید گردد.

د) استفاده از Url.Action در کلاس‌های کمکی برنامه خارج از یک کنترلر

فرض کنید قصد تهیه یک Html Helper سفارشی را به کمک کدنویسی در یک کلاس مجزا دارید. در اینجا نیز نباید Urlها را دستی تولید کرد. در Html Helperهای سفارشی نیز می‌توان به کمک متد UrlHelper.GenerateUrl، به همان امکانات Url.Action دسترسی یافت:

```
public static class Extensions
    {
        public static string MyLink(this HtmlHelper html, ...)
        {
            string url = UrlHelper.GenerateUrl(null, "actionName", "controllerName",
                                               null,
                                               html.RouteCollection,
                                               html.ViewContext.RequestContext,
                                               includeImplicitMvcValues: true);
//...
```

نظرات خوانندگان

نویسنده: محسن
تاریخ: ۱۳۹۱/۰۴/۱۰ ۱۵:۶

سلام و خسته نباشید خدمت شما آقای نصیری. وبلاگتون واقعا مفید و عالیه
ممنون از زحماتتون بابت مطالبی که به اشتراک میگذارید
یه سوال از خدمتتون داشتم
من روی یه پروژه MVC کار میکنم. به این صورت کار کردم که یک View کلی دارم که توی اون نمایش اطلاعات و "ایجاد" و "حذف" و "ویرایش" همه یکجا انجام میشن. این View را کاملا با Jquery کار کردم و Insert , Update , Delete کلا توسط Jquery انجام میشه. اما توی یکی از Viewهای دیگه که Strongly Type هم هستش، نمیخام به طور کامل از Jquery استفاده کنم. به این صورت که من اطلاعات جدول مربوطه رو توسط Jquery از پایگاه داده میگیرم و در یک گرید نمایش میدم. توی این گرید برای هر ردیف دوتا لینک "حذف" و "ویرایش" وجود داره. برای حذف هم با Jquery کارمو انجام میدم. اما برای "ایجاد" یک فرم از نوع فرم‌های MVC دارم که داخل یه dialog از نوع Jquery قرارشون دادم. برای ایجاد هم مشکلی نیست. اما برای ویرایش، نمیدونم که چطوری باید اطلاعات رو از پایگاه داده لود کنم که خودش مستقیما داخل TextBoxها قرار بگیره. البته توسط Jquery اومدم تک تک textboxها رو به صورت زیر مقدار دادم. حالا نمیدونم کارم درسته یا روش بهتری هست واسه این کار. اگر نیاز هست تا سورس برنامه رو واستون بفرستم

```
function editmode(val) {
        $.ajax({ url: "/User/SelectUser", data: { Username: val }, type: "post", dataType: "json",
success: function (data) {
            if (data != "timeout") {
                if (data.isRedirect) { window.location.href = data.redirectUrl; return; }
                try {
                    $("#dvEdit").dialog({ modal: 'true', title: 'ویرایش', hide: 'clip' });
                    //alert($("#dvEdit").html());

                    $("#UsernameEdit").val(data.Username);
                    $("#FirstNameEdit").val(data.FirstName);
                    $("#LastNameEdit").val(data.LastName);
                    $("#NationalCodeEdit").val(data.NationalCode);
                    $("#EmailEdit").val(data.Email);
                    $("#PhoneNoEdit").val(data.PhoneNo);
                    $("#MobileNoEdit").val(data.MobileNo);
                    $("#CreationDateEdit").val(data.CreationDate);
                    $("#CreationDateEdit").prop('disabled', true);
                    $("#LastActivityDateEdit").val(data.LastActivityDate);
                    $("#LastActivityDateEdit").prop('disabled', true);
                    $("#LastLoginDateEdit").val(data.LastLoginDate);
                    $("#LastLoginDateEdit").prop('disabled', true);
                    $("#IsLockedOutEdit").val(data.IsLockedOut);
                    $("#AddressEdit").val(data.Address);


                }
                catch (err) {
                    $("#dverr").show(); $("#lblErr").html(err);
                }
            }
            else
                AjaxTimeout();
        }
        , error: function (req, textstatus, errorthrown) { AjaxError(req, textstatus, "#dverr",
"#lblErr"); }
        , complete: function (xhr, e) { AjaxComplete(xhr, "#dverr", "#lblErr"); }
        });
    }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۰ ۱۶:۵۹

فرم ادیت رو به صورت strongly typed از نوع یک partial view مستقل درست کنید.
سیس در کنترلر مرتبط، یک اکشن متد را مخصوص رندر کردن این partial view در نظر بگیرید. کار آن دریافت اطلاعات مرتبط

با Model ارسالی به همین partial view است. سپس در آخر کار آن هم خواهیم داشت:

```
return PartialView("_MyPartialViewName", data);
```

حالا فراخوانی این اکشن متد توسط jQuery Ajax سبب پر شدن خودکار فیلدهای فرم strongly typed شما هم می‌شود. در حین درخواست، اطلاعات مدل از بانک اطلاعاتی دریافت شده و به Partial view ارسال می‌شود. چون strongly typed است، فیلدهای آن به صورت خودکار پر شده و نهایتا کل partial view به صورت یک رشته آماده شده در اختیار شما خواهد بود. بنابراین، قسمت عمده‌ای از کدهای سمت کاربر فوق کاهش پیدا می‌کنند. چون یک view کامل حاضر و آماده از سرور دریافت شده است که باید به صفحه توسط jQuery اضافه شود.

+
لطفا اینجا رو تبدیل به یک انجمن عمومی نکنید. عنوان بحث ساخت Url بود ... بعد تغییر جهت پیدا کرد به یک عنوان دیگر.
با تشکر از توجه شما.

نویسنده: Afshin Akbari
تاریخ: ۱۳۹۱/۰۵/۲۰ ۲:۱۸

سلام جناب نصیری ممنون از مطالب مفیدتون.
سوال:
همون طور که مرسوم هست معمولا کدهای jquery رو در داخل فایل مجزا قرار میدن که در این صورت دیگه mvc وجود نداره که بخوایم از @url استفاده کنیم میخوام بدونم برای این مشکل راه حلی وجود داره.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۲۰ ۸:۳۷

در View مورد نظر یک section تعریف کنید به نحو زیر (که در آن می‌شود متغیرها را تعریف کرد):

```
@Section scriptVariables {
    <script type="text/javascript">
        var variableA = '@MyVarA';
    </script>
}
```

سپس در فایل layout سایت، پیش از الحاق اسکریپت اصلی آن‌را فراخوانی کنید:

```
<head>
    <meta charset="utf-8" />
    @RenderSection("scriptVariables", false)

    <script src="@Url.Content("~/Scripts/custom.js")" type="text/javascript"></script>
</head>
```

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۳:۵۱

سلام؛ من توی پروژم 2 تا Area دارم.یک هلپر برای نمایش گالری هم دارم که توی اون هلپر از Url.Action استفاده میکنم.جالب اینجاست که این هلپر توی یک Area مسیر اکشن رو درست بر می‌گردونه ولی برای Area دومی که همون هلپر رو استفاده میکنم اسم اکشن (که Index هست) رو بر نمیگردونه.اینه :

```
Url.Action("Index", "ShowGallery", null, Request.Url.Schem)
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۴:۲۹

نام Area را باید ذکر کنید

```
@Url.Action("Action", "MyController", new { area = "OtherArea" })
```

و یا بهتر است از T4MVC استفاده کنید تا این مسایل را به صورت خودکار لحاظ کند.

عنوان: **استفاده از OpenID در وب سایت جهت احراز هویت کاربران**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۴/۰۸ ۲۱:۲
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, C#, MVC, Security, OpenID

قبلا شرح مختصری در زمینه OpenID در اینجا گفته شد.
حال می‌خواهیم این امکان را در پروژه خود بکار ببریم، جهت این کار باید ابتدا یک پروژه ایجاد کرده و از کتابخانه‌های سورس باز موجود استفاده کرد.
1- ابتدا در ویژوال استودیو یا هر نرم افزار دیگر یک پروژه MVC ایجاد نمایید.

2- نوع Internet Application و برای View Engine سایت Razor را انتخاب نمایید.

3- کتابخانه DotNetOpenId سورس باز را می‌توانید مستقیما از این آدرس دانلود نموده یا از طریق Package Manager Console و با نوشتن Install-Package DotNetOpenAuth به صورت آنلاین این کتابخانه را نصب نمایید.

4- مدل‌های برنامه را مانند زیر ایجاد نمایید

```
using System;
using System.Collections.Generic;
using System.ComponentModel.DataAnnotations;
using System.Globalization;
using System.Security.Cryptography;
using System.Text;
using System.Web.Mvc;
using System.Web.Security;

namespace OpenIDExample.Models
{
    #region Models

    public class ChangePasswordModel
    {
        [Required]
```

```
        [DataType(DataType.Password)]
        [Display(Name = "Current password")]
        public string OldPassword { get; set; }

        [Required]
        [ValidatePasswordLength]
        [DataType(DataType.Password)]
        [Display(Name = "New password")]
        public string NewPassword { get; set; }

        [DataType(DataType.Password)]
        [Display(Name = "Confirm new password")]
        [Compare("NewPassword", ErrorMessage = "The new password and confirmation password do not
match.")]
        public string ConfirmPassword { get; set; }
    }

    public class LogOnModel
    {
        [Display(Name = "OpenID")]
        public string OpenID { get; set; }

        [Required]
        [Display(Name = "User name")]
        public string UserName { get; set; }

        [Required]
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }

        [Display(Name = "Remember me?")]
        public bool RememberMe { get; set; }
    }

    public class RegisterModel
    {
        [Display(Name = "OpenID")]
        public string OpenID { get; set; }

        [Required]
        [Display(Name = "User name")]
        public string UserName { get; set; }

        [Required]
        [DataType(DataType.EmailAddress)]
        [Display(Name = "Email address")]
        public string Email { get; set; }

        [Required]
        [ValidatePasswordLength]
        [DataType(DataType.Password)]
        [Display(Name = "Password")]
        public string Password { get; set; }

        [DataType(DataType.Password)]
        [Display(Name = "Confirm password")]
        [Compare("Password", ErrorMessage = "The password and confirmation password do not match.")]
        public string ConfirmPassword { get; set; }
    }

    #endregion Models

    #region Services

    // The FormsAuthentication type is sealed and contains static members, so it is difficult to
    // unit test code that calls its members. The interface and helper class below demonstrate
    // how to create an abstract wrapper around such a type in order to make the AccountController
    // code unit testable.

    public interface IMembershipService
    {
        int MinPasswordLength { get; }

        bool ValidateUser(string userName, string password);

        MembershipCreateStatus CreateUser(string userName, string password, string email, string
OpenID);

        bool ChangePassword(string userName, string oldPassword, string newPassword);
```

```
        MembershipUser GetUser(string OpenID);
    }

    public class AccountMembershipService : IMembershipService
    {
        private readonly MembershipProvider _provider;

        public AccountMembershipService()
            : this(null)
        {
        }

        public AccountMembershipService(MembershipProvider provider)
        {
            _provider = provider ?? Membership.Provider;
        }

        public int MinPasswordLength
        {
            get
            {
                return _provider.MinRequiredPasswordLength;
            }
        }

        public bool ValidateUser(string userName, string password)
        {
            if (String.IsNullOrEmpty(userName)) throw new ArgumentException("Value cannot be null or
empty.", "userName");
            if (String.IsNullOrEmpty(password)) throw new ArgumentException("Value cannot be null or
empty.", "password");

            return _provider.ValidateUser(userName, password);
        }

        public Guid StringToGUID(string value)
        {
            // Create a new instance of the MD5CryptoServiceProvider object.
            MD5 md5Hasher = MD5.Create();
            // Convert the input string to a byte array and compute the hash.
            byte[] data = md5Hasher.ComputeHash(Encoding.Default.GetBytes(value));
            return new Guid(data);
        }

        public MembershipCreateStatus CreateUser(string userName, string password, string email, string
OpenID)
        {
            if (String.IsNullOrEmpty(userName)) throw new ArgumentException("Value cannot be null or
empty.", "userName");
            if (String.IsNullOrEmpty(password)) throw new ArgumentException("Value cannot be null or
empty.", "password");
            if (String.IsNullOrEmpty(email)) throw new ArgumentException("Value cannot be null or
empty.", "email");

            MembershipCreateStatus status;
            _provider.CreateUser(userName, password, email, null, null, true, StringToGUID(OpenID), out
status);
            return status;
        }

        public MembershipUser GetUser(string OpenID)
        {
            return _provider.GetUser(StringToGUID(OpenID), true);
        }

        public bool ChangePassword(string userName, string oldPassword, string newPassword)
        {
            if (String.IsNullOrEmpty(userName)) throw new ArgumentException("Value cannot be null or
empty.", "userName");
            if (String.IsNullOrEmpty(oldPassword)) throw new ArgumentException("Value cannot be null or
empty.", "oldPassword");
            if (String.IsNullOrEmpty(newPassword)) throw new ArgumentException("Value cannot be null or
empty.", "newPassword");

            // The underlying ChangePassword() will throw an exception rather
            // than return false in certain failure scenarios.
            try
            {
                MembershipUser currentUser = _provider.GetUser(userName, true /* userIsOnline */);
                return currentUser.ChangePassword(oldPassword, newPassword);
            }
```

```
            catch (ArgumentException)
            {
                return false;
            }
            catch (MembershipPasswordException)
            {
                return false;
            }
        }

        public MembershipCreateStatus CreateUser(string userName, string password, string email)
        {
            throw new NotImplementedException();
        }
    }

    public interface IFormsAuthenticationService
    {
        void SignIn(string userName, bool createPersistentCookie);

        void SignOut();
    }

    public class FormsAuthenticationService : IFormsAuthenticationService
    {
        public void SignIn(string userName, bool createPersistentCookie)
        {
            if (String.IsNullOrEmpty(userName)) throw new ArgumentException("Value cannot be null or
empty.", "userName");

            FormsAuthentication.SetAuthCookie(userName, createPersistentCookie);
        }

        public void SignOut()
        {
            FormsAuthentication.SignOut();
        }
    }

    #endregion Services

    #region Validation

    public static class AccountValidation
    {
        public static string ErrorCodeToString(MembershipCreateStatus createStatus)
        {
            // See http://go.microsoft.com/fwlink/?LinkID=177550 for
            // a full list of status codes.
            switch (createStatus)
            {
                case MembershipCreateStatus.DuplicateUserName:
                    return "Username already exists. Please enter a different user name.";

                case MembershipCreateStatus.DuplicateEmail:
                    return "A username for that e-mail address already exists. Please enter a different
e-mail address.";

                case MembershipCreateStatus.InvalidPassword:
                    return "The password provided is invalid. Please enter a valid password value.";

                case MembershipCreateStatus.InvalidEmail:
                    return "The e-mail address provided is invalid. Please check the value and try
again.";

                case MembershipCreateStatus.InvalidAnswer:
                    return "The password retrieval answer provided is invalid. Please check the value
and try again.";

                case MembershipCreateStatus.InvalidQuestion:
                    return "The password retrieval question provided is invalid. Please check the value
and try again.";

                case MembershipCreateStatus.InvalidUserName:
                    return "The user name provided is invalid. Please check the value and try again.";

                case MembershipCreateStatus.ProviderError:
                    return "The authentication provider returned an error. Please verify your entry and
try again. If the problem persists, please contact your system administrator.";

                case MembershipCreateStatus.UserRejected:
```

```
                    return "The user creation request has been canceled. Please verify your entry and
try again. If the problem persists, please contact your system administrator.";

                default:
                    return "An unknown error occurred. Please verify your entry and try again. If the
problem persists, please contact your system administrator.";
            }
        }
    }

    [AttributeUsage(AttributeTargets.Field | AttributeTargets.Property, AllowMultiple = false,
Inherited = true)]
    public sealed class ValidatePasswordLengthAttribute : ValidationAttribute, IClientValidatable
    {
        private const string _defaultErrorMessage = "'{0}' must be at least {1} characters long.";
        private readonly int _minCharacters = Membership.Provider.MinRequiredPasswordLength;

        public ValidatePasswordLengthAttribute()
            : base(_defaultErrorMessage)
        {
        }

        public override string FormatErrorMessage(string name)
        {
            return String.Format(CultureInfo.CurrentCulture, ErrorMessageString,
                name, _minCharacters);
        }

        public override bool IsValid(object value)
        {
            string valueAsString = value as string;
            return (valueAsString != null && valueAsString.Length >= _minCharacters);
        }

        public IEnumerable<ModelClientValidationRule> GetClientValidationRules(ModelMetadata metadata,
ControllerContext context)
        {
            return new[]{
                new
ModelClientValidationStringLengthRule(FormatErrorMessage(metadata.GetDisplayName()), _minCharacters,
int.MaxValue)
            };
        }
    }

    #endregion Validation
}
```

5- در پروژه مربوطه یک Controller به نام AccountController ایجاد نمایید. و کدهای زیر را برای آنها وارد نمایید.

```
using System.Web.Mvc;
using System.Web.Routing;
using System.Web.Security;
using DotNetOpenAuth.Messaging;
using DotNetOpenAuth.OpenId;
using DotNetOpenAuth.OpenId.RelyingParty;
using OpenIDExample.Models;

namespace OpenIDExample.Controllers
{
    public class AccountController : Controller
    {
        private static OpenIdRelyingParty openid = new OpenIdRelyingParty();

        public IFormsAuthenticationService FormsService { get; set; }

        public IMembershipService MembershipService { get; set; }

        protected override void Initialize(RequestContext requestContext)
        {
            if (FormsService == null) { FormsService = new FormsAuthenticationService(); }
            if (MembershipService == null) { MembershipService = new AccountMembershipService(); }

            base.Initialize(requestContext);
        }

        // **************************************
        // URL: /Account/LogOn
```

```
        // ************************************

        public ActionResult LogOn()
        {
            return View();
        }

        [HttpPost]
        public ActionResult LogOn(LogOnModel model, string returnUrl)
        {
            if (ModelState.IsValid)
            {
                if (MembershipService.ValidateUser(model.UserName, model.Password))
                {
                    FormsService.SignIn(model.UserName, model.RememberMe);
                    if (Url.IsLocalUrl(returnUrl))
                    {
                        return Redirect(returnUrl);
                    }
                    else
                    {
                        return RedirectToAction("Index", "Home");
                    }
                }
                else
                {
                    ModelState.AddModelError("", "The user name or password provided is incorrect.");
                }
            }

            // If we got this far, something failed, redisplay form
            return View(model);
        }

        // ************************************
        // URL: /Account/LogOff
        // ************************************

        public ActionResult LogOff()
        {
            FormsService.SignOut();

            return RedirectToAction("Index", "Home");
        }

        // ************************************
        // URL: /Account/Register
        // ************************************

        public ActionResult Register(string OpenID)
        {
            ViewBag.PasswordLength = MembershipService.MinPasswordLength;
            ViewBag.OpenID = OpenID;
            return View();
        }

        [HttpPost]
        public ActionResult Register(RegisterModel model)
        {
            if (ModelState.IsValid)
            {
                // Attempt to register the user
                MembershipCreateStatus createStatus = MembershipService.CreateUser(model.UserName,
model.Password, model.Email, model.OpenID);

                if (createStatus == MembershipCreateStatus.Success)
                {
                    FormsService.SignIn(model.UserName, false /* createPersistentCookie */);
                    return RedirectToAction("Index", "Home");
                }
                else
                {
                    ModelState.AddModelError("", AccountValidation.ErrorCodeToString(createStatus));
                }
            }

            // If we got this far, something failed, redisplay form
            ViewBag.PasswordLength = MembershipService.MinPasswordLength;
            return View(model);
        }
```

```
        // ************************************
        // URL: /Account/ChangePassword
        // ************************************

        [Authorize]
        public ActionResult ChangePassword()
        {
            ViewBag.PasswordLength = MembershipService.MinPasswordLength;
            return View();
        }

        [Authorize]
        [HttpPost]
        public ActionResult ChangePassword(ChangePasswordModel model)
        {
            if (ModelState.IsValid)
            {
                if (MembershipService.ChangePassword(User.Identity.Name, model.OldPassword,
model.NewPassword))
                {
                    return RedirectToAction("ChangePasswordSuccess");
                }
                else
                {
                    ModelState.AddModelError("", "The current password is incorrect or the new password
is invalid.");
                }
            }

            // If we got this far, something failed, redisplay form
            ViewBag.PasswordLength = MembershipService.MinPasswordLength;
            return View(model);
        }

        // ************************************
        // URL: /Account/ChangePasswordSuccess
        // ************************************

        public ActionResult ChangePasswordSuccess()
        {
            return View();
        }

        [ValidateInput(false)]
        public ActionResult Authenticate(string returnUrl)
        {
            var response = openid.GetResponse();
            if (response == null)
            {
                //Let us submit the request to OpenID provider
                Identifier id;
                if (Identifier.TryParse(Request.Form["openid_identifier"], out id))
                {
                    try
                    {
                        var request = openid.CreateRequest(Request.Form["openid_identifier"]);
                        return request.RedirectingResponse.AsActionResult();
                    }
                    catch (ProtocolException ex)
                    {
                        ViewBag.Message = ex.Message;
                        return View("LogOn");
                    }
                }

                ViewBag.Message = "Invalid identifier";
                return View("LogOn");
            }

            //Let us check the response
            switch (response.Status)
            {
                case AuthenticationStatus.Authenticated:
                    LogOnModel lm = new LogOnModel();
                    lm.OpenID = response.ClaimedIdentifier;
                    //check if user exist
                    MembershipUser user = MembershipService.GetUser(lm.OpenID);
                    if (user != null)
                    {
                        lm.UserName = user.UserName;
                        FormsService.SignIn(user.UserName, false);
```

```
                    }

                    return View("LogOn", lm);

                case AuthenticationStatus.Canceled:
                    ViewBag.Message = "Canceled at provider";
                    return View("LogOn");
                case AuthenticationStatus.Failed:
                    ViewBag.Message = response.Exception.Message;
                    return View("LogOn");
            }

            return new EmptyResult();
        }
    }
}
```

6- سپس برای Action به نام LogOn یک View می‌سازیم، برای Authenticate نیازی به ایجاد View ندارد چون قرار است درخواست کاربر را به آدرس دیگری Redirect کند. سپس کدهای زیر را برای View ایجاد شده وارد می‌کنیم.

```
@model OpenIDExample.Models.LogOnModel
@{
    ViewBag.Title = "Log On";
}
<h2>
    Log On</h2>
<p>
    Please enter your username and password. @Html.ActionLink("Register", "Register")
    if you don't have an account.
</p>
<script src="@Url.Content("~/Scripts/jquery.validate.min.js")" type="text/javascript"></script>
<script src="@Url.Content("~/Scripts/jquery.validate.unobtrusive.min.js")"
type="text/javascript"></script>
<form action="Authenticate?ReturnUrl=@HttpUtility.UrlEncode(Request.QueryString["ReturnUrl"])"
method="post" id="openid_form">
<input type="hidden" name="action" value="verify" />
<div>
    <fieldset>
        <legend>Login using OpenID</legend>
        <div class="openid_choice">
            <p>
                Please click your account provider:</p>
            <div id="openid_btns">
            </div>
        </div>
        <div id="openid_input_area">
            @Html.TextBox("openid_identifier")
            <input type="submit" value="Log On" />
        </div>
        <noscript>
            <p>
                OpenID is service that allows you to log-on to many different websites using a single
                indentity. Find out <a href="http://openid.net/what/">more about OpenID</a> and
                <a href="http://openid.net/get/">how to get an OpenID enabled account</a>.</p>
        </noscript>
        <div>
            @if (Model != null)
            {
                if (String.IsNullOrEmpty(Model.UserName))
                {
                <div class="editor-label">
                    @Html.LabelFor(model => model.OpenID)
                </div>
                <div class="editor-field">
                    @Html.DisplayFor(model => model.OpenID)
                </div>
                <p class="button">
                    @Html.ActionLink("New User ,Register", "Register", new { OpenID = Model.OpenID })
                </p>
                }
                else
                {
                    //user exist
                <p class="buttonGreen">
                    <a href="@Url.Action("Index", "Home")">Welcome , @Model.UserName, Continue..." </a>
                </p>
```

```
                }
            }
        </div>
    </fieldset>
</div>
</form>
@Html.ValidationSummary(true, "Login was unsuccessful. Please correct the errors and try again.")
@using (Html.BeginForm())
{
    <div>
        <fieldset>
            <legend>Or Login Normally</legend>
            <div class="editor-label">
                @Html.LabelFor(m => m.UserName)
            </div>
            <div class="editor-field">
                @Html.TextBoxFor(m => m.UserName)
                @Html.ValidationMessageFor(m => m.UserName)
            </div>
            <div class="editor-label">
                @Html.LabelFor(m => m.Password)
            </div>
            <div class="editor-field">
                @Html.PasswordFor(m => m.Password)
                @Html.ValidationMessageFor(m => m.Password)
            </div>
            <div class="editor-label">
                @Html.CheckBoxFor(m => m.RememberMe)
                @Html.LabelFor(m => m.RememberMe)
            </div>
            <p>
                <input type="submit" value="Log On" />
            </p>
        </fieldset>
    </div>
}
```

پس از اجرای پروژه صفحه ای شبیه به پایین مشاهده کرده و سرویس دهنده OpenID خاص خود را می‌توانید انتخاب نمایید.

7- برای فعال سازی عملیات احراز هویت توسط FormsAuthentication در سایت باید تنظیمات زیر را در فایل web.config انجام دهید.

```
<authentication mode="Forms">
      <forms loginUrl="~/Account/LogOn" timeout="2880" />
</authentication>
```

خوب تا اینجا کار تمام است و کاربر در صورتی که در سایت OpenID نام کاربری داشته باشد می‌تواند در سایت شما Login کند.
جهت مطالعات بیشتر ودانلود نمونه کدهای آماده می‌توانید به لینک‌های (^ و ^ و ^ و ^ و ^ و ^ و ^) مراجعه کنید.
کد کامل پروژه را می‌توانید از اینجا دانلود نمایید.

منبع

نظرات خوانندگان

نویسنده: ahmadalli
تاریخ: ۱۳۹۱/۰۴/۰۸ ۲۳:۱۲

خوب این کدهای شما برای نسخه‌های قدیمی MVC هست. من نصفش رو درست متوجه نشدم. اگر میشه برای نسخه ۴ یا ۳ هم مثال بزارید.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۴/۰۹ ۱:۲۵

اگه میشه کد برنامه را ضمیمه کنید...

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۴/۰۹ ۲:۴

کدی که ابتدا گذاشته بودم ظاهرا خیلی قدیمی بود اون با کدهای جدید تعویض کردم و لینک دانلود کد هم برای دوستان قرار دادم

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۴/۰۹ ۲:۲۱

اصلاح شد

نویسنده: saleh
تاریخ: ۱۳۹۱/۰۴/۱۷ ۰:۳۳

فرضاً اگه شخصی که با OID لوگین کرده بخواد در فروم سایت من فعالیت کنه چه جوری به اطلاعات پستها و اعمالش پی ببرم؟ چون این شخص در سایت ثبت نام نکرده و طبیعتاً اکانتی در سایت نداره!

در قسمت اول مقاله به این موضوعات اشاره نکردید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۷ ۱۱:۴۴

بحث فوروم نوشته شده شما با طراحی یک سیستم جدید (که در اینجا مد نظر است) متفاوت است. سورس فوق را مطالعه کنید تا با نحوه یکپارچگی آن با سیستم membership دات نت آشنا شوید.

نویسنده: احمدعلی شفیعی
تاریخ: ۱۳۹۱/۰۴/۱۸ ۱۲:۳۸

ممنونم

نویسنده: کامران
تاریخ: ۱۳۹۲/۰۲/۱۲ ۲۳:۲۱

سلام دوست عزیز.
قربان برای ASP.NET Webforms هم میتونید مقاله ای رو آماده کنید؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۳ ۰:۴

یک پروژه دیگر هم در سایت در مورد پروتکل OAuth هست.

نویسنده: کامران
تاریخ: ۱۳۹۲/۰۲/۱۳ ۰:۹

ممنون از جوابتون، دوست عزیز این هم سمپل و کدهاش با MVC هست، مقالاتی برای Webforms پیدا کردم به زبان لاتین اما درست توضیح داده نشده متاسفانه

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۲/۱۳ ۰:۲۵

سلام
دوست گلم یه مثال هست گذاشتم توی این آدرس

Google OpenID Authentication در ASP.NET با استفاده از DotNetOpenAuth

عنوان: **استفاده از ادیتور CKEditor در صفحات ASP.NET**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۴/۱۰ ۹:۵۴
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

در مواردی مانند همین وبلاگ احتیاج به داشتن محل نوشتن، در محیط ادیتور مانندی میباشد که برای این مورد میتوان از ادیتور CKEditor استفاده کرد. بعد از دانلود محتویات مربوطه (که در اینجا از dll مربوط به asp.net استفاده نشده بلکه فایل‌های ادیتور مستقیما مورد استفاده قرار گرفته است)، کافیست لینک‌های مربوط به جاوا اسکریپت و استایل را به هدر اضافه کرد:

```
<link href="admin/editor/contents.css" rel="stylesheet" type="text/css" />
<script src="admin/editor/ckeditor.js" type="text/javascript"></script>
```

و در کد صفحه، بسته به زبان مورد نیاز، یک اسکریپت را به صفحه اضافه نمود؛ در اینجا فارسی استفاده شده:

```
var clientId = "txtPostBody";
var strEditor = strBodyFa = "CKEDITOR.replace( '" + clientId + "',{skin : 'office2003' , language :
'fa' , contentsLangDirection : 'rtl'});";
ClientScript.RegisterStartupScript(GetType(), clientId + "_Editor", strBodyFa, true);
```

در مثال بالا برای باکس متنی به بنام txtPostBody، کد مربوط ایجاد و به صفحه افزوده شده است.
نکته: پوشه مربوط به ادیتور در محلی که در دسترس صفحه باشد باید قرار گیرد که در اکثر موارد قسمت ادمین وب یا اپلیکشن است

نظرات خوانندگان

نویسنده: امیر
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۱:۴۵

دوست عزیز سلام
مطالب مفیدی داری که مشتاقانه اونها رو پیگیری می‌کنم
راجع به این ادیتور یه سوال داشتم
ایا استفاده از سورس بدون لایسنس امکانپذیره؟
یا حتما لایسنس می‌خواد؟
من از کجا می‌تونم سورسش رو داشته باشم؟

بسیار بسیار متشکرم
گودرزی

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۶:۵۶

سلام، میتونید برای اطلاعات دقیق‌تر بخود سایت ckeditor.com/license رجوع کنید ولی تا اونجایی که من اطلاع دارم تا کد رو تغییر ندید استفاده ازش تو پروژه‌ها رایگان است ولی مورد گالری اون رو باید بخرید. ممنونم

نویسنده: امیر
تاریخ: ۱۳۹۱/۰۴/۱۸ ۱۸:۲۲

با تشکر از پاسخ شما.
مهمترین چیزی که من از ادیتور می‌خوام، قابلیت آپلود عکسه!
شما در همین باکس ارسال نظر این قابلیت رو دارید در حالی که ckeditor چنین قابلیتی رو نداره.

لطفا منو راهنمایی کنید.
سپاسگزارم

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۴/۱۸ ۱۹:۳

سلام
CKEditor مستقیما این امکان نداره اما می‌تونین از CKFinder که روی CKEditor نصب میشه و برای این کار طراحی شده استفاده کنید
موفق ومؤید باشید

نویسنده: بهزاد شیرانی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۱۴:۴۳

آیا از لحاظ امنیتی استفاده از ckEditor مشکلی نداره؟
به خاطر این که تا اونجایی که می‌دونم اگر بخوایم به روشی که شما گفتید از این کنترل استفاده کنیم باید تنظیمات امنیتی صفحه (ValidateRequest) رو غیر فعال کنیم و در این صورت کاربر هر نوع کد HTML که بخواد می‌تونه توی اون وارد کنه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۲۶ ۱۴:۵۲

- ValidateRequest به صورت پیش فرض به هر نوع تگی عکس العمل نشون می‌ده؛ نه فقط به یک سری تگ خاص.
- در MVC می‌تونید فقط به ازای یک خاصیت، AllowHtml رو فعال کنید و نه به اجبار به ازای کل صفحه یا کل سایت (مثل وب فرم‌ها).
- بعد از اینکه AllowHtml رو فعال کردید، نیاز خواهید داشت ورودی کاربر را اندکی تمیز کنید .

نویسنده: بهزاد شیرانی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۱۵:۳۹

ممنون از پاسختون
ولی من از WebForm استفاده می‌کنم
فکر می‌کنم همون تمیز کردن کدها راه خوبی باشه

نویسنده: حسین
تاریخ: ۱۳۹۳/۰۷/۰۹ ۱۲:۲۲

با سلام؛ من ویراشگر متنی که در قسمت تماس با استفاده کرده اید را نیاز دارم. در صورت امکان لینک آن را به ایمیل من ارسال کنید. با تشکر فراوران

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۹ ۱۲:۳۶

RedActor هست. البته بهتر است از Froala استفاده کنید؛ چون پشتیبانی بهتری دارد.

عنوان: **مدیریت ساده چهار عمل اصلی تکراری داده در صفحات**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۴/۱۰ ۱۰:۲۳
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

در بیشتر مواردی که در قسمت ادمین سایت می‌گذرد، مثل ریاضیات که شامل چهار عمل اصلی است، افزودن، ویرایش، حذف، و انتخاب نیز، همین وظیفه را در عملیات‌های کار با داده، انجام می‌دهند. البته در این بین اعتبارسنجی را نباید فراموش کرد:

```
0. Validation
1. Insert
2. Edit
3. Delete
4. Select
```

برای مثال میتوان از یک کلاس مشترک برای این کار که از Page ارث‌بری کرده است، استفاده و بصورت یک کلاس abstract وظیفه پیاده سازی را به صفحه مورد نظر محول کرد. برای نمونه کلاس ساده زیر را در نظر بگیرید:

```
public abstract class PageStateMachine : Page
    {
      ...
        protected abstract ... Save();
        protected abstract ... Select();
        protected abstract ... Delete();
        protected abstract ... Validation();
      ...
```

و برای روال گردانی هم مطمئنا این متدها باید به ترتیب یکسان در همه صفحات مورد استفاده قرار گیرد، پس لازم است حالت‌های صفحه به این کلاس پاس شود تا بر اساس حالت صفحه، متدها بترتیب صدا زده شوند:

```
public enum PageState
    {
        Save = 0,
        Select = 2,
        Delete = 8,
        Cancel = 16
    }
```

که در همین کلاس می‌توان متدی برای این کار در نظر گرفت:

```
public void Go(PageState pageState)
        {
            switch (pageState)
            {
                case PageState.Save:
                    {
                        var error = Validation();
                        Message = error.HasError ? error.Message : Save().Message;
                        if (!error.HasError)
                        {
                            ClearControls();
                            RefreshList();
                        }

                        break;
                    }
                case PageState.Select:
                    {
                        Message = Select().Message;
                        break;
                    }
                    break;
                case PageState.Delete:
                    {
                        Message = Delete().Message;
                        ClearControls();
                        RefreshList();
                        break;
```

```
                }
            case PageState.Cancel:
                {
                    Message = string.Empty;
                    ClearControls();
                    break;
                }
        }
    }
```

ادامه دارد

نظرات خوانندگان

نویسنده: اردلان شاه قلی
تاریخ: ۱۳۹۱/۰۸/۲۵ ۱۱:۲۷

سلام .
پس از سپاس فراوان می‌خواستم بدانم ایا ممکنه این موضوع را با مبحث Command Patern بیامیزید. فکر کنم چیز خوبی ازش دربیاید.
خیلی خیلی سپاس

عنوان: **راه اندازی وب سایت سریع و سبک با Nancy**
نویسنده: سلمان عرب عامری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۷:۴۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Nancy, Sinatra, HTTP

Nancy یک فریم ورک سبک برای ساخت سرویس‌های مبتنی بر HTTP بر روی .Net و Mono و در واقع پیاده سازی Sinatra در Ruby برای .net است. با استفاده از این کتابخانه شما می‌توانید به سادگی درخواست‌های مختلف کاربران را از طریق وب پاسخ دهید. از ویژگی‌های این کتابخانه امکان اجرای آن بدون نیاز به وجود وب سرور و به صورت Standalone می‌باشد.
بهتر است وارد عمل شویم و ببینیم این سیستم چگونه عمل می‌کند. برای شروع ما خود Asp .net را به عنوان میزبان در نظر می‌گیریم.
- یک پروژه خالی Asp .net ایجاد کنید. (Asp .net Empty Web Application)
- وارد خط فرمان Package Manager شوید و عبارت زیر را وارد کنید:

```
Install-Package Nancy.Hosting.Aspnet
```

- پروژه‌های Nancy از تعدای ماژول تشکیل شده اند که می‌توانیم برای هر ماژول آدرس وب جداگانه ای در نظر بگیریم.
یک کلاس به پروژه اضافه کنید و نام آن را TestModule بگذارید.
در متن کلاس عبارات زیر را تایپ کنید:

```
public class TestModule : NancyModule
{
    public TestModue()
    {
        Get["/"] = x => "Welcome to my site!";
        Get["/Hello/"] = x=> "Hello Nancy!";
        Get["/Bye/{name}"] = x=> "Good bye " + x.name;
    }
}
```

حال کلید F5 را زده و برنامه را اجرا کنید.
حالا در مرورگر خودتان عبارت http://localhost:12345/Hello را تایپ کنید. توجه کنید که به جای 12345 باید شماره پورتی که وب سرور دات نت اجراست تایپ کنید.
همانطور که متوجه شدید ما در خطوط بالا تعیین کرده ایم که برای درخواست‌های از نوع Get که مسیر ریشه سایت را درخواست می‌کنند عبارت Welcome to my site! ارسال شود. همچنین برای درخواست هایی که مسیر /Hello را درخواست می‌کنند عبارت Hello Nancy نمایان می‌شود.
نکته جالب برای درخواست‌های /Bye است. در اینجا ما یک پارامتر به نام name تعریف کرده ایم و گفته ایم که درخواست‌های Bye که در ادامه آنها عبارتی وجود دارد به صورت Good bye به همراه عبارت بازگردانده شوند.
همین عملیات برای درخواست‌های Put و Head و سایر انواع درخواست وجود دارد.
برای اینکه درخواست‌ها از مسیر خاصی فراخوانی شوند کافی است در هنگام اعلان سازنده کلاس ماژول، مسیر را تعیین کنید.

```
public class TestModule : NancyModule
{
    public TestModule() : base("/test")
    {
        Get["/user"] = x=> "It is test for user";
    }
}
```

این درخواست‌ها از مسیری شبیه این مسیر فراخوانی خواهند شد:

```
http://localhost:12345/test/user
```

همانطور که برای پردازش درخواست از عبارات کوتاه استفاده کردیم می‌توانیم از توابع هم استفاده کنیم.

```
Get["/hello/{username}"] = x=> {
     // some code
     // ...
     // ...
     return "Hello, " + username;
};
```

همچنین امکان بازگرداندن کدهای وضعیت به طور مستقیم وجود دارد.

```
Get["/user"] = x=> return 200;
```

و یا

```
Get["/user] = x=> return HttpStatusCode.OK;
```

همانطور که گفته شد امکان میزبانی پروژه‌های Nancy از داخل برنامه‌های دات نتی هم وجود دارد. در مقاله بعدی به این موضوع خواهیم پرداخت.

نظرات خوانندگان

نویسنده: بهمن
تاریخ: ۱۳۹۱/۰۴/۱۱ ۸:۴۴

سلام ممنون از اینکه این مطلب را ارسال کردید فقط یه سوالی داشتم اونم اینکه آیا در بیشتر سایتها که آدرس url آنها بدین شکل هست چه asp و چه php از این روش برای خلاصه کردن آدرس سایت استفاده میکنند ....؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۰:۴۹

این تکنیک اصطلاحا Url Rewriting نام دارد.

نویسنده: سلمان عرب عامری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۲:۲۴

همانطور که جناب نصیری گفتند، به این روش که آدرس‌ها را کاربر پسند می‌کنیم اصطلاحا Url rewriting می‌گویند. این نوع آدرس دهی باعث خوانا شدن، و همچنین بهتر ایندکس شدن در موتورهای جستجو می‌شود.
در دات نت پروژه MVC آدرس‌ها را طبق این اصول پیاده می‌کند و کاربر می‌تواند قاعده آدرس دهی خود را به صورت یک Route تعریف کند.
در Asp .net سنتی هم کتابخانه‌های Url Rewriting وجود دارند که برای شما این کار را انجام می‌دهند. ضمن اینکه خود شما هم با نوشتن HttpModule می‌توانید این کار را انجام دهید.

عنوان: T4MVC : یکی از الزامات مدیریت پروژه‌های ASP.NET MVC
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۹:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, T4MVC

تعریف ActionLink زیر را درنظر بگیرید:

```
@Html.ActionLink("text", "Index", "Home")
```

پارامترهای دوم و سوم آن که به نام‌های یک اکشن متد و کنترلر آن اشاره می‌کنند، توسط رشته‌ها تعریف شده‌اند. مشکلاتی هم که با رشته‌ها در حالت کلی وجود دارند به شرح زیر است:
الف) می‌توان نام کنترلر یا نام متد را در برنامه تغییر داد. به این ترتیب تمام ActionLink هایی که در برنامه به این کنترلر اشاره می‌کردند از کار می‌افتند (تکرار رشته‌ها به علاوه refactoring friendly نبودن آن‌ها).
ب) برای نوشتن رشته‌ها intellisense کارآیی ندارد.
ج) امکان بروز اشتباهات تایپی در این بین بسیار زیاد است.

راه حل متداولی که برای حل این نوع مشکلات وجود دارد، تعریف یک کلاس عمومی و معرفی رشته‌ها به صورت فیلدهایی ثابت در آن‌ها می‌باشند و سپس استفاده از این فیلدها بجای استفاده مستقیم از رشته‌ها.
و ... چقدر خوب می‌شد اگر ابزاری وجود می‌داشت که کلاس‌های کنترلرهای ما را آنالیز می‌کرد و خودش این ثوابت رشته‌ای را از آن‌ها استخراج و کلاس‌های عمومی یاد شده را تشکیل می‌داد!
خوشبختانه نیازی به اختراع مجدد چرخ نیست و اینکار توسط پروژه‌ی سورس بازی به نام T4MVC انجام شده است. برای دریافت آن به سایت زیر مراجعه نمائید:
https://t4mvc.codeplex.com
این پروژه توسط David Ebbo از اعضای تیم ASP.NET MVC تهیه شده است.

پس از دریافت پروژه، تنها به دو فایل زیر از آن نیاز داریم:
T4MVC.tt و T4MVC.tt.settings.t4

دو فایل فوق را به درون پوشه‌ای از پروژه جاری MVC خود کپی کنید (مثلا یک پوشه T4MVC را ایجاد و این دو فایل را به آن اضافه کنید). بلافاصله این فایل‌های t4 وارد عمل شده و کلاس‌های کنترلرها، Viewها، تصاویر و غیره را آنالیز و ... ثوابت رشته‌ای معادل آن‌ها را تولید می‌کنند.
اکنون برای استفاده از این کلاس‌های تولید شده می‌توان به صورت زیر عمل کرد:

اصلاح ActionLinkها و حذف رشته‌های موجود در آن‌ها

اینبار بجای اینکه بنویسیم:

```
@Html.ActionLink("text", "Index", "Home")
```

می‌توان نوشت:

```
@Html.ActionLink("text", result: MVC.Home.Index())
```

و یا:

```
@Html.ActionLink("text", MVC.Home.ActionNames.Index, MVC.Home.Name)
```

برای دسترسی به امکانات آن با نام کلاس MVC شروع می‌کنیم و سپس برای مثال به نام کنترلر Home رسیده و توسط ActionNames آن به تمام اکشن متدهای موجود در آن می‌توان دسترسی داشت.
البته این پروژه بسیار فراتر از تولید فیلدهای strongly typed معادل رشته‌ها است. همانطور که ملاحظه می‌کنید، یک سری overload را هم به متدهای پیش فرض ASP.NET MVC اضافه کرده است و حتی بجای معرفی رشته معادل اکشن متد Index، خود این اکشن متد را می‌توان معرفی کرد (آرگومانی از نوع ActionResult را هم اضافه کرده است). نمونه دیگر آن به نحو زیر می‌تواند باشد:

```
@Url.Action(result: MVC.Article.Delete())
```

که بجای مورد متداول ذیل قابل استفاده است:

```
@Url.Action("Delete", "Article")
```

امکان معرفی بهتر نام Partial Viewها

برای مثال اگر پیشتر یک Partial View را به این شکل تعریف می‌کردید:

```
@{ Html.RenderPartial("_ViewPage1"); }
```

اکنون امکان معرفی آن به نحو زیر فراهم شده است:

```
@{ Html.RenderPartial(MVC.Home.Views._ViewPage1); }
```

همچنین تمام این کلاس‌ها در کنترلرها نیز قابل دسترسی هستند:

```
return PartialView(Views._ViewPage1);
```

اینجا دیگر نیازی به ذکر کلاس MVC نبوده و می‌توان کار را با کلاس جدید Views شروع کرد. یا اگر نیازی به ذکر نام اکشن متدی در کنترلر جاری بود می‌توان از ActionNames مستقیما استفاده کرد.
و یا بجای:

```
return RedirectToAction(actionName: "Index", controllerName: "Menu");
```

می‌توان نوشت:

```
return RedirectToAction(actionName: MVC.Menu.ActionNames.Index, controllerName: MVC.Menu.Name);
```

و یا حتی به صورت خلاصه‌تر به نحو زیر:

```
return RedirectToAction(result: MVC.Menu.Index());
```

این overload جدیدی است که با T4MVC به پروژه اضافه می‌شود و یک Action Result را می‌تواند بپذیرد. به این ترتیب به صورت خودکار نام کنترلر و متد Index آن تنها در یک پارامتر لحاظ می‌شوند. اگر نیاز به ارسال پارامتری هم به متد Index وجود داشت، همینجا می‌توان اینکار را انجام داد (بجای استفاده از anonymously typed objects متداول).

و یا بجای مسیر دهی به شکل زیر:

```
return PartialView("~/Views/CommentsArchive/_LatestCommentsInfo.cshtml", data);
```

می‌توان نوشت:

```
return PartialView(MVC.CommentsArchive.Views._LatestCommentsInfo, data);
```

همچنین برای مسیر دهی viewهای قرار گرفته در پوشه shared می‌توان از MVC.Shared.Views شروع کرد.

امکان معرفی بهتر عناصر استاتیک سایت

این مورد نیز بسیار جالب توجه است. توسط کلاس Links آن می‌توان به محتویات استاتیک (تصاویر، فایل‌های css و غیره) پوشه‌های Content و Scripts هم دسترسی یافت و حتی این موارد را نیز refactor کرد:

```
<img src="@Links.Content.Images.arrow_right_png" alt="arrow" />
<script src="@Links.Scripts.jquery_1_5_1_min_js" type="text/javascript"></script>
```

امکان تعریف بهتر پارامترها و مقادیر route

بجای اینکه routeValues را همانند سابق با anonymously typed objects مقدار دهی کنیم:

```
Html.ActionLink(linkText: "عنوان",
                      actionName: "Index",
                      controllerName: "Comments",
                      routeValues: new
                      {
                           userName = @Model.FriendlyName
                      },
                      htmlAttributes: null))
```

اینبار می‌توان نوشت:

```
Html.ActionLink(linkText: "عنوان",
                      result: MVC.Comments.Index(userName: @Model.FriendlyName)
                      htmlAttributes: null))
```

در overload جدیدی که ملاحظه می‌کنید، هم دسترسی به متد Index یک کنترلر از نوع strongly typed است و هم امکان تعریف پارامترهای آن به نحو منطقی‌تری فراهم شده است (متد Comments.Index واقعا وجود خارجی داشته و پارامتری از نوع userName را می‌پذیرد).

چند نکته جانبی

-این ابزار بر اساس Reflection کار می‌کند (البته فقط در حین تشکیل خودکار کلاس‌های مورد نیاز؛ وگرنه ثوابتی را که ایجاد می‌کند کامپایل شده و در زمان اجرا سرباری را به برنامه اضافه نمی‌کنند). بنابراین اگر کلاسی به پروژه اضافه شده است، کامپایل کردن آن‌را فراموش نکنید.

-اگر تغییری در فایل‌های View، در تعداد و نام آن‌ها صورت گرفت، روی فایل T4MVC.tt کلیک راست کرده و گزینه‌ی اجرای آن‌را انتخاب کنید. پس از این‌کار، مجددا کامپایل پروژه را فراموش نکنید.

-در فایل T4MVC.tt.settings.t4 یک سری تنظیمات پیش فرض قرار دارند. برای مثال اگر علاقمندید که به این فایل‌های تولید شده خودکار، فضای نام سفارشی خاصی را اضافه کنید می‌شود آرایه ReferencedNamespaces آن‌را مقدار دهی کرد.

- overloadهای جدید ActionResult دار آن نسبت به نمونه‌های استاندارد موجود، بسیار منطقی‌تر به نظر می‌رسند.
- توضیحات کامل امکانات T4MVC را در مستندات رسمی آن می‌توانید مطالعه کنید.

و ... اگر یک مدت با آن کار کنید خواهید گفت: «من قبلا چطور با ASP.NET MVC کار می‌کردم؟!»

نظرات خوانندگان

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۱:۱۶

خیلی ممنون ... واجب شد که ازش استفاده کنیم دیگه :دی

نویسنده: یوسف نژاد
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۲:۵

با تشکر. عالی بود. مخصوصا بخش «من قبلا چطور با ASP.NET MVC کار می‌کردم؟!» واقعا.

نویسنده: افشار محبی
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۶:۱

عاشق روش‌های Refactoring Friendly هستم! در یک پروژه در دست انجام MVC بلافاصله شروع به استفاده از این ابزار کردیم... ابزار خوبی به نظر می‌رسد.

نویسنده: شهروز جعفری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۳۳

اگه در routeValues بخواهیم چند پارامتر بدم چ جوری باید عملل کرد من به صورت زیر میدم syntax error میده

```
@Html.ActionLink(linkText: item.Name,result: MVC.Word.Details(id : item.ID, name : item.Name.ToSeoUrl()
))
```

نویسنده: شهروز جعفری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۱۹:۳۷

پدا کردم مرسی

```
Html.ActionLink("Delete Dinner", MVC.Dinners.Delete(Model.DinnerID)
    .AddRouteValue("foo", 17)
    .AddRouteValue("bar", "abc"))
```

نویسنده: ali
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۱:۱۴

سلام یک سوال داشتم

من از طریق jquery یک iupdatio رو صفحه انجام دادم.

```
<script type="text/javascript">
        function LaodWordInfo(id) {
            showProgress();
            $.ajax({
                type: "Post",
                url: "test/Info",
                data: JSON.stringify({ ID: id }),
                contentType: "application/json; charset=utf-8",
                dataType: "json",
                complete: function (xhr, status) {
                    var data = xhr.responseText;
                    if (status === 'error' || !data) {

                    } else {
                        var dialog = $("#dialog");
                        dialog.html(data);
                        dialog.dialog("open");
                    }
                    hideProgress();
```

```
                    return false;
                }
            }
);
        }

        function showProgress() {
            $('#Progress').css("display", "block");
        }

        function hideProgress() {
            $('#Progress').css("display", "none");
        }
        $(function () {
            $("#dialog").dialog({

                autoOpen: false,
                show: "fade",
                hide: "fade",
                width: 550,
                title: "WordInfo",
                resizable: false
            });
        });
    </script>
```

این باید یه دیالوگ پر کنه نمایش بده.
ولی از وقتی T4MVC استفاده کردم تو کروم درست نشون میده ولی فایرفاکس error زیرو میده
"NetworkError: 404 Not Found - http://localhost:6012/test/test/Info"
به آدرس دقت کنید دوبار test اورده

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۱:۲۴

در کد فوق که از T4MVC استفاده نشده ولی نحوه تعریف Url شما اشتباه است. قبلا هم در مورد نحوه تعریف صحیح Url در ASP.NET MVC مطلب نوشتم . حتما باید از Url.Action استفاده کنید.
برای اینکه بتونید از Url.Action در اسکریپت‌های خودتون استفاده کنید پایین View جاری یک section به نام مثلا JavaScript درست کنید. در اینجا می‌شود داخل کدهای جاوا اسکریپتی هم از Razor استفاده کرد. سپس این section را در layout در قسمت header صفحه include کنید (در قسمت 14 سری MVC این سایت در مورد این تکنیک کاملا توضیح دادم).

نویسنده: رضا.ب
تاریخ: ۱۳۹۱/۰۴/۱۲ ۲:۴۱

نمونه‌های شبیه این باز هم بودند که مسئله‌ای که سالهاست دات نت با Reflection قابل حل کرده. ولی بازم تو جدیدترین تکنولوژی‌ها از مقادیر رشته‌ای استفاده میکنه. آیا هدف خاصی پشت این منطق هست؟ یا صرفا بی‌توجهی و قصور تیم توسعه هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۲ ۸:۳۶

Reflection اندکی کند است. در حین توسعه برنامه اگر ویژوال استودیو مرتبا بخواهد کوئری‌های مبتنی بر reflection از اسمبلی‌های شما بگیرد، سرعت کار شما کم خواهد شد و نهایتا ممکن است سبب ناپایداری آن هم بشود (اگر پروژه جاری بزرگ باشد).

نویسنده: افشار محبی
تاریخ: ۱۳۹۱/۰۴/۱۲ ۲۳:۷

در پروژه ما کنترلرها در یک class library جداگانه نگهداری می‌شوند. در این حالت T4MVC دچار مشکل می‌شود...

http://forums.asp.net/p/1683368/5049673.aspx

من T4MVC را به خود class library محل نگهداری Controllerها منتقل کردم و مشکلاتم حل شد.

نویسنده: حسین توسلی
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۷:۱۴

با سلام و خسته نباشید خدمت دوستان
آقا من نتونستم این فایل‌ها رو دانلود کنم، خواهش میکنم یه راهی بگین تا منم بتونم دانلودشون کنم.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۸:۴۷

از اینجا دریافت کنید.

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۱/۰۸/۲۰ ۸:۵۹

مهندس سلام.
من موقع اضافه کردن T4mvc از طریق نوگت تحت mvc4 فایلی که T4MVC.tt.settings.t4 نام داشته باشد را نمی‌بینم. کجای کارم ایراد دارد؟ با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۲۰ ۹:۱۰

اگر علاقمند هستید که اخبار آن‌را دنبال کنید نیاز است سورس کنترل آن‌را مرور کنید:
http://t4mvc.codeplex.com/SourceControl/list/changesets

```
24cc121c697f, by Michael Swain, Oct 22 8:39 PM

XmlSettings Patch

Patch to modify T4MVC to use an XML settings file instead of a static
include. This allows for future compatability as long as setting types
do not change.
```

نام جدید آن **xml**. T4MVC.tt.settings شده.
+ این فایل به همراه بسته NuGet آن نیست و باید از سورس کنترل دریافت شود (داخل پوشه T4MVCHostMvcApp\T4MVC Files).

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۱/۰۸/۲۰ ۹:۱۹

سپاس مهندس.

نویسنده: SMF
تاریخ: ۱۳۹۱/۱۲/۱۰ ۱۰:۳۲

با سلام

من تو V.S. 2012 تعدادی کنترل جدید به پروژه اضافه کردم ولی T4MVC لیست نمی‌کردشون، کامپایل مجدد کل سولوشن هم کمکی نکرد؛ بعد روی فایل tt راست کلیک و Run Custom Tool که زدم T4MVC خودشو به روز کرد؛ گفتم شاید به درد دوستانی که به مشکل مشابه برخوردن، بخوره ...

نویسنده: cooli
تاریخ: ۱۳۹۱/۱۲/۱۷ ۱۲:۵۰

دو سوال
1 - در ورژنی که بنده از NuGet نصب کردم (3.5.1) ، T4MVCExtensions هم نصب گردید ، کار T4MVCExtensions چیست ؟
2 - بعد از اجرا ، تمام کلاس‌های Controllerهای پروژه من partial شدند و تمام Actionهای من virtual ، چرا ؟ آیا مشکلی در آینده برای Controllerها و Actionها پیش نخواهد آمد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۱۷ ۱۲:۵۶

- یک سری متد کمکی اضافی برای کار با T4MVC که به صورت پیش فرض در MVC وجود ندارد؛ خصوصا برای کار با متدهایی که پارامتر Result دارند و متدهایی مانند MVC.Comments.Index را قبول می‌کنند.
- خیر. این روش کلی کار کتابخانه T4MVC است برای فراهم آوردن امکان overriding مسایل توکار خودش به همراه افزودن یک سری تعاریف قابل استفاده در کنترلر جاری. مثلا زمانی که می‌نویسند View.MyPartialViewName، نیاز است کنترلر جاری partial تعریف شود تا به رشته یاد شده که توسط T4MVC اضافه می‌شود دسترسی یافت.
در کل کار کردن با T4MVC علاوه بر دسترسی strongly typed به مسیرها و فایل‌های موجود پروژه، سرعت نهایی رندر شدن را هم بالا می‌برد؛ چون مسیرهایی را که ایجاد می‌کند، نیاز به کمترین حدس‌زدن و سعی و خطا را توسط موتور View جاری برای یافتن منبع مورد نظر دارد.

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۲/۲۰ ۹:۱۶

فقط من پوشه رو اضافه کردم ولی زمانی که میخام مثل شما تو اکشن لینک بنویسم نمیشناسه مسیرها رو.خودش نمیاره مثل میخام به شاخه ویو دسترسی داشته باشم اصلا نمیشناسه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۰ ۹:۱۹

run custom tool رو اجرا کردید؟

نویسنده: مولانا
تاریخ: ۱۳۹۲/۰۱/۲۶ ۱۱:۵۹

با سلام.
من T4MVC را بر روی پروژه اصلی قرار دادم.

فایل‌های resource من در یک پروژه جدا قرار دارند. مدلهای پروژه من هم در یک پروژه جدا قرار دارند. حال برای تعریف خاصیت اعتبارسنجی روی یکی از خاصیت‌های ویومدل آیا می‌شود T4MVC را بکار برد؟ (بجای استفاده از magic string)

```
[StringLength(25, ErrorMessageResourceType = typeof(ValidationErrorsResource), ErrorMessageResourceName
= "InvalidStringLength")]
public string FatherName { get; set; }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۲۶ ۱۲:۴۰

از T4ResX استفاده کنید.

نویسنده: مولانا
تاریخ: ۱۳۹۲/۰۱/۳۱ ۱۶:۱۸

با سلام.
من پوشه ای با عنوان SiteContent در root پروژه دارم. چگونه می‌توانم از طریق T4MVC به آن دسترسی داشته باشم. درحالت عادی وقتی Links را تایپ میکنم فقط منابع را برای پوشه Content و Scripts شناسایی میکند. باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۳۱ ۱۷:۳۹

مراجعه کنید به راهنمای مفصل آن . نقل قولی از آن:

```
Use StaticFilesFolders to tweak the list of folders from which links are generated

<!-- Folders containing static files for which links are generated (e.g. Links.Scripts.Map_js) -->
<StaticFilesFolders>
    <FileFolder>Scripts</FileFolder>
    <FileFolder>Content</FileFolder>
</StaticFilesFolders>
```

نویسنده: علی
تاریخ: ۱۳۹۲/۰۴/۱۵ ۱۹:۴۶

با سلام بنده داخل پروژه ام از t4mvc استفاده کردم امروز صبح یک partialview اضافه کردم وقتی روی T4MVC.tt راست کلیک کردم و run custom tool را زدم با خطای :

```
Error1Running transformation: System.ArgumentException: The file 'T4MVC.tt' is not open.
   at System.Runtime.InteropServices.Marshal.ThrowExceptionForHRInternal(Int32 errorCode, IntPtr
errorInfo)
   at System.Runtime.InteropServices.Marshal.ThrowExceptionForHR(Int32 errorCode)
   at Microsoft.VisualStudio.NativeMethods.ThrowOnFailure(Int32 hr, Int32[] expectedHRFailure)
   at Microsoft.VisualStudio.Shell.RunningDocumentTable.FindDocument(String moniker, IVsHierarchy&
hierarchy, UInt32& itemid, UInt32& docCookie)
   at Microsoft.XmlEditor.XmlHelper.GetCanonicalNameWithoutView(IServiceProvider site, String fname)
   at Microsoft.XmlEditor.XmlFileInfoService.RenameOrRemoveCacheFileInfo(String fname)
   at Microsoft.XmlEditor.Package.GetOrCreateFileInfo(String fname, String text)
   at Microsoft.XmlEditor.Package.GetOrCreateFileInfo(String fname, IVsTextLines buffer)
   at Microsoft.XmlEditor.FactoryBase.GetDesignerFactory(String monikerTheFileOnDisk, String
monikerToUse, IVsTextLines existingBuffer, Guid logicalView)
   at Microsoft.XmlEditor.Chooser.ChooseEditorFactory(String pszMkDocument, IVsHierarchy pHier, UInt32
itemid, IntPtr punkDocDataExisting, Guid& rguidLogicalView, Guid& pguidEditorTypeActual, Guid&
pguidLogicalViewActual)
   at EnvDTE.ProjectItem.Open(String ViewKind)
   at
Microsoft.VisualStudio.TextTemplating07E2522337018FCFC0AADC805726764A22758F53E05D3026845E2AA5E0226E57B7
7322743831B8A035E8EFE18D3CC1F808985A2958CCE46A2085D3520D2EC605.GeneratedTextTransformation.XmlSettings.
Init(ITextTemplatingEngineHost host) in e:\Project\GRP\GRP.Web\T4MVC.tt:line 2120
   at
Microsoft.VisualStudio.TextTemplating07E2522337018FCFC0AADC805726764A22758F53E05D3026845E2AA5E0226E57B7
7322743831B8A035E8EFE18D3CC1F808985A2958CCE46A2085D3520D2EC605.GeneratedTextTransformation.XmlSettings.
Load[T](ITextTemplatingEngineHost host) in e:\Project\GRP\GRP.Web\T4MVC.tt:line 2093
   at
Microsoft.VisualStudio.TextTemplating07E2522337018FCFC0AADC805726764A22758F53E05D3026845E2AA5E0226E57B7
7322743831B8A035E8EFE18D3CC1F808985A2958CCE46A2085D3520D2EC605.GeneratedTextTransformation.MvcSettings.
Load(ITextTemplatingEngineHost host) in e:\Project\GRP\GRP.Web\T4MVC.tt:line 1786
   at
Microsoft.VisualStudio.TextTemplating07E2522337018FCFC0AADC805726764A22758F53E05D3026845E2AA5E0226E57B7
7322743831B8A035E8EFE18D3CC1F808985A2958CCE46A2085D3520D2EC605.GeneratedTextTransformation.TransformTex
t() in e:\Project\GRP\GRP.Web\T4MVC.tt:line 39e:\Project\GRP\GRP.Web\T4MVC.tt21201GRP.Web
```

مواجه شدم t4mvc را uninstall کردم دوباره نصب کردم موقع generate دوباره همین پیغام را میدم. لطفا یکی به من کمک کنه.ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۵ ۲۰:۱۳

مشکلات مرتبط رو اینجا گزارش کنید .

نویسنده: محمد آزاد
تاریخ: ۱۳۹۲/۰۴/۱۶ ۱۴:۵۹

یکی از مشکلاتی که من با این ابزار خوب دارم اینه که بعد از هر تغییر در کنترلرها یا اضافه کردن یک کنترلر جدید حتما باید run custom tool اجرا کنم...می خواستم بدنم آیا راهی نیست که بشه این قضیه خودکار انجام بشه ؟ (مثلا بعد از Build پروژه)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۴/۱۶ ۱۶:۲۷

add-in برای اینکار هست. ولی به شخصه ترجیح می‌دهم خودکار اجرا نشود؛ چون در یک پروژه با تعداد فایل‌های بالا، پردازش آن مدتی زمان می‌برد و ممکن است در این بین، تغییرات انجام شده شامل تغییری در تعداد یا نام فایل‌ها یا اکشن متدها نباشند. تغییرات در سایر لایه‌ها باشند. در این حالت اصلا ضرورتی ندارد که با هر Build یکبار اینکار تکرار شود.

نویسنده: کامی
تاریخ: ۱۳۹۲/۰۵/۲۶ ۰:۱۰

سلام؛ من از T4 استفاده کردم ولی وقتی پروژه رو publish میکنم و اون رو روی هاست upload می کنم t4 رو نمی‌شناسه و تمام مواردی رو که از t4 استفاده کردم رو error میده. آیا باید تنظیمات خاصی انجام بدم که این مشکل رو حل کنم؟ لطفا راهنمایی کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۲۶ ۰:۲۶

کل کاری که T4MVC انجام می‌ده، اضافه کردن یک سری متد الحاقی جدید است به پروژه به علاوه تبدیل یک سری از مسیرها به معادل‌های constant آن و اندکی تغییر در متدهای کنترلرها برای سازگار شدن با متدهای الحاقی جدید. بنابراین حاصل این عملیات به صورت کامپایل شده در پوشه bin پروژه شما موجود است. این پوشه را کلا بر روی سرور ارسال کنید.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۰۸/۲۲ ۰:۴۸

«من قبلا چطور با ASP.NET MVC کار می‌کردم؟!»

نویسنده: رویا حیدری
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۷:۱۶

من در پروژم Controller ها رو در یک لایه و Viewها را در لایه‌ی دیگری دارم. از T4MVC هم در لایه‌ی Controller استفاده کردم. اما بعد از زدن Run Custom Tool مسیرهایی که منتهی به یک view میشه ساخته نمیشه و Error میده! (آنهایی که RedirectToAction هستند به درستی ساخته می‌شه)

```
VDoc.Web.Controllers.TopicController.ViewsClass' does not contain a definition for '_Edit' and no
extension method '_Edit' accepting a first argument of type
'VDoc.Web.Controllers.TopicController.ViewsClass' could be found (are you missing a using directive or
an assembly reference?)
```

آیا برای استفاده از T4 در لایه‌های مختلف باید کار خاصی انجام بدم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۳ ۱۸:۴

- ViewsRootFolder داخل فایل تنظیمات آن، قابل تنظیم است.
- همچنین بحثی هم در اینجا شده با یک چنین راه حلی

نویسنده: احمد ولی پور
تاریخ: ۱۳۹۳/۰۷/۱۵ ۰:۴۲

من سه تا فایل T4MVC.tt و T4MVC.tt.settings.xml و حتی T4MVC.tt.hooks.t4 رو تو یه پوشه تو سطح اول میذازم ولی همش خطای زیر رو میده:

Could not load file or assembly 'System.Web.Mvc, Version=3.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35' or one of its dependencies. The located assembly's manifest definition does (not match the assembly reference. (Exception from HRESULT: 0x80131040

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۱۵ ۱:۱۸

تنظیمات فایل‌های کانفیگ برنامه شما (bindingRedirect ها) صحیح نیست یا ناقص است. اطلاعات بیشتر

نویسنده: reza110
تاریخ: ۱۳۹۳/۰۸/۲۸ ۱۵:۴۳

با سلام
T4MVC یکسری تغییرات در برنامه ما ایجاد می‌کنه از جمله مثلا یک کانستراکتور برای کنترلر Home اضافه می‌کنه و من چگونه باید کدی مثل کد زیر رو داخل کنترل ام بنویسیم که خطای تکراری نگیره؟

```
private readonly ICacheService _cacheService;
public HomeController() : this(MvcApplication.CacheService)
{
}
public HomeController(ICacheService cacheService)
{
      _cacheService = cacheService;
}
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۸ ۱۸:۱۳

- روش « poor man's dependency injection » اساسا زیر سؤال هست. روش صحیح تزریق وابستگی‌ها در برنامه‌های ASP.NET MVC در اینجا بحث شده‌است.
- اگر نمی‌خواهید T4MVC سازنده‌ی پیش فرض ایجاد کند، فایل‌های T4 آن‌را ویرایش کنید .

عنوان: **مدیریت ساده چهار عمل اصلی تکراری داده در صفحات - قسمت 2**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۴/۱۱ ۲۱:۴۶
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

در ادامه این بحث مثال محصولات برای ثبت و ویرایش و حذف محصول با توجه به متدهایی که باید پیاده سازی شود:

```
public partial class product : PageStateMachine
 {
  protected void btnSave_Click(object sender, EventArgs e)
  {
    Go(PageState.Save);
  }
  protected override ... Save()
  {
    try
    {
        if (Id > 0)
        {
            // get from bank and update
        }
        else
        {
            // create new one and add it to context
        }
    }
    catch (Exception ex)
    {
        // handle exception
    }
  }

  protected void btnSelect_Click(object sender, EventArgs e)
  {
    Go(PageState.Select);
  }
  protected override ... Select()
  {
    try
    {
      // get from bank and show in details
    }
    catch (Exception ex)
    {
        // handle exception
    }
  }

  protected void btnDelete_Click(object sender, EventArgs e)
  {
    Go(PageState.Delete);
  }
  protected override StateMachineMessage Delete()
  {
    try
    {
      // remove from bank
    }
    catch (Exception ex)
    {
        // handle exception
    }
  }
 }
```

به این ترتیب روال ذخیره و بازیابی و ویرایش و حذف به صورتی توسط PageStateMachine مدیریت شده و کافیست متدها در سمت کد پشت صفحه بدرستی نوشته شود .

عنوان: **مقدمه ای بر AutoMapper**
نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۷:۲۲
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net MVC, AutoMapper, ASP.Net, MVC

AutoMapper کتابخانه ای ساده و سبک برای نگاشت اطلاعات یک شی به شی دیگر به صورت خودکار هست و...
اگه این پست رو مطالعه کرده باشید یه مشکل امنیتی بنام «Mass Assignment» مطرح شد.برای رفع این مشکل یک روش استفاده از ViewModel بود.
فرض کنید Model ما

```
public class User
    {
        public int Id { get; set; }

        public string FirstName { get; set; }

        public string LastName { get; set; }

        public string UserName { get; set; }

        public string Password { get; set; }

        public bool IsAdmin { get; set; }

        public virtual ICollection<BlogPost> BlogPosts { get; set; }
    }
```

و ViewModel ما

```
public class UserViewModel
    {
        public string FirstName { get; set; }

        public string LastName { get; set; }

        public string Password { get; set; }
    }
```

باشه(توجه کنید در واقع من برای View ی مورد نظرم فقط به نام , نام خانوادگی و پسورد نیاز دارم)
برای استفاده UserViewModel بعنوان Model در View ی مورد نظر باید شی UserViewModel رو با اطلاعات شی User مقدار دهی کنیم مثلا با کدی مثل این در کنترلر.

```
public ActionResult Index(int id = 1)
        {
            var user = _userService.GetById(id);
            var userViewModel = new UserViewModel
                {
                    FirstName = user.FirstName,
                    LastName = user.LastName,
                    Password = user.Password
                };

            return View(userViewModel);
        }
```

رهایی از نوشتن اینجور کدهای تکراری و خسته کننده باعث پیدایش AutoMapper شد...
برای استفاده از AutoMapper از نوگت استفاده میکنیم.

```
PM> Install-Package AutoMapper
```

در شروع برنامه نگاشت‌ها رو تعریف میکنم.یک روش ابداعی تعریف نگاشت‌ها در یک کلاس استاتیک و فراخوانی اون تو متد Application_Start هست.

```
public static class AutoMapperWebConfiguration
    {

        public static void Configure()
        {
            ConfigureUserMapping();
        }

        private static void ConfigureUserMapping()
        {
            Mapper.CreateMap<User, UserViewModel>();
        }
    }
```

اولین پارامتر نوع مبدا و دومین پارامتر نوع مقصد هست.
برای انجام نگاشت هم از متد Map استفاده میکنیم.

```
public ActionResult Index(int id=1)
        {
            var user = _userService.GetById(id);
            var userViewModel=new UserViewModel();
            AutoMapper.Mapper.Map(user, userViewModel);

            return View(userViewModel);
        }
```

همنطور که میبنید با نوشتن چندین خط کد عملیات نگاشت اطلاعات یک شی به شی دیگه انجام شد.
ادامه دارد...

نظرات خوانندگان

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۷:۳۲

دمت گرم مدتیه معطل همین مپینگ‌ها به صورت دستی هستم

نویسنده: سروش ترک زاده
تاریخ: ۱۳۹۱/۰۴/۱۳ ۱۷:۳۳

سلام
ممنون،
راه حل ساده واقعا لذت بخش هست...

نویسنده: daniel
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۱:۱۳

تو Compact Framwork از valueinjecter باید استفاده کرد.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۳:۲

خیلی ممنونم ... میشه این سمپل رو ضمیمه کنید؟!

نویسنده: رضا.ب
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۳:۴۶

تشکر بابت معرفی.
کاربرد این Auto Mapper در اینجور نگاشت‌ها معنی میده فقط؟ یعنی نگاشت یه شیِ جزء به شیِ کل؟ یا مثلا کاربردهایی مثل ORM هم میتواند داشته باشد؟

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۴ ۹:۲۱

یکی از مهمترین کاربردهاش نگاشت از یک مدل بزرگ و پیچیده به یک مدل ساده هست که بهش Flattening میگن و کاربردهای دیگه مثلا Projection و... که همنطور که آخر پست گفتم تو پست‌های بعدی تکمیلش میکنم.
برای مطالعه بیشتر همه به این ویکی مراجعه کنید.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۴/۱۴ ۱۲:۱۶

سمپل خاصی نیست کدها رو از یک پروژه کشیدم بیرون...
تنها قسمتی که تو این پست نذاشتم(و البته نیاز هم نبود) View ش بود.

```
@model DotnetDevBlog.Web.Models.UserViewModel

@{
    ViewBag.Title = "Index";
}

<h2>Index</h2>
@Html.DisplayForModel()
```

نویسنده: سعید یزدانی
تاریخ: ۱۳۹۲/۰۴/۰۲ ۱۶:۱۹

با سلام
ممنون از مطلب مفیدتون
میخواستم بدونم اگه از چند model بخواهیم استفاده کنیم باید چکار کنیم؟

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۴/۰۴ ۱۰:۵۸

از این کلاس کمکی استفاده کن

نویسنده: hossein101211
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۳:۵۸

با سلام میخواستم ببینم آیا امکان استفاده از automapper بدین صورت هم وجود داره؟

```
public void Edit_news_ajax(News news)
        {
            using (var Context = new ProCamContext())
            {
                var q=Context.News.Find(news.id);
                AutoMapper.Mapper.Map(news,q);
               //NewsRepository.EditNews(news.id, q);
                Context.SaveChanges();
            }//end news
        }
```

با تشکر

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۴:۳۰

اگه q از نوع news هست که نیاز ی به AutoMapper نداری !

```
AutoMapper.Mapper.Map(news,q);
```

این خط مشکلی نداره

نویسنده: hossein101211
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۵:۵

من میخوام محتویات news رو (که توسط model binding پر شده) رو در سطری از جدولم بریزم (با اون آیدی که با find پیدا کردم جایگزین بشه) اینکار با AutoMapper امکان پذیر هست؟
برای اینکار نیاز به create.map هم دارم؟
با تشکر

نویسنده: محمد صاحب
تاریخ: ۱۳۹۲/۰۴/۱۱ ۱۱:۲۷

اگه نوع هاشون متفاوته بله باید Mapper.CreateMap رو تعریف کنی.

عنوان: **بهبود سرعت نمایش صفحات در ASP.NET MVC با حذف View Engines اضافی**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۳:۴۳
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, ASP.Net MVC

در ASP.NET MVC امکان استفاده از چند View Engine به صورت همزمان وجود دارد و همچنین هربار که قرار است Viewایی رندر شود، از تمام این‌ها تا یافتن موتور مناسب نمایش View جاری کوئری می‌گیرد. بدیهی است هرچقدر تعداد موتورهای ثبت شده در اینجا بیشتر باشند، زمان بیشتری نیز برای یافتن موتور نمایشی مناسب صرف خواهد شد؛ خصوصا اگر موتور مناسب در آخر لیست ثبت شده باشد.

در ASP.NET MVC 3 دو موتور نمایشی به صورت پیش فرض نصب هستند (WebForms and Razor). بنابراین اگر صرفا از Razor استفاده می‌کنید، می‌توان موتور اول را کلا از سیستم پردازشی برنامه حذف کرد. برای اینکار تنها کافی است در فایل global.asax.cs برنامه بنویسیم:

```
protected void Application_Start() {
    ViewEngines.Engines.Clear();
    ViewEngines.Engines.Add(new RazorViewEngine());
    ...
}
```

این موارد را توسط Glimpse بهتر می‌توان بررسی کرد. Glimpse یک پروفایلر سمت سرور ASP.NET است و دارای نسخه مخصوص ASP.NET MVC نیز می‌باشد. برای نصب آن باید از طریق NuGet اقدام کرد و حتما دقت داشته باشید که نسخه MVC آن باید نصب شود تا برگه‌های Routing و View آن ظاهر شوند.

پس از نصب از طریق NuGet، به صورت خودکار اسمبلی‌های لازم به پروژه اضافه شده و همچنین فایل web.config برنامه نیز ویرایش می‌شود. در انتهای این فایل سطر ذیل مشخص می‌کند که Glimpse فعال باشد یا خیر.

```
<glimpse enabled="true" />
```

پس از نصب، برنامه را اجرا کرده و به آدرس http://localhost/ **glimpse.axd** مراجعه کنید تا صفحه تنظیمات آن ظاهر شود. تنها کاری که باید در اینجا صورت گیرد کلیک بر روی دکمه Turn Glimpse On است.

به این ترتیب یک کوکی به مرورگر اضافه شده و اکنون پس از بازگشت به صفحه اصلی برنامه و refresh کامل صفحه، در کنار سمت راست پایین صفحه، آیکن آن ظاهر خواهد شد.

بر روی این آیکن کلیک نمائید تا در برگه‌ی View آن، انواع Viewهایی که درگیر نمایش صفحه جاری بوده‌اند، مشخص شوند:

همانطور که ملاحظه می‌کنید در اینجا دو موتور پیش فرض فعال بوده و پس از سعی و خطای صورت گرفته، در انتهای کار Razor انتخاب شده است. اکنون اگر نکته حذف موتورهای نمایشی اضافی را اعمال کنیم به تصویر زیر خواهیم رسید:

/

Ajax | Binding | Config | Environment | Execution | MetaData | Remote | Request | Routes | Server

Ordinal	Requested View	Master Override	Partial	View Engine	Check Cache	Found	Details
1	Index		False	RazorViewEngine	True	False	Not Found In: RazorViewEngine cache
2	Index		False	RazorViewEngine	False	True	Key / Value: ViewData --, Model --, TempData --

هم تعداد سعی و خطاها کمتر شده و هم تعداد فایل‌هایی که بررسی شده است به حداقل رسیده (برای مثال در حالتیکه موتور WebForms فعال باشد، چهار فایل با پسوندهای مختلف در مکان‌های پیش فرض نیز حتما جستجو خواهند شد).

نظرات خوانندگان

نویسنده: شهروز جعفری
تاریخ: ۱۳۹۱/۰۴/۱۳ ۲۳:۵۸

یه سوال شاید نه چندان حرفه ای
شما گفتید که ا مکان استفاده از چند View Engine به صورت همزمان وجود دارد .آیا میشه تو یک ویو این کارو انجام داد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۴ ۰:۱

خیر. همزمانی در اینجا به معنای امکان استفاده همزمان از فایل‌های مستقل aspx و cshtml یا vbhtml است.

نویسنده: شهروز جعفری
تاریخ: ۱۳۹۱/۰۴/۱۴ ۰:۱۹

یه سوال نامرتبط Ninject چیه؟کاره همون structuremap میکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۴ ۰:۲۷

بله. از این نوع کتابخانه‌ها زیاد است. Ninject، StructureMap، AutoFac و غیره. خود مایکروسافت هم یک نمونه دیگر به نام Unity دارد.

عنوان: **نحوه ایجاد یک اسلایدشو به صورت داینامیک**
نویسنده: پویا امینی
تاریخ: ۱۳۹۱/۰۴/۱۵ ۱۷:۲۹
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, jQuery, DynamicSlideShow

در این آموزش قصد دارم نحوه ایجاد یک Slideshow به صورت داینامیک را با ASP.NET طراحی کنم(منظور از ایجاد Slideshow به صورت داینامیک این است که عکس‌ها را به صورت داینامیک از DB بخواند).

اولین گام این است که Plugin مورد نظر را دریافت کنید که من از پلاگین Orbit استفاده کرده ام

ابتدا یک DataBase با نام DynamicSlideShow ایجاد و یک جدول با ساختار زیر با نام Picturesدرون آن ایجاد می‌کنیم

Id	int	☐
PicturePath	nvarchar(50)	☐
PictureText	nvarchar(200)	☑
		☐

گام بعدی ایجاد یک پروژه Asp.Net با زبان C# و اضافه کردن فایلهای پلاگین به پروژه و ایجاد یک Handler برای بازیابی داده‌ها از DB است. ساختار Solutionما باید به صورت زیر باشد

برای اینکه بتوانیم با DB ارتباط برقرار کنیم از EF استفاده می‌کنیم به همین منظور ابتدا یک Model به نام DynamicSlideShowModel ایجاد می‌کنیم

در گام بعد بر روی GenerateFromDatabase انتخاب کرده و بر روی دکمه Next کلیک می‌کنیم و در مراحل بعد ابتدا DB مربوط به مثال (DynamicSlideShow) و جدول آن را انتخاب می‌کنیم . حال یک Model به درون پوشه App_Code اضافه شده است

در ادامه برای واکشی رکوردهای موجود در جدول Pictures کدهای زیر را درون Handler می‌نویسیم

```
var ctx = new DynamicSlideShowEntities();
  var list = ctx.Pictures.ToList();
  string str = JsonConvert.SerializeObject(list);
  context.Response.Write(str);
```

در این کدها تنها نکته‌ای که احتیاج به توضیح دارد این است که ابتدا یک لیست از رکوردها را از جدول Pictures واکشی می‌کنیم و برای پاس دادن به کلاینت ما آن‌ها را به فرمت Json تبدیل کرده ایم که برای تبدیل از کتابخانه آماده Newtonsoft.Json.dll استفاده کرده ایم

حال باید با استفاده از jQuery کدهای درون Handler را اجرا کنیم؛ برای همین منظور یک صفحه با نام default.aspx ایجاد کرده و کدهای زیر را درون آن می‌نویسیم

```
<head runat="server">
    <title>Dynamic SlideShow</title>
    <link href="CSS/orbit-1.2.3.css" rel="stylesheet" type="text/css" />
    <script src="Script/jquery-1.7.1.min.js" type="text/javascript"></script>
    <script src="Script/jquery.orbit-1.2.3.min.js" type="text/javascript"></script>
    <script type="text/javascript">
        $(function () {
            $.ajax({
                url: "Handler.ashx",
                contentType: "application/json; charset=utf-8",
                success: function (data) {
                    $.each(data, function (i, b) {
                        var str = '<img src="' + b.PicturePath + '" alt="' + b.PictureText + '"/>';
                        $("#featured").append(str);
                    });
                    $('#featured').orbit();
                },
                dataType: "json"

            });
            });
    </script>
</head>
<body>
    <form id="form1" runat="server">
    <div id="featured">
    </div>
    </form>
</body>
```

در اینجا ابتدا با استفاده از متد ajax کتابخانه jQuery کدهای درون Handler را اجرا کرده و به ازای هر المان موجود در جدول یک تگ img به صفحه اضافه می‌کنیم.

نظرات خوانندگان

نویسنده: مسعود زیانی
تاریخ: ۱۳۹۱/۰۴/۱۷ ۱۰:۴۶

خیلی عالی بود ممنون

نویسنده: میثم
تاریخ: ۱۳۹۱/۰۴/۱۹ ۱۳:۳۱

مرسی خیلی عالی بود ، این عکس‌ها یکی یکی از سمت سرور اضافه می‌شوند یا همه با هم ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۹ ۱۳:۴۶

کاری که در اینجا انجام شده ساخت div زیر به صورت پویا است (بر اساس لیست تصاویر دریافت شده از سرور):

```
<div id="featured">
     <img src="img1.jpg" alt="img1" />
     <img src="img2.jpg" alt="img2" />
     <img src="img3.jpg" alt="img3" />
</div>
```

و سپس فراخوانی متد زیر روی آن:

```
$('#featured').orbit();
```

نویسنده: صادق
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۳:۱

با سلام
من کدهای بالا رو همه رو درست انجام دادم ولی در فایل handler متاسفانه Tolist(); نمیشه. به فایل ضمیمه دقت کنید. با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۳:۶

ذکر using System.Linq را فراموش کردید.

نویسنده: صادق
تاریخ: ۱۳۹۱/۰۶/۱۰ ۱۳:۲۷

ممنونم از شما استاد عزیز

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۱۴

یه سوال واسه من پیش اومد مهندس
ممکنه ما 100 تصویر داشته باشیم که بالطبع لود همه آنها یکباره درست نیست
آیا میشه کاری کرد که در هر درخواست که اسلایدشو تعویض میشه تصویر دینامیک لود کرد یا مثلا هر دفعه سه تصویر لود کرد و به پلاگین اضافه کرد تا نمایش بده؟
چون اینجوری همه تصاویر به یکباره لود شده و سرعت خیلی پایین میاد

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۲:۲۱

می‌تونید از ایده‌های افزونه‌هایی تحت عنوان « jQuery loading images on demand » استفاده کنید. مثلا: (^)

نویسنده: حمید خرازان
تاریخ: ۱۳۹۲/۰۱/۰۳ ۲۲:۵۷

سلام،مهندس جان من وقتی از این کد استفاده میکنم فقط رکورد اول رو واسم میاره و بقیه رو undefined میده، میدونید دلیلش چیه؟

نویسنده: مهدی آزادی
تاریخ: ۱۳۹۲/۰۱/۱۸ ۱۰:۳

عالی بود.

پاینده باشین.

نویسنده: پویا امینی
تاریخ: ۱۳۹۲/۰۱/۱۸ ۱۶:۳۴

دوست عزیز باید کدتون رو ببینم و نظر بدم.

نویسنده: Abolfazl
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۲:۲۸

با سلام ..
من همه چیز را همان طوری که شما گفتید انجام دادم ولی نمی‌دونم چرا اجرا نی کنه ..بعد از لود صفحه یک لحظه نشون میده بعد فقط فلش که برای حرکت تصویر است نشون میده و اون هم در گوشه بالای صفحه :
این هم کد :::

```
using System;
using System.Collections.Generic;
using System.Data.SqlClient;
using System.Linq;
using System.Web;


public class GallerySite
{
    int iD;

    public int ID
```

```
    {
        get { return iD; }
        set { iD = value; }
    }
    string imagePath;

    public string ImagePath
    {
        get { return imagePath; }
        set { imagePath = value; }
    }
    string imageText;

    public string ImageText
    {
        get { return imageText; }
        set { imageText = value; }
    }
    public List<GallerySite> GetImage()
    {
        List<GallerySite> _Gallery = new List<GallerySite>() { };
        SqlConnection cnn = new SqlConnection("Data Source=.;Initial
Catalog=RoshanZamirDataBase;Integrated Security=True");
        cnn.Open();
        SqlCommand cmd = new SqlCommand("Select * From Gallery", cnn);
        SqlDataReader datareadfewr = cmd.ExecuteReader();
        if (datareadfewr.HasRows)
        {
            while (datareadfewr.Read())
            {
                _Gallery.Add(new GallerySite()
                {
                    ID = Convert.ToInt32(datareadfewr["ID"]),
                    ImagePath = (string)datareadfewr["ImagePath"],
                    ImageText=(string)datareadfewr["ImageText"]
                });
            }
        }
        return _Gallery;
    }
}

.................................................................................................
.........................



    <link href="orbit-1.2.3.css" rel="stylesheet" />
    <script src="jquery-1.8.3.min.js"></script>
    <script src="jquery.orbit-1.2.3.min.js"></script>
    <script type="text/javascript">
        $(function () {
            $.ajax({
                url: "Handler.ashx",
                contentType: "application/json; charset=utf-8",
                success: function (data) {
                    $.each(data, function (i, b) {
                        var str = '<img src="' + b.ImagePath + '" alt="' + b.ImageText + '"/>';
                        $("#featured").append(str);
                    });
                    $('#featured').orbit();
                },
                dataType: "json"
            });
        });
    </script>



.................................................................................................
...................

Handler :



public class Handler : IHttpHandler {

    public void ProcessRequest (HttpContext context) {
```

```
        var _Gallery = new GallerySite();
        var List = _Gallery.GetImage();
        string str = JsonConvert.SerializeObject(List);
        context.Response.Write(str);
    }
    public bool IsReusable {
        get {
            return false;
        }
    }
}
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۳:۱۴

با نحوه دیباگ اسکریپت‌های jQuery آشنا هستید؟ افزونه فایرباگ رو نصب کنید. بعد مراجعه کنید به قسمت مثلا network آن. بعد مشاهده کنید، آیا چیزی رد و بدل شده؟ آیا در کنسول اسکریپت‌های مرورگر خطایی لاگ شده. آیا مثلا CSSهای سایت روی این افزونه تاثیر داشته. مرورگر کروم هم در قسمت developer tools آن شبیه به فایرباگ رو داره. دکمه F12 آن.

نویسنده: Abolfazl
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۴:۵۳

من نتونستم درستش کنم ..یه راه دیگه وجود نداره :؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۳ ۱۷:۳۷

کد رو دیباگ کن:

```
var str = '<img src="' + b.ImagePath + '" alt="' + b.ImageText + '"/>';
```

ببین اینجا نهایتا چه رشته‌ای تشکیل میشه.
آیا src درستی به مرورگر ارائه شده؟ مثلا آیا c:\path\data\image\01.jpg است؟ یا اینکه به نحو صحیحی به صورت یک مسیر نسبی از ریشه سایت مانند siteimage/01.jpg مقدار دهی شده؟

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۸:۳۹

سلام؛ من از روش بالا در mvc استفاده کردم می‌خواستم ببینم آیا کار صحیحی انجام دادم یا خیر ؟
در مورد Http Hndler و Http Module هم اطلاعات کمی دارم و آیا استفاده از این‌ها در mvc کار صحیحی است یا خیر ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۹:۲

نیازی نیست. مطلب فوق در مورد وب فرم‌ها نوشته شده. برای انتقال آن به MVC بجای فایل ashx یک اکشن متد با خروجی Json خواهید داشت.

عنوان: **محدود کردن کاربرها به آپلود فایل‌هایی خاص در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۶ ۰:۱
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Security, ASP.Net MVC

یک ادیتور آنلاین را تصور کنید که کاربران در قسمت ارسال تصویر آن قرار است فقط فایل‌های png، jpg و gif ارسال کنند و نه مثلا فایل test.aspx و موارد مشابه. در اینجا برای محدود کردن نوع فایل‌های آپلود شده می‌توان از فیلترهای سفارشی ASP.NET MVC کمک گرفت:

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web.Mvc;

namespace SecurityModule
{
    public class AllowUploadSpecialFilesOnlyAttribute : ActionFilterAttribute
    {
        readonly List<string> _toFilter = new List<string>();
        readonly string _extensionsWhiteList;
        public AllowUploadSpecialFilesOnlyAttribute(string extensionsWhiteList)
        {
            if (string.IsNullOrWhiteSpace(extensionsWhiteList))
                throw new ArgumentNullException("extensionsWhiteList");

            _extensionsWhiteList = extensionsWhiteList;
            var extensions = extensionsWhiteList.Split(',');
            foreach (var ext in extensions.Where(ext => !string.IsNullOrWhiteSpace(ext)))
            {
                _toFilter.Add(ext.ToLowerInvariant().Trim());
            }
        }

        bool canUpload(string fileName)
        {
            if (string.IsNullOrWhiteSpace(fileName)) return false;

            var ext = Path.GetExtension(fileName.ToLowerInvariant());
            return _toFilter.Contains(ext);
        }

        public override void OnActionExecuting(ActionExecutingContext filterContext)
        {
            var files = filterContext.HttpContext.Request.Files;
            foreach (string file in files)
            {
                var postedFile = files[file];
                if (postedFile == null || postedFile.ContentLength == 0) continue;

                if (!canUpload(postedFile.FileName))
                    throw new InvalidOperationException(
                        string.Format("You are not allowed to upload {0} file. Please upload only these
files: {1}.",
                                      Path.GetFileName(postedFile.FileName),
                                      _extensionsWhiteList));
            }

            base.OnActionExecuting(filterContext);
        }
    }
}
```

توضیحات کدهای فوق:

برای تهیه فیلتر محدود سازی نوع فایل‌های قابل ارسال به سرور، با ارث بری از ActionFilterAttribute شروع خواهیم کرد.
سپس با تحریف متد OnActionExecuting آن، توسط filterContext.HttpContext.Request.Files می‌توان به کلیه فایل‌های درحال ارسال به سرور در طی درخواست جاری، دسترسی یافت.
به این ترتیب از طریق مقدار خاصیت postedFile.FileName می‌توان به پسوند فایل در حال ارسال رسید و بر این اساس امکان

ارسال فایل‌های غیرمجاز را در نیمه راه با صدور یک استثناء سلب کرد.

برای استفاده از این فیلتر سفارشی تهیه شده نیز می‌توان به نحو زیر عمل کرد:

```
[AllowUploadSpecialFilesOnly(".jpg,.gif,.png")]
public ActionResult ImageUpload(HttpPostedFileBase file)
```

در اینجا پسوند فایل‌های مجاز قابل ارسال، توسط یک کاما از هم جدا خواهند شد.

یک نکته تکمیلی:

اگر کاربر قرار است تنها تصویر ارسال کند، بررسی پسوند فایل لازم است اما کافی نیست. برای این منظور می‌توان از کلاس Image واقع شده در فضای نام System.Drawing نیز کمک گرفت:

```
public static bool IsImageFile(HttpPostedFileBase photoFile)
{
    using (var img = Image.FromStream(photoFile.InputStream))
    {
         return img.Width > 0;
    }
}
```

در اینجا اگر فایل ارسالی تصویر نباشد، به صورت خودکار یک استثناء صادر خواهد شد.

نظرات خوانندگان

نویسنده: حسین مرادی نیا
تاریخ: ۱۳۹۱/۰۴/۱۷ ۱۴:۵

سلام
همانطور که گفتید چک کردن پسوند فایل الزامی است ولی کافی نیست.برای همین از کلاس Image استفاده کردید که اگر فایل ارسال شده یک فایل تصویری نبود استثنایی صادر شود. حالا برای فایل‌ها با فرمت‌های دیگه چیکار کنیم؟ (مثلا ممکن است بخواهیم به کاربر اجازه ارسال فایل‌های zip و یا rar بدهیم و مواردی از این دست).

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۱۷ ۱۴:۱۶

برای بررسی محتوایی خاص، نیاز به parser مخصوص این نوع فایل‌ها است تا بتواند بررسی کند محتوای دریافتی معتبر است یا نه و عموما کسی محتوای این نوع فایل‌ها را بررسی نمی‌کند. بنابراین بررسی پسوند در اکثر موارد کافی است.
مشکل این نیست که فایل rar واقعا rar است یا نه. مشکل این است که این فایل ارسالی، قابلیت اجرا را از طریق فراخوانی آدرس آن در مرورگر، نداشته باشد. بحث بررسی پسوند هم به همین دلیل است. فایل aspx را می‌شود از طریق فراخوانی در مرورگر بر روی سرور اجرا کرد ولی فایلی با پسوند rar این قابلیت را ندارد.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۵/۰۹ ۲:۱

ممنون ، استثنایی که صادر می‌شود را چطور میشه ؟ در بدنه‌ی Action مربوطه مدیریت کرد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۹ ۸:۲۳

لازم نیست مدیریت کنید. هدف این بوده که نظم جاری را تغییر دهد. این استثناء توسط ELmah دریافت و ثبت خواهد شد. همچنین کاربر به یکی از صفحات پیش فرض خطای برنامه هدایت می‌شود و متوجه خواهد شد که خطایی رخ داده است.
ولی در کل می‌شود IExceptionFilter را نیز پیاده سازی و مدیریت کرد:

```
public class CustomFilter : FilterAttribute, IExceptionFilter
{
    public void OnException(ExceptionContext filterContext)
    {
```

نویسنده: رضوی
تاریخ: ۱۳۹۱/۰۵/۲۸ ۸:۵۳

سلام

چند نکته تکمیلی برای امنیت بیشتر
1- بهتر است که فایل روی دیتابیس ذخیره شود، در غیر اینصورت نام فایل و پسوند توسط سیستم تعیین شود.
2- نام و پسوند ارسالی در نظر گرفته نشود.
3- با اندازه گیری سایز فایل GIF نمیتوان به معتبر بودن آن اطمینان داشت.

http://www.pyxsoft.com/en/hacking-hackers.html
/http://www.exploit-db.com/exploits/16181

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۷ ۱۱:۱۶

یک نکته‌ی تکمیلی

نحوه‌ی فقط خواندنی کردنی دسترسی به فایل‌های یک پوشه (گرفتن دسترسی اجرا)

```
<location path="upload">
  <system.webServer>
    <handlers accessPolicy="Read" />
  </system.webServer>
</location>
```

عنوان: **ارسال فایل در ASP.NET MVC و اعتبار سنجی سمت کاربر**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۳ ۱۸:۱۶
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, ASP.Net MVC

پیشنیازها:
- نحوه ارسال فایل به سرور توسط ASP.NET MVC
- نحوه اعتبار سنجی سمت سرور ارسال فایل‌ها

در ASP.NET MVC برای آپلود فایل‌ها عموما عنوان می‌شود که از تگ input به نحو زیر استفاده شود:

```
<input type="file" name="file" />
```

مشکلی که با این روش وجود دارد، عدم فعال شدن اعتبار سنجی سمت کاربر در حد مثلا «لطفا یک فایل را انتخاب کنید» است.
برای فعال سازی آن می‌توان از همان روش unobtrusive معرفی شده در ASP.NET MVC 3 به نحو زیر به صورت دستی استفاده کرد:

```
<input type="file" data-val="true" data-val-required="لطفا فایلی را انتخاب کنید" name="file" />
@Html.ValidationMessage("file")
```

در اینجا ویژگی‌های data-val و data-val-required به صورت خودکار اعتبار سنجی سمت کاربر را فعال خواهند کرد (شبیه به required field validator عمل می‌کنند).

از این نکته خصوصا در طراحی html helperهای سفارشی نیز می‌توان استفاده کرد.
برای مثال فرض کنید مشغول طراحی یک کنترل captcha هستید. قسمتی از آن مربوط به دریافت ورودی از کاربر است. به علاوه، پیغام خطایی هم که باید نمایش داده شود نیز باید توسط کاربر قابلیت سفارشی شدن را داشته باشد (و نمی‌توان آن‌را توسط یک attribute به سادگی به یک مدل خاص انتساب داد).
در این حالت تنظیم data-val و data-val-required به کمک anonymously typed objects پارامتر htmlAttributes میسر نیست (چون بین حروف آن dash وجود دارد) و باید از overload و نوع dictionary دار آن به نحو زیر استفاده کرد:

```
htmlHelper.TextBox("CaptchaInputText", string.Empty,
                           new Dictionary<string, object>
                           {
                             { "data-val", "true"},
                             { "data-val-required", validationErrorMessage},
                           })
```

نظرات خوانندگان

نویسنده: مولانا
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۷:۳۰

با سلام.
آیا امکان آپلود فایل بوسیله ajax نیز وجود دارد. و یا این روش معایبی دارد؟ با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۷ ۱۸:۵۴

بله. مرورگرهایی که XMLHttpRequest Level 2 را پشتیبانی کنند (از IE 10 به بعد مثلا)، امکان Ajax upload توکار به همراه گزارش درصد آپلود را بدون نیاز به فلش یا سیلورلایت، دارند. یک نمونه پیاده سازی آن

عنوان: **مسدود کردن آدرس IP با استفاده از IHttpModule در Asp.Net**
نویسنده: مجید مقصودی پور
تاریخ: ۱۳۹۱/۰۴/۲۵ ۱۸:۲۸
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, IHttpModule, IP Block

باسلام:

هدف این مقاله بیشتر آشنایی با HttpModule در قالب یک پروژه می‌باشد.

قصد دارم در این مقاله یک روش برای مسدود کردن IP هایی که به هر روشی در سایت شما اقداماتی غیر عادی انجام دادن و شما قصد دارید سایت شما به اونها نمایش داده نشه بیان کنم.

دقت کنید که شما میتونید با تشخیص کاربر متخلف مثل کاربری که بیش از 5 بار اقدام به وارد کردن نام کاربری و رمز عبور کرده کنید و IP کاربر رو در جایی مانند DataBase ذخیره کنید و یک زمان براش ثبت کنید و تا 15 دقیقه بعد از اون دسترسی به سایت رو ازش بگیرید.

برای شروع من کلاس مربوطه رو با نام IPBlockModule در پوشه App_Code میسازم:

```
using System;
using System.Data;
using System.Configuration;
using System.Web;
using System.Web.Security;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Web.UI.WebControls.WebParts;
using System.Web.UI.HtmlControls;

public class IPBlockModule : IHttpModule
{
       public IPBlockModule()
       {
              // TODO: Add constructor logic here
       }
    public void Dispose()
    {
        //Dispose
    }

    public void Init(HttpApplication context)
    {
        context.BeginRequest += new EventHandler(Application_BeginRequest);
    }

    private void Application_BeginRequest(object source, EventArgs e)
    {
        HttpContext context = ((HttpApplication)source).Context;
        string ipAddress = context.Request.UserHostAddress;
        if (IsBlockedIpAddress(ipAddress))
        {
            context.Response.StatusCode = 403;
            context.Response.Write("Forbidden : The server understood the request, but It is refusing
to fulfill it.");
        }
    }

    private bool IsBlockedIpAddress(string ipAddress)
    {
        //Here I have stored Ip addresses in String[]. you can also Store in database.
        string[] IPs = {            "117.196.35.121",
                                    "117.196.35.122",
                                    "117.196.35.123",
                                    "117.196.35.124",
                                    "127.0.0.1"
                        };

        foreach(string IP in IPs)
        {
            if(IP == ipAddress)
                return true;
        }
        return false;

    }

}
```

و در فایل web.config این کلاس رو اضافه میکنیم :

```
<configuration>
	<system.web>
		<compilation debug="true" targetFramework="4.0"/>
		<httpModules>
			<add name="IPBlockModule" type="IPBlockModule"/>
		</httpModules>
	</system.web>
</configuration>
```

میتونید متغیر آرایه ای IPs رو از پایگاه داده بخونید و کوئری رو با شرط مدت زمان سپری شده که معمولا 15 دقیقه هست بگیرید.

موفق باشید.

نظرات خوانندگان

نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۴/۲۵ ۲۲:۳۰

سلام. با تشکر از شما.
بسیار مطلب خوب و کاربردی بود.
در همین رابطه مطلب خوبی در اینجا خوانده بودم که برای مطالعه بیشتر پیشنهاد می‌کنم.
همچنین متد IsBlockedIpAddress را می‌توان با عبارت LINQ زیر جایگزین کرد:

```
private static bool IsBlockedIpAddress(string ipAddress)
        {
            string[] ips = {
                                "117.196.35.121",
                                "117.196.35.122",
                                "117.196.35.123",
                                "117.196.35.124",
                                "127.0.0.1"
                            };

            return ips.Any(ip => ip == ipAddress);
        }
```

نویسنده: Nima
تاریخ: ۱۳۹۱/۰۴/۲۵ ۲۲:۳۵

سلام دوست عزیز

آیا بلاک کردن ip کار دقیقی است؟ از این نظر که ده‌ها نفر در اینترنت ممکن است با یک ip به سایت ما وصل بشن و این کار باعث میشه همه اونا بلاک بشن

نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۴/۲۵ ۲۲:۴۳

محدودیت زمانی 15 دقیقه را برای همین منظور اعمال می‌کنند.

نویسنده: مجید مقصودی پور
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۲:۱۶

باسلام:
میشه گفت یک راه برای جلوگیری از هکرها میباشد هکرها میتونه هم انسان باشه و یا هم یک برنامه کاربردی (روبوت) که یک عملیات رو در یک بازه زمانی زیاد تست میکنه.
که این بحث در این مورد در اینجا نمیگنجه و بیشتر من در این مقاله هدفم استفاده از HTTP Module بود.
موفق باشید.

نویسنده: مجید مقصودی پور
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۲:۱۸

باسلام:
باتشکر از شما دوست گرامی.
یک نکته رو توجه داشته باشید که راه‌ها و روش‌های زیادی برای پیاده سازی وجود داره من خودم سعی میکنم ساده‌ترین روش رو برای پیاده سازی متدها انتخاب کنم و هر برنامه نویسی میتونه اونو بسته به معماری که ازش استفاده میکنه تغییر بده.
راه‌های رسیدن به خدا زیاده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۶:۵۹

از این نمونه‌ها من در این سایت زیاد دارم. کسانی که مرتبا روزی چند بار سایت رو زیر حمله می‌گیرند.

Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ
Banned IP: 46.165.199.150 Details...		07/16/2012	09:11 ق.ظ

Error	User	Date	Time
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ
Banned IP: 31.29.56.1 Details...		07/16/2012	02:46 ب.ظ

نویسنده: مجید مقصودی پور
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۷:۲

شما برای جلوگیری از این IPها کاری میکنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۷:۵

بله. یک فیلتر آنتی داس درست کردم که شبانه روز دسترسی این نوع مزاحم‌ها رو حذف می‌کنه. تا الان هر روز به طور میانگین حداقل 2 بار در زمان‌های مختلف به این سایت حمله شده.

نویسنده: مجید مقصودی پور
تاریخ: ۱۳۹۱/۰۴/۲۶ ۱۷:۷

اگر در قالب یک مقاله ارائه بدید خیلی مفید واقع میشه.
باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۷ ۱۰:۲۲

تقریبا بر همین اساس پیاده سازی شده. اطلاعات در کش سیستم ذخیره می‌شن و به این ترتیب IPهای مهاجم رو میشه تشخیص داد.

نویسنده: امیر
تاریخ: ۱۳۹۱/۰۴/۲۷ ۱۴:۲۶

با درود و عرض ادب

استاد نصیری

سایت ما هر چند وقت یک بار با ipهای مختلفی هک می‌شود

همه جور محدودیت و enjection هم اعمال کردم.

به نظرتون این مشکل رو چی کار باید کرد.؟

ممنون از راهنمایی تون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۷ ۱۴:۲۹

نمیشه همینطوری ندیده اظهار نظر کرد. نیاز به code review داره.

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۴/۲۷ ۱۵:۱۹

سلام

البته توی این آدرس به فارسی توضیح داده شده همون روش

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۵/۱۶ ۱۰:۲۳

مهندس نصیری من IP هایی که به نحوی دارن بازدید اضافی انجام میدن و مشکوک هستن طبق همین روش اونارو توی لیست سیاه میذارم و دسترسی اونهارو به سایت قطع می‌کنم آیا روش درستیه؟
یعنی دیگه نمی‌تونن بیان توی سایت اما این امکان گذاشتم که بعدا اونهارو از لیست سیاه در بیارم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۶ ۱۳:۵۷

نه. IPهای کاربران در ایران عموما ثابت نیست. نصف روز بسته شود کافی است.

عنوان: **نحوه استفاده صحیح از لوسین در ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۲۹ ۹:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Design patterns, Lucene.NET

بر مبنای پیاده سازی متداولی که در n هزار سایت اینترنتی می‌توان یافت، نحوه کار با جستجوگر لوسین حدودا به این شکل است:

```
var directory = FSDirectory.Open(new DirectoryInfo(Environment.CurrentDirectory + "\\LuceneIndex"));
using (var searcher = new IndexSearcher(directory, readOnly: true))
{
      //do something ...

       searcher.Close();
       directory.Close();
}
```

و ... اینکار به این شکل غلط است!

مطابق مستندات رسمی لوسین، این کتابخانه thread-safe است. به این معنا که در آن واحد چندین و چند کاربر می‌توانند از یک وهله از شیءهای Reader و Searcher استفاده کنند و نباید به ازای هر جستجو، یکبار این اشیاء را ایجاد و تخریب کرد. البته در اینجا تنها یک Writer در آن واحد می‌تواند مشغول به کار باشد.

مشکلاتی که به همراه باز و بسته کردن بیش از حد IndexSearcher وجود دارد، مصرف بالای حافظه است (به ازای هر کاربر مراجعه کننده، یکبار باید ایندکس‌ها در حافظه بارگذاری شوند) و همچنین تاخیر اولیه این بارگذاری و کندی آن‌را نیز باید مدنظر داشت.

نتیجه گیری:

برای کار با جستجوگر لوسین نیاز است از الگوی Singleton استفاده شود و تنها یک وهله از این اشیاء بین تردهای مختلف به اشتراک گذاشته شود.

نظرات خوانندگان

نویسنده: حسین غلامی
تاریخ: ۱۳۹۱/۱۰/۱۱ ۱:۲۹

آقای نصیری میشه نمونه ای رو با استفاده از این الگو ، مثال بزنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۱۱ ۱۰:۹

مراجعه کنید به مثال Auto Complete .

عنوان: **استفاده از reCAPTCHA در ASP.NET**
نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۴/۲۹ ۱۹:۳۹
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Security, CAPTCHA

CAPTCHA جهت تشخیص انسان در وب استفاده می‌شود. در حقیقت می‌توان CAPTCHA را مانعی مابین منابع یک سایت و روبات‌های مخرب دانست. گوگل یک CAPTCHA رایگان جهت استفاده توسعه دهندگان وب مهیا کرده است که استفاده از آن را به چند دلیل می‌توان مثبت دانست:

فراگیر بودن آن و استفاده سایت‌های مختلف از آن (بیش از 200.000 سایت).
سهولت استفاده برای کاربران.
سهولت استفاده توسط برنامه نویسان.
دسترسی پذیری مناسب بدلیل وجود تلفظ و پخش عبارت، بصورت صوتی.
امنیت بالا. بسیاری از CAPTCHAها به سادگی قابل شکستن هستند!

حال اجازه دهید نگاهی به نحوه استفاده از reCAPTCHA در ASP.NET داشته باشیم:

1. پس از اینکه به حساب کاربری خود در Google وارد شدید به این صفحه + مراجعه کنید تا یک API Key جدید برای سایت خود ایجاد کنید. برای استفاده از reCAPTCHA اولین پیش نیاز داشتن یک API Key معتبر است.

2. دانلود reCAPTCHA ASP.NET Library. این کتابخانه شامل کنترل reCAPTCHA است. پس از دریافت، **Recaptcha.dll** را توسط گزینه **Add Reference** به پروژه خود اضافه کنید.

3. جهت استفاده از این کنترل، در بالای صفحه از دستور زیر استفاده کنید:

```
<%@ Register TagPrefix="recaptcha" Namespace="Recaptcha" Assembly="Recaptcha" %>
```

سپس کنترل reCAPTCHA را در محل مناسب قرار دهید:

```
<recaptcha:RecaptchaControl
    ID="recaptcha"
    runat="server"
    PublicKey="your_public_key"
    PrivateKey="your_private_key"
    />
```

4. حال جهت بررسی و اعتبار سنجی فرم باید از متد IsValid کلاس Page استفاده شود. نمونه کد مورد نظر در VB.NET بصورت زیر است:

```
Sub btnSubmit_Click(ByVal sender As Object, ByVal e As EventArgs)
          If Page.IsValid Then
              lblResult.Text = "You Got It!"
              lblResult.ForeColor = Drawing.Color.Green
          Else
              lblResult.Text = "Incorrect"
              lblResult.ForeColor = Drawing.Color.Red
          End If
      End Sub
```

نظرات خوانندگان

نویسنده: زهرا
تاریخ: ۱۳۹۱/۰۴/۳۱ ۹:۱۴

با سلام

آیا این dll اپن سورس هست؟ از کجا میشه فهمید که درون این dll کدهای مخرب وجود ندارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۳۱ ۹:۳۶

دقت کردید در code.google.com هاست شده؟ یعنی سورس باز است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۳۱ ۹:۳۸

اگر دسترسی به گوگل کد ندارید، می‌شود از طریق NuGet نیز آن‌را دریافت کرد.

نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۴/۳۱ ۹:۴۸

سلام. جهت تکمیل نظر آقای نصیری و همچنین پاسخ به شما در مورد کدهای مخرب باید عرض کنم که بیش از 200.000 سایت معتبر در مقاطع زمانی مختلف از این سرویس تا به حال بهره گرفته اند. در این لیست می‌توان به سایت‌های بزرگی چون Twitter، Facebook و eBay اشاره کرد.
جای نگرانی نیست.

نویسنده: naser
تاریخ: ۱۳۹۱/۰۵/۱۲ ۱۹:۵۸

سلام. من از این سیستم استفاده کردم . رو تست لوکال، ناقص لود میشه. با کروم و فایرفاکس هم تست کردم . قضیه چیه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۲ ۲۳:۴۳

این مورد بیشتر به تداخل css کلی سایت با این کنترل مربوط می‌شود. بهترین کار استفاده از firebug است. ابتدا در برگه net آن بررسی کنید پیام 404 ایی مشاهده می‌شود؟ ممکن است چیزی از قلم افتاده باشد. سپس به وسیله firebug به صورت live شروع به ویرایش css المان انتخاب شده کنید تا به نتیجه برسید.

نویسنده: na3er
تاریخ: ۱۳۹۱/۰۵/۱۳ ۱۳:۴۳

ممنون از راهنمایی تون. مشکلم حل شد مینویسم که اگه دوستام دیگه هم به این مشکل خوردن بدونن از چیه. اولین مشکل از این بود که من به سلول‌های جدولم طول داده بودم که کمی بزرگ باشه(height) که برش داشتم و دومین مورد هم مربوط به تگ body بود که direction=rtl بود. این دو مود نباید در مورد جایی که این کنترل قرار داره اعمال بشه. ممنون استاد نصیری.

نویسنده: حسین
تاریخ: ۱۳۹۱/۰۵/۱۷ ۲۲:۳۱

سلام
دوستان من نمیتونم فایل دل ال الشو دانلود کنم، تو رو خدا فایلشو بذارین منم بردارم. با اینکه تو لاگین جی میلم هستم ازم پسورود میخاد که هر چی میزنم میگه اشتباهه ... ! با اینکه من داخل جی میلم شدم و در تب دیگه مرورگرم دارم میل هامو میخونم.
دوستان منتظرم کمکم کنید لطفا. با تشکر از مطالب مفیدتون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۷ ۲۲:۳۶

بالاتر توضیح دادم. از NuGet استفاده کنید. (دسترسی به گوگل کد با IP ایرانی میسر نیست؛ ولی NuGet محدودیتی ندارد)

نویسنده: پژمان
تاریخ: ۱۳۹۳/۰۲/۲۱ ۱۴:۵۲

با عرض سلام.
ببخشید من کارهایی که گفته بودیدو انجام دادم، و در کد پشت باتن enterLogin نوشتم :

```
protected void enterLogin_Click( object sender, EventArgs e )
        {
            if(Page.IsValid)
            {
                ScriptManager.RegisterStartupScript(this, GetType(), "showalert", "alert('خوش آمدید
!');", true);

            }

            ScriptManager.RegisterStartupScript(this, GetType(), "showalert", "alert('خطا !');", true);
```

CAPTCHA نمایش داده میشود اما بررسی نمیکنه که عدد وارد شده درست است یا نه؟ میشه بهم بگید مشکله کار کجاست؟

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۲/۲۱ ۱۶:۵۶

کد شما قسمت *else* *نداره* یا حداقل اینجا نوشته نشده. بنابراین حتی اگر صفحه معتبر هم باشه، قسمت نمایش خطا اجرا میشه.

عنوان: **واکشی اطلاعات به صورت chunk chunk (تکه تکه) و نمایش در ListView**
نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۳:۳۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Entity framework

یکی از مسائلی که در هنگام کار با کنترلهای داده‌ای نظیر GridView , ListView و .. با آن روبرو هستیم مسئله صفحه بندی می‌باشد و در بسیاری از موارد، کل اطلاعات در هر درخواست، بارگذاری میشود. در حالیکه روش بهینه به این صورت است که با توجه به PageSize و Index رکورد، می‌توان تعداد رکورد مورد نظر در همان صفحه را بارگذاری کرد، نه کل رکوردها را.
در این مثال که از Ef Code First و الگوی Unit Of Work استفاده کرده ام ، قصد نمایش اطلاعات در یک ListView و صفحه بندی آن را بصورت chunk chunk دارم.
برای آشنایی بیشتر با الگوی Unit Of Work می‌توانید به مقاله UOW در همین سایت مراجعه کنید.

ابتدا یک کنترل ListView روی صفحه ایجاد می‌کنیم. برای آشنایی بیشتر با این کنترل و بررسی قابلیتهای آن میتوان از این مقاله معرفی کنترل‌های ListView و DataPager استفاده کنید.

سپس با توجه به الگوی Unit Of Work از یک مدل ساده استفاده می‌کنیم. همچنین برای بارگذاری اطلاعات به صورت صفحه به صفحه، نیاز به داشتن Index رکورد و PageSize، جهت محاسبه تعداد رکورد مورد نیاز داریم.

```
public class User
    {
        public Int64 UserId { get; set; }
        public String UserName { get; set; }
     }
```

```
public interface IUserService
    {
        int GetCustomerCount();
        List<User> GetCustomers(int StartIndex, int PageSize);
    }
```

```
public class ImplUserService : IUserService
    {
        IUnitOfWork _uow;
        IDbSet<User> _user;

        public ImplUserService(IUnitOfWork uow)
        {
            _uow = uow;
            _user = uow.Set<User>();
        }

        public int GetCustomerCount()
        {
            int totalCount = _user.ToList().Count;
            return totalCount;
        }

        public List<User> GetCustomers(int StartIndex, int PageSize)
        {
            return _user.OrderBy(i => i.UserId).Skip(StartIndex).Take(PageSize).ToList();
        }
  }
```

در کدهای بالا متد GetCustomerCount تعداد کل رکوردهایی که باید واکشی شوند را مشخص می‌کند. همچنین می‌توان این تعداد را در cache یا viewState ذخیره کرد تا در دفعات بعدی در صورت خالی نبودن مقدار Cacheیا viewState ، نیاز به محاسبه مجدد count نداشته باشیم.

متد GetCustomer که کار اصلی را انجام میدهد از دو پارامتر استفاده می‌کند: StartIndex نقطه شروع و PageSize تعداد رکورد مورد نظر. در اینجا از دستورات Linq استفاده شده و دستور Skip مشخص میکند از کدام شماره رکورد به بعد شروع به واکشی

نماید و دستور Take مشخص می‌کند که چه تعداد رکورد را واکشی نماید.

حالا به سراغ کدهای HTML می‌رویم. در آنجا علاوه بر ListView نیاز به DataPager جهت صفحه بندی و Object DataSource جهت کنترل بارگذاری اطلاعات به صورت chunk chunk داریم.

```
<asp:ListView ID="ListView1" runat="server" DataSourceID="ObjectDataSource1">
            <ItemTemplate>
                <tr style="background-color: #DCDCDC; color: #000000;">
                    <td>
                        <asp:Label ID="UserIdLabel" runat="server" Text='<%# Eval("UserId") %>' />
                    </td>
                    <td>
                        <asp:Label ID="UserNameLabel" runat="server" Text='<%# Eval("UserName") %>' />
                    </td>
                </tr>
            </ItemTemplate>


            <LayoutTemplate>
                <table runat="server">
                    <tr runat="server">
                        <td runat="server">
                            <table id="itemPlaceholderContainer" runat="server" border="1"
style="background-color: #FFFFFF;
                                border-collapse: collapse; border-color: #999999; border-style: none;
border-width: 1px;">
                                <tr runat="server" style="background-color: #DCDCDC; color: #000000;">
                                    <th runat="server">
                                        UserId
                                    </th>
                                    <th runat="server">
                                        UserName
                                    </th>
                                </tr>
                                <tr id="itemPlaceholder" runat="server">
                                </tr>
                            </table>
                        </td>
                    </tr>
                    <tr runat="server">
                        <td runat="server" style="text-align: center; background-color: #CCCCCC;
                            color: #000000;">
                            <asp:DataPager ID="DataPager1" runat="server" PageSize="2">
                                <Fields>
                                    <asp:NextPreviousPagerField ButtonType="Button"
ShowFirstPageButton="True" ShowNextPageButton="False"
                                        ShowPreviousPageButton="False" />
                                    <asp:NumericPagerField />
                                    <asp:NextPreviousPagerField ButtonType="Button"
ShowLastPageButton="True" ShowNextPageButton="False"
                                        ShowPreviousPageButton="False" />
                                </Fields>
                            </asp:DataPager>
                        </td>
                    </tr>
                </table>
            </LayoutTemplate>
        </asp:ListView>
```

همانطور که مشاهده میکنید در DataPager ، مقدار PageSize مشخص شده است.اما Object DataSource

```
<asp:ObjectDataSource ID="ObjectDataSource1" runat="server" EnablePaging="True"
SelectCountMethod="GetCustomerCount"
            SelectMethod="GetCustomers" TypeName="UserService.ImplUserService" EnableViewState="False"
            MaximumRowsParameterName="PageSize" StartRowIndexParameterName="StartIndex">
```

که فکر می‌کنم با توجه به متدهای تعریف شده در بالا ، تعریف Object DataSource کاملاً گویا میباشد.

نکته مهم:

اگر الان برنامه را اجرا کنید با خطای No parameterless constructor defined for this object ر وبرو خواهید شد که جهت

حل این مشکل از کد زیر استفاده میکنیم.

```
protected void ObjectDataSource1_ObjectCreating(object sender, ObjectDataSourceEventArgs e)
        {
            e.ObjectInstance = ObjectFactory.GetInstance<IUserService>();
        }
```

در واقع نیاز است تا یک وهله از کلاس مورد نظر را به Object DataSource معرفی کنیم.
حال با اجرای برنامه و Trace آن متوجه خواهید شد که با کلیک بر روی شماره صفحه، تنها به تعداد رکوردهای همان صفحه، واکشی خواهیم داشت.

با تشکر از راهنمایی‌های آقای نصیری، امیدوارم این مقاله برای دوستان مفید باشه. منتظر نظرات دوستان هستم و اینکه چه جوری بتونیم اینکار رو با jquery ajax و به صورت سبکتر انجام بدیم.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۴:۲۵

یک نکته تکمیلی: در اینجا بهتر است جهت محاسبه تعداد رکوردها از متد Count مستقیما بر روی dbSet استفاده کنید (نیازی به ToList ندارد).

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۴:۳۴

بله کاملاً درسته
ممنون مهندس.

نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۵:۵۵

سلام و تشکر از شما بخاطر این مطلب. پیشنهاد می‌کنم جهت توسعه این ترفند (همانطور که اشاره کردید) نگاهی به + و + و + بیاندازید.
موفق باشید.

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۴/۳۰ ۱۶:۳۰

با سلام و تشکر از شما بابت نظرتون
scrolling هم راه خوبیه برای اینکار که کاربردهای زیادی داره
ممنونم.

عنوان: **Upload چند فایل بطور هم زمان در ASP.NET 4.5**
نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۴/۳۱ ۰:۱۲
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, HTML5, ASP.Net 4.5

چندی پیش امکان بارگذاری چندین فایل بطور هم زمان روی سرور با استفاده از کنترل‌های Telerik یا DevExpress مهیا می‌شد. همچنین به کمک jQuery تکنیک هایی وجود داشت. اما در HTML5 می‌توان از تگ زیر استفاده کرد:

```
<input type="file" multiple="multiple" name="FileUpload1" id="FileUpload1" />
```

یکی از امکانات جدید ASP.NET4.5 سازگاری کنترل‌های سمت سرور با HTML5 است. از این رو به کنترل FileUpload خصوصیاتی از قبیل **HasFiles** و **PostedFiles** و **AllowMultiple** اضافه شده است:

```
<asp:FileUpload ID="FileUpload1" runat="server" AllowMultiple="true" />
```

این کنترل در مرورگرهای متفاوت بصورت‌های مختلفی نمایش داده می‌شود. نمایش در مرورگر Chrome بصورت زیر خواهد بود:

و در Opera:

با این تفاسیر در سمت سرور، کار دشواری پیش روی ما نخواهد بود:

```
protected void Upload_Click(object sender, EventArgs e)
{
  if (FileUpload1.HasFiles)
  {
    string rootPath = Server.MapPath("~/App_Data/");
    foreach (HttpPostedFile file in FileUpload1.PostedFiles)
    {
      file.SaveAs(Path.Combine(rootPath, file.FileName));
      Label1.Text += String.Format("{0}<br />", file.FileName);
    }
  }
}
```

البته از آنجایی که هدف از این مطلب معرفی یکی از قابلیت‌های جدید HTML5 و ASP.NET4.5 است، کد بالا بسیار ساده نوشته شده است و فایل‌های ارسال شده به سرور را در پوشه App_Data و بدون در نظر گرفتن مسائل عرف درباره Upload فایل، ذخیره می‌کند.

عنوان: **رمزنگاری Connection String از طریق خط فرمان**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۵/۰۲ ۱۹:۱۶
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Security

برای رمز نگاری Connection String ابتدا Command Prompt را از مسیر زیر باز کنید

```
Start Menu\Programs\Microsoft Visual Studio 2010\Visual Studio Tools\Visual Studio Command Prompt
```

و سپس دستور زیر را برای رمزنگاری Connection String وارد کنید :

```
aspnet_regiis.exe -pef "connectionStrings" "sulotion path"
```

در اینجا پارامتر اول (pef) مشخص می‌کند که Application ما از نوع File System Website است،پارامتر دوم(connectionStrings) نام کلید موردنظری است که می‌خواهیم Encrypt شود،در اینجا می‌توانیم هر کلیدی را Encrypt کنیم
به عنوان مثال فرض کنید تنظیمات SMTP را در Web.Config بدین صورت داریم :

```
<system.net>
    <mailSettings>
      <smtp deliveryMethod="Network">
        <network host="smtp.gmail.com" port="587" userName="username@gmail.com" password="password" />
      </smtp>
    </mailSettings>
  </system.net>
```

برای رمزنگاری بدین صورت عمل می‌کنیم :

```
aspnet_regiis.exe -pef "system.net/mailSettings/smtp" "sulotion path"
```

و بعد از Encrypt کردن توسط دستور فوق بدین صورت در Web.Config نمایش داده می‌شود :

```
<system.net>
    <mailSettings>
      <smtp configProtectionProvider="RsaProtectedConfigurationProvider">
        <EncryptedData Type="http://www.w3.org/2001/04/xmlenc#Element"
          xmlns="http://www.w3.org/2001/04/xmlenc#">
          <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#tripledes-cbc" />
          <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
            <EncryptedKey xmlns="http://www.w3.org/2001/04/xmlenc#">
              <EncryptionMethod Algorithm="http://www.w3.org/2001/04/xmlenc#rsa-1_5" />
              <KeyInfo xmlns="http://www.w3.org/2000/09/xmldsig#">
                <KeyName>Rsa Key</KeyName>
              </KeyInfo>
              <CipherData>

<CipherValue>QuOFQrT6XwxDhQjFnM3EByleyWqYY6lA1cGK1Dzli/hrDOYSj35ADk4MB3PeLOMVYh76kB8vch0/iZKAZaJlNUPKi/
iZjEzE755B3sILKGLxfkH3j3qKHB0x1WN65L6zBXgzufphCVaNRobQXOl5J3E0Df8VCf/bERZu741HLPs=</CipherValue>
              </CipherData>
            </EncryptedKey>
          </KeyInfo>
          <CipherData>

<CipherValue>NdwBe0mWZ+Yg/DEzNuiDfXlGpicoH1ZMn54FTrLuVsY3rawS/k6KPID3bZvOWB/XYseTYFGhqs7FUEqIYMvWjJYYmD
Azk6dd4iv9y6ch3ZcXWQ/R5TkQLWoLQPYgdwGI3uJNs22t28xUISm1wS0uDbizCM2Io+DzSQe8N4Ih2MP9mb2NCbZ4BZEBCPvCevpSp
dEjGN9v7hk=</CipherValue>
          </CipherData>
        </EncryptedData>
      </smtp>
    </mailSettings>
  </system.net>
```

در اینجا Encryption از نوع RSAProtectedConfigurationProvider انتخاب شده است که می‌توانیم آنرا به DataProtectionConfgurationProvider تغییر دهیم که مورد اول از روش رمز نگاری کلید عمومی RSA استفاده می‌کند.
برای Decrypt کردن هم فقط کافی است پارامتر اول دستور aspnet_regiis را به pdf تغییر دهیم :

```
aspnet_regiis.exe -pdf "system.net/mailSettings/smtp" "sulotion path"
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۳ ۱۳:۱۲

البته خیلی‌ها دسترسی به سرور برای انجام این نوع کارها را ندارند. به همین جهت امکان انجام آن از طریق برنامه نویسی هم مهیا شده: (^)

نویسنده: صمد بلاج
تاریخ: ۱۳۹۱/۱۰/۰۸ ۹:۱۵

سلام؛من میخوام کانکشن استرینگم رو رمز نگاری کنم.
از روش EF استفاده کردم و تحت ویندوز هست آیا این روش مناسب هستش؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۸ ۹:۳۹

بله. با انواع فایل‌های کانفیگ برنامه‌های دات نت سازگار است.

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۳/۲۱ ۱۸:۴۱

سلام؛ بنده سایتم رو رویه لوکال پاپلیش کردم بعد مطالبی رو که بالا نوشته شده رو اعمال کردم و سایت رو ریختم رویه هاست.خطای ران تایم ارور میده.باید ابتدا کانکشن استرینگ رو رمز بکنم بعد عمل پاپلیش رو انجام بدم یا اینکه باید عمل دیگه ای انجام بشه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۱ ۲۰:۳۸

حاصل این رمزنگاری بر اساس کلیدهای موجود در کامپیوترهای مختلف متفاوت خواهد بود. بنابراین روی سرور باید رمزنگاری شود. اگر دسترسی مدیریتی دارید، روش خط فرمان مناسب است؛ اگر خیر، این روش با برنامه نویسی هم قابل اجرا است .

نویسنده: مرتضی مختاری
تاریخ: ۱۳۹۲/۰۳/۲۳ ۱۹:۲۶

سلام؛ بنده از طریق کد میخوام کانکشن استرینگ رو رمز بکنم ولی هنگام ذخیره confg.Save(ConfigurationSaveMode.Full);
خطایی میده به این عنوان
System.Security.Cryptography.CryptographicException: Object already exists
چطوری می‌تونم اصلاحش کنم.با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۲۳ ۲۰:۲۵

یک سری بررسی سطح دسترسی و همچنین محافظت نشده بودن ابتدا باید انجام شود:

```
var config =  ConfigurationManager.OpenExeConfiguration(ConfigurationUserLevel.None);
var provider = "RsaProtectedConfigurationProvider";
var connStrings = config.ConnectionStrings;
if (connStrings != null)
{
    if (!connStrings.SectionInformation.IsProtected) // اگر محافظت شده نبود
    {
```

```
        if (!connStrings.ElementInformation.IsLocked) // اگر از طرف هاست قفل نشده بود
        {
            // Protect the section.
            connStrings.SectionInformation.ProtectSection(provider);

            connStrings.SectionInformation.ForceSave = true;
            config.Save(ConfigurationSaveMode.Full);
        }
    }
}
```

نویسنده: بهاره
تاریخ: ۱۳۹۲/۰۶/۲۲ ۱۰:۳۲

سلام
اگر پروژه از نوع sharepoint 2003 باشد،آیا این روش موثر است؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۲ ۱۱:۱۶

شیرپوینت هم یک برنامه ASP.NET است.

نویسنده: بهاره
تاریخ: ۱۳۹۲/۱۱/۱۱ ۲۲:۱۸

اگر ویندوز اپلیکیشن بود چی؟ می‌شه هم روش کامندی و هم برنامه نویسیش رو بگید؟ ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۱ ۲۲:۴۷

فرقی نمی‌کنه. روش کدنویسی آن که در نظرات مطرح شده با تمام فایل‌های کانفیگ استاندارد کار می‌کند.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۲/۲۲ ۲:۵۶

لینکی که معرفی کردین فقط رمزنگاری رو انجام میده . برای decrypt کردن چه کاری باید انجام بدیم ؟ و اینکه بنده از EF database first استفاده میکنم . چه موقع و کجا باید این عمل encrypt و decrypt رو انجام بدم ؟ ضمن اینکه از هاست اشتراکی استفاده میکنم .
متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۲ ۹:۷

لازم نیست برای decrypt آن کاری انجام دهید. خودکار است. نیاز به تنظیم خاصی در برنامه ندارد.

عنوان: **باگ Directory Traversal در سایت**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۳ ۹:۴۴
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Security

من فایل‌های سایت جاری رو در مسیر استاندارد app_data ذخیره سازی می‌کنم. علت هم این است که این پوشه، جزو پوشه‌های محافظت شده‌ی ASP.NET است و کسی نمی‌تواند فایلی را مستقیما از آن دریافت و یا سبب اجرای آن با فراخوانی مسیر مرتبط در مرورگر شود.

این مساله تا به اینجا یک مزیت مهم را به همراه دارد: اگر شخصی مثلا فایل shell.aspx را در این پوشه ارسال کند، از طریق مرورگر قابل اجرا و دسترسی نخواهد بود و کسی نخواهد توانست به این طریق به سایت و سرور دسترسی پیدا کند.

برای ارائه این نوع فایل‌ها به کاربر، معمولا از روش خواندن محتوای آن‌ها و سپس flush این محتوا در مرورگر کاربر استفاده می‌شود. برای نمونه اگر به لینک‌های سایت دقت کرده باشید مثلا لینک‌های تصاویر آن به این شکل است:

http://site/file?name=image.png

Image.png نام فایلی است در یکی از پوشه‌های قرار گرفته شده در مسیر app_data.

File هم در اینجا کنترلر فایل است که نام فایل را دریافت کرده و سپس به کمک FilePathResult و return File آن‌را به کاربر ارائه خواهد داد.

تا اینجا همه چیز طبیعی به نظر می‌رسد. اما ... مورد ذیل چطور؟!

Bad file request: ~/a.s. Details...
Bad file request: ~/. Details...
Bad file request: ~/.. Details...
Bad file request: ~/ Details...
Bad file request: ~/web.config Details...
You are not allowed to upload script.sql file. Please upload only these files: .jpg,.gif,.png. Details...
Bad file request: ~/bin/myblog.zip Details...
Bad file request: ../../bin/myblog.zip Details...
Bad file request: ../../../bin/myblog.zip Details...
Bad file request: ../../../../bin/myblog.zip Details...
Bad file request: ../../../../../bin/myblog.zip Details...
Bad file request: ../../../../../../bin/myblog.zip Details...
Bad file request: ../../../../../../bin.zip Details...
Bad file request: ../../../../../bin.zip Details...
Bad file request: ../../../../bin.zip Details...
Bad file request: ../../../bin.zip Details...
Bad file request: ../../bin.zip Details...
Bad file request: ../bin.zip Details...
Bad file request: ~/bin.zip Details...
Bad file request: ~/web.config? Details...
Bad file request: ~/web.config?.jpg Details...
Bad file request: ~/web.config Details...

لاگ خطاهای فوق مرتبط است به سعی و خطای شب گذشته یکی از دوستان جهت دریافت فایل web.config برنامه!
متدهای Server.MapPath یا متد return File و امثال آن تمامی به کاراکتر ویژه ~ (اشاره‌گر به ریشه سایت) به خوبی پاسخ می‌دهند. به عبارتی اگر این بررسی امنیتی انجام نشده باشد که کاربر چه مسیری را درخواست می‌کند، محتوای کامل فایل web.config برنامه به سادگی قابل دریافت خواهد بود (به علاوه هر آنچه که در سرور موجود است).

چطور می‌شود با این نوع حملات مقابله کرد؟
دو کار الزامی ذیل حتما باید انجام شوند:
الف) با استفاده از متد Path.GetFileName نام فایل را از کاربر دریافت کنید. به این ترتیب تمام زواید وارد شده حذف گردیده و فقط نام فایل به متدهای مرتبط ارسال می‌شود.
ب) بررسی کنید مسیری که قرار است به کاربر ارائه شود به کجا ختم شده. آیا به c:\windows اشاره می‌کند یا مثلا به c:\myapp\app_data ؟
اگر به لاگ فوق دقت کرده باشید تا چند سطح بالاتر از ریشه سایت هم جستجو شده.

نتیجه گیری:

اگر در برنامه‌های وب خود (فرقی نمی‌کند مرتبط به چه فناوری است)، نام فایلی را از کاربر جهت ارائه محتوایی به او دریافت و از این نام فایل بدون هیچ نوع بررسی خاصی، مستقیما در برنامه استفاده می‌کنید، برنامه شما به مشکل امنیتی Directory Traversal مبتلا است.

پ.ن.
-1 این باگ امنیتی در سایت وجود داشت که توسط یکی از دوستان در روزهای اول آن گزارش شد؛ ضمن تشکر!
-2 از این نوع اسکن‌ها در لاگ‌های خطاهای سایت جاری زیاد است. برای مثال به دنبال فایل‌هایی مانند DynamicStyle.aspx و css.ashx یا theme.ashx می‌گردند. حدس من این است که در یکی از پرتال‌های معروف یا افزونه‌های این نوع پرتال‌ها فایل‌های یاد شده دارای باگ فوق هستند. فایل‌های ashx عموما برای flush یک فایل یا محتوا به درون مرورگر کاربر در برنامه‌های ASP.NET Web forms مورد استفاده قرار می‌گیرند.

نظرات خوانندگان

نویسنده: فربد صالحی
تاریخ: ۱۳۹۱/۰۵/۰۳ ۱۱:۱۷

تا جایی که من متوجه شدم شما در کنترلر File، اسم فایل رو دریافت می‌کنید و اون رو به مسیری داخل App_Data نگاشت میدید و بعد فایل رو از اون مسیر به کاربر return می‌کنید. اگه به این صورته سوالی واسه من پیش اومده:
همونطور که خودتون مثال زدید مهاجم ممکنه به جای اسم فایل یک مسیر مثلا ~/web.config رو به کنترلر بفرسته، خب اگه با این مسیر به صورت یک اسم برخورد بشه مثلا میشه :

```
~/App_Data/~/web.config
```

درسته؟ یعنی در این صورت هم باز از ریشه، فایل web.config برمیگرده؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۳ ۱۱:۴۰

- این مورد چطور؟

```
var path = Server.MapPath("~/App_Data/../web.config");
```

حتما یکبار خروجی آن‌را دیباگ کنید؛ جالب است.
کاربر هم بجای مسیر یک تصویر یا فایل، مسیر زیر را وارد کرده:

```
../web.config
```

+ عرض کردم در راه‌حل‌های عنوان شده.
اولین بررسی دریافتی از کاربر باید این مورد باشد:

```
var fileName = Path.GetFileName("~/web.config");
```

و نه استفاده مستقیم از نام دریافتی از وب.
خروجی متد فوق (web.config خالی) دیگر به ریشه سایت و یا هیچ مسیری اشاره نخواهد کرد.

نویسنده: فربد صالحی
تاریخ: ۱۳۹۱/۰۵/۰۳ ۱۳:۴۵

اتفاقا خودم به استفاده از ".." توجه کردم، ولی چون تست نکردم این مورد رو ، تصورم این بود که تو سی شارپ جواب نمیده

نویسنده: رحمت اله رضایی
تاریخ: ۱۳۹۱/۰۵/۰۳ ۲۱:۴۹

هدف تست دیشب مشکل دیگری بود که در مطلبی جدا به آن خواهم پرداخت. اما برای اینکه بدانید برای این مورد اخیر چقدر بی توجهی می‌شود کافیست در گوگل صفحاتی را جستجو کنید که آدرسهایی اینگونه دارند و سعی کنید با تغییر آدرس، فایل web.config را دانلود کنید :

```
.../download.aspx?file=...
.../download.ashx?file=...
.../get.aspx?file=...
.../get.ashx?file=...
.../download.aspx?path=...
```

```
.../download.ashx?path=...
...
```

به این صورت جستجو کنید :

```
inurl:"/download.ashx?path="
```

سایتهای فارسی زیادی هم میتوانید پیدا کنید که این مشکل را دارند.

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۰۴ ۷:۵۸

من این مشکل رو قبلاً در BlogEngine.NET دیده بودم که برطرفش کردن.

نویسنده: فرهاد یزدان پناه
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۴:۳۰

آیا ذخیره محتواهای ارسالی (فایل، تصویر و ...) در پایگاه داده بهتر نیست؟
همچنین پشتیبان گیری را راحت می‌کند.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۴:۳۴

بله. همینطوره. به شرطی‌که امکانات سخت افزاری مهیا چنین اجازه‌ای رو به شما بدهد. مثلا سرور اختصاصی داشته باشید با RAM و CPU قابل قبول. به عبارتی در شبکه‌های خصوصی شرکت‌ها، نه سایت‌های عمومی با حداقل‌هایی که در اختیار دارند. یا پردازش ابری ... مثلا برای سازمان‌هایی که می‌تونند هزینه کنند یا دسترسی به این امکانات دارند.

نویسنده: محمد صاحب
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۵:۱۵

ممنونم خیلی جالب بود...
یک روش هم میتونه Encrypt کردن نام فایل باشه که البته سربار خودش رو داره...

نویسنده: فرهاد یزدان پناه
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۵:۵۳

بله در فضاهایی همچون Azure محدودیت هایی بر روی اندازه پایگاه داده و ... وجود دارد. ولی نه در هر نوع محیط ابری.
در ضمن الان شما برای جلوگیری از خیلی مسائل مجبور شدید یک Handler جهت پراکسی اطلاعات (چه ارسالی و چه دریافتی) ایجاد کنید که از خیلی لحاظ مشابه ذخیره فایل‌ها در پایگاه داده باشه.
در ضمن مگه اندازه فایل‌های ارسالی و ... چقدر است که نیازمند مقدار زیادی پردازش باشه. استفاده از مکانیزم‌های Cache موجود در asp.net هم می‌تونه کمک کنه.
حرف شما از خیلی لحاظ صحیحه و شکی در اون نیست ولی میشه به راه حل‌های دیگری هم فکر کرد.

نویسنده: رضوی
تاریخ: ۱۳۹۱/۰۵/۰۵ ۱۵:۸

سلام

باگهایی از این قبیل به باگهای **File Inclusion** معروفند که به دو دسته Remote File Inclusion و Local File Inclusion تقسیم می‌شوند و هکرها با استفاده از آنها می‌تواند به بسیاری از اطلاعات سرور دست پیدا کنند و یا بدون آپلود، شل بگیرند.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۰۵ ۱۵:۵۹

البته در گزارشات متداول منتشره، هر دو عبارت به جای هم بکار برده می‌شوند و استفاده از file inclusion بیشتر برای سایت‌های PHP و لینوکسی مرسوم است؛ چیزی مثل این:

```
http://site.com/forcedownload.php?file=../../../../../../../../etc/passwd
```

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۵/۱۷ ۱۳:۵۳

بله چندتایی از این سایتها رو منم تونستم ببینم، ولی نکته جالب اینه که سایتهای غیر ایرانی این مسئله رو حل کردند

نویسنده: میثم جوادی
تاریخ: ۱۳۹۱/۰۵/۳۰ ۲۰:۳۵

یه روش دیگه واسه دانلود Web.config

عنوان: Google OpenID Authentication در ASP.NET با استفاده از DotNetOpenAuth
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۵/۰۴ ۵:۴۷
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, C#, Security, OpenID, DotNetOpenAuth

در پست‌های قبلی OpenID چیست؟
استفاده از OpenID در وب سایت جهت احراز هویت کاربران با مفاهیم OpenID تاحدودی آشنا شدیم
در این پست می‌خواهیم با یک مثال ساده نشان دهیم که سایت ما چگونه با استفاده از OpenID Authentication می تواند از اکانت گوگل استفاده کرده و کاربر در وب سایت ما شناسایی شود.
برای اینکار ابتدا
1- کتابخانه DotNetOpenAuth را از طریق NuGet به لیست رفرنس‌های پروژه وب خود اضافه نمایید
2- یک صفحه بنام Login.aspx یا هر نام دلخواهی را به پروژه خود اضافه نمایید. در نهایت کد Html صفحه شما به ید به صورت زیر باشد.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="OAuthWebTest.Login"
%>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
    <div id="loginform">
        <div id="NotLoggedIn" runat="server">
            <asp:ImageButton ID="ButtonLoginWithGoogle" runat="server" ImageUrl="Google.JPG"
                OnCommand="ButtonLoginWithGoogle_Click"
CommandArgument="https://www.google.com/accounts/o8/id" />
            <p />
            <asp:Label runat="server" ID="LabelMessage" />
        </div>
    </div>
    </form>
</body>
</html>
```

در کد Html بالا به خصوصیت CommandArgument از کنترل ImageButton دقت نمایید که دارای مقدار "https://www.google.com/accounts/o8/id" می باشد. در واقع این آدرس باید برای اکانت گوگل جهت احراز هویت توسط OpenID استفاده شود. (آدرس API گوگل برای استفاده از این سرویس)
3- در قسمت کد نویسی صفحه کدهای زیر را وارد نمایید.

```
using System;
using System.Web.UI;
using System.Web.UI.WebControls;
using DotNetOpenAuth.OpenId.RelyingParty;

namespace OAuthWebTest
{
    public partial class Login : Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            OpenIdRelyingParty openIdRelyingParty = new OpenIdRelyingParty();
            var response = openIdRelyingParty.GetResponse();
            if (response == null) return;
            switch (response.Status)
            {
                case AuthenticationStatus.Authenticated:
                    NotLoggedIn.Visible = false;
                    Session["GoogleIdentifier"] = response.ClaimedIdentifier.ToString();
                    Response.Redirect("Default.aspx");
                    break;
                case AuthenticationStatus.Canceled:
                    LabelMessage.Text = "Cancelled.";
                    break;
```

```
                case AuthenticationStatus.Failed:
                    LabelMessage.Text = "Login Failed.";
                    break;
            }
        }

        protected void ButtonLoginWithGoogle_Click(object src, CommandEventArgs e)
        {
            string discoveryUri = e.CommandArgument.ToString();
            OpenIdRelyingParty openId = new OpenIdRelyingParty();
            var returnToUrl = new UriBuilder(Request.Url) { Query = "" };
            var request = openId.CreateRequest(discoveryUri, returnToUrl.Uri, returnToUrl.Uri);
            request.RedirectToProvider();
        }
    }
}
```

همانگونه که مشاهده می‌کنید در رویداد کلیک دکمه لوگین ابتدا آدرس یکتا گوگل (مقدار ImageButton اختصاص داده شده به کنترل CommandArgument) به همراه آدرس صفحه جاری وب سایت توسط متد CreateRequest از شی openId ترکیب شده و یک درخواست (Request) ساخته شده و در نهایت برای سرویس دهنده گوگل ارسال می‌شود. با اجرای پروژه با تصویر زیر روبرو می‌شوید بروی کلید لوگین کلیک نمایید.

در واقع با کلیک روی دکمه مورد نظر تکه کدی که در بالا شرح داده شد اجرا شده و ما را به صفحه ای شبیه تصویر پایین هدایت می‌کند

در صورتی که کلید Sign In انتخاب شود شما به سایت (همین برنامه) اجازه داده اید که از اطلاعات حساب کاربری شما استفاده کند. پس از کلیک به برنامه اصلی (طبق آدرس بازگشت تعیین شده در رویداد ButtonLoginWithGoogle_Click) در رویداد PageLoad صفحه لاگین اطلاعات بازگشتی از سایت سرویس دهنده (در اینجا گوگل) چک می‌شود و با توجه با پاسخ مورد نظر عملیاتی انجام می‌شود، مثلا در صورتی که شما تایید کرده باشید اطلاعات شناسایی شما توسط گوگل در کلیدی به نام GoogleIdentifier در سشن ذهیره شده و شما به صفحه اصلی سایت هدایت می‌شوید.

دموی پروژه در این آدرس قرار داده شده است.

سورس پروژه از این آدرس قابل دسترسی است.

توجه: در این آدرس شرح داده شده که چگونه دسترسی سایت‌های دیگر به اکانت گوگل خود را قطع کنید

در پست‌های آینده اتصال به تویتر و ف ی س بوک و سایت‌های دیگر توضیح داده خواهد شد.

نظرات خوانندگان

نویسنده: مجتبی چنانی
تاریخ: ۱۳۹۱/۰۵/۰۴ ۹:۰

با سلام و تشکر از پست خوبتون

در زمانی که ما از سیستم‌های ورود و ثبت کاربران شرکت‌های دیگر استفاده می‌کنیم آیا می‌توانیم لاگ گرفته یا اینکه برای خودمان یک صفحه داشته باشیم تا ورود و خروج‌های اکانت‌های درون وبسایتمان را بررسی کنیم یا اینکه خودمان باید این بخش را کدنویسی کنیم؟

یک سئوال دیگر این است که زمانی که از openid های شرکت‌های دیگه استفاده می‌کنیم فقط احراز هویت را از این سرویس‌ها دریافت می‌کنیم یا اینکه در همه صفحات و دیگر کارهای کاربر نظارت به صورت خودکار انجام می‌شود یا اینکه باز هم باید کدنویسی کنیم؟

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۵/۰۴ ۱۴:۹

در صورتی که بخواهید لاگ بندازید باید خود کد نویسی کرده و در دیتابیس ذخیره کنید
بله می‌توان در هر صفحه با استفاده از کلیدی که در سشن کاربر ذخیره کرده اید (با توجه به اینکه سشن کاربران جداست) می‌توانید وجود کاربر را چک کنید

نویسنده: سینا علیزاده
تاریخ: ۱۳۹۲/۰۹/۰۳ ۱۳:۲۹

آقا عالی بود ممنون
میشه برای facebook و توییتر هم بزارید ، واقعا بهش نیاز دارم ممنون میشم
کارتون عالیه موفق باشید

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۰۳ ۱۳:۳۵

نگاهی هم به پروژه DotNetAuth داشته باشید.

نویسنده: دادخواه
تاریخ: ۱۳۹۲/۱۰/۱۶ ۱۱:۳۱

سلام.

این سرویس چطوری برای اکانت‌های یاهو کار می‌کنه؟

عنوان: **نکاتی در مورد ELMAH**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۵/۰۶ ۱:۲۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, ELMAH

سفارشی سازی ایمیل ارسالی :

در مورد ELMAH(Erro Logging Module And Handlers) آقای نصیری چندین مطلب نوشته اند (+ و + و ...)
قبل از ارسال ایمیل توسط ELMAH رخدادی به نام Mailing اجرا (Raise) می‌شود. اگر برای این رخداد یک Event Handler ایجاد کنیم، می‌توانیم جزئیات مربوط به خطایی که قرار است ارسال شود را تغییر دهیم. به عنوان مثال می‌توانیم بگوئیم اگر Error ما از نوع Application Exception بود آنرا به آدرس دیگری ارسال کن و .... برای ایجاد یک Event Handler برای رخداد Mailing ابتدا فایل Global.asax رو به پروژه اضافه کنید و کد زیر را به آن اضافه کنید :

```
void ErrorMailModuleName_Mailing(object sender, Elmah.ErrorMailEventArgs e)
{

}
```

دقت داشته باشید که در کد فوق به جای ErrorMailModuleName نام HTTPModule را بنویسید، این ماژول در فایل Web.Config قرار دارد :

```
<httpModules>
      <add name="ErrorMail" type="Elmah.ErrorMailModule, Elmah" />
</httpModules>
```

که در کد فوق ErrorMail می‌باشد در نهایت :

```
void ErrorMail_Mailing(object sender, Elmah.ErrorMailEventArgs e)
    {

    }
```

به عنوان مثال در کد زیر اگر Error از نوع Application Exception بود Error Log به آدرس sir1afifi@gmail.com نیز ارسال می‌شود :

```
void ErrorMail_Mailing(object sender, Elmah.ErrorMailEventArgs e)
    {
        if (e.Error.Exception is ApplicationException)
        {
            e.Mail.To.Add("sir1afifi@gmail.com");
        }
    }
```

ذخیره Errorها در دیتابیس SQL SERVER :

ELMAH خطاهای Log شده را در جدولی با نام ELMAH_Error ذخیره می‌کند، اگر به داخل پوشه ELMAH توجه کرده باشید یک فایل با نام SQLServer.sql وجود دارد که حاوی اسکریپت مربوط به ساخت جدول فوق می‌باشد. با اجرای این اسکریپت جدول مربوطه همرا با سه SP با نام‌های ELMAH_LogError ، ELMAH_GetErrorXml ، ELMAH_GetErrorsXml ساخته می‌شوند. بعد از ساخت جدول مربوطه باید تگ زیر را در فایل Web.Config بنویسیم :

```
<errorLog type="Elmah.SqlErrorLog, Elmah"
            connectionStringName="..." />
```

connectionStringName هم نام کانکشن استرینگ را در این قسمت قرار می‌دهیم به عنوان مثال با داشتن کانکشن استرینگ زیر :

```
<connectionStrings>
    <add name="conn" connectionString="Data Source=.;Initial Catalog=test;User
ID=user1;Password=123456;"/>
  </connectionStrings>
```

به این صورت connectionStringName برابر با مقدار name یعنی conn می‌شود.

نظرات خوانندگان

نویسنده: رضا
تاریخ: ۱۳۹۱/۰۷/۰۳ ۱۱:۳۰

برای استفاده از Elmah در یک پروژه MVC باید پکیج Elmah.MVC رو دریافت کنیم یا همون Elmah?

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۰۳ ۱۱:۳۷

توضیحات بیشتر در اینجا

نویسنده: امیر
تاریخ: ۱۳۹۲/۰۵/۳۰ ۹:۳۱

با سلام؛ چند سوال:
چطوری بدون استفاده از ممبر شیپ نتونه هر کسی این آدرس زد این صفحه بیاد. یه جورایی یا پسورد ازش بگیره یا سطح دسترسی براش تعریف کنیم.

2- در کدوم قسمت فایل وب کانفیگ تغییر بدم که اسم فایل elamh.axd تغییر کند مثلا بشه test.axd

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۳۰ ۹:۳۷

به این دو سؤال در مطالب پیشنیاز بحث جاری پاسخ داده شده‌اند:
معرفی ELMAH
مدیریت خطاها در یک برنامه ASP.NET MVC

نویسنده: امیر
تاریخ: ۱۳۹۲/۰۵/۳۰ ۱۰:۰

ممنون آقای نصیری .من خوندم ولی نگفته بودین کدوم قسمت رو تغییر بدیم تا نام elamh بشه. فقط گفته شد تغییر داده شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۳۰ ۱۰:۳۵

در همان مطلب معرفی عنوان شده، داریم:

```
<add verb="POST,GET,HEAD" path="elmah.axd" type="Elmah.ErrorLogPageFactory, Elmah" />
```

در اینجا مقدار path را بجای elmah.axd تغییر بدید به هر نام دیگری که علاقمند بودید.

نویسنده: امیر
تاریخ: ۱۳۹۲/۰۵/۳۱ ۱۲:۱۵

چطوری بدون استفاده از ممبر شیپ نتونه هر کسی این آدرس زد (www.yoursite.com/elmah.axd)این صفحه بیاد. یه جورایی یا پسورد ازش بگیره یا سطح دسترسی براش تعریف کنیم.چون سطح دسترسی من توسط کد نویسی هست

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۳۱ ۱۲:۲۲

در همان مطلب پیشنیاز در مورد location بحث شده، یا حداقل اشاره شده (و اگر کدهای آن ذکر نشده، چون یک مبحث مقدماتی است و نه تکمیلی). در یک مبحث تکمیلی فرض بر این است که شما حداقل یک فصل از مباحث ابتدایی Forms Authentication رو مطالعه کردید. مثلا می‌دونید تگ location را باید به چه صورتی تعریف کرد، یا چطور باید Role به آن اضافه کرد:

```
<location path=".............">
  <system.web>
    <authorization>
      <allow roles="Administrators" />
      <deny users="*" />
    </authorization>
  </system.web>
</location>
```

عنوان: **سازگار کردن لینک‌های قدیمی یک سایت با ساختار جدید آن در ASP.NET MVC**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۴ ۲۱:۲۴
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, ASP.Net MVC

اگر پیشتر سایتی را در آدرس مشخصی در اینترنت داشته‌اید و اکنون تنها نرم افزار آن تغییر کرده است، اما نحوه ارائه خدمات آن خیر، لازم است بتوانید شرایط ذیل را مدیریت کنید:
- موتورهای جستجو مدام اطلاعات قبلی خود را به روز می‌کنند. اگر آدرس قبلی مقاله‌ای در سایت شما http://site/year/month/day/title بوده، برای نمونه گوگل هر از چندگاهی مجددا به این آدرس مراجعه می‌کند تا حداقل مطمئن شود وجود خارجی دارد یا خیر (این نکته را از لاگ‌های خطای سایت استخراج کردم).
- سایت‌های زیادی هستند که پیشتر به سایت شما و مطالب آن لینک داده‌اند. نمی‌توانید از آن‌ها درخواست کنید لطفا بانک اطلاعاتی خود را به روز کنید.
- اگر فید قبلی سایت شما http://site/feeds/posts بوده و اکنون چیز دیگری است، باز هم نمی‌توانید از همه درخواست کنید اطلاعات خود را به روز کنند. عده‌ای اینکار را خواهند کرد و تعداد زیادی هم خیر.

برای مدیریت یک چنین مواردی می‌توان از امکانات مسیریابی موجود در ASP.NET MVC استفاده کرد؛ که نمونه‌ای عملی از آن‌را جهت سازگاری سایت جاری با هاست قبلی آن (بلاگر) در ادامه مطالعه خواهید نمود:

الف) سازگار سازی لینک‌های قدیمی برچسب‌های سایت با ساختار جدید آن

در بلاگر آدرس‌های برچسب‌ها، به صورت http://site/search/label/name تعریف شده است. در سایت جاری برچسب‌ها توسط کنترلر Tag مدیریت می‌شوند. برای هدایت آدرس‌های قدیمی (موجود در موتورهای جستجو یا ثبت شده در سایت‌هایی که به ما لینک داده‌اند) می‌توان از تعریف مسیریابی ذیل در فایل global.asax استفاده کرد:

```
routes.MapRoute(
                "old_bloger_tags_list", // Route name
                "search/label/{name}", // URL with parameters
                new { controller = "Tag", action = "Index", name = UrlParameter.Optional, area = "" }
// Parameter defaults
            );
```

به این ترتیب به صورت خودکار تمامی آدرس‌های شروع شده با http://site/search/label پالایش شده و سپس قسمت name آن‌ها جدا سازی می‌شود. این نام به متدی به نام Index در کنترلر Tag که دارای پارامتری به نام name است ارسال خواهد شد.

ب) از دست ندادن خوانندگان قدیمی فیدهای سایت

دو نوع فید کلی در بلاگر وجود دارد: http://site/feeds/posts/default و http://site/feeds/comments/default؛ اما در سایت جاری فیدها توسط کنترلری به نام Feed ارائه می‌شوند. برای سازگار سازی آدرس‌های قدیمی و هدایت آن‌ها به صورت خودکار به کنترلر فید می‌توان از دو تعریف مسیریابی ذیل استفاده کرد:

```
routes.MapRoute(
                "old_bloger_posts_feeds_list", // Route name
                "feeds/posts/default", // URL with parameters
                new { controller = "Feed", action = "Posts", name = UrlParameter.Optional, area = "" }
// Parameter defaults
            );

routes.MapRoute(
                "old_bloger_comments_feeds_list", // Route name
                "feeds/comments/default", // URL with parameters
                new { controller = "Feed", action = "Comments", name = UrlParameter.Optional, area = ""
} // Parameter defaults
            );
```

در اینجا دو آدرس ذکر شده به کنترلر Feed و متدهای Posts و Comments آن هدایت خواهند شد و به این نحو کاربران قدیمی سایت هیچگونه تغییری را احساس نکرده و باز هم فیدخوان‌های آن‌ها، بدون مشکل کار خواهند کرد.

ج) پردازش لینک‌های قدیمی مطالب سایت و هدایت آن‌ها به آدرس‌های جدید

این مورد اندکی مشکل‌تر از موارد قبلی است:

```
routes.MapRoute(
                "old_bloger_post_urls",
                "{yyyy}/{mm}/{title}",
                new { controller = "Post", action = "OldBloggerLinks" },
                new { yyyy = @"\d{4}", mm = @"\d{1,2}" }
            );
```

برای نمونه آدرس مقاله‌ای مانند http://site/2012/05/ef-code-first-15.html را درنظر بگیرید. سه قسمت سال، ماه و عنوان آن، حائز اهمیت هستند. این‌ها را در اینجا به کنترلر Post و متد OldBloggerLinks آن هدایت خواهیم کرد. همچنین برای سال و ماه آن نیز قید تعریف شده است. سال عددی 4 رقمی است و ماه عددی یک تا دو رقمی.

کدهای متد OldBloggerLinks را در اینجا مشاهده می‌کنید:

```
public virtual ActionResult OldBloggerLinks(int yyyy, int mm, string title)
        {
            var oldUrl = string.Format(CultureInfo.InvariantCulture,
"http://www.dotnettips.info/{0}/{1}/{2}", yyyy, mm.ToString("00"), title);
            var blogPost = _blogPostsService.FindBlogPost(oldUrl);
            if (blogPost != null)
                return RedirectToActionPermanent(actionName: ActionNames.Index, controllerName:
MVC.Post.Name,
                                                 routeValues: new { id = blogPost.Id, name =
blogPost.Title.GetPostSlug() });
            return this.Redirect("/");
        }
```

در اینجا چون ساختار لینک‌ها کلا تغییر کرده است، ابتدا بر اساس پارامترهای دریافت شده، لینک قدیمی بازسازی می‌شود. سپس به بانک اطلاعاتی مراجعه شده و لینک قدیمی به همراه شماره مطلب مرتبط با آن یافت می‌شود (یک فیلد oldUrl برای مطالب قدیمی در بانک اطلاعاتی وجود دارد). در آخر هم به کمک متد RedirectToActionPermanent آدرس رسیده به آدرس جدید مطلب در سایت ترجمه و هدایت خواهد شد. Permanent بودن آن برای به روز رسانی خودکار اطلاعات موتورهای جستجو مفید است.

نتیجه گیری

به کمک امکانات مسیریابی توکار ASP.NET MVC می‌توان ساختار قدیمی یک سایت را به ساختار جدید آن ترجمه کرد. به این ترتیب لینک‌های قدیمی ثبت شده در صدها سایت اینترنتی که به سایت ما اشاره می‌کنند، مجددا بدون مشکل قابل استفاده بوده و همچنین موتورهای جستجو نیز امکان به روز رسانی اطلاعات خود را خواهند یافت.

عنوان: CAPTCHAfa
نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۲:۴۶
آدرس: www.dotnettips.info
گروه‌ها: CAPTCHA, reCAPTCHA, CAPTCHAfa, ASP.Net, MVC, ASP.Net MVC

حتماً با CAPTCHA آشنا هستید. فرایندی که در طی آن متنی نمایش داده می‌شود که عمدتاً فقط یک انسان قادر به درک و پاسخگویی به آن است. با این کار از ارسال داده‌های بیهوده توسط ربات‌ها جلوگیری می‌شود.
reCAPTCHA ایده‌ای است که با نمایش کلمات واقعی و اسکن شده از کتاب‌های قدیمی، بخشی از مشکلات را حل کرده و از کاربران اینترنت برای شناسایی کلماتی که رایانه توانایی خواندن آن‌ها را ندارد استفاده می‌کند. (شکل زیر)

با وارد کردن درست هر کلمه، بخشی از یک کتاب، روزنامه، و یا مجله‌ی قدیمی در رایانه شناسایی و به فرمت دیجیتال ذخیره می‌شود. به این شکل شما در دیجیتالی کردن متون کاغذی سهیم هستید.
پروژه‌ی reCAPTCHA توسط گوگل حمایت می‌شود و در این آدرس قرار دارد.
به تازگی تیمی از دانشکده فنی دانشگاه تهران به همراه انستیتو تکنولوژی ایلینویز شیکاگو، پروژه ای بر همین اساس اما برای متون فارسی با عنوان CAPTCHAfa تولید کرده اند (شکل زیر) که در این آدرس در دسترس است. امیدوارم این پروژه به گونه ای تغییر کنه که برای دیجیتالی کردن متن‌های پارسی استفاده بشه. در حال حاضر، این پروژه از کلماتی از پیش تعریف شده استفاده می‌کنه.

متاسفانه این پروژه در حال حاضر فقط توسط برنامه‌های PHP قابل استفاده است. از این رو بر آن شدم تا اون رو برای برنامه‌های ASP.NET (هم Web Forms و هم MVC) آماده کنم. برای استفاده از CAPTCHAfa نیاز به یک کلید خصوصی و یک کلید عمومی دارید که از این آدرس قابل دریافت است.
کدهای پروژه‌ی Class Library به شرح زیر است.

```
// =-=-=-=
```

```
            Stream dataStream = null;
            WebResponse response = null;
            string responseFromServer = string.Empty;

            try
            {
                WebRequest request = WebRequest.Create(CAPTCHAFA_VERIFY_SERVER);

                request.Method = "POST";
                request.ContentType = "application/x-www-form-urlencoded";

                byte[] byteData = Encoding.UTF8.GetBytes(data);

                request.ContentLength = byteData.Length;

                dataStream = request.GetRequestStream();

                dataStream.Write(byteData, 0, byteData.Length);

                dataStream.Close();

                response = request.GetResponse();

                dataStream = response.GetResponseStream();

                reader = new StreamReader(dataStream);

                responseFromServer = reader.ReadToEnd();
            }
            finally
            {
                if (reader != null)
                {
                    reader.Close();
                }

                if (dataStream != null)
                {
                    dataStream.Close();
                }

                if (response != null)
                {
                    response.Close();
                }
            }

            return responseFromServer;
        }
    }
}
```

استفاده در پروژه‌های ASP.NET Web Forms

ابتدا ارجاعی به فایل Captchafa.dll ایجاد کنید و سپس در روال Page_Load، کد زیر را قرار دهید. این کار برای تزریق اسکریپ CAPTCHAfa به صفحه استفاده می‌شود.

```
if (!IsPostBack)
{
    litCaptcha.Text = Captchafa.CaptchafaGetHtml();
}
```

litCaptcha، یک کنترل Literal است که اسکریپت تولید شده، به عنوان متن آن معرفی می‌شود.

بررسی صحت مقدار وارد شده توسط کاربر (مثلاً در روال Click یک دکمه) به صورت زیر است.

```
Captchafa captchaFa = new Captchafa();

bool isAnswerCorrect = captchaFa.IsAnswerCorrect();

if (isAnswerCorrect)
{
    // پاسخ صحیح است
}
else
{
   // پاسخ صحیح نیست
```

```
}
```

استفاده در پروژه‌های ASP.NET MVC

ابتدا ارجاعی به فایل Captchafa.dll ایجاد کنید. در ASP.NET MVC بهتره تا فرایند کار رو در یک HTML helper کپسوله کنیم.

```
public static class CaptchaHelper
{
    public static MvcHtmlString Captchafa(this HtmlHelper htmlHelper)
    {
        return MvcHtmlString.Create(Captcha.Captchafa.CaptchafaGetHtml());
    }
}
```

بررسی صحت مقدار وارد شده توسط کاربر (پس از ارسال فرم به Server) به صورت زیر است.

```
[HttpPost]
[ActionName("Index")]
public ViewResult CaptchaCheck()
{
    Captchafa captchaFa = new Captchafa();

    bool isAnswerCorrect = captchaFa.IsAnswerCorrect();

    if (isAnswerCorrect)
    {
        ViewBag.IsAnswerCorrect = true;
    }
    else
    {
        ViewBag.IsAnswerCorrect = false;
    }

    return View();
}
```

و در نهایت، کدهای View (از سینتکس موتور Razor استفاده شده است).

```
@using CaptchafaDemoMvc.Helpers;

@{
    ViewBag.Title = "Index";
}

<form action="/" method="post">

@Html.Captchafa();

<input type="submit" id="btnCaptchafa" name="btnCaptchafa" value="آزمایش" />

@{
    bool isAnswerExists = ViewBag.IsAnswerCorrect != null;
}

@if (isAnswerExists)
{
    if ((bool)ViewBag.IsAnswerCorrect == true)
    {
       <span id="lblResult">پاسخ صحیح است</span>
    }
    else
    {
       <span id="lblResult">پاسخ صحیح نیست</span>
    }
}
</form>
```

دموی پروژه رو در این آدرس قرار دادم. پروژه‌ی نمونه نیز از این آدرس قابل دریافت است.

پ.ن: به زودی برخی بهبودها رو بر روی این پروژه انجام میدم.

نظرات خوانندگان

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۵:۹

اول دیدم هر چی تست میکنم میگه صحیح نیست! ... بعد دیدم باید فاصله باشه بین دو کلمه که صحیح است! در صورتی که تو recaptcha اصلی فاضله اهمیت نداره! ... چرا اینجوریه؟!

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۵:۳۸

چون در زبان فارسی، حروف وقتی به هم بچسبن، شکلشون تغییر می‌کنه. بنابراین طبیعیه که "مسلسل" با "مسل سل" فرق کنه.

نویسنده: ایمان نعمتی
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۵:۴۰

کار جالبی کردی بهروز جان
اما به نظر میرسه خود پروژه خیلی جالب نباشه
اینکه از یه سری کلمات از پیش تعیین شده استفاده میکنه و برای نمایش هم فقط دوتا خط به شکل بیضی روی کلمات میندازه زیاد جالب نیست به نظرم. شبیه به ری‌کپتچا خواسته کار کنه اما خب چیز خوبی از کار درنیامده

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۵:۵۲

برای شروع، کار بسیار ارزشمندیه. ضمن اینکه تنوع کلمات مورد استفاده بسیار بالاست. خیلی مشتاقم که زودتر برای دیجیتالی کردن متون فارسی بشه ازش استفاده کرد، اما نیاز به یک متولی قوی و پشتیبانی خوب داره. مثل کاری که سایت گنجور انجام میده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۵:۵۳

نوشتن مشابه آن با دات نت زیاد سخت نیست. برای نمونه « بررسی نحوه برنامه نویسی سایت نستعلیق آنلاین » می‌تونه ایده خوبی برای شروع باشه. من این مطلب رو به همراه مطلب « تبدیل عدد به حروف » کنار هم گذاشتم و یک captcha فارسی ازش تهیه کردم.

نویسنده: ایمان نعمتی
تاریخ: ۱۳۹۱/۰۵/۱۵ ۱۹:۵۳

چه جالب!
امیدوارم استفاده از کپتچای فارسی در سایت‌های فارسی مرسوم بشه

نویسنده: علیرضا اسم‌رام
تاریخ: ۱۳۹۱/۰۵/۱۶ ۹:۵۶

با سلام. جهت حل مشکل کلمات از پیش تعریف شده، می‌توان از Google Translate استفاده کرد.
متاسفانه هنوز فرصت نکردم تا جزییات را دقیق ببینم، اما امیدوارم تلفظ و پخش کلمات بصورت صوتی خوب عمل کند!

نکته دیگر اینکه در مورد پیچیدگی تصویر CAPTCHA، بهترین روش استفاده از نقطه و پاره خط جهت تشویش تصویر است. معمولاً

بیشترین خطاها در OCR مربوط به این دو شکل پایه‌ی هندسی هستند. البته این موضوع بیشتر در مورد کلمات لاتین مطرح است.
شاید در مورد کلمات فارسی بیضی هایی که دوستان روی متن انداختند پیشچیدگی را افزایش دهد. آقای ایمان نعمتی لطفاً فراموش نکنیم که هدف از تشویش تصویر، سختی کار برای انسان نیست بلکه پیچیده کردن کار روبات است.
در مجموعه بسیار موضوع خوبی است. اگر دوستانی علاقه به این موضوع داشته باشند ایده و مطلب بسیار خوبی وجود دارد، هرچند پیاده سازی نشده است اما فکر می‌کنم ارزش انتشار و مطالعه را داشته باشد. البته جهت پیاده سازی آن چالشی بر سر راه نیست و فقط کمی وقت آزاد و البته تخصص نیاز دارد.

این مطلب را در این سایت میشه منتشر کرد؟ (با توجه به اینکه هنوز پیاده سازی ندارد و صرفاً بیان الگوریتم است.)

نویسنده: هوشنگ
تاریخ: ۱۳۹۱/۰۵/۱۸ ۱۱:۱۱

"با وارد کردن درست هر کلمه، بخشی از یک کتاب، روزنامه، و یا مجله‌ی قدیمی در رایانه شناسایی و به فرمت دیجیتال ذخیره می‌شود. به این شکل شما در دیجیتالی کردن متون کاغذی سهیم هستید. "

این منطقی نیست ، با همان شیوه ای که کلمه تایپ شده کاربر با عکس مقایسه میشه ، به همان طریق میشه متون کاغذی رو دیجیتالی کرد و نیازی به استفاده از این شیوه برای دیجتالی کردن نیست

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۱۸ ۱۲:۴

دوست من.
تمامی کلمات در یک متن (مخصوصاً اگر اون متن قدیمی باشه و برخی حروف یک کلمه پاک یا ناخوانا باشند) توسط رایانه و برنامه‌های OCR قابل تشخیص نیست.
مدل‌های CAPTCHAیی مانند reCAPTCHA که بر اساس دیجیتالی کردن متون پیاده سازی شدند، دو کلمه رو به کاربر نشان میدن:
1) کلمه ای که توسط رایانه به درستی تشخیص داده شده
2) کلمه یا بخشی از اون که رایانه نتونسته اون رو به درستی تشخیص بده.
در صورتی که کاربر، کلمه ای که توسط رایانه تونسته تشخیص داده بشه و برای رایانه مشخص هست رو صحیح وارد کنه، فرض بر این گرفته میشه که کلمه‌ی دوم یا بخشی از کلمه‌ی دوم که قابل تشخیص نبوده رو هم تونسته به درستی وارد کنه.
پس از این، میانگینی از تعداد عبارت ورودی کاربران برای کلمه‌ی نامشخص گرفته میشه و عبارتی که بیشترین فراوانی رو داشته به عنوان مورد قابل قبول ثبت میشه.

نویسنده: بهروز راد
تاریخ: ۱۳۹۱/۰۵/۲۸ ۱۲:۵۰

کامپوننت رو آپدیت کردم. نسخه‌ی اولیه بر اساس وفاداری به کدهای PHP و برگردانی از اونها بود. در نسخه‌ی جدید، متد CaptchafaGetHtml به GetHtml تغییر پیدا کرد، مدیریت خطاها اضافه شد و کلیدهای خصوصی و عمومی باید در زمان استفاده به کامپوننت پاس داده شوند. نسخه‌ی جدید را از لینک انتهای مقاله دریافت کنید.

نویسنده: بابک
تاریخ: ۱۳۹۳/۰۸/۲۸ ۱۵:۲۳

آدرس ظاهرا به آدرس زیر تغییر کرده است:
http://www.captchafa.net

نویسنده: محمد رضا صفری

تاریخ: ۱۳۹۳/۱۱/۲۰ ۱۳:۱۲

فکر کنم که کلا پروژه تعطیل شده ...

عنوان: **ایجاد یک Repository در پروژه برای دستورات EF**
نویسنده: میثم ثوامری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۱:۴۷
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Design patterns, Entity framework, VB.NET

ایجاد یک Pattern در پروژتون میتونه نظم, سرعت و زیبایی خاصی به کدتون بده. با وجود frameworkهای و Patternهایی مسه MVC و MVVM برنامه نویسان را وادار کنه که همه Actionهای یک پروژه رو به سمت کلاینت ببرن.تو یک فرصت دیگه در مورد فریمورک Knockout حتما تاپیک میزارم.امروز میخوام یک Pattern با استفاده از یک Interface و codefirst model براتون بزارم.

گام اول:ایجاد که class property

```
Public Class Employee
    Public Property ID As Integer
    Public Property Fname As String
    Public Property Bdate As DateTime
End Class
```

گام دوم:ایجاد بانک با استفاده از CodeFirst

```
Imports System.Data.Entity
Public Class EmployeeDBContext : Inherits DBContext
    Public Property Employees As DbSet(Of Employee)
End Class
```

گام سوم: ایجاد repository با استفاده از interface

```
Interface EmployeeRepository
    ReadOnly Property All As List(Of Employee)
    Function Find(id As Integer) As Employee
    Sub InsertOrUpdate(p As Employee)
    Sub Delete(id As Integer)
    Sub Save()
End Interface
```

گام چهارم: تعریف کلاس برای implement کردن از iInterface

```
Public Class EmployeeClass : Implements EmployeeRepository
    Private DB As New EmployeeDBContext
    Public ReadOnly Property All As List(Of Employee) Implements EmployeeRepository.All
        Get
            Return DB.Employees.ToList()
        End Get
    End Property

    Public Sub Delete(id As Integer) Implements EmployeeRepository.Delete
        Dim query = DB.Employees.Single(Function(q) q.ID = id)
        DB.Employees.Remove(query)
    End Sub

    Public Function Find(id As Integer) As Employee Implements EmployeeRepository.Find
        Return DB.Employees.Where(Function(q) q.ID = id)
    End Function

    Public Sub InsertOrUpdate(p As Employee) Implements EmployeeRepository.InsertOrUpdate
        If p.ID = Nothing Then
            DB.Employees.Add(p)
        Else
            DB.Entry(p).State = Data.EntityState.Modified
        End If
    End Sub

    Public Sub Save() Implements EmployeeRepository.Save
        DB.SaveChanges()
    End Sub
```

```
End Class
```

برای استفاده تو پروژه براحتی میتونید یک instance از classتون ایجاد کنید و ..

```
Dim cls As New EmployeeClass
```

```
Public Sub BindGrid()
        GridView1.DataSource = cls.All
        GridView1.DataBind()
    End Sub
```

موفق باشید

نظرات خوانندگان

نویسنده: علیرضا صالحی
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۲:۱۳

برای مواردی که خروجی یک لیست (تعدادی آیتم) باشد از Property استفاده نمی‌شود. مثلا برای All باید از Method استفاده کنید. Properties vs. Methods
بهتر است برای خروجی متد هایی مانند All نیز به جای لیست از IEnumerable یا IQueryable استفاده کنید.
متدهای Update و Insert نیز به طور جداگانه تعریف شوند. (قرار است هر متد تنها یک وظیفه داشته باشد)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۲:۲۱

- در مورد آرایه بحث شده در MSDN. ضمن اینکه استفاده از متد عموما برای حالتیکه عملیات قابل توجهی در بدنه آن قرار است صورت گیرد، توصیه می‌شود . البته در اینجا چون عملیات دریافت اطلاعات از بانک اطلاعاتی می‌تواند سنگین درنظر گرفته شود، استفاده از متد ارجحیت دارد. خواص نمایانگر اطلاعاتی سبک و با دسترسی سریع هستند.
- خروجی لیست بهتر است. (^) + اگر ReSharper جدید را نصب کنید استفاده از IEnumerable را نیز توصیه نمی‌کند ؛ چون ممکن است چندین بار رفت و برگشت به بانک اطلاعاتی در این بین صورت گیرد.
- مشکلی ندارد. خود EF Code first چنین متدی را دارد. (^) بحث کلاس تک وظیفه‌ای متفاوت است با متدی که نهایتا قرار است اطلاعات یک رکورد را در بانک اطلاعاتی تغییر دهد (اگر نبود ثبتش کند؛ اگر بود فقط همان رکورد مشخص را به روز رسانی کند).

نویسنده: ناشناس
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۲:۳۷

با سلام
دلیل استفاده از Interface EmployeeRepository چیه؟
دقیقا دلیل استفاده Interface اینجا چیه؟
با تشکر از مطلب خوبتون.

نویسنده: ناشناس
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۲:۴۵

اینجا شاید استفاده از IQueryable بهتر باشه.
شاید کاربر بخواد قبل از نمایش اطلاعات اونو فیلتر کنه یا اینکه بهتره دو متد Find داشته باشی یکی با خروجی یک آیتم و دیگری با خروجی چندین آیتم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۲:۴۸

خیر (^). طراحی یک لایه سرویس که خروجی IQueryable دارد نشتی دار درنظر گرفته شده و توصیه نمی‌شود. اصطلاحا leaky abstraction هم به آن گفته می‌شود؛ چون طراح نتوانسته حد و مرز سیستم خودش را مشخص کند و همچنین نتوانسته سازوکار درونی آن‌را به خوبی کپسوله سازی و مخفی نماید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۲:۵۰

به دو دلیل:
- استفاده از امکان تزریق وابستگی‌ها

- امکان نوشتن ساده‌تر آزمون‌های واحد با فراهم شدن زیر ساخت mocking اشیاء

نویسنده: ناشناس
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۳:۲

در همین پست خود شما تعداد زیاد رکوردها رو مثال زدیدو این پیاده سازی از این موضوع رنج میبره. و در مورد IQueryable قبول دارم و گفتم که بهتر است از دو یا چند متد find استفاده کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۳:۲۵

منظور از آن مطلب این بود که از ابزاری که در اختیار دارید درست استفاده کنید. اگر قرار است دو یا چند جستجو را انجام دهید، اینکارها بله باید با IQueryable داخل یک متد انجام شود، اما خروجی متد فقط باید لیست حاصل باشد؛ نه IQueryable ایی که انتهای آن باز است و سبب نشتی لایه سرویس شما در لایه‌های دیگر خواهد شد.

نویسنده: میثم ثوامری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۹:۵۴

برای برنامه نویسا پیدا کردن یک property راحتره در ضمن از property برای تزریق یا بازیابی اطلاعات از یک object استفاده میکنن.
IQueryable در واقع توسعه یافته IEnumerable.تفاوت عمدشون در LINQ operators که در IQueryable استفاده میشه.اگر هم بخوایم دلیل پیشنهادی داده باشیم اونم اینه که مدیریت حافظه در IQueryable رعایت شده در حالی که Lisrtها کامپایلرو مجاب به اجنام دستور تا انتها میکنن.

نویسنده: میثم ثوامری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۱۹:۵۹

مهندس با نظر دوستمون موافقم
IQueryable بهترین انتخاب برای remote data source که میشه به database یا webserviceها اشاره کرد.بطور کل اگر شما از ORM مسه linqtosql استفاده میکنید
IQueriable : کوئری شمارو به دستورات sql در database server تبدیل میکنه
IEnumerable: همه رکوردهای شما قبل از اینکه بسمت دیتابیس برن بصورت object در memory نگهداری میشن.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۵/۱۹ ۲۰:۱۱

IQueryable فقط یک expression است. هنوز اجرا نشده. (expose آن از طریق وب سرویس اشتباه است و به مشکلات serialization برخواهید خورد.)
زمانیکه First، ToList و امثال آن روی این عبارت فراخوانی شود تبدیل به SQL شده و سپس بر روی بانک اطلاعاتی اجرا می‌شود. به این deferred execution یا اجرای به تعویق افتاده گفته می‌شود.
اگر این عبارت را در اختیار لایه‌های دیگر قرار دهید، یعنی انتهای کار را بازگذاشته‌اید و حد و حدود سیستم شما مشخص نیست. شما اگر IQueryable بازگشت دهید، در لایه‌ای دیگر می‌شود یک join روی آن نوشت و اطلاعات چندین جدول دیگر را استخراج کرد؛ درحالیکه نام متد شما GetUsers بوده. بنابراین بهتر است به صورت صریح اطلاعات را به شکل List بازگشت دهید، تا انتهای کار باز نمانده و طراحی شما نشتی نداشته باشد.

نویسنده: محمد عامریان
تاریخ: ۱۳۹۱/۰۸/۱۸ ۱۷:۶

با سلام من یک معماری طراحی کردم به شکل زیر

ابتدا یک اینترفیس به شکل زیر دارم

```
using System;
using System.Collections;
using System.Linq;

namespace Framework.Model
{
    public interface IContext
    {
        T Get<T>(Func<T, bool> prediction) where T : class;
        IEnumerable List<T>(Func<T, bool> prediction) where T : class;
        void Insert<T>(T entity) where T : class;
        int Save();
    }
}
```

بعد یک کلاس ازش مشتق شده

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Data;
using System.Data.Entity;
using System.Linq;
using System.Text;

namespace Framework.Model
{
    public class Context : IContext
    {
        private readonly DbContext _dbContext;

        public Context(DbContext context)
        {
            _dbContext = context;
        }

        public T Get<T>(Func<T,bool> prediction) where T : class
        {
            var dbSet = _dbContext.Set<T>();
            if (dbSet!= null)
                return dbSet.Single(prediction);

            throw new Exception();
        }

        public void Insert<T>(T entity) where T : class
        {
            var dbSet = _dbContext.Set<T>();
            if (dbSet != null)
            {
                _dbContext.Entry(entity).State = EntityState.Added;
            }
        }

        public int Save()
        {
            return _dbContext.SaveChanges();
        }


        IEnumerable IContext.List<T>(Func<T, bool> prediction)
        {
            var dbSet = _dbContext.Set<T>();
            if (dbSet != null)
                return dbSet.Where(prediction).ToList();

            throw new Exception();
        }
    }
}
```

سپس یک کلاش context دارم که مستقیما از dbcontext مشتق شده

```
using System.Data.Entity;
using DataModel;

namespace Model
{
    public class EFContext : DbContext
    {
        public EFContext(string db): base(db)
        {

        }

        public DbSet<Product> Products { get; set; }
    }
}
```

و سپس کلاس دارم که اومده پیاده سازی کرده context که خودم ساختمو

```
using System;
using System.Collections.Generic;
using System.Data.Entity;
using System.Linq;
using System.Text;

namespace Model
{
    public class Context : Framework.Model.Context
    {
        public Context(string db): base(new EFContext(db))
        {

        }
    }
}
```

در پروژه دیگری اومدم یک کلاس context جدید ساختم

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;

namespace Biz
{
    public class Context : Model.Context
    {
        public Context(string db) : base(db)
        {

        }
    }
}
```

و در کنترلر هم به این شکل ازش استفاده کردم

```
using System.Web.Mvc;
using Framework.Model;

namespace ProductionRepository.Controllers
{
    public class BaseController : Controller
    {
        public IContext DataContext { get; set; }

        public BaseController()
        {
            DataContext = new
Biz.Context(System.Configuration.ConfigurationManager.ConnectionStrings["Database"].ConnectionString);
        }
    }
}
```

```
using System.Web.Mvc;
using DataModel;
using System.Collections.Generic;

namespace ProductionRepository.Controllers
{
    public class ProductController : BaseController
    {
        public ActionResult Index()
        {
            var x = DataContext.List<Product>(s => s.Name != null);
            return View(x);
        }

    }
}
```

و این هم تست

```
using NUnit.Framework;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Web.Mvc;

namespace TestUnit
{
    [TestFixture]
    public class Test
    {
        [Test]
        public void IndexShouldListProduct()
        {
            var repo = new Moq.Mock<Framework.Model.IContext>();
            var products = new List<DataModel.Product>();
            products.Add(new DataModel.Product { Id = 1, Name = "asdasdasd" });
            products.Add(new DataModel.Product { Id = 2, Name = "adaqwe" });
            products.Add(new DataModel.Product { Id = 4, Name = "qewqw" });
            products.Add(new DataModel.Product { Id = 5, Name = "qwe" });
            repo.Setup(x => x.List<DataModel.Product>(p => p.Name !=
null)).Returns(products.AsEnumerable());
            var controller = new ProductionRepository.Controllers.ProductController();
            controller.DataContext = repo.Object;
            var result = controller.Index() as ViewResult;
            var model = result.Model as List<DataModel.Product>;
            Assert.AreEqual(4, model.Count);
            Assert.AreEqual("", result.ViewName);

        }
    }
}
```

نظرتون چیه آقای نصیری

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۱۸ ۱۷:۱۳

موارد 1 و 2 عنوان شده در این مطلب رو تکرار کرده: (^)

نویسنده: مجید پارسا
تاریخ: ۱۳۹۳/۰۷/۱۲ ۱۲:۹

با سلام؛ سوالی که وجود داره اینه که با استفاده از repository pattern چطور میتونیم join بزنیم. با توجه به نظرات قبلی توصیه شده است که از خروجی IQueryable نباید برای لایه داده استفاده شود.
در این صورت در هنگام نوشتن دستورات join ابتدا تمامی رکوردهای جداول مورد نظر توسط الگوی repository به حافظه load می‌شود، با توجه به ماهیت linq to object بودن کوئری مورد نظر(join) اجرای برنامه به لحاظ زمانی و مصرف حافظه از

کارایی خوبی برخوردار نخواهد بود.
در این حالت یا می‌باست از خیر کارایی بالاتر گذشت یا از خروجی IQueryable استفاده کرد که در تضاد با پیشنهاد دوستان گرامی می‌باشد.
آیا در این حالت منطقی است joinهای پر استفاده را با خروجی IEnumerable در repository مربوط به خودش نوشت یا راهکار دیگری وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۱۲ ۱۲:۱۹

- الگوی مخزن عمومی (Generic repository pattern)، لایه داده برنامه نیست. زمانیکه از یک ORM استفاده می‌کنید، لایه داده برنامه همان ORM است.
- الگوی مخزن عمومی، عمده‌ی کارش مخفی کردن ساز و کار ORM مورد استفاده از لایه سرویس برنامه است (^).
- اگر از الگوی عمومی مخزن استفاده می‌کنید، سطح دسترسی آن‌را internal تعریف کنید تا محدود شود به لایه سرویس برنامه. داخل لایه سرویس برنامه به هر نحوی که علاقمندید از آن استفاده کنید. نهایتا این لایه سرویس است که خروجی IList یا IEnumerable نهایی را در اختیار مصرف کننده قرار می‌دهد.

نویسنده: مجید پارسا
تاریخ: ۱۳۹۳/۰۷/۱۲ ۱۶:۸

با تشکر، از آنجا که من اولین بار است که به شکل حرفه‌ای برنامه نویسی سه لایه را تجربه می‌کنم با توجه به توضیحات شما این طور متوجه شدم که پیاده سازی کلاس‌های Repository در لایه سرویس صورت گیرد اگر اشتباه نکنم.

در صورت امکان بیشتر موضوع رو باز کنید (منظورم آماتوری تره)
نمونه برنامه‌های سه لایه موجود در اینترنت پیدا کردم در حد CRUD ساده و با استفاده از الگوی مخزن عمومی بوده. مانند مثال‌های سایت asp.net در صورت معرفی نمونه کاملتر و واقعی‌تر ممنون میشوم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۱۲ ۱۷:۲۹

مراجعه کنید به مسیر راه EF Code first ، انتهای مطلب، قسمت لایه بندی پروژه‌های EF Code first

عنوان: **فشرده سازی خروجی فایلهای استاتیک سایت**
نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۶/۱۵ ۱۱:۱۸
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, GZip, Deflate

زمانیکه صفحه‌ای لود می‌شود، به همراه خود محتویاتی از قبیل تصویر و فایلهای جاوا اسکریپت را نیز به سیستم کاربر حمل می‌کند. در ASP.NET می‌توان از خاصیت فشرده سازی gzip , gzip جهت ارائه محتویات سرور استفاده کرد:

برای پیاده سازی آن می‌توان از رویداد Application_PreRequestHandlerExecute در فایل Global.asax استفاده کرد:

```
using System;
using System.IO.Compression;
using System.Web;
using System.Web.UI;

namespace GZipTest
{
    public class Global : HttpApplication
    {
        void Application_PreRequestHandlerExecute(object sender, EventArgs e)
        {
            var app = sender as HttpApplication;
            if (app == null) return;

            var acceptEncoding = app.Request.Headers["Accept-Encoding"];
            var prevUncompressedStream = app.Response.Filter;

            if (!(app.Context.CurrentHandler is Page ||
              app.Context.CurrentHandler.GetType().Name == "SyncSessionlessHandler") ||
            app.Request["HTTP_X_MICROSOFTAJAX"] != null)
                return;

            if (string.IsNullOrEmpty(acceptEncoding))
                return;

            acceptEncoding = acceptEncoding.ToLower();

            if (acceptEncoding.Contains("deflate") || acceptEncoding == "*")
```

```
            {
                // defalte
                app.Response.Filter = new DeflateStream(prevUncompressedStream,
CompressionMode.Compress);
                app.Response.AppendHeader("Content-Encoding", "deflate");
            }
            else if (acceptEncoding.Contains("gzip"))
            {
                // gzip
                app.Response.Filter = new GZipStream(prevUncompressedStream, CompressionMode.Compress);
                app.Response.AppendHeader("Content-Encoding", "gzip");
            }
        }
    }
}
```

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۵ ۱۳:۲۲

در مورد تصویر ارسالی:
در این سایت از این روش (System.Web.Optimization) استفاده می‌شود. روش یاد شده در IIS7 خروجی فشرده شده با GZip نیز دارد؛ اما نه در IIS6. ولی در هر دو حالت، کش را تنظیم می‌کند:

↑ TYPE	SIZE (KB)	GZIP (KB)	COOKIE RECEIVED (bytes)	COOKIE SENT (bytes)	HEADERS	URL	EXPIRES (Y/M/D)
⊞ doc (1)	28.4K						
⊟ js (3)	256.9K						
js	36.8K	14.8K				http://www.google-analytics.com/ga.js	2012/9/5
js	179.2K					http://www.dotnettips.info/Scripts/js?...	2013/9/4
js	40.8K					http://www.dotnettips.info/Content/redactor/js?...	2013/9/4

به همین جهت در بار بعدی مشاهده سایت، دیگر درخواست اضافه‌ای جهت دریافت اسکریپت‌ها و شیوه نامه‌ها ارسال نخواهد شد:

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۶/۱۵ ۱۳:۲۸

ممنون،راهکارهایی که شما قبلا برای کاهش درخواست بین کلاینت و سرور داده بودید حداقل برای بنده خیلی کارگشا بوده و این مطلب برای استفاده بهتر از امکان فشرده سازی آورده شده.
پ ن:عکس برای درک مسئله آورده شده بود!

نویسنده: علیرضا صالحی
تاریخ: ۱۳۹۱/۰۶/۱۵ ۱۵:۲۴

روی ویندوز سرورهای 2008 (IIS 7 و بالاتر) این فشرده سازی توسط IIS فقط با تنظیم ساده ای در web.config انجام می‌شود. در اغلب هاستینگ هایی هم که من استفاده کردم این قابلیت فعال بوده است و نیازی به کدنویسی نیست. و البته بستگی به نوع پروژه هم ندارد.

```
<httpCompression directory="%SystemDrive%\inetpub\temp\IIS Temporary Compressed Files">
<scheme name="gzip" dll="%Windir%\system32\inetsrv\gzip.dll" doStaticCompression="true"/>
<staticTypes>
<add mimeType="text/*" enabled="true" />
<add mimeType="message/*" enabled="true" />
<add mimeType="application/x-javascript" enabled="true" />
<add mimeType="*/*" enabled="false" />
</staticTypes>
</httpCompression>
    <urlCompression doStaticCompression="true" doDynamicCompression="false" />
```

اطلاعات بیشتر :

HTTP Compression

Scheme

URL Compression

نویسنده: احسان
تاریخ: ۱۳۹۱/۰۶/۱۵ ۲۳:۴۲

آقای نصیری ممنونم از این مطلبتون اما ی سوال :
من روی سرورم IIS 6 دارم و بدون هیچ گونه تنظیم خاصی می‌بینم که در سربرگ Net، همه Responseها بطور خودکار gzip شدن. بدون اینکه کدی نوشته باشم یا کاری کرده باشم.

↑ TYPE	SIZE (KB)	GZIP (KB)
⊟ doc (1)	10.9K	
doc	10.9K	3.6K
⊟ js (2)	122.6K	
js	94.8K	32.6K
js	27.8K	4.6K
⊟ css (2)	3.4K	
css	1.9K	0.7K
css	1.5K	0.6K
⊞ cssimage (5)	19.2K	

آیا خود IIS6 بطور خودکار این فشرده سازی را انجام می‌دهد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۱۵ ۲۳:۴۵

بله. سرور دار یک سری تنظیمات رو می‌تونه اعمال کنه تا اینکار برای تمام سایت‌های هاست شده خودکار شود.

نویسنده: احسان
تاریخ: ۱۳۹۱/۰۶/۱۶ ۰:۰

ممنون از توجهتون.

نویسنده: میلاد
تاریخ: ۱۳۹۱/۰۶/۳۱ ۱۴:۲۳

سلام و ممنون از مطالب مفیدتون،
من به تازگی وب سایتی ساختم و با خوندن این مطلب ، فشرده سازی رو روی سایت انجام دادم.
اما مشکل اینجاست که این فشرده سازی تنها روی سرور IIS لوکال جواب میده و وقتی که سایت رو روی هاست آپلود می‌کنم ،

هیچ فشرده سازی انجام نمیشه.
لطفا راهنماییم کنید.
با تشکر

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۱/۰۶/۳۱ ۱۶:۲۴

همونطور که مهندس نصیری در کامنت اول این پست گفتند این تنظیمات بر روی iis نسخه 7 به بالا به این صورت هم جواب میده ولی برای نسخه‌های پائین باید از iis خود سرور کمک گرفت ولی در کل منظور از ایجاد این پست معرفی این نوع اعمال، برای سایتها است

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۳۱ ۱۶:۵۴

+ Header **Append** عموما روی IIS6 مشکل ساز است و بهتر است از AddHeader استفاده شود.

نویسنده: میلاد
تاریخ: ۱۳۹۱/۰۶/۳۱ ۲۳:۵۴

سلام
ممنون از پاسختون...
با AddHeader هم امتحان کردم... اما موفق نشدم... لطفا یک نگاهی به تصویر ضمیمه بیاندازید... شاید چیزی دستگیرتون بشه:

Request Headers

GET [illegible]/script/smoothDivScroll.css HTTP/1.1

Client

Accept: text/css,*/*;q=0.1
Accept-Encoding: gzip, deflate
Accept-Language: en-us,en;q=0.5
User-Agent: Mozilla/5.0 (Windows NT 6.1; rv:15.0) Gecko/20100101 Firefox/15.0.1

Cookies / Login

Cookie
ASP.NET_SessionId=vcr5comle4tzouramit1ksds

Miscellaneous

Referer: [illegible]

Transport

Connection: keep-alive
Host: [illegible]

Get SyntaxView | Transformer | Headers | TextView | ImageView | HexView
Raw | JSON | XML

Response Headers

HTTP/1.1 200 OK

Cache

Date: Fri, 21 Sep 2012 20:15:49 GMT

Entity

Content-Length: 2296
Content-Type: text/css
ETag: "d9e9b9c2f796cd1:0"
Last-Modified: Thu, 20 Sep 2012 06:18:33 GMT

Miscellaneous

Accept-Ranges: bytes
Server: Microsoft-IIS/7.5
X-Powered-By: ASP.NET
X-Powered-By-Plesk: PleskWin

Open in Google Docs Viewer
Open link in new tab
Open link in new window

Open link in new incognito window

Download file
Copy link address
Edit PDF File on PDFescape.com

با تشکر.
Open in Google Docs Viewer
Open link in new tab
Open link in new window
Open link in new incognito window

Download file
Copy link address
Edit PDF File on PDFescape.com

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۰۱ ۰:۳۰

شما که از IIS7.5 استفاده می‌کنید یا از این روش یا از روش‌های کش کردن اطلاعات استفاده کنید.

نویسنده: میلاد
تاریخ: ۱۳۹۱/۰۷/۰۱ ۰:۵۲

سلام و تشکر
از روش جناب صالحی هم استفاده کردم اما جواب نداد... سوالی که الان برام ایجاد شده اینه که آیا ممکنه هاست من استفاده از فشرده سازی رو غیرفعال کرده باشه؟
و اینکه آیا مگر کش کردن و فشرده سازی مقوله‌های جدایی نیستند؟!
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۰۱ ۰:۵۵

- ممکنه این مورد روی IIS7.5 شما فعال نباشد. باید با پشتیبانی هاست صحبت کنید.
- هستند. ولی اگر کش کردن را فعال کنید، کاربر یکبار که فایل استاتیک شما را دریافت کرد، بار بعد درخواستی را به سرور ارسال نمی‌کند. به همین جهت در کل بار کمتر و نتیجه بهتری نسبت به فشرده سازی دارد که هر بار باید اعمال شود.

نویسنده: میلاد
تاریخ: ۱۳۹۱/۰۷/۰۱ ۱:۴

خیلی ممنون از صبر و شکیباییتون...
فقط یک سوال دیگه :
اگر این قابلیت از طرف هاست غیر فعال باشه ، روی کدهایی که درون رویداد Application_PreRequestHandlerExecute نوشته میشن هم تاثیر داره؟
اگر جواب منفی هست ، پس چرا به نظر میاد اصلا این رویداد روی هاست اجرا نمیشه؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۷/۰۱ ۱:۱۰

- خیر.
- من دسترسی به سرور و کدهای شما برای دیباگ ندارم. از ELMAH استفاده کنید. شاید خطایی در کار باشد که با ELMAH لاگ می‌شود. شاید اصلا این روال فراخوانی نمی‌شود که بهتر است این نوع کدها را به یک http module منتقل کرد و نه اینکه مستقیما داخل global.asax.cs نوشت.

نویسنده: شاهین کیانی
تاریخ: ۱۳۹۲/۱۱/۱۱ ۲:۴۴

سلام من از این کد چطوری استفاده کنم در برنامم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۱ ۹:۳۹

همانطور که در متن نوشتند، این کدها رو باید به فایل global.asax.cs برنامه اضافه کنید. عینا داخلش قرار بدید کار می‌کند.

عنوان: **بار کردن ساعت و تاریخ فعلی سرور با JQuery Ajax**
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۷:۱۴
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, jQuery, Ajax

در این مطلب می‌خواهم شما را با نحوه بار گزاری ساعت و تاریخ سیستم سرور با استفاده از JQuery Ajax آشنا کنم.
در بعضی از سایتها با استفاده از جاوا اسکریپت تاریخ و ساعت جاری سیستم کلاینت به او نشان داه می‌شود.
این روش یک مزیت دارد: اول اینکه این کدها سمت کلاینت اجرا میشن و برای سرور بار اضافی ایجاد نمیکنن.
و یک عیب هم دارد: در صورتی که ساعت و تاریخ روی سیستم کلاینت تنظیم نباشد، همین ساعت و تاریخ نادرست برای او نمایش داده می‌شود. همین عیب می‌تواند باعث افت کیفیت وب سایت شود.

اما راهی هست که تاریخ و ساعت سیستم سرور برای کاربر نشان داده شود و آن هم استفاده از JQuery Ajax هست. به صورتی که هر ثانیه درخواستی برای یک handler فرستاده می‌شود و آن handler نیز ساعت و تاریخ روی سرور را باز می‌گرداند و این مقدار بازگشته شده را می‌توان در تگی از صفحه وب نمایش داد.

مثال: ابتدا یک صفحه aspx می‌سازیم و تگ زیر را در آن قرار می‌دهیم:

```
<p id="datetime"></p>
```

ساعت و تاریخ بار شده از سرور در این تگ باید نشان داده شود.

سپس کدهای اسکریپت زیر را می‌نویسیم:

```
var auto_referesh = setInterval
 (
     function()
    {
         $.post
         (
            "GetDateTime.ashx",
             function (result)
            {
                $('#datetime').html(result);
            }
        );
     }, 1000


 );
```

با نوشتن این کدها هر ثانیه یک بار، بوسیله Ajax درخواستی برای یک handler به اسم GetDateTime.ashx فرستاده می‌شود.
وظیفه این handler برگرداندن تاریخ و ساعت فعلی سیستم سرور است. بعد از دریافت مقدار این مقدار از این handler، آنرا در تگ با شناسه datetime قرار می‌دهیم.

کد استفاده شده در handler هم به این صورت است:

```
<%@ WebHandler Language="C#" Class="GetDateTime" %>

using System;
using System.Web;

public class GetDateTime : IHttpHandler {

    public void ProcessRequest (HttpContext context) {
        context.Response.ContentType = "text/plain";
        context.Response.Write(DateTime.Now.ToString());
    }

    public bool IsReusable {
```

```
        get {
            return false;
        }
    }

}
```

در انتها فایل ضمیمه این مثال را از این لینک دریافت کنید:

AjaxDateTime.zip

نظرات خوانندگان

نویسنده: صابر فتح اللهی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۰

سلام
روش شما خوبه اما عیبی که داره اینه که هر ثانیه ساعت از سرور میگیره و باعث سربار میشه
به نظر من روش بهتر اینه از همون کد طرف کلاینت استفاده بشه اما زمان اولیه از سرور خونده بشه بعد طرف کلاینت این ساعت خودمون اضافه کنیم و ساعت شبیه سازی کنیم
در صورتی هم که کاربر به صفحه دیگری بره یا صفحه رفرش کنه دوباره ساعت از سرور خونده میشه و الگوریتم ساعت طرف کلاینت به کار می‌افته
به نظر من این روش بهینه‌تر هست حالا باز ببینیم دوستان چه نظری دارن
موفق وموید باشید

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۳۷

سلام
بله این روش سربار اضافی داره. روشی که گفتید خیلی بهتر هست.
ولی خوب مسئله اینه که چطور میشه داده نوع DateTime در ASP.Net رو با jQuery خوند؟
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۰:۵۵

روش دوم: خود وب سرور هم با درخواست‌های Head تاریخ رو ارسال می‌کنه. به این صورت هم قابل خواندن است:

```
<script src="Scripts/jquery.min.js" type="text/javascript"></script>
    <script type="text/javascript">
        $(document).ready(function () {
            try {
                var result = $.ajax({ 'type': 'HEAD', 'url': '/' }).success(function () {
                    var date1 = new Date(result.getResponseHeader('Date'));
                    alert(date1);
                });
            }
            catch (err) {
                //...
            }
        });
    </script>
```

و مهم‌ترین مزیتش این است که با تمام وب سرورهای استاندارد کار می‌کنه و فرقی نمی‌کنه کد شما PHP است یا ASP.NET.
+ این رو هم باید درنظر داشت که حین پردازش تاریخ دریافتی از وب سرور باید مسایل GMT را هم لحاظ کرد تا تاریخ و زمان دریافتی با زمان ایران تطابق پیدا کند.

نویسنده: احمد احمدی
تاریخ: ۱۳۹۱/۰۶/۲۵ ۱۲:۵

سلام
همونطور که جناب فتح اللهی فرمودند، بهتر هست که یک بار زمان را از سرور دریافت کنید و سپس بروزرسانی را بوسیله جاوا اسکریپت انجام دهید.
بنده در ارتباط با این بحث، یک کلاس ساده نوشتم که می‌تونه عملیات بروزرسانی را انجام بده.

```
var MyTime = function () {
    var date;
    var tag;

    var init = function (hour, minute, seconds, tagId) {
        var constructor = getConstructorString(hour, minute, seconds);
        date = new Date(constructor);
        tag = document.getElementById(tagId);
        //console.log('MyTime : Init(%s, %s, %s, %s)', hour, minute, seconds, tagId);
    };

    var run = function () {
        update();
        window.setInterval(update, 1000);
        //console.log('MyTime : Run');
    };

    var update = function updateClock() {
        var h = date.getHours();
        var m = date.getMinutes();
        var s = date.getSeconds();

        s++;
        if (s == 60) { m++; s = 0; };
        if (m == 60) { h++; m = 0; };
        if (h == 13) h = 1;

        var constructor = getConstructorString(h, m, s);
        date = new Date(constructor);

        h = (h < 10) ? ("0" + h) : h;
        m = (m < 10) ? ("0" + m) : m;
        s = (s < 10) ? ("0" + s) : s;

        tag.innerHTML = h + ":" + m + ":" + s;
        //console.log('MyTime : update');
    };

    var getConstructorString = function (hour, minute, seconds) {
        //console.log('MyTime : getConstructorString');
        return '01/01/2000 ' + hour + ':' + minute + ':' + seconds;
    };

    return {
        Init: init,
        Run: run
    };
}
```

روش کار به این صورت هست که شما یک بار متد Init را به همراه پارامترهای (ساعت، دقیقه، ثانیه، آی‌دی تگ) فراخوانی می‌کنید، در نهایت برای اجرای ساعت، یک بار هم متد Run را فراخوانی می‌کنید.

مثال :

```
myTime = new MyTime();
myTime.Init(12, 59, 50, 'clock');
myTime.Run();
```

امیدوارم مفید بوده باشه .

عنوان: **یک نکته از ASP.NET 4.5 GridView**
نویسنده: ا.ح.م.د
تاریخ: ۱۳۹۱/۰۶/۲۷ ۲۳:۹
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, GridView, Paging

تا قبل از ASP.NET 4.5 ، هنگام کار با GridView رسم بر این بوده که به خاصیت DataSource ، یک منبع داده (مانند SqlDataSource و ...) را Bind کرده و متد DataBind را صدا نموده و نتیجه نمایش داده می‌شد.
اما با استفاده از ویژگی‌های جدید اضافه شده(هر چند با تأخیر نسبت به Gridهای پیشرفته دیگر) کار با این کنترل راحت‌تر و خواناتر شده است. یکی از این ویژگی‌ها را با هم بررسی می‌کنیم:
با استفاده از ویژگی SelectMethod میتوان متدی را به GridView معرفی کرد که وظیفه منبع داده را انجام داده و هنگام Bind فراخوانی شده و گرید را پر کند:
مثال:

```
<asp:GridView ID="gvCities"
                runat="server"
                AutoGenerateColumns="False"
                ItemType="WebApplication3.City"
                SelectMethod="GetAllCities">
                <Columns>
                    <asp:TemplateField HeaderText="نام">
                        <ItemTemplate><%#: Item.Name %></ItemTemplate>
                    </asp:TemplateField>
                </Columns>
     </asp:GridView>
```

نکته مهم در این کد ItemType است. با استفاده از این خاصیت به جای اینکه مانند قبل نام فیلدهایی که قرار است در گرید نمایش داده شود را بصورت string معرفی کنیم (مثلا در اینجا Eval("Name") ، اگر نام فیلد را غلط بنویسیم هنگام کامپایل خطایی صادر نمی‌شود)، آنرا بصورت Strongly Type نوشته و از اشتباه جلوگیری می‌کنیم.(+)
کد متد:

```
public IQueryable<City> GetAllCities()
        {
            var context = new EFContext();
            var q = from c in context.City
                    orderby c.Name
                    select c;
            return q;
        }
```

و سپس دستور زیر را فراخوانی می‌کنیم:

```
gvCities.DataBind();
```

اگر بخواهیم در گرید Paging داشته باشیم بصورت زیر عمل می‌کنیم:

```
<asp:GridView ID="gvCities"
                runat="server"
                AutoGenerateColumns="False"
                AllowPaging="True"
                PageSize="10"
                ItemType="WebApplication3.City"
                SelectMethod="GetAllCities">
                 <Columns>
                    <asp:TemplateField HeaderText="نام">
                        <ItemTemplate><%#: Item.Name %></ItemTemplate>
                    </asp:TemplateField>
                </Columns>
  </asp:GridView>
```

که در اینجا دو خصوصیت AllowPaging و PageSize را مقدار دهی کرده ایم. این خصوصیت‌ها اجازه صفحه بندی را به گرید

می‌دهند.حال برای اینکه متد نیز برای صفحه بندی آماده شود باید سه آرگومان به آن اضافه کنیم:(نام پارامترها باید دقیقا موارد زیر باشد)

1- startRowIndex: نقطه شروع صفحه بندی را مشخص می‌کند.

2- maximumRows: تعداد سطرهایی که گرید باید نمایش دهد را مشخص می‌کند.

3- totalRowCount: این پارامتر باید در تابع مقدار دهی شود (مانند مثال) تا مشخص شود نتیجه Query چند رکورد است و در نهایت گرید تعداد صفحات را بر این اساس نمایش می‌دهد.

و برای اینکه صفحه بندی را در Query هم لحاظ کنیم از دو تا بع Skip و Take استفاده شده است.

```
public IQueryable<City> GetAllCities(int startRowIndex, int maximumRows, out int totalRowCount)
 {
        var context = new EFContext();
        var q = from c in context.City
                    select c;

        totalRowCount = q.Count();

        return q.OrderBy(x=>x.Name).Skip(startRowIndex).Take(maximumRows);
 }
```

نکته مهم در این متد IQueryable بودن آن است که باعث واکشی داده‌ها بصورت صفحه به صفحه میشود.

دستورات SQL تولید شده در پروفایلر:

همانطور که مشاهده می‌کنید دو دستور SQL تولید شده ، یکی برای بازگرداندن تعداد رکوردها و یکی هم برای واکشی داده‌ها به

اندازه تعداد رکوردهای مجاز در هر صفحه.

نظرات خوانندگان

نویسنده: saeid
تاریخ: ۱۳۹۱/۰۶/۳۰ ۱:۲۲

با سلام خدمت دوست عزیزم و تشکر از زحمت شما. سوالم این بود که واقعا این مورد در ورژن‌های قبلی امکان پذیر نبوده؟!
http://books.google.com/books?id=wAsBOIZFXV4C&pg=PA338&lpg=PA338&dq=SelectMethod+gridview+asp.net+startRowIndex+maximumRows+totalRowCount&source=bl&ots=sA1GqNuQr9&sig=vzOaA6EQ6h1w3iOTQ45dppyMwtw&hl=en&sa=X&ei=yztaUKTFJYSJhQf7o4CoDQ&ved=0CFAQ6AEwBQ#v=onepage&q=SelectMethod%20gridview%20asp.net%20startRowIndex%20maximumRows%20totalRowCount&f=false

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۶/۳۰ ۸:۶

نه به صورت Strongly Type Binding که توضیح داده شد.

نویسنده: saeid
تاریخ: ۱۳۹۱/۰۶/۳۱ ۱۳:۲۴

بله درسته. ممنون از لینکی که به اشتراک گذاشتید. راستش مطلب رو بطور کامل نخونده بودم.

عنوان: **قابلیت های کاربردی ASP.NET WebFroms - قسمت اول**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۷/۱۶ ۲۰:۴۱
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, ASP.Net 4.5

قابلیت CompositeScript پس از گذشت مدتی که از توسعه پروژه مورد نظرتان می‌گذرد احتمالاً فایل‌های javascript زیادی در پروژه شما استفاده می‌شود که هم مدیریت و هم بار سنگینی بر سرعت بارگذاری اولیه سایت شما به دلیل زمان بارگذاری فایل‌های javascript خواهد گذاشت. در ASP.NET چندین روش برای مدیریت فایل‌های javascript , css وجود دارد.
ساده‌ترین روش استفاده از امکانات خود ASP.Net است. با قابلیت CompositeScript براحتی می‌توانید فایل‌های javascript خود را با هم ادغام و در یک فایل ScriptResource.axd برای Client ارسال نمایید. برای این کار کافیست از تگ جدید CompositeScript در کنترل ScriptManager استفاده نمایید. در کد زیر این قابلیت نمایش داده شده است:

```
<asp:ScriptManager ID="ScriptManager" runat="server" ScriptMode="Release">
                <CompositeScript>
                    <Scripts>
                        <asp:ScriptReference Path="~/Scripts/jquery-1.7.2.min.js" />
                        <asp:ScriptReference Path="~/Scripts/jquery.ui.core.min.js" />
                        <asp:ScriptReference Path="~/Scripts/jquery.ui.widget.min.js" />
                        <asp:ScriptReference Path="~/Scripts/jquery.ui.mouse.min.js" />
                        <asp:ScriptReference Path="~/Scripts/jquery.ui.sortable.min.js" />
                    </Scripts>
                </CompositeScript>
 </asp:ScriptManager>
```

علاوه بر آن حتما خاصیت ScriptMode کنترل ScriptManager را بر روی Release تنظیم نمایید تا از حداکثر کارایی و کش فایل‌ها استفاده نمایید

دسترسی به ScriptManager در صفحات دیگر و الصاق فایل‌های خاص

برخی از فایل‌های Javascript فقط در صفحات خاصی استفاده شده اند و لازم نیست در هر صفحه بارگذاری شود برای این کار کافی است فایل مورد نظر را در صفحه خاص به ScriptManager اضافه نمایید. البته ابتدا لازم است به ScriptManager دسترسی داشته باشیم. کد زیر نحوه دسترسی به آن را نمایش داده است:

```
ScriptManager scriptManager = ScriptManager.GetCurrent(this.Page);
scriptManager.CompositeScript.Scripts.Add(new ScriptReference("~/Scripts/jquery.json.min.js"));
```

ادامه دارد...

نظرات خوانندگان

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۱/۰۷/۱۷ ۸:۴۸

با سلام و تشکر از مطلب خوبتان
در مورد فایلهای css هم روشی وجود دارد؟

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۰۷/۱۷ ۹:۳۳

با استفاده از ScriptManager خیر! اما در مقاله‌های بعدی با روش دیگری توضیح خواهم داد.

نویسنده: میثم فغفوری
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۳:۱۵

ممنون عالی بود فقط یک سوال:
من با افزونه YSlow که بررسی میکنم ظاهرا فایل ایجاد شده رو کش میکنه به مدت 1 سال حالا اگر تو این مدت تغییری تو این اسکریپت‌ها داده باشیم آپدیت میکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۳:۴۸

اگر به انتهای لینک آن دقت کنید، یک کوئری استرینگ اضافه شده. مقدار آن در حقیقت هش فایل‌های مورد استفاده است. با تغییری در این فایل‌ها، این هش هم تغییر خواهد کرد. مرورگر با مشاهده‌ی آدرس جدید، درخواست جدیدی را به سرور برای دریافت فایل‌های به روز شده ارسال می‌کند.

عنوان: **نحوه اضافه کردن Auto-Complete به جستجوی لوسین در ASP.NET MVC و Web forms**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۰۶ ۹:۵۸
آدرس: www.dotnettips.info
گروه‌ها: Lucene.NET, ASP.Net, MVC

پیشنیازها:

چگونه با استفاده از لوسین مطالب را ایندکس کنیم؟
چگونه از افزونه jQuery Auto-Complete استفاده کنیم؟
نحوه استفاده صحیح از لوسین در ASP.NET

اگر به جستجوی سایت دقت کرده باشید، قابلیت ارائه پیشنهاداتی به کاربر توسط یک Auto-Complete به آن اضافه شده‌است. در مطلب جاری به بررسی این مورد به همراه دو مثال Web forms و MVC پرداخته خواهد شد.

قسمت عمده مطلب جاری با پیشنیازهای یاد شده فوق یکی است. در اینجا فقط به ذکر تفاوت‌ها بسنده خواهد شد.

الف) دریافت لوسین

از طریق NuGet آخرین نگارش را دریافت و به پروژه خود اضافه کنید. همچنین Lucene.NET Contrib را نیز به همین نحو دریافت نمائید.

ب) ایجاد ایندکس

کدهای این قسمت با مطلب برجسته سازی قسمت‌های جستجو شده، یکی است:

```
using System.Collections.Generic;
using System.IO;
using Lucene.Net.Analysis.Standard;
using Lucene.Net.Documents;
using Lucene.Net.Index;
using Lucene.Net.Store;
using LuceneSearch.Core.Model;
using LuceneSearch.Core.Utils;

namespace LuceneSearch.Core
{
    public static class CreateIndex
    {
        static readonly Lucene.Net.Util.Version _version = Lucene.Net.Util.Version.LUCENE_30;

        public static Document MapPostToDocument(Post post)
        {
            var postDocument = new Document();
            postDocument.Add(new Field("Id", post.Id.ToString(), Field.Store.YES,
Field.Index.NOT_ANALYZED));
            var titleField = new Field("Title", post.Title, Field.Store.YES, Field.Index.ANALYZED,
Field.TermVector.WITH_POSITIONS_OFFSETS);
```

```
            titleField.Boost = 3;
            postDocument.Add(titleField);
            postDocument.Add(new Field("Body", post.Body.RemoveHtmlTags(), Field.Store.YES,
Field.Index.ANALYZED, Field.TermVector.WITH_POSITIONS_OFFSETS));
            return postDocument;
        }

        public static void CreateFullTextIndex(IEnumerable<Post> dataList, string path)
        {
            var directory = FSDirectory.Open(new DirectoryInfo(path));
            var analyzer = new StandardAnalyzer(_version);
            using (var writer = new IndexWriter(directory, analyzer, create: true, mfl:
IndexWriter.MaxFieldLength.UNLIMITED))
            {
                foreach (var post in dataList)
                {
                    writer.AddDocument(MapPostToDocument(post));
                }

                writer.Optimize();
                writer.Commit();
                writer.Close();
                directory.Close();
            }
        }
    }
}
```

تنها تفاوت آن اضافه شدن titleField.Boost = 3 می‌باشد. توسط Boost به لوسین خواهیم گفت که اهمیت عبارات ذکر شده در عناوین مطالب، بیشتر است از اهمیت متون آن‌ها.

ج) تهیه قسمت منبع داده Auto-Complete

```
namespace LuceneSearch.Core.Model
{
    public class SearchResult
    {
        public int Id { set; get; }
        public string Title { set; get; }
    }
}
```

```
using System.Collections.Generic;
using System.IO;
using Lucene.Net.Index;
using Lucene.Net.Search;
using Lucene.Net.Store;
using LuceneSearch.Core.Model;
using LuceneSearch.Core.Utils;

namespace LuceneSearch.Core
{
    public static class AutoComplete
    {
        private static IndexSearcher _searcher;

        /// <summary>
        /// Get terms starting with the given prefix
        /// </summary>
        /// <param name="prefix"></param>
        /// <param name="maxItems"></param>
        /// <returns></returns>
        public static IList<SearchResult> GetTermsScored(string indexPath, string prefix, int maxItems
= 10)
        {
            if (_searcher == null)
                _searcher = new IndexSearcher(FSDirectory.Open(new DirectoryInfo(indexPath)), true);

            var resultsList = new List<SearchResult>();
            if (string.IsNullOrWhiteSpace(prefix))
                return resultsList;
```

```
            prefix = prefix.ApplyCorrectYeKe();

            var results = _searcher.Search(new PrefixQuery(new Term("Title", prefix)), null, maxItems);
            if (results.TotalHits == 0)
            {
                results = _searcher.Search(new PrefixQuery(new Term("Body", prefix)), null, maxItems);
            }

            foreach (var doc in results.ScoreDocs)
            {
                resultsList.Add(new SearchResult
                {
                    Title = _searcher.Doc(doc.Doc).Get("Title"),
                    Id = int.Parse(_searcher.Doc(doc.Doc).Get("Id"))
                });
            }

            return resultsList;
        }
    }
}
```

توضیحات:

برای نمایش Auto-Complete نیاز به منبع داده داریم که نحوه ایجاد آن‌را در کدهای فوق ملاحظه می‌کنید. در اینجا توسط جستجوی سریع لوسین و امکانات PrefixQuery آن، به تعدادی مشخص (maxItems)، رکوردهای یافت شده را بازگشت خواهیم داد. خروجی حاصل لیستی است از SearchResultها شامل عنوان مطلب و Id آن. عنوان را به کاربر نمایش خواهیم داد؛ از Id برای هدایت او به مطلبی مشخص استفاده خواهیم کرد.

د) نمایش Auto-Complete در ASP.NET MVC

```
using System.Text;
using System.Web.Mvc;
using LuceneSearch.Core;
using System.Web;

namespace LuceneSearch.Controllers
{
    public class HomeController : Controller
    {
        static string _indexPath = HttpRuntime.AppDomainAppPath + @"App_Data\idx";

        public ActionResult Index(int? id)
        {
            if (id.HasValue)
            {
                //todo: do something
            }
            return View(); //Show the page
        }

        public virtual ActionResult ScoredTerms(string q)
        {
            if (string.IsNullOrWhiteSpace(q))
                return Content(string.Empty);

            var result = new StringBuilder();
            var items = AutoComplete.GetTermsScored(_indexPath, q);
            foreach (var item in items)
            {
                var postUrl = this.Url.Action(actionName: "Index", controllerName: "Home", routeValues:
new { id = item.Id }, protocol: "http");
                result.AppendLine(item.Title + "|" + postUrl);
            }

            return Content(result.ToString());
        }
    }
}
```

```
@{
```

```
    ViewBag.Title = "جستجو";
    var scoredTermsUrl = Url.Action(actionName: "ScoredTerms", controllerName: "Home");
    var bulletImage = Url.Content("~/Content/Images/bullet_shape.png");
}
<h2>
    جستجو</h2>

<div align="center">
    @Html.TextBox("term", "", htmlAttributes: new { dir = "ltr" })
    <br />
    را وارد نمائید lu جهت آزمایش
</div>

@section scripts
{
    <script type="text/javascript">
        EnableSearchAutocomplete('@scoredTermsUrl', '@bulletImage');
    </script>
}
```

```
function EnableSearchAutocomplete(url, img) {
    var formatItem = function (row) {
        if (!row) return "";
        return "<img src='" + img + "' /> " + row[0];
    }

    $(document).ready(function () {
        $("#term").autocomplete(url, {
            dir: 'rtl', minChars: 2, delay: 5,
            mustMatch: false, max: 20, autoFill: false,
            matchContains: false, scroll: false, width: 300,
            formatItem: formatItem
        }).result(function (evt, row, formatted) {
            if (!row) return;
            window.location = row[1];
        });
    });
}
```

توضیحات:

- ابتدا ارجاعاتی را به jQuery، افزونه Auto-Complete و اسکریپت سفارشی تهیه شده، در فایل layout پروژه تعریف خواهیم کرد. در اینجا سه قسمت را مشاهده می‌کنید: کدهای کنترلر، View متناظر و اسکریپتی که Auto-Complete را فعال خواهد ساخت.
- قسمت مهم کدهای کنترلر، دو سطر زیر هستند:

```
result.AppendLine(item.Title + "|" + postUrl);
return Content(result.ToString());
```

مطابق نیاز افزونه انتخاب شده در مثال جاری، فرمت خروجی مدنظر باید شامل سطرهایی حاوی متن قابل نمایش به همراه یک Id (یا در اینجا یک آدرس مشخص) باشد. البته ذکر این Id اختیاری بوده و در اینجا جهت تکمیل بحث ارائه شده است.
return Content هم سبب بازگشت این اطلاعات به افزونه خواهد شد.
- کدهای View متناظر بسیار ساده هستند. تنها نام TextBox تعریف شده مهم می‌باشد که در متد جاوا اسکریپتی EnableSearchAutocomplete استفاده شده است. به علاوه، نحوه مقدار دهی آدرس دسترسی به اکشن متد ScoredTerms نیز مهم می‌باشد.
- در متد EnableSearchAutocomplete نحوه فراخوانی افزونه autocomplete را ملاحظه می‌کنید.
جهت آن، به راست به چپ تنظیم شده است. با 2 کاراکتر ورودی فعال خواهد شد با وقفه‌ای کوتاه. نیازی نیست تا انتخاب کاربر از لیست ظاهر شده حتما با عبارت جستجو شده صد در صد یکی باشد. حداکثر 20 آیتم در لیست ظاهر خواهند شد. اسکرول بار لیست را حذف کرده‌ایم. عرض آن به 300 تنظیم شده است و نحوه فرمت دهی نمایشی آن‌را نیز ملاحظه می‌کنید. برای این منظور از متد formatItem استفاده شده است. آرایه row در اینجا در برگیرنده اعضای Title و Id ارسالی به افزونه است. اندیس صفر آن به عنوان دریافتی اشاره می‌کند.
همچنین نحوه نشان دادن عکس العمل به عنصر انتخابی را هم ملاحظه می‌کنید (در متد result مقدار دهی شده).
window.location را به عنصر دوم آرایه row هدایت خواهیم کرد. این عنصر دوم مطابق کدهای اکشن متد تهیه شده، به آدرس یک صفحه اشاره می‌کند.

۵) نمایش Auto-Complete در ASP.NET WebForms

قسمت عمده مطالب فوق با وب فرم‌ها نیز یکی است. خصوصا توضیحات مرتبط با متد EnableSearchAutocomplete ذکر شده.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="LuceneSearch.WebForms.Default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <meta charset="utf-8" />
    <meta name="viewport" content="width=device-width" />
    <title>جستجو</title>
    <link href="Content/Site.css" rel="stylesheet" type="text/css" />
    <script src="Scripts/jquery-1.7.1.min.js" type="text/javascript"></script>
    <script src="Scripts/jquery.autocomplete.js" type="text/javascript"></script>
    <script src="Scripts/custom.js" type="text/javascript"></script>
</head>
<body dir="rtl">
    <h2>
        جستجو</h2>
    <form id="form1" runat="server">
    <div align="center">
        <asp:TextBox runat="server" dir="ltr" ID="term"></asp:TextBox>
        <br />
        را وارد نمائید lu جهت آزمایش
    </div>
    </form>
    <script type="text/javascript">
        EnableSearchAutocomplete('Search.ashx', 'Content/Images/bullet_shape.png');
    </script>
</body>
</html>
```

```
using System.Text;
using System.Web;
using LuceneSearch.Core;

namespace LuceneSearch.WebForms
{
    public class Search : IHttpHandler
    {
        static string _indexPath = HttpRuntime.AppDomainAppPath + @"App_Data\idx";

        public void ProcessRequest(HttpContext context)
        {
            string q = context.Request.QueryString["q"];
            if (string.IsNullOrWhiteSpace(q))
            {
                context.Response.Write(string.Empty);
                context.Response.End();
            }

            var result = new StringBuilder();
            var items = AutoComplete.GetTermsScored(_indexPath, q);
            foreach (var item in items)
            {
                var postUrl = "Default.aspx?id=" + item.Id;
                result.AppendLine(item.Title + "|" + postUrl);
            }

            context.Response.ContentType = "text/plain";
            context.Response.Write(result.ToString());
            context.Response.End();
        }

        public bool IsReusable
        { get { return false; } }
    }
}
```

در اینجا بجای Controller از یک Generic handler استفاده شده است (Search.ashx).

```
result.AppendLine(item.Title + "|" + postUrl);
context.Response.Write(result.ToString());
```

در آن، عنوان مطالب یافت شده به همراه یک آدرس مشخص، تهیه و در Response نوشته خواهند شد.

کدهای کامل مثال فوق را از اینجا می‌توانید دریافت کنید:

LuceneSearch.zip

همچنین باید دقت داشت که پروژه MVC آن از نوع MVC4 است (VS2010) و فرض براین می‌باشد که IIS Express 7.5 را نیز پیشتر نصب کرده‌اید.

کلمه عبور فایل: dotnettips91

نظرات خوانندگان

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۰/۰۷ ۱۲:۵۹

ممنون از این مطلب مفید. آیا با استفاده روش فوق میشه چند جدول رو با هم ایندکس کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۷ ۱۳:۳

بله. در اینجا مشکلی با ثبت اطلاعات از چندین جدول مختلف، در یک ایندکس وجود ندارد. برای مدیریت جستجوی بهتر روی آن‌ها یک روش این است که در متد MapPostToDocument ، فیلد دیگری را به نام مثلا TableName اضافه کنید تا بشود در حین جستجو از آن استفاده کرد.

نویسنده: محمد حبیبی
تاریخ: ۱۳۹۱/۱۰/۲۵ ۱۴:۴

من یک تکست باکس ایجاد کردم که از همین Autocomplete شما استفاده میکنه. منتها یک دکمه جستجو هم قرار دادم که با کلیک کردن روی اون دوباره از لوسین استفاده میکنه و چندین رکورد رو هم بازگشت میده.
اما چون باید یک سری رشته که نتیجه‌ی لوسین هست رو برگردونم از return Content استفاده میکنم ولی نتایج رو در یک صفحه‌ی جدید و بدون Master Page میاره.
نحوه‌ی نمایش نتایج اطلاعات لوسین در یک View که حاوی یک Master Page هم هست چطوریه؟
مثلا نتایج زیر همون تکست باکس نمایش داده بشه.
آیا باید نوع اکشن چیز دیگه ای باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۲۵ ۱۴:۲۱

نتیجه جستجوی لوسین مثلا در مطلب فوق لیستی از SearchResultها است. همین لیست را در اکشن متد بازگشت دهید. سپس در View متناظر یک حلقه درست کرده و حاصل را به صورت دلخواهی فرمت کنید.
علت استفاده از return Content در مثال بالا، نیاز افزونه جی‌کوئری استفاده شده به خروجی ساده متنی است. در حالت‌های دیگر از return View معمولی استفاده کنید.

نویسنده: مرتضی
تاریخ: ۱۳۹۱/۱۱/۰۵ ۱۲:۴۵

سلام آقای نصیری ممنون از اینکه اطلاعتتون رو در اختیار دیگران قرار میدید بنده از کدهای بالا تویه سایتم استفاده کردم و جستجوی سایت مثل ساعت داره کار میکنه فقط تویه ظاهرش به مشکل برخوردم بنده میخوام اون box رو که اطلاعات توش نمایش داده میشه از سمت راست تکس باکس تراز بشه مثل همون عکسی که بالا گذاشتید از بنده از سمت چپ تراز شده و یه سوال دیگه, اون عکس loading رو هنگام جستجو نمایش نمیده آدرس دهی اون هم درسته البته تکس باس بنده position:absolute هستش نمیدونم ماله اینه که نمایش نمیده یا مثلا میره یه قسمت دیدگی از صفحه. ممنون میشم راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۰۵ ۱۳:۱

این موارد را باید با اصلاح اسکریپت یا css مربوط به auto-complete مدیریت کنید.
مثلا محل قرارگیری منوی بازشونده به صورت زیر مقدار دهی شده. این را در فایل jquery.autocomplete.js یافته و اصلاح کنید:

```
left: offset.left - options.width + 125
```

یا تصویر loading در css به نحو زیر تعیین شده:

```
.ac_loading
{
background: white url('Images/indicator.gif') left center no-repeat;
}
```

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۱/۰۹ ۱۷:۴۲

سلام؛ من یه مشکلی دارم وقتی بعد از جستجو گزینه ای رو انتخاب می‌کنی صفحه refresh می‌شه و متن انتخاب شده پاک می‌شه میشه لطفا منو راهنمایی کنید مرسی

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۰۹ ۱۸:۶

بله. به همین نحو طراحی شده. زمانیکه یک گزینه انتخاب می‌شود، سطر زیر، کاربر را به صفحه متناظر هدایت می‌کند:

```
window.location = row[1];
```

این سطر در سمت کلاینت، مساوی Response.Redirect سمت سرور است. می‌تونید اینجا متن انتخابی رو به صورت مثلا یک کوئری استرینگ تعریف کنید و بعد در صفحه‌ای دیگر دریافت و نمایش بدید.

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۱۱/۱۷ ۱:۵

بسیار عالی بود . تنها یک مشکل برای من در ارتباط با کدهای آموزشی که برای استفاده از لوسین در اینترنت پیدا کردم وجود داره و اون هم اینکه در اونها کلاس و توابع به صورت استاتیک تعریف و در کنترلرها و یا کلاس‌های لایه سرویس فراخوانی شدن . این مسئله باعث اشکال در نوشتن آزمون واحد برای کنترلرها و یا متدها نمیشه ؟

آیا در استاتیک معرفی کردن کلاس و توابع علت و مزیت خاصی وجود داره ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۱۷ ۸:۵۵

- اگر از لایه سرویس استفاده می‌کنید که نهایتا با یک سری اینترفیس در کنترلرها کار خواهید کرد.
- بله. توضیح دادم در مطلب « نحوه استفاده صحیح از لوسین در ASP.NET ».

نویسنده: میهمان
تاریخ: ۱۳۹۱/۱۱/۲۱ ۲۰:۱۵

با سلام
امکانش هست که یه توضیح هم در مورد نحوه کار مستقیم با دیتابیس بدین؟
کدام قسمت‌ها باید تغییر کند و کدام قسمت‌ها باید حذف و اضافه شوند
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۱ ۲۰:۳۰

لطفا اولین پیشنیاز عنوان شده را مطالعه کنید.

نویسنده: میهمان
تاریخ: ۱۳۹۱/۱۱/۲۲ ۰:۱۷

با سلام مجدد؛ تمامی پیشنیازهای این مقاله را مطالعه کردم و خط به خط کدهای پروژه ای را که زحمت کشیدید و تهیه کرده اید را بررسی کردم. شما در اولین پیشنیاز عنوان شده فرموده بودید " کتابخانه Lucene مستقل است از منبع داده مورد استفاده و تنها اطلاعاتی با فرمت شیء Document معرفی شده به آن‌را می‌شناسد. " و در پروژه، در ابتدا کل داده‌ها را به یک document تبدیل کردید و سپس آنها را به IndexWriter اضافه کردید. و سپس جستجوهای خود را روی این دیکشنری انجام می‌دادید.
حالا سوال من این است: من در ابتدا باید تمامی اطلاعات را از طریق عنوان شده به یک فایل دیکشنری تبدیل کنم و هر مطلب جدید که اضافه میشود ، به این دیکشنری نیز اضافه شود ؟ اگر تعداد رکرودها بالای میلیون باشد؟ راهی وجود ندارد که ما مستقیم به خود دیتابیس کار کنیم ؟ و یا نه من ابتدا باید روی دیتابیس کلمه مورد نظر را جستجو کتم و پس از یافتن آن، آنرا به دیکشنری اضافه کنم ؟ بهترین راه چیست که برای داده‌های بالا جوابگو باشد
با تشکر فراوان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۱/۲۲ ۰:۵۱

- لوسین مستقل است از بانک اطلاعاتی. همچنین یکبار باید این ایندکس را تهیه کنید. اگر تعداد رکوردهای شما بالا است، فقط همان بار اول است که کار تهیه زمانبر خواهد بود. برای دفعات بعد در حد اضافه کردن چند سند لوسین به آن یا به روز رسانی و حذف است و کار دیگری ندارد.
- پس از تهیه ایندکس، جستجوی لوسین کاری به بانک اطلاعاتی شما ندارد. بر روی ایندکس خودش انجام می‌شود و نیازی به جستجوی مجدد در بانک اطلاعاتی شما نیست. یک سیستم مستقل است.
این روش متداول کار با لوسین است و حالت دیگری هم ندارد. این مستقل بودن هم یک مزیت است. برای مثال SQL Server CE یا خیلی از بانک‌های اطلاعاتی دیگر Full Text Search توکار ندارند. اینجا لوسین خوب جواب می‌ده.
ضمن اینکه من در یک دمو استفاده از لوسین برای ایندکس کردن کل اطلاعات ویکی‌پدیا رو دیدم. تهیه ایندکس آن یک روز کار برده بوده (با توجه به حجم اطلاعات بالای ویکی پدیا)، اما جستجوی آن فوق العاده سریع و با کیفیت بود. این ویدیو رو در اینجا می‌تونید مشاهده کنید:
Full-text search with Lucene and neat things you can do with it

نویسنده: morteza
تاریخ: ۱۳۹۱/۱۲/۰۱ ۲۲:۳۰

سلام آقای نصیری میخواستم ببینم آیا شما اطلاعی از نحوه غیر فعال کردم اینتر تویه این پلاگین دارید بنده میخوام وقتی کاربر اینتر رو فشار داد جستجوی معمولی انجام بشه و اگه خواست از مواردی که توسط autocomplate براش آمده توسط موس یکی رو انتخاب کنه و به صفحه مربوطه هدایت بشه ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۰۱ ۲۲:۴۹

کلید enter با case KEY.RETURN در این افزونه جی‌کوئری مدیریت می‌شود.

نویسنده: علیرضا پایدار
تاریخ: ۱۳۹۱/۱۲/۲۰ ۱۰:۵۵

یک سوال خدمت شما داشتم اینکه :

وقتی بخوایم از 2 جدول متفاوت جستجو کنیم به نظر شما اطلاعات هر جدول را جداگانه ایندکس کنیم(منظورم همان کاری که متد CreateFullTextIndex شما انجام میده) یا خیر؟
البته منظور دو جدولی که با هم رابطه دارند.

چندتا لینک در مورد لوسین:

http://www.thebestcsharpprogrammerintheworld.com/blogs/how-to-create-and-search-a-lucene-net-index-in-4-simple-steps-using-c-sharp-step-1.aspx

http://www.codeproject.com/Articles/272309/Lucene-Search-Programming

http://www.codeproject.com/Articles/320219/Lucene-Net-ultra-fast-search-for-MVC-or-WebForms

ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۰ ۱۱:۳۲

من همه رو داخل یک ایندکس ثبت می‌کنم. فقط یک فیلد اضافه‌تر به نام «نام جدول» مورد نظر نیاز هست تا بشود روی آن کوئری خاص گرفت یا اینکه کلا روی تمام رکوردها جستجو کرد به یکباره.

نویسنده: مهمان
تاریخ: ۱۳۹۲/۰۳/۱۹ ۷:۵۵

ضمن خسته نباشید
اگر نیاز داشته باشیم که بعد از انتخاب گزینه کاربر به صفحه دیگه هدایت نشه و فقط آیتم انتخابی در تکست باکس بمونه باید چه کاری انجام داد. مرسی

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۹ ۱۷:۱۳

سطر window.location رو حذف کنید.

نویسنده: imo0
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۰:۵۵

سلام؛ تویه اولین پیشنیازها که کار با لوسین رو گفته بودین توابعی مثله SearchByPartialWords و Query و غیره ایجاد شده ولی تو اینجا یه جور دیگه جستجو کردین توی ایندکس‌ها . میخواستم بدونم او توابع رو مگه واسه سرچ تو ایندکس‌ها ننوشتین؟ تازه اینجا شما دارین متن Title رو میفرستین سمت کاربر . این highlight کردن عبارت جستجو شدش پس چی شد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۱:۳۵

- این هم یک روش دیگر هست. از آن روش‌های ذکر شده در پیشنیازها هم می‌توانید استفاده کنید.
- highlight کردن عبارت جستجو شد ه در عنوان بازگشت داده شده، توسط افزونه auto-complete انجام می‌شود؛ خودکار است.

نویسنده: imo0
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۱:۴۶

تو اون تابع Query که نوشتین یه دونه FastVectorHighlighter ایجاد کردین . اگه highlight توسط auto-complete خودکار انجام میشه پس نقش این چیه؟ میشه بگین کدوم روش بهتره ؟ یا فرقی نمیکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۲۷ ۱۱:۵۲

دو بحث وجود داره:
- برجسته سازی قسمتی از عبارت جستجو شده در لیست نمایش داده شده توسط افزونه auto-complete. این مورد خودکار است و توسط افزونه انجام می‌شود.
- برجسته سازی قسمتی از عبارت جستجو شده در نتایج یک جستجوی کامل بعدی که قرار است highlight آن توسط ما با کدنویسی خاصی انجام شود. مراجعه کنید به این مطلب برای توضیحات بیشتر.

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۲/۲۲ ۱۵:۵۶

با سلام
من از لوسین در asp.net web form استفاده کردم و کاملا هم راضی هستم . فقط یه مشکلی داشتم. اگه بخوام غیر از عبارت مورد نظر برای جستجو یه مقدار دیگه رو هم بفرستم باید چکار کنم؟
مثلا یه Dropdown داشته باشیم وبخواییم مثلا مقدار categoryid رو هم پاس بدیم تا به query مربوطه اضافه بشه.من dropdown رو اضافه کردم اما مقدارش رو نمیشه در هندلر مربوط به Search دریافت کرد مثل q.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۲ ۱۷:۲۶

افزونه‌ی مورد استفاده در آن به نام jquery.autocomplete.js سورسش پیوست هست. ajax را در آن جستجو کنید به مورد ذیل خواهید رسید:

```
$.ajax({
    //...
    data: $.extend({
              q: lastWord(term),
              limit: options.max
              }, extraParams),
       //...
});
```

در اینجا q مشخص است و extraParams آن متغیر. بنابراین برای ارسال اطلاعات اضافی باید extraParams را مانند کدهای ذیل مقدار دهی کرد. نحوه‌ی دریافت عناصر آن در سمت سرور، مانند نحوه‌ی دریافت q است.

```
$(document).ready(function () {
        $("#term").autocomplete(url, {
            // .....
            extraParams: { id: 12, xyz: "test" }
        }).result(function (evt, row, formatted) {
            // .....
        });
    });
```

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۸:۱۱

اگر بخوام لیستی که برای نتیجه جستجو باز میشه رو Custom کنم مثلاً یه عکس بهش اضافه کنم یا یه جدول ساخته شده با HTML رو توش نشون بدم با یه سری جزئیات ، آیا امکان پذیره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۷ ۱۹:۳۰

نحوه سفارشی سازی هر آیتم لیست را در مطلب فوق، در متد formatItem می‌توانید مشاهده کنید.

نویسنده: علیرضا
تاریخ: ۱۳۹۳/۰۴/۰۳ ۱۰:۵۸

با سلام؛ تکست باکسی که می‌خام جستجو از طریق اون انجام بشه در یک مستر پیج قرار داره و زمانی که صفحه و مستر پیج در یک پوشه قرار دارند مشکلی نیست ولی اگر صفحه در یک پوشه دیگر قرار گیرد جواب نمی‌ده. توضیح اینکه از متد ResolveClientUrl هم استفاده کردم ولی جواب نداد :

```
EnableSearchAutocomplete('<%=ResolveClientUrl( "~/Handlers/find.ashx")%>',
'<%=ResolveClientUrl("Images/bullet_shape.png")%>');
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۰۳ ۱۱:۱۴

- نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery
- بررسی کنید آیا ID جعبه متنی در آن صفحه، term است یا چیز دیگری ؟ اگر تغییر کرده، ClientIDMode=Static بهتر است تنظیم شود.

نویسنده: علیرضا
تاریخ: ۱۳۹۳/۰۴/۰۳ ۱۸:۳۱

با تشکر از پاسخ شما ، id درست بود و با فایرباگ هم بررسی کردم و خطای زیر رو می‌داد :

```
TypeError: $(...).autocomplete is not a function
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۰۳ ۱۸:۵۴

این خطا یعنی اسکریپت‌های اصلی به صفحه پیوست نشده‌اند. برای تعریف آن‌ها هم از ResolveClientUrl استفاده کنید:

```
<script type="text/javascript" src='<%: ResolveClientUrl("~/path....") %>'></script>
```

نویسنده: محمّد زارع
تاریخ: ۱۳۹۳/۱۰/۱۲ ۱۰:۲۸

سلام. من برای ثبت چندتا جدول مختلف در یک ایندکس فیلد TableId هم گرفتم برای اینکه مشخص بشه سند مربوط به کدوم جدوله. حالا راهی هست که به جز Id فیلد دیگه ای هم برای بروزرسانی یا حذف یک سند ارسال کرد؟ مثلا با Id و TableId یه سند رو حذف یا بروزرسانی کرد نه با Id خالی.
یعنی با این دستور:

```
indexWriter.DeleteDocuments(new Term("Id", id.ToString()))
```

مثلاً یه محصول داشته باشیم و یه مقاله با آیدی یکسان چجوری میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۲ ۱۰:۴۱

متد DeleteDocuments یک کوئری هم قبول می‌کند:

```
var query = new BooleanQuery();
query.Add(new BooleanClause(new TermQuery(new Term("id", id.ToString(CultureInfo.InvariantCulture))),
Occur.SHOULD));
// query.Add(...
// query.Add(...
// ...
writer.DeleteDocuments(query);
```

عنوان: Best Practice هایی برای ASP.NET MVC - قسمت اول
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۸/۱۲ ۱۸:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, Tips

در سایت جاری مطالب زیادی درباره ASP.NET MVC نوشته شده است. این مطلب و قسمت بعدی آن مروری خواهد داشت بر Best Practiceها در ASP.NET MVC.

استفاده از NuGet Package Manager برای مدیریت وابستگی‌ها

درباره اهمیت NuGet برای مصرف کنندگان قبلا این مطلب نوشته شده است.
بجای صرف وقت برای اینکه بررسی کنیم آیا این نسخه‌ی جدید کتابخانه‌ی X یا اسکریپت jQuery آمده است یا خیر، می‌توان این وظیفه را به NuGet سپرد. علاوه بر این NuGet مزیت‌های دیگری هم دارد؛ مثلا تیم‌های برنامه نویسی می‌توانند کتاب خانه‌های مشترک خودشان را در مخزن‌های سفارشی NuGet قرار دهند و توزیع و Versioning آن‌را به NuGet بسپارند.

تکیه بر Abstraction (انتزاع)

Abstraction در طراحی سیستم‌ها منجر به تولید نرم افزار هایی Loosely coupled با قابلیت نگهداری بالا و همچنین فراهم شدن زمینه برای نوشتن Unit Test می‌شود.
اگر به مطالب قبلی وب سایت برگردیم در مطلب چرا ASP.NET MVC گفته شد که :
2) دستیابی به کنترل بیشتر بر روی اجزای فریم ورک :
در طراحی ASP.NET MVC همه‌جا interfaceها قابل مشاهد هستند. همین مساله به معنای افزونه پذیری اکثر قطعات تشکیل دهنده ASP.NET MVC است؛ برخلاف ASP.NET web forms. برای مثال تابحال چندین routing engine، view engine و غیره توسط برنامه نویس‌های مستقل برای ASP.NET MVC طراحی شده‌اند که هیچکدام با ASP.NET web forms میسر نیست. برای مثال از view engine پیش فرض آن خوشتان نمی‌آید؟ عوضش کنید! سیستم اعتبار سنجی توکار آن‌را دوست ندارید؟ آن‌را با یک نمونه بهتر تعویض کنید و الی آخر ...
به علاوه طراحی بر اساس interfaceها یک مزیت دیگر را هم به همراه دارد و آن هم ساده سازی mocking (تقلید) آن‌ها است جهت ساده سازی نوشتن آزمون‌های واحد.

از کلمه‌ی کلیدی New استفاده نکنید

هر جا ممکن است کار وهله سازی اشیاء را به لایه و حتی Framework دیگری بسپارید. هر زمان اشیاء نرم افزار خودتان را با کلمه‌ی new وهله سازی می‌کنید اصل Abstraction را فراموش کرده اید. هر زمان اشیاء نرم افزار را مستقیم وهله سازی می‌کنید در نظر داشته باشید می‌توانید آنها را به صورت وابستگی تزریق کنید.
در همین رابطه مطالب زیر را از دست ندهید :
تزریق وابستگی (dependency injection) به زبان ساده
تزریق وابستگی (Dependency Injection) و توسعه پذیری

از HttpContext مستقیما استفاده نکنید (از HttpContextBase استفاده کنید)

از .NET 4 به بعد فضای نامی تعریف شده که در بر دارنده‌ی کلاس‌های انتزاعی (Abstraction) خیلی از قسمت‌های اصلی ASP.NET است. یکی از مواردی که در توسعه‌ی ASP.NET معمولا زیاد استفاده می‌شود، شیء HttpContext است . استفاده از HttpContextBase را به استفاده از HttpContext ترجیح دهید. اهمیت این موضوع در راستای اهمیت انتزاع (Abstraction) می‌باشد.

از "رشته‌های جادویی" اجتناب کنید

استفاده از رشته‌های جادویی در خیلی از جاها کار را ساده می‌کند؛ بعضی وقت‌ها هم به آنها نیاز است اما مشکلات زیادی دارند :

رشته‌ها معنای باطنی ندارند (مثلا : دشوار است که از روی نام یک ID مشخص کنم این ID چطور به ID دیگری مرتبط است و یا اصلا ربط دارد یا خیر)
با اشتباهات املایی یا عدم رعایت حروف بزرگ و کوچک ایجاد مشکل می‌کنند.
به Refactoring واکنش خوبی نشان نمی‌دهند. (برای درک بهتر این مطلب را بخوانید.)
برای درک بهتر 2، یک مثال بررسی می‌شود؛ اولی از رشته‌های جادویی برای دسترسی به داده در ViewData استفاده می‌کند و دومی refactor شده‌ی مثال اول است که از strongly type مدل برای دسترسی به همان داده استفاده می‌کند.

```
<p>
    <label for="FirstName">First Name:</label>
    <span id="FirstName">@ViewData["FirstName"]</span>
</p>
```

مثال دوم :

```
<p>
    <label for="FirstName">First Name:</label>
    <span id="FirstName">@Model.FirstName</span>
</p>
```

مثلا مثال دوم مشکلات رشته‌های جادویی را ندارد.
در رابطه با Magic strings این مطلب را مطالعه بفرمایید.

از نوشتن HTML در کدهای "Backend" خودداری کنید

با توجه به اصل جداسازی وابستگی‌ها (Separation of Concerns) وظیفه‌ی کنترلرها و دیگر کدهای backend رندر کردن HTML نیست. (ساده سازی کنترلر ها) البته در نظر داشته باشید که قطعا تولید HTML در متدهای کمکی کلاس هایی که "تنها" وظیفه‌ی آنها کمک به Viewها جهت تولید کد هست ایرادی ندارد. این کلاس‌ها بخشی از View در نظر گرفته می‌شوند نه کدهای "backend".

در Viewها "Business logic" انجام ندهید

معکوس بند قبلی هم کاملا صدق می‌کند ، منظور این است که Viewها تا جایی که ممکن است باید حاوی کمترین Business logic ممکن باشند. در واقع تمرکز Viewها باید استفاده و نحوه‌ی نمایش داده ای که برای آن‌ها فراهم شده باشد نه انجام عملیات روی آن.

استفاده از Presentation Model را به استفاده مستقیم از Business Objectها ترجیح دهید

در مطالب مختلف وب سایت اشاره به اهمین ViewModelها شده است. برای اطلاعات بیشتر بند ج آموزش 11 از سری آموزش‌های ASP.NET MVC را مطالعه بفرمایید.

Ifهای شرطی را در Viewها را در متدهای کمکی کپسوله کنید

استفاده از شرطها در View کار توسعه دهنده را برای یک سری اعمال ساده می‌کند اما می‌تواند باعث کمی کثیف کاری هم شود. مثلا:

```
@if(Model.IsAnonymousUser) {
    <img src="@Url.Content("~/content/images/anonymous.jpg")" />
} else if(Model.IsAdministrator) {
    <img src="@Url.Content("~/content/images/administrator.jpg")" />
```

```
} else if(Model.Membership == Membership.Standard) {
    <img src="@Url.Content("~/content/images/member.jpg")" />
} else if(Model.Membership == Membership.Preferred) {
    <img src="@Url.Content("~/content/images/preferred_member.jpg")" />
}
```

می‌توان این کد که تا حدودی شامل منطق تجاری هم هست را در یک متد کمکی کپسوله کرد :

```
public static string UserAvatar(this HtmlHelper<User> helper)
{
    var user = helper.ViewData.Model;
    string avatarFilename = "anonymous.jpg";
    if (user.IsAnonymousUser)
    {
        avatarFilename = "anonymous.jpg";
    }
    else if (user.IsAdministrator)
    {
        avatarFilename = "administrator.jpg";
    }
    else if (user.Membership == Membership.Standard)
    {
        avatarFilename = "member.jpg";
    }
    else if (user.Membership == Membership.Preferred)
    {
        avatarFilename = "preferred_member.jpg";
    }
    var urlHelper = new UrlHelper(helper.ViewContext.RequestContext);
    var contentPath = string.Format("~/content/images/{0}", avatarFilename)
    string imageUrl = urlHelper.Content(contentPath);
    return string.Format("<img src='{0}' />", imageUrl);
}
```

اکنون برای نمایش آواتار کاربر به سادگی می‌توان نوشت :

```
@Html.UserAvatar()
```

به این ترتیب کد ما تمیزتر شده ، قابلیت نگهداری آن بالاتر رفته ، منطق تجاری یک بار و در یک قسمت نوشته شده از این کد در جاهای مختلف سایت می‌توان استفاده کرد و اگر لازم به تغییر باشد با تغییر در یک قسمت همه جا اعمال می‌شود.

منتظر قسمت بعدی باشید.

نظرات خوانندگان

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۰۸/۱۲ ۱۹:۴۵

نکات جالبی بود .
یک نکته که به شخصه اون رو تجربه کرده ام (بخصوص در مورد استفاده از best practiceها) وضعیتی است که اصطلاحاً بهش overkilling میگن (اگر اشتباه نکنم) ، یعنی بعضی وقتا دیگه زیاده روی میشه . مثلا در به کاربردن اینترفیس‌ها و یا loose coupling .
برای مثال در مورد تشخیص زمان استفاده از اینترفیس در این کتاب ، یکی از بهترین راهکارها استفاده از آزمون واحد معرفی شده .
بسیار عالی میشه در صورت امکان در این مورد هم نکاتی را ذکر کنید .

نویسنده: افشار محبی
تاریخ: ۱۳۹۱/۰۸/۲۴ ۸:۲۶

خیلی مفید و عملی بود. ممنون.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۰۸/۲۴ ۱۴:۱۹

عالی بود ممنون،
بهتر نیست به جای این if و elseها از Dictionary استفاده کنیم؟

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۱/۰۸/۲۴ ۱۴:۲۰

خواهش می‌کنم.
چرا همینطوره.
کدها همه مثال هستند و هدف این مطلب چیز دیگری بوده.

عنوان: **بستن دسترسی درخواست‌های PROPFIND در ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۱۴ ۲۱:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Security

هر از چندگاهی که به لاگ‌های خطای برنامه مراجعه می‌کنم، درخواست‌هایی (حملاتی) با این مشخصات ثبت شده:

```
REQUEST_METHOD:  OPTIONS
REQUEST_METHOD:  PROPFIND
HTTP_USER_AGENT:  Microsoft-WebDAV-MiniRedir/6.1.7600
SCRIPT_NAME:  /ipc$
```

برای بستن این نوع درخواست‌های ویژه (که عموما برای دسترسی به اطلاعات شیرپوینت و یا سرور بکار می‌روند)، فقط کافی است فایل web.config برنامه را به نحو زیر اصلاح کنیم:

```
<system.web>
    <httpHandlers>
      <add path="*" verb="OPTIONS, PROPFIND" type="System.Web.DefaultHttpHandler" />
    </httpHandlers>
</system.web>

<system.webServer>
    <handlers>
      <add name="propf" path="*" verb="OPTIONS, PROPFIND" type="System.Web.DefaultHttpHandler" />
    </handlers>
</system.webServer>
```

نظرات خوانندگان

نویسنده: الهام
تاریخ: ۱۳۹۱/۰۸/۱۵ ۸:۴۳

سلام آقای نصیری

ببخشید میشه توضیح بدین این اصلاحیه چه عملی انجام میده؟ به نظر میرسه این نوع درخواستها که هنوز هندل میشن

با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۸/۱۵ ۸:۵۱

بله. پیغام "Not Implemented 501" را به درخواست‌های PROPFIND ارائه می‌ده.

عنوان: **کلاس کمکی جهت مشاهده آیتم های موجود در حافظه کش و حذف آنها**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۸/۱۹ ۱۶:۲۰
آدرس: www.dotnettips.info
برچسب‌ها: #ASP.Net, C

مواقع بسیاری پیش می‌آید که در زمان کار با یک نرم افزار تحت وب زمان اشکال زدایی پیش می‌آید که به دلیل موجود بودن داده در حافظه کش برنامه نویس نمی‌تواند داده‌های واقعی را ببیند و داده‌های موجود در حافظه کش را مشاهده می‌کند (بیشتر مواقعی که از طریق بانک اطلاعاتی مستقیما اقدام به حذف و اضافه داده می‌کنیم) در این بخش یک کلاس آماده کرده ام که همیشه خودم در نرم افزار هایم استفاده می‌کنم.

شما می‌توانید این کلاس را به یک GridView یا کنترل‌های دیگر بایند کرده و کلیدهای موجود در حافظه کش را مشاهده کنید، و در صورتی که خواستید یک کلید خاص را از حافظه کش حذف نمایید (البته این کلاس بیشتر برای مدیر نرم فزار کاربرد دارد).

می‌توانید فایل مورد نظر را از طریق لینک کلاس کمکی جهت مشاهده آیتم‌های موجود در حافظه کش و حذف آنها دانلود نمایید.
در کلاس زیر هر کدام از قسمت‌ها را شرح می‌دهیم.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Web;
using System.Web.Caching;

namespace PWS.BLL
{
    /// <summary>
    /// کلاس  آیتم‌های حافظه کش
    /// </summary>
    [DataObject(true)]
    public class CacheItems
    {
#region Constructors (2)

        /// <summary>
        /// سازنده اصلی
        /// </summary>
        /// <param name="cacheItem">عنوان آیتم ذخیره شده در حافظه کش</param>
        public CacheItems(String cacheItem)
        {
            CacheItem = cacheItem;
        }

        /// <summary>
        /// سازنده پیش فرض
        /// </summary>
        public CacheItems(){}

#endregion Constructors

#region Properties (2)

        /// <summary>
        /// کش کانتکست جاری
        /// </summary>
        /// <value>
        /// The cache.
        /// </value>
        private static Cache Cache
        {
            get {return HttpContext.Current.Cache; }
        }

        /// <summary>
        /// عنوان آیتم ذخیره شده در حافظه کش
        /// </summary>
        public String CacheItem{ get; set;}

#endregion Properties

#region Methods (4)
```

```
// Public Methods (3)

        /// <summary>
        /// لیست تمام آیتم‌های ذخیره شده در حافظه کش
        /// </summary>
        /// <returns></returns>
        public List<CacheItems> GetCaches()
        {
            var items = new List<CacheItems>();
            //بازیابی کل کلیدهای موجود در حافظه کش و اضافه کردن آن به لیست مربوطه
            var enumerator = Cache.GetEnumerator();
            while (enumerator.MoveNext())
            {
                 items.Add(new CacheItems(enumerator.Key.ToString()));
            }
            return items;
        }

        /// <summary>
        /// حذف آیتم جاری از حافظه کش
        /// </summary>
        public void RemoveItemFromCache()
        {
            RemoveItemFromCache(CacheItem);
        }

        /// <summary>
        /// حذف کردن یک آیتم از حافظه کش
        /// </summary>
        /// <param name="key">کلید ذخیره شده در حافظه کش</param>
        public static void RemoveItemFromCache(string key)
        {
            PurgeCacheItems(key);
        }
// Private Methods (1)

        /// <summary>
        /// حذف کردن یک ایتم از حافظه کش با پشوند وارد شده
        /// </summary>
        /// <param name="prefix">پیشوندی از کلید موجود در حافظه کش</param>
        private static void PurgeCacheItems(String prefix)
        {
            prefix = prefix.ToLower();
            var itemsToRemove = new List<String>();
           //لیست آیتم‌های موجود در حافظه کش
            var enumerator = Cache.GetEnumerator();
            while (enumerator.MoveNext())
            {


                //در صورتی که کلید مورد نظر با پارامتر وارد شده شروع شده باشد آن را به یک لیست اضافه
می‌کنیم


                if (enumerator.Key.ToString().ToLower().StartsWith(prefix))
                    itemsToRemove.Add(enumerator.Key.ToString());
            }
             //لیست مورد نظر را پیمایش کرده و گزینه‌های آن را از حافظه کش حذف می‌کنیم
            foreach (var itemToRemove in itemsToRemove)
                Cache.Remove(itemToRemove);
        }

#endregion Methods
    }
}
```

موفق ومومید باشید

نظرات خوانندگان

نویسنده: سی شارپ 2012
تاریخ: ۱۳۹۲/۰۸/۲۵ ۱۳:۲۶

چگونه از این کلاس استفاده کنیم ؟
لطفا مثال بزنید

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۸/۲۶ ۱۲:۹

به نظرم کاملا واضحه
شما می‌تونین متد GetCaches را به یک گرید ویو بایند کنید.
با این کار شما لیست ایتم‌های موجود در کش دارید
در مرحله بعد برای حذف هر ایتم از کش می‌توانید ایتم انتخاب شده را به تابع PurgeCacheItems(String prefix) پاس داده و ایتم مورد نظر حذف کنید.
کد نوشته شده چیز چندان پیچیده ای نداره اگرم باز مشکل دارید لطفا مطرح کنید
موفق و موید باشید

عنوان: **صفحه کلید مجازی برای ورود اطلاعات**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۹/۰۶ ۹:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, JavaScript, ASP.Net MVC, ASP.Net 4.5, Security

همانگونه که اطلاع دارید یکی از روش‌های سرقت اطلاعات استفاده از نرم افزارهای جاسوس صفحه کلید (Key Logger) است، البته ثبت کلیدهای فشرده شده می‌تواند توسط سخت افزارهایی که سر راه سوکت صفحه کلید و کیس قرار می‌گیره، انجام بشه. در صورتی که چنین سخت افزاری (مخصوصا در کافی نت‌ها) روی کامپیوتر کاربر نصب باشه، یا توسط ویروس و بدافزارها اینگونه نرم‌افزارهایی روی سیستم کاربر قرار بگیره هر کلیدی که توسط کاربر روی صفحه کلید فشرده میشه توسط اینها ثبت شده و در موقع مناسب برای فرد سازنده به طریقی (ایمیل یا ارتباط از طریق برنامه‌های مبتنی بر سوکت) حتی بسیاری از این برنامه‌ها پا را فراتر گذاشته و عنوان پنجره ای که کلیدها در آن فشرده شده نیز ثبت می‌شود (توسط توابع API ویندوز- البته اگر دوستان مایل باشن و از نظر مدیریت سایت ایرادی نداشته باشه، نحوه طراحی این نوع برنامه‌های جاسوس سخت افزار، صفحه کلید، یا ماوس آموزش می‌دم توی همین سایت)، حال برای امنیت برنامه‌های تحت وب یا ویندوز چگونه می‌توان در زمان ورود اطلاعات حساس مانند کلمه عبور یا شماره کارت اعتباری این امنیت را برای کاربر ایجاد کرد که داده‌هایش توسط این سخت افزارها یا بدافزارها جایی ثبت نشود؟

بله، حدس شما درست است استفاده از صفحه کلیدهای مجازی میشه گفت یکی از بهترین راه‌های ممکن هست، چون در این روش‌ها کلید به صورت سخت افزاری فشرده نمی‌شود (کلید فشرده شده به صف پیام‌های ویندوز نمی‌رود) در نتیجه نرم افزارها یا سخت افزارهای جاسوس نمی‌توانند این اطلاعات را ثبت کنند. و کاربر با خیال راحت می‌تواند داده‌های خود را وارد نمایند (تاکید می‌کنم این روش فقط جلو این نرم افزارها یا سخت افزارها را می‌گیرد و تضمینی برای اینکه در زمان ارسال داده‌های شما لو نرود ندارد).

خوب حال چه باید کرد؟

یک راه می‌تواند پیاده‌سازی صفحه کلید مجازی با کدهای طرف کلاینت مانند جاوا اسکریپت و وی‌بی اسکریپت است، اما گروهی پلاگینی را توسعه داده‌اند که با چند خط کدنویسی ساده به راحتی می‌توانید یک صفحه کلید مجازی چندزبانه (با هر زبانی که دلتون می‌خواد) داشته باشید و از اون در برنامه‌های خودتون استفاده کنید.

نحوه‌ی نصب:

ایتدا فایل‌های مورد نیاز را از سایت سازنده که شامل فایل جاوا اسکریپت ، فایل استایل و یک تصویر (آخرین نسخه) یا از این آدرس به صورت کامل (در حال حاضر نسخه 1.49) دریافت کرده، پس از دریافت فایل‌ها آنها را در هاست خود بارگزاری (آپلود) نمائید. سپس کدهای زیر را در صفحه‌ای که می‌خواهید صفحه کلید نمایش یابد در بین تگ <head> ... و </head> قرار دهید.

```
<script type="text/javascript" src="keyboard.js" charset="UTF-8"></script>
<link rel="stylesheet" type="text/css" href="keyboard.css">
```

حالا فقط کافی است به inputboxها و یا هر ورودی دیگر خود class=keyboardInput بدهید.

مثال:

```
<input type="text" value="" class="keyboardInput">
```

در نهایت کد صفحه شما باید اینگونه باشد:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title></title>
    <script type="text/javascript" src="keyboard.js" charset="UTF-8"></script>
    <link rel="stylesheet" type="text/css" href="keyboard.css"/>
</head>
<body>
    <input type="text" value="" class="keyboardInput"/>
</body>
```

```
</html>
```

با این کار پس از اجرای صفحه مورد نظر خروجی شما مانند تصویر زیر خواهد بود، جهت محدود کردن کلیدها و عملیات دلخواه و سفارشی سازی با پارامترهای دلخواه می‌تواند از دموهای موجود در سایت سازنده بهره بگیرید.

موفق وموید باشید.

نظرات خوانندگان

نویسنده: مجتهدی
تاریخ: ۱۳۹۱/۰۹/۰۶ ۱۸:۰

سلام
تا جایی که یادم میاد، برنامه هایی هم بودن که نقاطی که موس روشون کلیک کرده بود رو ذخیره می‌کردن (چطوریش رو نمی‌دونم!)
اگه همچین برنامه هایی وجود داشته باشن حتی اگه مثل صفحات پرداخت آنلاین بانک‌ها که جای ۱۰ کاراکتر عدد، تغییر می‌کنه، با آنالیز شماره کارت‌ها و مسائلی از قبیل اینکه هر بانک با شماره خاصی شروع می‌شه و... ، فکر کنم بدست آوردن اطلاعات حساس کار سختی نباشن...
درست می‌گم؟

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۹/۰۷ ۱۱:۵۸

دوست من کار نشد نداره، امنیت نسبی هست
بله میشه این کار انجام داد به راحتی امکان پذیر هست، اما ذخیره نقاط چه کاربردی می‌تونی داشته باشه؟

نویسنده: امیرحسین مرجانی
تاریخ: ۱۳۹۱/۰۹/۰۷ ۱۶:۴۶

مطلب مفیدی بود.
ممنونم

نویسنده: nojirom
تاریخ: ۱۳۹۱/۱۰/۱۹ ۱۵:۵۸

با تشکر از آقای فتح اللهی عزیز بابت این مطلب فقط زبان پیش فرض کیبورد راچگونه فارسی تنظیم کنیم

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۰/۲۰ ۷:۳۷

سلام
دوست من در فایل اسکریپت مزبوطه اوایل فایل این خط پیدا کرده

```
this.VKI_kts = this.VKI_kt = "US International";  // Default keyboard layout
```

و آن را به هر زبانی که دلخواه شماست تغییر دهید مثلا برای فارسی اینگونه است:

```
this.VKI_kts = this.VKI_kt = "\u0641\u0627\u0631\u0633\u06cc";  // Default keyboard layout
```

البته من توی همون فایل "fa" جستجو کردم و عنوان اون در این قسمت قرار دادم
موفق وموید باشید

نویسنده: حیدری
تاریخ: ۱۳۹۱/۱۲/۱۸ ۲۰:۵۶

سلام

مرسی ، دستتون درد نکنه
ایشالا همیشه موفق باشید

نویسنده: مرتضی محمودی
تاریخ: ۱۳۹۲/۰۸/۲۴ ۲۰:۱۵

درود
منو راهنمایی کنین که چطور از این کلاس برای RadTextBox یا دیگر Inputهای تلریک استفاده کنم
باتشکر....

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۲۴ ۲۰:۵۷

چه فرقی می‌کنه؟ به خاصیت یا کلاس CSS اش مقدار keyboardInput رو نسبت بده. تمام این کنترل‌های مصنوعی، نهایتا به inputهای استاندارد HTML تبدیل و رندر می‌شن. وگرنه مرورگر نمی‌تونه اون‌ها رو تشخیص بده یا پردازش کنه.

نویسنده: مرتضی محمودی
تاریخ: ۱۳۹۲/۰۸/۲۴ ۲۲:۲

آخه سمت سرور که میره دیگه کلاس میپره. من تکست هام داخل AjaxPanel هست.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۸/۲۴ ۲۳:۱۲

دوباره باید بایندش کنی.

نویسنده: مهیار نظری پور
تاریخ: ۱۳۹۳/۰۴/۲۵ ۱۹:۱۳

بی نهایت درود؛ بنده، به یک کیبورد، به شکل کِی پَد (تنها اعداد، بصورت ماشین حسابی) نیازمندم.
برای درج در فیلد "شماره تماس" فُرم رزروشِن سایتم.
در نت خیلی گشتم، امّا چیزی نیافتم.
دقیقاً چیزی مثل کِی پَد درگاه آنلاین بانک، که برای درج شماره حساب و این‌ها بکار میره روی سایت.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۰۴/۲۷ ۰:۳۹

سلام
خوب میتوانید از همان کدهای درگاه بانک الگو برداری کنید
تلاشتون بکنید اگر نتوانستید من کدش براتون بگیرم
موفق و موید باشید

عنوان: **غیر فعال کردن یک دکمه در حین انجام پردازش های سمت سرور**
نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۹/۱۲ ۵:۴۹
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, JavaScript, Security

بعضی مواقع بهتر است یک دکمه در حال انجام پردازش‌های سمت سرور غیر فعال شود و وقتی عملیات سمت سرور به پایان رسید این دکمه دوباره فعال شود. غیر فعال کردن یک دکمه به این دلیل انجام می‌شود که از postbackهای مجدد در حین postback شدن صفحه جلوگیری شود.
فرض کنید در رویداد کلیک یک دکمه کدی نوشته اید که اطلاعات یک دانشجو را ذخیره کند. کاربر نرم افزار بعد از یک بار کلیک روی دکمه، درخواستی به سرور می‌فرستد و این باعث می‌شود کدهای درون رویداد کلیک دکمه اجرا شوند، به دلیل این که کاملا این کدها اجرا نشده اند، صفحه هم postback نشده است. کاربر به هر دلیل صبر نمی‌کند تا این پردازش تمام شود (شاید او نمی‌داند عملیاتی در سمت سرور در حال اجرا شدن هست چون پیغامی به او نشان داده نشده است)، در نتیجه باز هم روی همین دکمه کلیک می‌کند و باعث می‌شود در حین پردازش کدهای درون دکمه توسط سرور، دوباره درخواستی به سمت سرور فرستاده شود و باز هم کدهای درون همین دکمه به اجرا در آیند. به عبارت بهتر در حین postback شدن صفحه دوباره درخواست postback کردن صفحه را می‌دهد و سرور هم در حین انجام کدهای درون دکمه با درخواست قبلی، دوباره کدهای درون رویداد کلیک را اجرا می‌کند. در این مثال این کار باعث می‌شود کدهای رویداد کلیک دکمه، به تعداد کلیک‌های انجام شده اجرا شوند. در نتیجه به همین تعداد می‌تواند یک دانشجوی مستقل در دیتابیس ذخیره شود. این می‌تواند مشکلات بسیاری را همراه داشته باشد.

به همین دلیل بهتر است با کلیک روی این دکمه این کارها اتفاق بیفتند:
الف) غیرفعال کردن دکمه در حین انجام پردازش‌های سمت سرور
ب) نشان دادن یک پیغام به کاربر در حین انجام پردازش‌های سمت سرور
ج) فعال کردن دکمه بعد از انجام پردازش‌های سمت سرور

برای غیرفعال کردن دکمه در حین انجام پردازش‌های سمت سرور نمی‌توان از کدهای سمت سرور استفاده کرد. چون تا کاملا صفحه postback نشود نمی‌توان این کدها را به صفحه اعمال کرد. پس این گزینه کنار می‌رود.
راه حل بسیار خوب استفاده از جاوا اسکریپت است. مثال زیر را ببینید:

```
<asp:Button runat="server"
            ID="btnProcess"
            Text="پردازش"
            onclick="btnProcess_Click"
            OnClientClick="this.disabled = true; this.value = 'در حال پردازش اطلاعات ...';"
            UseSubmitBehavior="false"
             />
<asp:Label runat="server" ID="lblMessage" Text=""></asp:Label>
```

در رویداد OnClientClick کارهای (الف) و (ب) انجام می‌شوند و با false کردن مقدار رویداد UseSubmitBehavior کار (ج) انجام می‌شود.

و در رویداد کلیک دکمه کد زیر را بنویسید:

```
protected void btnProcess_Click(object sender, EventArgs e)
{
    // insert student in database
    System.Threading.Thread.Sleep(2000);
    lblMessage.Text = "پردازش اطلاعات به پایان رسید";
}
```

در این رویداد باید یک دانشجو اضافه شود. برای ایجاد یک پردازش سمت سروری دو ثانیه ای از متد Sleep استفاده شده است.

کد برگرفته شده از : dotnetforum.lk

نظرات خوانندگان

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۱/۰۹/۱۲ ۶:۹

به نظر میاد وقتی تو صفحه validator داشته باشیم و validator جلوی postback شدن صفحه رو تحت شرایطی میگیره؛ دکمه در حالت غیر فعال باقی خواهد موند

نویسنده: پژمان پارسائی
تاریخ: ۱۳۹۱/۰۹/۱۲ ۷:۳۱

در صورتی که بتونیم با بررسی‌های سمت سروری جلوی postbackهای همزمان رو بگیریم امنیت بالاتری خواهیم داشت.
(منظورم استفاده از custom validator هست که در سمت سرور و بسته به شرایطی که گفتید validate می‌شود)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۱۲ ۸:۳۸

برای رفع این مشکل (غیرفعال کردن سازگار با اعتبار سنجی سمت کاربر) باید به نحو زیر عمل کرد:

```
UseSubmitBehavior="false"
OnClientClick="if ((typeof(Page_ClientValidate) == 'function') && (Page_ClientValidate()==false)) {
return false;} this.disabled=true;"
```

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۹/۲۴ ۲۰:۳۶

ممنون از کار شما
من این روش تست کردم اما زمانی که با یک تاییده همراه باشه کد تقریبا از کار می‌افته، صفحه به سرور پست میشه اما عملیاتی صورت نمی‌گیره.
کدی که نوشتم:

```
$("input[type=submit]").click(function () {
                if ((typeof (Page_ClientValidate) == 'function') && (Page_ClientValidate() == false)) {
                     return false;
                }
                if (!confirm("آیا مطمئن هستید؟"))
                    return false;
                this.disabled = true;
                this.value = 'در حال پردازش اطلاعات ...';
                return true;
            });
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۲۴ ۲۱:۱

در jQuery برای لغو ارسال به سرور، بهتر است بجای return false از e.preventDefault استفاده شود:

```
....click(function(e) {
        e.preventDefault() // stop the automatic form submission
```

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۹/۲۵ ۸:۳۰

سلام این روش استفاده کردم (با return و بدون return)
اما جواب نمی‌ده و فرم ارسال میشه، عملیات هم انجام میشه چه کاربر تایید کنه یا نکنه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۰۹/۲۵ ۹:۴۱

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

<asp:Content ID="HeaderContent" runat="server" ContentPlaceHolderID="HeadContent">
    <script type="text/javascript">
        $(document).ready(function () {
            $("input[type=submit]").click(function (e) {
                if ((typeof (Page_ClientValidate) == 'function') && (Page_ClientValidate() == false)) {
                    return false;
                }
                if (!confirm("آیا مطمئن هستید؟")) {
                    return false;
                }
                this.disabled = true;
                this.value = 'در حال پردازش اطلاعات ...';

                __doPostBack($(this).attr('name'), '');
            });
        });
    </script>
</asp:Content>
<asp:Content ID="BodyContent" runat="server" ContentPlaceHolderID="MainContent">
    <p>
        <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
        <asp:RequiredFieldValidator ID="RequiredFieldValidator1" runat="server"
ControlToValidate="TextBox1"
            ErrorMessage="RequiredFieldValidator"></asp:RequiredFieldValidator>
        <br />
        <asp:Button ID="Button1" runat="server" Text="Button" OnClick="Button1_Click" />
    </p>
</asp:Content>
```

نویسنده: بهاره
تاریخ: ۱۳۹۲/۱۱/۱۷ ۱۹:۴۴

سلام
آیا روشی هست که بشه جلوی کلیک کاربر روی دکمه‌ی refresh مرورگر (IE) را گرفت؟
امکانش هست که این دکمه را غیرفعال کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۷ ۱۹:۵۴

« الگوی PRG در ASP.NET MVC »

نویسنده: بهاره
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۳:۵

ممنون. ولی من از MVC استفاده نمی‌کنم. پروژه در قالب SharePoint 2007 است. Redirect به 303 در asp.net وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۳:۲۸

- مهم داشتن یک ایده اولیه برای شروع و جستجو هست.
- معادل return RedirectToAction در آن کدها Response.Redirect است در وب فرم‌ها. مابقی اصول الگوی PRG تفاوتی نمی‌کند. البته Redirect در اینجا از status code=302 استفاده می‌کند؛ ولی اکثر مرورگرها آن‌را همانند 303 پردازش خواهند کرد.

نویسنده: امیر نوروزیان
تاریخ: ۱۳۹۳/۱۰/۱۹ ۱۳:۳۲

با سلام
در mvc چطوری این روش استفاده کنیم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۹ ۱۳:۳۶

نکته‌ی آن در این مطلب بحث شده: « استفاده از دکمه‌های CSS توئیتر در ASP.NET MVC - قسمت دوم »

عنوان: **تغییر اندازه تصاویر #1**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۹/۱۴ ۱۱:۳۰
آدرس: www.dotnettips.info
گروه‌ها: #ASP.Net, C

برای برنامه نویسان همیشه این امکان هست که تصاویری را که از کاربر دریافت می‌کنند تغییر اندازه دهند، مثلا در همین سایت تصاویری از کاربران جهت نمایش در پروفایل آنها دریافت می‌شود، در همین سایت نیز این اتفاق می‌افتد مثلا تصاویر پروفایل کاربران با اندازه‌های متفاوتی نشان داده می‌شود.

برای انجام این کارها میتوان به دو طریق عمل کرد:

تغییر اندازه تصویر در زمان ذخیره‌سازی
در زمانی که می‌خواهیم تصویر را به بازدید کننده نشان دهیم

در حالت 1 زمانی که تصویری را از کاربر دریافت می‌کنیم با توجه به اینکه تصویر را با چه اندازه‌هایی در نرم افزار نیاز داریم تغییر اندازه داده و تک تک ذخیره می‌کنیم، این روش کل عملیات در زمان ثبت و تنها یکبار اتفاق می‌افتد، این روش جای بیشتری از منابع (مانند هارد دیسک یا دیتابیس) سرور را اشغال می‌کند اما در عوض می‌توان گفت سرعت بالاتری دارد، در روش دوم زمانی که بازدید کننده از سایت (نرم افزار) بازدید می‌کند تصویر اصلی با توجه به نیاز تغییر اندازه داده شده و برای کاربر ارسال می‌شود (در واقع کاربر آن را مشاهده می‌کند)، این روش فضای کمتری از منابع را اشغال می‌کند اما در زمان اجرا عملیات اضافی برای هر کاربری (البته با کش کردن این عملیات کم می‌شود) انجام می‌شود.
در هر دو روش گفته شده در هر صورت ما باید متد (توابعی) برای تغییر اندازه تصویر داشته باشیم که در زیر نحوه نوشتن آن را شرح خواهیم داد.

```
using System;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Imaging;
using System.IO;

namespace PWS
{
    public static class Helpers
    {
        /// <summary>
        /// تغییر اندازه تصویر
        /// </summary>
        /// <param name="imageFile">آرایه بایتی از تصویر مورد نظر</param>
        /// <param name="targetSize">اندازه تصویر خروجی</param>
        /// <param name="format">فرمت تصویر خروجی</param>
```

```
        /// <returns></returns>
        public static byte[] ResizeImageFile(this byte[] imageFile, Int32 targetSize, ImageFormat
format)
        {
            if (imageFile == null)
                throw new Exception("لطفا تصویر اصلی را مشخص نمایید");
            //باز کردن تصویر اصلی به عنوان یک جریان
            using (var oldImage = Image.FromStream(new MemoryStream(imageFile)))
            {
                //محاسبه اندازه تصویر خروجی با توجه به اندازه داده شده
                var newSize = CalculateDimensions(oldImage.Size, targetSize);
                //ایجاد تصویر جدید
                using (var newImage = new Bitmap(newSize.Width, newSize.Height,
PixelFormat.Format24bppRgb))
                {
                    using (var canvas = Graphics.FromImage(newImage))
                    {
                        canvas.SmoothingMode = SmoothingMode.AntiAlias;
                        canvas.InterpolationMode = InterpolationMode.HighQualityBicubic;
                        canvas.PixelOffsetMode = PixelOffsetMode.HighQuality;
                        //تغییر اندازه تصویر اصلی و قرار دادن آن در تصویر جدید
                        canvas.DrawImage(oldImage, new Rectangle(new Point(0, 0), newSize));
                        var m = new MemoryStream();
                        //ذخیره تصویر جدید با فرمت وارد شده
                        newImage.Save(m, format);
                        return m.GetBuffer();
                    }
                }
            }
        }

        /// <summary>
        /// محاسبه ابعاد تصویر خروجی
        /// </summary>
        /// <param name="oldSize">اندازه تصویر اصلی</param>
        /// <param name="targetSize">اندازه تصویر خروجی</param>
        /// <returns></returns>
        private static Size CalculateDimensions(Size oldSize, Int32 targetSize)
        {
            var newSize = new Size();
            if (oldSize.Height > oldSize.Width)
            {
                newSize.Width = Convert.ToInt32(oldSize.Width * (targetSize / (float)oldSize.Height));
                newSize.Height = targetSize;
            }
            else
            {
                newSize.Width = targetSize;
                newSize.Height = Convert.ToInt32(oldSize.Height * (targetSize / (float)oldSize.Width));
            }
            return newSize;
        }
    }
}
```

در متد ResizeImageFileتصویر اصلی به عنوان یک جریان باز می‌شود. (سطر 23)

اندازه تصویر خروجی با توجه به اندازه وارد شده توسط متد CalculateDimensions تعیین می‌شود؛ روال کار متد CalculateDimensions اینگونه است که اندازه عرض و ارتفاع تصویر اصلی بررسی می‌شود و با توجه به اینکه کدام یک از اینها بزرگتر است تغییر اندازه در آن صورت می‌گیرد، مثلا در صورتی که عکس ارتفاع بیشتری نسبت به عرض تصویر داشته باشد تصویر تغییر اندازه داده شده نیز با توجه به تناسب ارتفاع تغییر داده می‌شود و بالعکس. (سطر 26)

پس از تغییر اندازه تصویر جدیدی در حافظه ایجاد می‌شود. (سطر 28)

سپس تنظیمات گرافیکی لازم بروی تصویر جدید اعمال می‌شود. (سطر 30 تا 34)

تصویر اصلی با توجه به اندازه جدید تغییر کرده و در تصویر جدید قرار می‌گیرد. (سطر 36)

در نهایت تصویر جدید با فرمت وارد شده در متد ذخیره شده و به عنوان خروجی متد برگشت داده می‌شود. (سطر 37 تا 40)

خروجی این متد نیز آرایه بایتی می‌باشد که به سادگی می‌توانید از آن برای ذخیره تصاویر در دیتابیس استفاده نمایید.

نحوه استفاده از این تابع در ASP.NET می‌تواند به صورت زیر باشد :

```
byte[] oldImage = FileUploadImage.FileBytes;
byte[] target = oldImage.ResizeImageFile(100, ImageFormat.Jpeg);
```

در واقع فراخوانی مذکور تصویر ورودی را به اندازه 100 پیکسل تغییر داده و در ارایه بایتی به نام target ذخیره می‌کند.
در بخش بعد در زمان نمایش تصویر به کاربر آن را تغییر اندازه خواهیم داد.
لازم به ذکر است که کدهای تغییر اندازه از StarterKitهای میکروسافت کپی‌برداری شده است.

نظرات خوانندگان

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۰۹/۱۴ ۱۳:۲۴

بسیار عالی

در asp.net MVC کلاس webImage در فضانام System.Web.Helpers این امکانات رو در اختیار توسعه دهنده قرار میده .

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۰۹/۱۴ ۱۵:۰

بله دیدم جالب بود واسم، اما تغییر اندازه ندیدم کدوم متده؟

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۰۹/۱۴ ۱۸:۴

متد WebImage.Resize

نویسنده: پ.م
تاریخ: ۱۳۹۳/۰۲/۲۷ ۱۴:۳۶

لطفا در مورد اینکه در زمان واکشی تصاویر از دیتابیس (کش کنیم) توضیح دهید
چون خیلی سرعت کار با سایت پایین میاد
مثلا هر بار کاربر که بخواد از صفحه ای به صفحه ای دیگه بره تصاویر از نو بارگزاری میشن و این سرعت سایت رو کند میکنه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۷ ۱۴:۳۹

به مطلب و مآخذ « مدیریت درخواست‌های شرطی در ASP.NET MVC » مراجعه کنید. نکات کلی در بحث عنوان شده. در مآخذ آن پیاده سازی‌های مخصوص وب فرم‌ها هم هست.

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۵/۳۱ ۱۹:۱۷

سلام؛ موقع تغییر سایز یک تصویر png به png پس زمینه رو مشکی میکنه. آیا راهی هست در صورتی که تصویر اولیه png باشه تصویر نهایی هم بدون پس زمینه باشه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۲۰ ۱۳:۵

Resize PNG Image and Keep Transparent Background

عنوان: **آشنایی و بررسی ابزار MiniProfiler**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۱/۱۵ ۱۶:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Entity framework, ASP.Net MVC, EF Code First, Oracle, Statistics, stackoverflow, MVC

در کنار کتابخانه elmah که وظیفه ثبت تمامی خطاهای برنامه را دارد کتابخانه MiniProfiler امکان یافتن مشکلات کارایی و تنگناهای وب سایت را در اختیارمان قرار می‌دهد. دو قابلیت عمده که این ابزار فراهم می‌نمایید
امکان مشاهده و بررسی کوئری‌های خام ADO.NET از قبیل SQL Server,Oracle و LINQ-to-SQL و EF/First Code و...
نمایش زمان اجرای عملی صفحات

برای استفاده از این ابزار کافیست تا آن را از nuget دریافت نمایید

```
PM> Install-Package MiniProfiler
```

در ASP.NET MVC در صفحه _Layout قبل از بسته شدن تگ body تابع RenderIncludes را مانند زیر صدا بزنید تا در همه صفحات نمایش داده شود

```
@using StackExchange.Profiling;
<head>
 ..
</head>
<body>
  ...
  @MiniProfiler.RenderIncludes()
</body>
```

در کلاس global کد زیر را برای اجرای MiniProfiler اضافه نمایید

```
protected void Application_BeginRequest()
{
    if (Request.IsLocal)
    {
        MiniProfiler.Start();
    }
}

protected void Application_EndRequest()
{
    MiniProfiler.Stop();
}
```

برای پیکربندی MiniProfiler در web.config کد زیر را اضافه نمایید

```
<system.webServer>
  ...
  <handlers>
    <add name="MiniProfiler" path="mini-profiler-resources/*" verb="*"
         type="System.Web.Routing.UrlRoutingModule"
         resourceType="Unspecified"
         preCondition="integratedMode" />
  </handlers>
</system.webServer>
```

یا کتابخانه MiniProfiler.MVC را از nuget دریافت نمایید

```
PM> Install-Package MiniProfiler.MVC
```

با اضافه شدن این کتابخانه همه پیکربندی بصورت صورت خودکار انجام می‌گیرد. حال وب سایت را اجرا کنید در بالای صفحه

مانند شکل زیر مدت زمان بارگذاری صفحه نمایش داده می‌شود که با کلیک بر روی آن اطلاعات بیشتری را مشاهده می‌نمایید

اگر در اکشن اجرا شده کوئری اجرا شد باشد ستونی به نام query times نمایش داده می‌شود که تعداد کوئری‌ها و مدت زمان آن را نمایش می‌دهد

حال بر روی گزینه sql کلیک کنید که صفحه دیگری باز شود و کوئری خام آن را مشاهد نمایید اگر کوئری تکرار شده باشد در کنار آن با DUPLICATE متمایز شده است

برای مشاهده کوئری‌های Entity Framework/First Code کتابخانه MiniProfiler.EF را اضافه نمایید

```
PM> Install-Package MiniProfiler.EF
```

اگر بصورت دستی MiniProfiler را پیکربندی کرده باشید می‌بایست در Application_Start دستور زیر را اجرا نمایید

```
protected void Application_BeginRequest()
{
    if (Request.IsLocal)
    {
        MiniProfiler.Start();
        MiniProfilerEF.Initialize();
    }
}
```

در حالت پیشرفته‌تر اگر قصد داشته باشید زمان یک قطعه کد را جداگانه محاسبه نمایید بصورت زیر عمل نمایید

```
public ActionResult Index()
{

    var profiler = MiniProfiler.Current;

    using (profiler.Step("Step 1"))
    {
        //code 1
    }

    using (profiler.Step("Step 2"))
    {
        //code 2
    }

    return View();
}
```

با این کار زمان هر step را بصورت جداگانه محاسبه می‌نماید. در ASP.NET Webforms دقیقا به همین صورت استفاده می‌شود فقط

کافیست در masterpage اصلی یا اگر از masterpage استفاده نمی‌کنیم در صفحه مورد نظر تابع RenderIncludes را بصورت زیر صدا بزنیم

```
<%= StackExchange.Profiling.MiniProfiler.RenderIncludes() %>
```

امیدوارم مفید واقع شده باشد.

نظرات خوانندگان

نویسنده: سعید
تاریخ: ۱۳۹۱/۱۱/۱۵ ۱۸:۳

ممنون از شما. آخرین باری که miniprofiler.ef رو تست کردم با ef5 کار نمی‌کرد و خطا می‌داد. شما تست کردید با آخرین نگارش ef code first؟

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۱/۱۵ ۱۸:۱۴

نه هیچ مشکلی ندارد من خودم با ef code first 5 تست کردم

نویسنده: Alireza Godazchian
تاریخ: ۱۳۹۱/۱۱/۱۸ ۸:۴۶

با سلام؛ آقای مهندس خیلی عالی توضیح دادید و بنده هم تست کردم و جواب گرفتم.
با تشکر فراوان

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۱/۱۸ ۲۰:۵۲

سلام چرا تو دانت ای اس پی کار نمیکنه فقط تو ام وی سی جواب میده
MiniProfiler نمیشناسه

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۱/۱۸ ۲۲:۸

در ASP.NET هم بخوبی جواب می‌دهد آیا Handler را در Web.config اضافه نموده اید؟

نویسنده: هیمن صادقی
تاریخ: ۱۳۹۳/۰۴/۲۷ ۱۶:۵۳

با سلام و درود بی کران
من کار که فرمودید انجام دادم اما زمانی که اجرا می‌کنم خطا

```
No Entity Framework provider found for the ADO.NET provider with invariant name 'System.Data.Odbc'. Make sure"
the provider is registered in the 'entityFramework' section of the application config file. See
".http://go.microsoft.com/fwlink/?LinkId=260882 for more information
```

میدهد
از mvc4 , entity framework6 استفاده می‌کنیم
در صورت امکان روش رفع خطا می‌دانید بیان نماید تا بتوانم از این ابزار استفاده کنم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۲۷ ۱۸:۲۲

برای EF 6 یک بسته‌ی نیوگت جداگانه دارد. اطلاعات بیشتر

عنوان: مروری بر قابلیت جدید ASP.NET FriendlyUrls
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۱/۲۸ ۱۶:۲۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, ASP.Net 4.5, FriendlyUrls, SEO

احتمالا به اهمیت بحث SEO و آدرس‌های تمیز آن آشنا هستید. ASP.NET MVC از همان ابتدا از قابلیت URL Routing پشتیبانی می‌کرد و ASP.NET Web Forms هم از نسخه 4.0 .NET. این قابلیت را در خود اضافه نمود. با این حال نوشتن Url Routing برای وب سایت‌ها کمی مشکل و در ASP.NET کمتر استفاده می‌شود. علاوه بر آن تغییر آدرس‌های قدیمی با Url Routing بسیار وقت گیر می‌باشد. به همین خاطر تیم توسعه ASP.NET کتابخانه ای با نام Asp.NET FriendlyUrls که فعلا نسخه قبل از نهایی آن منتشر شده است را در اختیار توسعه دهندگان وب سایت‌ها قرار داده‌اند که با حداقل کد نویسی و بدون اضافه نمودن مسیریابی آدرس‌ها، تمام آدرس‌های قدیم و جدید را مطابق استاندارد SEO تغییر می‌دهد.

دریافت و نصب Asp.NET FriendlyUrls از Nuget

در Visual Studio 2012.2 در صورتی که یک پروژه جدید از نوع ASP.NET Web Forms ایجاد نمایید بصورت پیشفرض این کتابخانه نصب شده است. اما می‌توانید با دستور زیر نیز آن را از Nuget دریافت نمایید:

```
Install-Package Microsoft.AspNet.FriendlyUrls -Pre
```

پس از نصب، در کلاس Global می‌بایست کتابخانه را فعال نمایید:

```
public static class RouteConfig
    {
        public static void RegisterRoutes(RouteCollection routes)
        {
            routes.EnableFriendlyUrls();
        }
    }
```

و همچنین در Application_Start

```
void Application_Start(object sender, EventArgs e)
        {
            RouteConfig.RegisterRoutes(RouteTable.Routes);
        }
```

به همین راحتی؛ حال دو صفحه First.aspx و Second.aspx را به پروژه اضافه نمائید و وب سایت را اجرا کنید. بدون نوشتن پسوند aspx به صفحات دسترسی خواهید داشت:

از درون برنامه هم به راحتی می‌توانید صفحات را به همان صورت قبل بدون پسوند صدا بزنید:

```
Response.Redirect("~\\Second");
```

دسترسی به پارامترها نیز به صورت قبل امکان پذیر می‌باشد:

```
Response.Redirect("~\\First?Id=1001");
string Id = Request.QueryString["Id"] as string;
```

برای آدرس‌های پیشرفته‌تر، 3 متود الحاقی اضافه گردیده است. مثلا اگر آدرس زیر را داشته باشیم

```
http://localhost:57353/First/Products/NewProduct
```

Request.GetFriendlyUrlFileVirtualPath ()

مسیر مجازی فایل را بر می‌گرداند

Request.GetFriendlyUrlSegments()

بخش‌های آدرس را بر می‌گرداند

```
protected void Page_Load(object sender, EventArgs e)
{
    IList<string> urlSegments = Request.GetFriendlyUrlSegments();
}
```

urlSegments Count = 2
[0] "Products"
[1] "NewProduct"
Raw View

Request.GetFriendlyUrlFileExtension()

پسوند فایل آدرس را بر می‌گرداند

```
string extension = Request.GetFriendlyUrlFileExtension();
```

extension ".aspx"

کلاس FriendlyUrl دو متود استاتیک و یک خاصیت دارد

FriendlyUrl.Segments

شبیه Request.GetFriendlyUrlSegments() مجموعه بخش‌های آدرس صفحه جاری را بر می‌گرداند

FriendlyUrl.Resolve

آدرس صفحه جاری را بر می‌گرداند

FriendlyUrl.Href

امکان ساخت یک آدرس را با استفاده از مسیر مجازی، بخش‌ها و پارامترها، فراهم می‌نماید

```
string url = FriendlyUrl.Href("~/First", "Product", 1001);
```

url "/First/Product/1001"

در آخر، فضای نام کتابخانه، Microsoft.AspNet.FriendlyUrls می‌باشد.

نظرات خوانندگان

نویسنده: M.Q
تاریخ: ۱۳۹۱/۱۱/۳۰ ۱۹:۵۰

خیلی عالی بود ممنون

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۰۲ ۸:۴۸

دستت درد نکنه مهندس عالی بود، فکر کنم نسخه نهاییش اومده در این لینک

نویسنده: Rahmani
تاریخ: ۱۳۹۱/۱۲/۰۵ ۱۴:۱۶

وقتی با vs سایت رو run می‌کنم این قابلیت کار می‌کنه، اما وقتی می‌فرستمش روی iis رو سرور واقعی، خطای 404 میده.
سرچیدم ولی جواب خاصی پیدا نکردم. کمک لطفاً

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۰۵ ۱۴:۲۹

از چه IIS ایی استفاده می‌کنید؟ اگر IIS 6 است، این مطلب را مطالعه کنید.
به صورت خلاصه IIS 6 برنامه‌های ASP.NET و صفحات آن‌را بر اساس پسوند آن‌ها پردازش می‌کند. اگر پسوندی ندارند باید IIS را طوری تنظیم کرد که تمام درخواست‌های رسیده را به موتور ASP.NET هدایت کند.

نویسنده: Rahmani
تاریخ: ۱۳۹۱/۱۲/۰۵ ۱۵:۲۲

ببخشید. باید همونجا می‌گفتم.
IIS 7.5 روی Windows Server 2008

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۰۵ ۱۷:۵

- بررسی کنید آیا برنامه شما در IIS7 در حالت classic اجرا می‌شود یا integrated mode . اگر حالت کلاسیک است، همان روش IIS6 که عنوان شد باید طی شود.
- تنظیم زیر را به web.config برنامه اضافه کنید:

```
<system.webServer>
  <modules runAllManagedModulesForAllRequests="true" />
</system.webServer>
```

- این هات فیکس را نصب کنید: (^). البته اگر سرویس پک یک 2008 R2 نصب شده، نیازی به آن نیست.

نویسنده: Rahmani
تاریخ: ۱۳۹۱/۱۲/۰۶ ۱۳:۰

با نصب اون هات فیکس مشکل رفع شد.
ممنون آقای نصیری

نویسنده: پریسا
تاریخ: ۱۳۹۱/۱۲/۰۹ ۱۸:۵

چرا وقتی از FriendlyUrl استفاده می‌کنم به ازای هر style یا فایل .js
pageload صفحه اجرا میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۰۹ ۱۹:۱۸

می‌تونید مسیرهایی/فایل‌هایی رو که نباید وارد سیستم routing شوند از لیست حذف کنید. در همان متد RegisterRoutes باید تعاریف زیر اضافه شوند (اگر متد IgnoreRoute در دسترس نبود متد Ignore همین کار را انجام می‌دهد):

```
routes.IgnoreRoute("*.html|js|css|gif|jpg|jpeg|png|swf");
```

نویسنده: پرسا
تاریخ: ۱۳۹۱/۱۲/۱۰ ۹:۳۴

سلام کدهایی که فرمودین اضافه کردم

```
public static void RegisterRoutes(RouteCollection routes)
        {

            routes.EnableFriendlyUrls();
            routes.Ignore("*.html|js|css|gif|jpg|jpeg|png|swf");
        }
```

اما باز نشد با اجرای صفحه همچنان به ازای هر فایل js
page load صفحه اجرا می‌شود

```
public partial class Test : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        if (!IsPostBack)
        {
            IList<string> r = Microsoft.AspNet.FriendlyUrls.FriendlyUrl.Segment

        }

    }
}
```

r Count = 2
[0] "تست1"
[1] "تست2"
Raw View

```
protected void Page_Load(object sender, EventArgs e)
{
    if (!IsPostBack)
    {
        IList<string> r = Microsoft.AspNet.FriendlyUrls.FriendlyUrl.Segments;
    }
}
```

r Count = 2
[0] "تست1"
[1] "jquery-1.4.1.js"
Raw View

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۱۰ ۱۰:۲۵

- برای تعریف صحیح اسکریپت‌ها یا از متد ResolveUrl استفاده کنید:

```
<script src='<%= ResolveUrl("~/Scripts/jquery-1.4.1.js")%>' type="text/javascript" />
```

و یا روش بهتر استفاده از اسکریپت منیجر است .

نویسنده: پرسا
تاریخ: ۱۳۹۱/۱۲/۱۰ ۱۱:۵۳

خیلی لطف کردین فقط یه سوال دیگه
فایل‌های js اضافه کردم به اسکریپت منیجر درست شد
اما فایل css را چی کار کنم

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۲/۱۲ ۰:۱۹

تست کنید آیا این سیستم به css ایی که داخل ریشه روت سایت هم هست ایراد می‌گیره؟ اگه الان با js مشکل ندارید علت این است که سیستم فشرده سازی و یکی کردن اون‌ها رو به صورت یک زیر پوشه ارائه نمی‌ده.

نویسنده: پرسا
تاریخ: ۱۳۹۱/۱۲/۱۲ ۸:۴۸

بله من اگر آدرس Css در پروژه به صورت زیر باشد

```
<link href="css/StyleSheet.css" rel="stylesheet" type="text/css" />
```

باز برای این pageload صفحه را اجرا می‌کند

نویسنده: پریسا
تاریخ: ۱۳۹۲/۰۲/۲۱ ۱۷:۱۲

سلام
زمانی که از این frinly url استفاده می‌کنم جایی که subdomain دارم مثلا a.test.ir/gallery صفحه‌ی گالری

را پیدا نمی‌کنه در صورتی که بدون subdomain کار می‌کند

نویسنده: ایمان صفری
تاریخ: ۱۳۹۲/۰۵/۰۹ ۲۰:۲۰

با تشکر از شما من وقتی از Resolve استفاده می‌کنم مشکلی ندارم اما زمانیکه وارد یک Route تعریف شده میشم هم Css و فایلهای jquery که استفاده کردم به هم می‌ریزه و هم اینکه Url ای که وارد شدم با کلیک روی یک لینک دیگه به آدرس مورد جستجو اضافه میشه از کدهایی که هم آقای نصیری گفتند استفاده کردم ولی نتیجه ای نداد

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۰ ۰:۱۸

برای خطایابی و یافتن مشکل مطلب نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery می‌تونه مفید باشه.

نویسنده: حسین
تاریخ: ۱۳۹۲/۱۱/۰۸ ۱۶:۲۷

با سلام و تشکر از شما؛ این افزونه با medium trust کار نمیکنه راهی وجود داره که با medium Trust کار کنه؟ یه درد و دل! اینکه چرا بعضی هاست‌ها تنظیمات medium trust رو اعمال میکنن برام عجیبه! میگن به دلیل امنیتی! خب سایت هایی که full trust هستن چجوری مشکل امنیت پیدا نمیکنن! مثل همین موضوع من از asp.net identity هم نمیتونم استفاده کنم چون به owin گیر میده (به همون دلیل امنیتی!)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۰۸ ۱۶:۵۸

یکی از اعضای تیم ASP.NET در اینجا عنوان کرده که Medium Trust به صورت رسمی منسوخ شده درنظر گرفته می‌شود و در محصولات آتی ما به هیچ عنوان مدنظر نخواهد بود و هاست‌ها باید به روش OS-level isolation کوچ کنند.

نویسنده: سارا
تاریخ: ۱۳۹۲/۱۱/۲۰ ۱۰:۴۸

ببخشید من دقیقا منظورتون رو از کلاس global متوجه نمی‌شم....باید چنین کلاسی ایجاد کنیم یا باید با نصب به وجود بیاد؟...لطفا یه توضیح مختصر در این مورد بدین

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۰ ۱۱:۳۲

منظورشان همان فایل global.asax.cs معروف است (Add New Item -> Global Application Class).

نویسنده: متین
تاریخ: ۱۳۹۲/۱۱/۲۰ ۱۱:۵۸

سلام؛ من کتابخانه رو به پروژه اضافه کردم. اما برای فعال سازیش به مشکل برخوردم. EnableFriendlyUrls رو نمیشناسه و خطا میده

```
System.Web.Routing.RouteCollection' does not contain a definition for 'EnableFriendlyUrls'
and no extension method 'EnableFriendlyUrls' accepting a first argument of type
'System.Web.Routing.RouteCollection' could be found (are you missing a using directive or an assembly
reference?)
```

استفاده هم زمان این روش با url routing که مشکلی به وجود نمیاره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۲۰ ۱۲:۲۴

متد الحاقی است و نیاز به تعریف فضای نام مرتبط با آن در کلاس مورد استفاده می‌باشد (using Microsoft.AspNet.FriendlyUrls).

نویسنده: رضا
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۰:۴۵

من این رو اجرا می‌کنم توی صفحه ای به اسم viewSwitcher ارور میده
هیچ کدوم از کلمات این خط رو نمیشناسه

```
<%: CurrentView %> view | <a href="<%: SwitchUrl %>" data-ajax="false">Switch to <%: AlternateView
%></a>
```

مشکلش چیه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۱:۴۸

مشکلی نداره. یک نمونه برای آزمایش: FriendlyUrlsTests.rar

عنوان: **آشنایی با قابلیت جدید ASP.NET Web Forms Scaffolding**
نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۰۳ ۲۳:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Scaffolding Web Forms

مایکروسافت با افزایش سرعت به روز رسانی توسعه پروژه‌های سورس باز خود جهت پاسخ دادن به نیاز توسعه دهندگان و توسعه ویژوال استادیو مطابق با آخرین تکنولوژی‌های تولید وب سایت، می‌کوشد تعداد بیشتری از توسعه دهندگان را به سمت استفاده از تکنولوژی‌های خود سوق دهد.

سالها است که برنامه نویسان خبره با توجه به روش کاری خود از امکانات Code Generatorها برای تولید کدهای لایه‌های Data Access ، Logic و یا حتی User Interface استفاده می‌نمایند. پس از عرضه Entity Framework و تولید خودکار کدهای لایه های Data Access و Logic، این بار این امکان علاوه بر ASP.NET MVC در ASP.NET Web Forms نیز فراهم گردیده‌است تا بدون کد نویسی خسته کننده و تکراری، کدهای لایه رابط کاربر Create-Read-Update-Delete (CRUD) را نیز تولید نماییم.

شروع کار با ASP.NET Scaffolding

پیش نیاز این کار استفاده از Visual Studio 2012 به همراه Web Tools 2012.2 می‌باشد.

اول، ابزار Microsoft ASP.NET Scaffolding را از منوی Tools گزینه Extensions and Updates دریافت و نصب نمایید.

دوم پروژه جدیدی از نوع Visual C# ASP.NET Web Forms Application با فریم ورک 4.5 ایجاد نمایید.
از پنجره NuGet Package manager با دستور install کتابخانه ASP.NET Web Forms Scaffold Generator را دریافت نمایید

```
install-package Microsoft.AspNet.Scaffolding.WebForms -pre
```

کلاس Person را مانند زیر در فولدر Models ایجاد نمایید

```
public class Person
    {
        [ScaffoldColumn(false)]
        public int ID { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
    }
```

ویژگی ScaffoldColumn را برای ID، برابر false قرار دهید تا از ایجاد این ستون جلوگیری نمائید.
پروژه را Build نمایید.
بر روی پروژه راست کلیک و از گزینه Add، گزینه Scaffold... را انتخاب نمایید.

از پنجره Add Scaffold باز شده بر روی گزینه Add، کلیک کنید.

پنجره Add Web Forms Pages مانند زیر باز می‌شود که امکان انتخاب کلاس،Data Context و MasterPage فراهم می‌باشد.

از گزینه Data Context class گزینه New Data Context را انتخاب نمایید. صفحات مورد نیاز را در فولدر Views/Person ایجاد می‌نمایید.

کدهای تولید شده را می‌توانید بازبینی نمایید پروژه را اجرا تا خروجی کار را مشاهده نمایید.

نظرات خوانندگان

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۱/۱۲/۰۴ ۹:۳۸

سلام . ممنون بابت معرفی این Extensions . چه تفاوتی بین این Extensions و پروژه‌های Dynamic Data که در ASP هست ، وجود داره ؟ و یک سوال دیگه ، هدف خاصی برای استفاده از دات نت 4.5 وجود داره ؟

نویسنده: علی قمشلویی
تاریخ: ۱۳۹۱/۱۲/۰۴ ۹:۴۵

خیلی عالی بود

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۰۴ ۱۰:۲۶

این ابزار با استفاده از Dynamic Data عملیات تولید کدها را انجام می‌دهد و پیش نیاز آن دات نت 4.5 حتمالا بخاطر پشتیبانی بهتر از Entity Framework First Code و dllهای دیگر است.

نویسنده: محمد باقر سیف اللهی
تاریخ: ۱۳۹۱/۱۲/۰۴ ۱۰:۴۷

تشکر

نویسنده: فاتحی
تاریخ: ۱۳۹۱/۱۲/۰۷ ۱۱:۳۹

سلام
من Web Tools 2012.2 رو نصب کردم و تمام مراحلی را که توضیح دادید رو هم انجام دادم.
اما نمی‌تونم گزینه scaffold رو تو Add پیدا کنم.
به نظرتون مشکل از کجاست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۰۷ ۲۱:۱۵

لیست جامع به روز رسانی‌های VS 2012 رو بررسی کنید. شاید چیزی از قلم افتاده این بین.

نویسنده: مجتبی کاویانی
تاریخ: ۱۳۹۱/۱۲/۰۷ ۲۱:۵۲

برای نمایش گزینه Scaffold می‌بایست ابزار Microsoft ASP.NET Scaffolding را نصب نمایید

نویسنده: امیر
تاریخ: ۱۳۹۱/۱۲/۰۸ ۱:۳۱

سلام بعد از انجام تمام مراحل این خطا برام ظاهر میشه

Server Error in '/' Application.

The resource cannot be found. **Version Information:** Microsoft .NET Framework Version:4.0.30319; ASP.NET Version:4.0.30319.17929

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۲/۰۸ ۱۲:۱۷

برای کار با scaffolding نیاز هست که به EF Code first کاملا مسلط باشید وگرنه نمی‌تونید برنامه رو دیباگ کنید یا از خطاهاش سر دربیارید.

عنوان: **آموزش ساخت و کار با subdomain در حالت لوکال هاست**
نویسنده: ناصر فرجی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۲:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

برای پروژه ای نیاز بود تا از sub-domain استفاده شود. نحوه کار بدین صورت بود که اطلاعات باید بر اساس شهرها بارگذاری می‌شد؛ برای مثال اگر کاربر آدرس tehran.mysite.com را وارد کرد، اطلاعات مربوط به شهر تهران بارگذاری شود و برای سایر شهرها هم به همین ترتیب. در سرور که روش کار و ساخت ساب دومین کاملا مشخص است. اما برای لوکال باید به چه ترتیب عمل شود؟

مسلما برنامه پیش از آپلود حتما باید روی لوکال تست شود. برای ساخت ساب دومین روی لوکال هاست کافی ست به مسیر زیر رفته و فایل hosts را با دسترسی administrator باز کنید .

```
[windows drive]:\Windows\System32\drivers\etc
```

سپس به شیوه زیر ساب دومین‌های خود را در این فایل وارد کنید .

```
# localhost name resolution is handled within DNS itself.
127.0.0.1  localhost
127.0.0.1 tehran.localhost
127.0.0.1 mashhad.localhost
127.0.0.1 sabzevar.localhost
```

خوب، از این به بعد در صورتیکه کاربر هر کدام از ساب دومین‌های وارد شده در این فایل را در مرورگر خود وارد کند، به راحتی می‌توان در رویداد Application_BeginRequest در فایل Gloabal.asax مسیر url را بدست آورد و کاربر را به مسیر دلخواه هدایت کرد.
نکته مهمی که باید حتما توجه شود استفاده از دستور Request.Url.ToString(); است. هنگامیکه برنامه روی هاست در حال اجراست، قادر به برگرداندن مسیر به همراه با نام ساب دومین می‌باشد، اما این دستور روی لوکال، نام ساب دومین را برنمی‌گرداند! برای این کار روی لوکال استفاده از روش زیر استفاده کنید:

```
host = Request.Headers["HOST"].ToLower();
```

اکنون بعد از به دست آوردن مسیر کاربر می‌توانیم او را به مسیر دلخواه بفرستیم. مثلا من ابتدا با به دست آوردن نام شهر از روی url کاربر، کد متناظر با شهر را پیدا کرده و کد را به همراه آدرس به صفحه مورد نظر فرستاده و در آن صفحه با دریافت کد شهر، اطلاعات لازم را از دیتابیس فراخوانی می‌کنم:

```
void Application_BeginRequest(object sender, EventArgs e)
    {
        string host, url, newUrl;
        host = Request.Headers["HOST"].ToLower();   //for localhost
        //host = Request.Url.ToString();  //  for server
        url = Request.Url.PathAndQuery.ToLower();

        if (host.EndsWith(".localhost:8001") && (url == "" || url == "/"))
        {
            int cityId = City.GetCityIdByCityName(host.Replace(".localhost:8001", ""));
            newUrl = string.Format("~/postlist/{0}/", cityId);
            HttpContext.Current.RewritePath(newUrl);
            return;
        }

    }
```

نکته: در صورتی که در حال استفاده از vpn باشید قادر به دسترسی به subdomainهای خود در حالت لوکال نخواهید بود. پس برای تست برنامه، حتما این مورد را مد نظر داشته باشید.

موفق باشید.

نظرات خوانندگان

نویسنده: فواد عبداللهی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۲:۴۴

ممنون از آموزشتون
ولی یه سوال برام پیش اومد و اونم اینکه شما تو فایل hosts به صورت استاتیک اسامی رو تعریف کردین ولی در عمل بیشتر نیاز به داینامیک بودن این مطلب هست.

نویسنده: محسن
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۳:۱۹

IIS ماژولی به نام URL Rewrite برای اینکار داره. با برنامه نویسی هم قابل کنترل است.

نویسنده: ناصر فرجی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۳:۲۴

بله. روی لوکال هدف تست برنامه ست. برای هاست هم در صورتی که بخاید به صورت بی نهایت ساب دومین داشته باشید (شبیه سیستم هایی مثل بلاگفا) باید یک ip اختصاصی داشته باشید و تنظیماتی رو در DNS خودتون اعمال کنید. به این صورت کاربر هر زیر دامنه ای رو روی دامین شما باز کنه به صورت اتوماتیک به وب سایت شما هدایت میشه. در صورتی هم که تعداد ساب دومین هاتون محدوده که میتونید خودتون از طریق کنترل پنل دامنه اون‌ها رو ایجاد کنید.

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۲/۰۹/۲۰ ۲۰:۵۷

با سلام و تشکر از مطالب شما
میخواستم بدونم میشه یه پروژه از نوع mvc و یه پروژه از نوع web form رو به صورت subdomain رو iis نصب کرد؟
در واقع من میخوام cms سایت رو با mvc انجام بدم و خود سایت رو با web from و به صورت ساب دومین مثلاً:

```
cms.mysite.com
```

```
mysite.com
```

ممنون.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۹/۲۰ ۲۱:۱۲

مشکلی نداره. برای هر کدوم یک application pool جدا درست کنید تا تداخل نداشته باشند.

نویسنده: میثم فغفوری
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۹:۲۳

ممنون بابت مطلبتون.
متد Request.Headers["HOST"] بر روی هاست نام ساب دامین‌ها رو برمیگردونه؟ اگر نه چجوری میشه فهمید برنامه به صورت لوکال اجرا شده یا بر روی سرور؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۹:۲۸

از خاصیت IsLocal استفاده کنید. پشت صحنه‌ی Host .

نویسنده: رسول
تاریخ: ۱۳۹۲/۱۲/۱۴ ۱۱:۳۳

سلام.
من این روش را توی لوکال بدون مشکل انجام دادم
اما بر روی vps باید چیکار کنم؟
توی dns رفتم و new alias ساختم ولی کار نداد.

عنوان: **تغییر اندازه تصاویر #2**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۲:۳۵
آدرس: www.dotnettips.info
گروه‌ها: #ASP.Net, C

در ادامه مطلب تغییر اندازه تصاویر #1 ، در این پست می‌خواهیم نحوه تغییر اندازه تصاویر را در زمان درخواست کاربر بررسی کنیم.

در پست قبلی بررسی کردیم که کاربر می‌تواند در دوحالت تصاویر دریافتی از کاربران سایت را تغییر اندازه دهد، یکی در زمان ذخیره سازی تصاویر بود و دیگری در زمانی که کاربر درخواست نمایش یک تصویر را دارد.

خوب ابتدا فرض می‌کنیم برای نمایش تصاویر چند حالت داریم مثلا کوچک، متوسط، بزرگ و حالت واقعی (اندازه اصلی).
البته دقت نمایید که این طبقه بندی فرضی بوده و ممکن است برای پروژه‌های مختلف این طبقه بندی متفاوت باشد. (در این پست قصد فقط اشنایی با تغییر اندازه تصاویر است و شاید کد به درستی refactor نشده باشد).
برای تغییر اندازه تصاویر در زمان اجرا یکی از روش ها، می‌تواند استفاده از Handler باشد. خوب برای ایجاد Handler ابتدا در پروژه وب خود بروی پروژه راست کلیک کرده، و گزینه New Item را برگزینید، و در پنجره ظاهر شده مانند تصویر زیر گزینه Generic Handler را انتخاب نمایید.

پس از ایجاد هندلر، فایل کد آن مانند زیر خواهد بود، ما باید کدهای خود را در متد **ProcessRequest** بنویسیم.

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
```

```
namespace PWS.UI.Handler
{
    /// <summary>
    /// Summary description for PhotoHandler
    /// </summary>
    public class PhotoHandler : IHttpHandler
    {

        public void ProcessRequest(HttpContext context)
        {
            context.Response.ContentType = "text/plain";
            context.Response.Write("Hello World");
        }

        public bool IsReusable
        {
            get
            {
                return false;
            }
        }
    }
}
```

خوب برای نوشتن کد در این مرحله ما باید چند کار انجام دهیم.

1- گرفتن پارامترهای ورودی کاربر جهت تغییر سایز از طریق روش‌های انتقال مقادیر بین صفحات (در اینجا استفاده از Query String).

2-بازیابی تصویر از دیتابیس یا از دیسک به صورت یک آرایه بایتی.

3- تغییر اندازه تصویر مرحله 2 و ارسال تصویر به خروجی.

```
using System;
using System.Data.SqlClient;
using System.Diagnostics;
using System.Drawing;
using System.Drawing.Drawing2D;
using System.Drawing.Imaging;
using System.Globalization;
using System.IO;
using System.Web;

namespace PWS.UI.Handler
{
    /// <summary>
    /// Summary description for PhotoHandler
    /// </summary>
    public class PhotoHandler : IHttpHandler
    {

        /// <summary>
        /// بازیابی تصویر اصلی از بانک اطلاعاتی
        /// </summary>
        /// <param name="photoId">کد تصویر</param>
        /// <returns></returns>
        private byte[] GetImageFromDatabase(int photoId)
        {
            using (var connection = new SqlConnection("ConnectionString"))
            {
                using (var command = new SqlCommand("Select Photo From tblPhotos Where Id = @PhotoId",
connection))
                {
                    command.Parameters.Add(new SqlParameter("@PhotoId", photoId));
                    connection.Open();
                    var result = command.ExecuteScalar();
                    return ((byte[])result);
                }
            }
        }

        /// <summary>
        /// بازیابی فایل از دیسک
        /// </summary>
        /// <param name="photoId">با فرض اینکه نام فایل این است</param>
        /// <returns></returns>
```

```
        private byte[] GetImageFromDisk(string photoId /* or somting */)
        {
                using (var sourceStream = new FileStream("Original File Path + id", FileMode.Open,
FileAccess.Read))
                {
                    return StreamToByteArray(sourceStream);
                }
        }

        /// <summary>
        /// Streams to byte array.
        /// </summary>
        /// <param name="inputStream">The input stream.</param>
        /// <returns></returns>
        /// <exception cref="System.ArgumentException"></exception>
        static byte[] StreamToByteArray(Stream inputStream)
        {
            if (!inputStream.CanRead)
            {
                throw new ArgumentException();
            }

            // This is optional
            if (inputStream.CanSeek)
            {
                inputStream.Seek(0, SeekOrigin.Begin);
            }

            var output = new byte[inputStream.Length];
            int bytesRead = inputStream.Read(output, 0, output.Length);
            Debug.Assert(bytesRead == output.Length, "Bytes read from stream matches stream length");
            return output;
        }

        /// <summary>
        /// Enables processing of HTTP Web requests by a custom HttpHandler that implements the <see
cref="T:System.Web.IHttpHandler" /> interface.
        /// </summary>
        /// <param name="context">An <see cref="T:System.Web.HttpContext" /> object that provides
references to the intrinsic server objects (for example, Request, Response, Session, and Server) used
to service HTTP requests.</param>
        public void ProcessRequest(HttpContext context)
        {
            // Set up the response settings
            context.Response.ContentType = "image/jpeg";
            context.Response.Cache.SetCacheability(HttpCacheability.Public);
            context.Response.BufferOutput = false;

            // مرحله اول
            int size = 0;
            switch (context.Request.QueryString["Size"])
            {
                case "S":
                    size = 100; //100px
                    break;
                case "M":
                    size = 198; //198px
                    break;
                case "L":
                    size = 500; //500px
                    break;
            }
            byte[] changedImage;
            var id = Convert.ToInt32(context.Request.QueryString["PhotoId"]);
            byte[] sourceImage = GetImageFromDatabase(id);
            // یا
            //byte[] sourceImage = GetImageFromDisk(id.ToString(CultureInfo.InvariantCulture));

            //2 مرحله
            if (size != 0)  //غیر از حالت واقعی تصویر
            {
                changedImage = Helpers.ResizeImageFile(sourceImage, size, ImageFormat.Jpeg);
            }
            else
            {
                changedImage = (byte[])sourceImage.Clone();
            }

            // 3 مرحله
            if (changedImage == null) return;
            context.Response.AddHeader("Content-Length",
```

```
changedImage.Length.ToString(CultureInfo.InvariantCulture));
            context.Response.BinaryWrite(changedImage);
        }

        public bool IsReusable
        {
            get
            {
                return false;
            }
        }
    }
}
```

در این هندلر ما چند متد اضافه کردیم.

1- متد **GetImageFromDatabase** : این متد یک کد تصویر را گرفته و آن را از بانک اطلاعاتی بازیابی میکند. (در صورتی که تصویر در بانک ذخیره شده باشد)
2- متد **GetImageFromDisk** : این متد نام تصویر (**با فرض اینکه یک عدد می‌باشد**) را به عنوان پارامتر گرفته و آنرا بازیابی می‌کند (در صورتی که تصویر در دیسک ذخیره شده باشد.)
3- متد **StreamToByteArray** : زمانی که تصویر از فایل خوانده می‌شود به صورت Stream است این متد یک Stream را گرفته و تبدیل به یک آرایه بایتی می‌کند.

در نهایت در متد **ProcessRequest** تصویر خوانده شده با توجه به پارامترهای ورودی تغییر اندازه داده شده و در نهایت به خروجی نوشته می‌شود.

برای استفاده این هندلر، کافی است در توصیر خود به عنوان مسیر رشته ای شبیه زیر وارد نمایید:

```
PhotoHandler.ashx?PhotoId=10&Size=S
```

مانند

```
<img src='PhotoHandler.ashx?PhotoId=10&Size=S' alt='تصویر ازمایشی' />
```

پ.ن : هرچند می‌توانستیم کد هارا بهبود داده و خیلی بهینه‌تر بنویسیم اما هدف فقط اشنایی با عمل تغییر اندازه تصویر در زمان اجرا بود، امیدوارم اساتید من ببخشن.

نظرات اقای موسوی تا حدودی اعمال شد و در پست تغییراتی انجام شد.
موفق ومومید باشید

نظرات خوانندگان

نویسنده: مهدی موسوی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۳:۳۴

سلام.
چند نکته جهت بهبود کیفیت کد نوشته شده:

شما changedImage رو که byte[] هستش دارید، به چه دلیل اونو به Stream تبدیلش می‌کنید؟ چرا با استفاده از Context.Response.BinaryWrite همون آرایه changedImage رو بهش نمی‌دید؟ اینطوری دیگه به اون حلقه و همچنین تبدیل byte[] به Stream و مجددا خوندن از Stream نیازی نخواهد بود.
Request.QueryString["photoId"] رو بهتر نیست یک بار یک جا تعریف کنید و از اون در طول Function استفاده کنید؟ یا حتی Property ی جداگانه ای برای اینکار تعریف کنید و هر جا لازم بود اونو Call کنید؟
Dispose شدن Stream هایی که ایجاد کرده اید چی میشه؟
Catch کردن Exception کار صحیحی نیست وقتی قرار نیست کاری با اون Exception انجام بشه. از اون بدتر، در .NETهای 4.5 به قبل، مشکلات متعدد دیگه ای رو میتونه در پی داشته باشه.
در پیاده سازی StreamToByteArray، برخی از کدهایی که نوشته اید باید حقیقتا ASSERT باشن، نه اینکه در Runtime اونها رو چک کنید و ...

موفق باشید.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۷:۵۴

بله حق با شماست.
-1بله درسته، حواسم به چنین متدی نبود، در کد پست اعمال کردم.
-2 بله می‌توان اینکار را کرد و حتی بهبودهای بیشتری روی آن داد. اما قصد اموزش یک مبحث بود والا می‌توانستیم کل ifها و switchها را حذف کنیم.
-3 بله درسته حذفش می‌کنم.
-4 این متد نوشته من نیست و من ارجاع دادم به منبع اصلی.

موفق باشید

نویسنده: دانشجو
تاریخ: ۱۳۹۲/۰۱/۱۹ ۹:۱

دلیل استفاده از Handler برای تغیر اندازه تصویر چیه ؟ در روش قبلی شما با یک تابع این کار رو انجام دادین ، آیا تفاوتی بین این دو روش وجود داره ؟ کدوم بهتره استفاده بشه ؟ اگه ممکنه در مورد Handler توضیحی بدین . با تشکر

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۱۹ ۱۳:۱۴

سلام
دوست گلم توی پستی که شما ارجاع دادین گفته شده که برای تغییر اندازه تصاویر دو راه میشه در نظر گرفت:
-1 در زمان ثبت، تصویر تغییر اندازه داده شود.
-2 در زمان نیاز تغییر اندازه داده شود.

که در مورد هر کدوم توضیحاتی داده شده است، توی این پست با استفاده از هندلر در زمان اجرا (و استفاده از تابع تغییر اندازه تصویر) تصویر مورد نظر تغییر اندازه داده شده است.
در مورد هندلر هم لینک داده بودم توی پست.
موفق و موید باشید.

نویسنده: شاهین
تاریخ: ۱۳۹۲/۰۲/۰۵ ۲:۵۵

دوست عزیز یه مثال میذاری که چطور ازش استفاده کنم؟
هرکاری می‌کنم ارور میده میکه آدرس فایل اشتباهه

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۲/۰۵ ۱۰:۱۵

سلام
دوست من مشکلتون کجاست، لطفا کدهاتون بذارین که بهتر بتونم شمارو راهنمایی کنم.
کاری که باید بکنین
1- نوشتن متد ResizeImage
2-ایجاد یک هندلر مانند بالا
3-استفاده در نمایش تصاویر که مثالش آخر پست هست.

نویسنده: شاهین
تاریخ: ۱۳۹۲/۰۲/۰۵ ۱۵:۳۷

مرسی بابت جواب دادنتون این خط کد رو چطوری باید بنویسم؟
Original File Path + id

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۲/۰۵ ۱۶:۵۵

شما می‌تونین در پارامتر ورودی متد نام فایل و مسیر اون پاس بدین.

در خط 103 از برنامه بالا شما می‌توانید به جای کد زیر

```
byte[] sourceImage = GetImageFromDatabase(id);
```

از کدی شبیه به این استفاده نمایید

```
byte[] sourceImage = GetImageFromDisk(
     Path.Combine(context.Server.MapPath("~/uploads"),Path.GetFileName(id.ToString())));
```

که مسیر uploads یک مسیر دلخواه است که فایل‌های شما در ان قرار گرفته است. نحوه فراخوانی در این حالت به شکل زیر خواهد بود

```
PhotoHandler.ashx?PhotoId=test.jpg&Size=S
```

مانند

```
<img src='PhotoHandler.ashx?PhotoId=test.jpg&Size=S' alt='تصویر ازمایشی' />
```

البته باید تبدیل نوع ورودی در خط 102 تغییر کند(نام فایل از نوع رشته ای است)، هر چند می‌توان کدها را بهتر کرد.

پ.ن: نام فایل می‌تواند test.jpg یا هر نامی باشد که در فولدر مورد نظر وجود دارد. البته برای این کار باید تمیز کاری روی ورودی انجام شود که می‌توانید از این پست بهره بگیرید.
موفق و موید باشید

نویسنده: رضا گرمارودی
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۳:۳۳

سلام؛ هر دو مطلب شما را مطالعه کردم. همچنین لینک هندلر شما که به مایکروسافت و MSDN ارجا شده بود و در webform مشکلی ندارم اما در Mvc هندلر را نمی‌شناسد. قسمت زیر را هم به وبکانفیگ اضافه کردم ولی موثر واقع نشد. لطفا راهنمایی بفرمایید برای استفاده از هندلری همانند هندلر فوق در mvc چکار باید انجام داد.

```
<system.webServer>
      <handlers>
         <add name="ImageHandler" path="ImageHandler.ashx" verb="*" type="ImageHandler" />
      </handlers>
  </system.webServer>
```

نسخه مورد استفاده : mvc5 , visulstudio2013

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۳:۳۹

هندلر را می‌توان تبدیل به یک فیلتر و یا حتی یک ActionResult کرد. برای مثال در مورد تصاویر « اضافه کردن Watermark به تصاویر یک برنامه ASP.NET MVC در صورت لینک شدن در سایتی دیگر »

عنوان: **حذف هدرهای مربوط به وب سرور از طریق برنامه نویسی**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۱/۱۲/۱۹ ۲۳:۲۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, ASP.Net MVC, MVC, Security

در تکمیل این مطلب برای حذف هدرهای مربوط به وب سرور در برنامه‌های ASP.NET MVC از روش زیر می‌توانیم استفاده کنیم.

در حالت پیش فرض تمام پاسخهای که به سمت سرور ارسال میشوند به همراه خود یک سری جزئیات را نیز منتقل میکنند.

برای یک وب اپلیکیشن APS.NET MVC این هدرها را داریم :

Server: که توسط IIS اضافه میشود.

X-AspNet-Version: که در زمانFlush در httpresponse اضافه میشود.

X-AspNetMvc-Version: که توسط MvcHandler در System.Web.dll اضافه میشود.

X-Powered-By: این مورد نیز توسط IIS اضافه میشود.

هکرها از اینکه فریم ورک مورد استفاده چه چیزی است خوشحال خواهند شد: اگر سرور شما برای مدتی Update نشده باشد و یک آسیب پذیری امنیتی بزرگ برای ورژن فریم ورکی که استفاده می‌کنید پیدا شود در نتیجه به هکرها برای رسیدن به هدفشان

کمک کرده اید.

به علاوه این هدرها فضایی را برای تمام پاسخ‌ها در نظر میگیرند (البته در حد چندین بایت ولی در اینجا بحث برروی Optimization است).

برای حذف این هدرها باید مراحل زیر را انجام دهیم:

حذف کردن هدر Server : به Global.asax.cs رفته و رویداد Application_PreSendRequestHeaders با کد زیر را به آن اضافه کنید :

```
protected void Application_PreSendRequestHeaders(object sender, EventArgs e)
     {
         var app = sender as HttpApplication;
         if (app == null || !app.Request.IsLocal || app.Context == null)
             return;
         var headers = app.Context.Response.Headers;
         headers.Remove("Server");
     }
```

حذف کردن هدر X-AspNetMvc-Version: در فایل Global.asax.cs به رویداد Application_Start این کد زیر را اضافه کنید :

```
protected void Application_Start()
     {
         ...
         MvcHandler.DisableMvcResponseHeader = true;
         ...
     }
```

حذف کردن هدر X-AspNet-Version: به فایل Web.Config مراجعه کرده و این المنت را در داخل system.web اضافه کنید:

```
<system.web>
         ...
         <httpRuntime enableVersionHeader="false" />
         ...
</system.web>
```

حذف کردن هدر X-Powered-By: در داخل فایل Web.Config در داخل system.webServer این خطوط را اضافه کنید:

```
<system.webServer>
         ...
         <httpProtocol>
             <customHeaders>
                 <remove name="X-Powered-By" />
             </customHeaders>
         </httpProtocol>
         ...
</system.webServer>
```

با انجام مراحل فوق پاسخ‌های سرور سبک‌تر شده و در نهایت حاوی اطلاعات مهم در مورد ورژن فریم ورک نمی‌باشد.

نظرات خوانندگان

نویسنده: sysman
تاریخ: ۱۳۹۲/۰۱/۱۴ ۰:۲۲

ممنون از این مطلب مفید ولی من یک مشکلی دارم کاری که گفتید را انجام دادم وهمه چیز خوب انجام شد ولی در بخش GET modernizr-2.5.3.js باز هم اطلاعات سرور رو به من میدهد

نویسنده: علي
تاریخ: ۱۳۹۲/۰۱/۱۴ ۰:۳۶

فایل‌های استاتیک رو هم باید به موتور ASP.NET مپ کنید. تا زمانیکه مپ نباشند مستقیما توسط IIS سرو می‌شن و تنظیمات ASP.NET روی اون‌ها تاثیری نداره.

نویسنده: sysman
تاریخ: ۱۳۹۲/۰۱/۱۴ ۱۰:۴۰

نمی دونم این کاری که گفتین رو دقیقا چطور انجام بدهم ولی این کار باعث ایجاد کندی نمی‌شه؟ اگر در خود IIS تنظیمات این پست رو اعمال کنم به نتیجه میرسم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۱۴ ۱۱:۱۳

- برای فایل‌های جاوا اسکریپت توصیه من این است:
الف) اگر از Web forms استفاده می‌کنید: استفاده از Script manager (^ و ^)
ب) اگر از MVC استفاده می‌کنید: استفاده از Bundling & minification
در هر دو حالت نحوه ارائه اسکریپت‌ها تحت کنترل برنامه ASP.NET در خواهد آمد و مستقیما و بدون دخالت ASP.NET، توسط IIS توزیع نمی‌شوند.
- برای مپ کردن فایل‌های استاتیک به موتور ASP.NET می‌شود از StaticFileHandler استفاده کرد. اگر کش کردن اطلاعات استاتیک در سمت سرور فعال شود، این مساله بار اضافه‌ای را به سرور تحمیل نخواهد کرد.

```
<system.web>
   <httpHandlers>
      <add path="*.js" verb="*" type="System.Web.StaticFileHandler" />
   </httpHandlers>
```

نویسنده: یونس دوست
تاریخ: ۱۳۹۲/۰۶/۰۲ ۱۱:۳۴

سلام.من ازش استفاده کردم ولی موقع حذف Header مربوط به سرور ارور زیر رو میده:
This operation requires IIS integrated pipeline mode.
ضمنا من از iis 7.5 استفاده می‌کنم و توی Application Pools هم .Net AppPool Classic رو در حالت pipeline mode:Integrated قرار دادم هم DefaultAppPool ولی همچنان ارور رو دارم.مشکل از کجاست؟
ممنون.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۶/۰۲ ۱۱:۵۴

- محل تست کردن واقعی این کدها بر روی ویندوز سرور است و نه دیباگر و وب سرور آزمایشی ویژوال استودیو.
- شما اگر تا این حد دسترسی به IIS دارید، اصلا نیازی به کدنویسی برای حذف هدرهای وب سرور نخواهید داشت. به قسمت HTTP response headers کنسول مدیریتی مراجعه و مداخل موجود را حذف یا ویرایش کنید.
+ حذف هدر مربوط به نام Server کار ساده‌ای نیست. خیلی‌ها از روش HTTP module هم جواب نگرفته‌اند، اما با استفاده از URL Scan خود مایکروسافت قابل حذف است (این برنامه روی ویندوزهای سرور 2003 به بعد قابل نصب است). بعد از نصب به فایل C:\Windows\System32\inetsrv\urlscan\UrlScan.ini مراجعه و RemoveServerHeader را با 1 مقدار دهی کنید. ضمنا قبل از نصب URLScan تغییر زیر را هم امتحان کنید (بجای Remove از Set استفاده شده):

```
void OnPreSendRequestHeaders(object sender, EventArgs e)
{
   HttpContext.Current.Response.Headers.Set("Server", "CERN httpd");
}
```

نویسنده: پیام دات نت
تاریخ: ۱۳۹۳/۰۳/۱۵ ۲۰:۳۵

با سلام؛ ممنون از مطلب خوبتون. من تو IIS 8 Local اجرا کردم درست کار میکنه و بسیار عالیه. اما تو اجرای نهایی روی سرور هدر Server حذف نشد و از سمت سرور ارسال میشد. بعد از کلی جستجو به نتیجه زیر رسیدم.

```
<system.webServer>
    <rewrite>
      <outboundRules>
        <rule name="Remove RESPONSE_Server" >
          <match serverVariable="RESPONSE_Server" pattern=".+" />
          <action type="Rewrite" value="" />
        </rule>
      </outboundRules>
    </rewrite>
  </system.webServer>
```

نویسنده: دانش پژوه
تاریخ: ۱۳۹۳/۱۰/۰۲ ۱۱:۳۹

ممنون از مطلبتون

فقط یک نکته رو میخواستم بگم نام کوکی سشن بطور پیشفرض ASP.NET_SessionId هست و میشه از این نام فهمید که سرور asp.net هستش. از طریق کد زیر توی وب کانفیگ میتونید تغییرش بدید:

```
<system.web>
    <sessionState cookieName="foo" />
</system.web>
```

عنوان: **یک نکته کوچک راجع به تعریف Mime Type ها روی سرور**
نویسنده: ناصر فرجی
تاریخ: ۱۳۹۱/۱۲/۲۲ ۰:۲۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, IIS, CSS, Web fonts

همان طور که می‌دانید در css امکان استفاده از فونت‌های فارسی مهیاست. برای این کار کافیست با دستور زیر فونت را در فایل css خود تعریف کنیم و در صورتیکه فونت روی سیستم کاربر موجود نباشد ابتدا فونت روی سیستم دانلود شده و سپس نمایش داده می‌شود. استفاده از سه پسوند مختلف نیز برای مرورگرهای مختلف در نظر گرفته شده است تا خروجی در تمامی مرورگرها به درستی نمایش داده شود.

```
@font-face {
  src: url('Font/BYekan.eot?#') format('eot'),  /* IE6-8 */
       url('Font/BYekan.woff') format('woff'),  /* FF3.6+, IE9, Chrome6+, Saf5.1+*/
       url('Font/BYekan.ttf') format('truetype');  /* Saf3-5, Chrome4+, FF3.5, Opera 10+ */
}
```

در یک پروژه با مشکل عجیبی روبرو شدم . پروژه روی لوکال به خوبی کار می‌کرد اما بعد از آپلود روی سرور فونت‌ها بسیار دیر لود می‌شد. یعنی ابتدا تمامی اجزای صفحه لود شده و با تاخیری چند ثانیه ای فونت نمایش داده می‌شد. مرورگری که در حال کار روی آن برای تست پروژه بودم باید از پسوند woff مربوط به آن فونت استفاده می‌کرد.
بعد از تست پروژه با فایرباگ متوجه شدم که متاسفانه اصلا فونت روی سیستم دانلود نشده و خروجی 404 که مربوط به File Not Found است برگشت داده می‌شود. سعی کردم با دادن آدرس فونت در برنامه download manager فونت را دانلود کنم. اما باز هم فونت دانلود نمی‌شد! (یعنی فایل روی سرور موجود بود ولی هیچ دسترسی به آن ممکن نبود) برای دو پسوند دیگر یعنی eot و ttf هم این کار را تست کردم و متوجه شدم روی این دو پسوند مشکل خاصی وجود ندارد. با تماس با پشتیبانی هاست متوجه شدم اصلا این پسوند در Mime Type های سرور تعریف نشده است . بنابراین به صورت دستی Mime Type مورد نظر را روی سرور تعریف کردم و مشکل حل شد:

همچنین لیست کامل Mime Typesها در آدرس زیر موجود است:

لیست کامل Mime Types

نظرات خوانندگان

نویسنده: آموزش مجازی
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۵:۲۷

شما اگر این تنظیمات رو توی پروژه خودت انجام بدی ، احتیاجی نیست برای IIS حتما MimeType تعریف کنی. البته فقط قسمتهایی از این تنظیمات در مورد مساله ای که گفتی لازم هست اما من برای ایده گرفتن در مورد تنظیمات IIS 7 و به بالا در web.config که کلا مهجور مونده ، تنظیماتی که خودم معمولا می‌ذارم رو گذاشتم اینجا.

```
<system.webServer>
    <urlCompression doStaticCompression="true" doDynamicCompression="true" />
  <staticContent>
    <remove fileExtension=".js"/>
    <mimeMap fileExtension=".js" mimeType="text/javascript" />
    <mimeMap fileExtension=".mp4" mimeType="video/mp4" />
    <mimeMap fileExtension=".m4v" mimeType="video/m4v" />
    <mimeMap fileExtension=".ogg" mimeType="video/ogg" />
    <mimeMap fileExtension=".ogv" mimeType="video/ogg" />
    <mimeMap fileExtension=".webm" mimeType="video/webm" />
    <mimeMap fileExtension=".oga" mimeType="audio/ogg" />
    <mimeMap fileExtension=".spx" mimeType="audio/ogg" />
    <mimeMap fileExtension=".svg" mimeType="images/svg+xml" />
    <mimeMap fileExtension=".svgz" mimeType="images/svg+xml" />
    <mimeMap fileExtension=".eot" mimeType="application/vnd.ms-fontobject" />
    <mimeMap fileExtension=".otf" mimeType="font/otf" />
    <mimeMap fileExtension=".woff" mimeType="font/x-woff" />
  </staticContent>
```

نویسنده: آرمان فرقانی
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۶:۳۹

البته با توجه به پردازش سلسه مراتبی فایل‌های config در وب سرور احتمال دارد در برخی میزبان‌های اشتراکی برخی از قابلیت‌ها توسط مدیر سرور Lock شود.

نویسنده: ناصر فرجی
تاریخ: ۱۳۹۲/۰۱/۰۱ ۱۵:۴۱

بله. تمامی تنظیمات iis رو به نوعی میشه از طریق web.config هم انجام داد ولی برای اعمال سراسری تنظیمات بهتره از iis استفاده بشه.

عنوان: **نمایش یک فایل PDF پویا در یک iframe در ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۸ ۱۱:۲۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, PdfReport, PDF

عموما در برنامه‌های وب برای نمایش فایل‌های پویای باینری تولید شده، یا ابتدا آن‌ها را بر روی سخت دیسک ذخیره کرده و مسیر نهایی را به نحوی به کاربر نمایش می‌دهند و یا فایل را بدون ذخیره سازی، در مرورگر کاربر اصطلاحا Flush می‌کنند. حالت Flush سبب نمایش صفحه دیالوگ ذخیره سازی فایل گردیده و در همینجا Response خاتمه خواهد یافت.
برای نمونه در اینجا توسط متد inMemoryFile، یک فایل PDF در حافظه تشکیل شده و سپس به صورت یک Byte Array بازگشت داده می‌شود. در ادامه کار، این اطلاعات در مرورگر کاربر Flush خواهد شد:

```
using System.IO;
using System.Net.Mime;
using System.Web;

namespace WebApplication
{
    public class PdfHandler : IHttpHandler
    {
        private static byte[] inMemoryFile()
        {
            //تولید پویای فایل در حافظه و یا حتی خواندن از یک نمونه موجود
            return File.ReadAllBytes(@"D:\path\DynamicCrosstabSampleRpt.pdf");
        }

        public void ProcessRequest(HttpContext context)
        {
            var pdf = inMemoryFile();

            context.Response.Cache.SetCacheability(HttpCacheability.NoCache);
            context.Response.ContentType = MediaTypeNames.Application.Pdf;
            context.Response.AddHeader("Content-Length", pdf.Length.ToString());
            context.Response.AddHeader("content-disposition", "attachment;filename=test.pdf");
            context.Response.Buffer = true;
            context.Response.Clear();
            context.Response.OutputStream.Write(pdf, 0, pdf.Length);
            context.Response.OutputStream.Flush();
            context.Response.OutputStream.Close();
            context.Response.End();
        }

        public bool IsReusable
        {
            get  {  return false;  }
        }
    }
}
```

و برای نمایش آن در یک iframe در صفحه:

```
<iframe width="100%" src="PdfHandler.ashx" height="200px"></iframe>
```

نتیجه کار، نمایش صفحه دیالوگ ذخیره سازی فایل به کاربر است:

سؤال: فرض کنید Adobe reader بر روی سیستم نصب است و مرورگر با استفاده از Active-X آن می‌تواند این نوع فایل‌ها را نمایش دهد. آیا راهی وجود دارد تا بجای نمایش save popup dialog، این فایل توسط مرورگر نمایش داده شود؟

پاسخ: بلی. در کدهای فوق تنها کافی است یک سطر آن تغییر کند:

```
Response.AddHeader("content-disposition", "inline;filename=test.pdf");
```

در اینجا تنها نحوه مقدار دهی content-disposition تفاوت کرده است. حالت attachment سبب نمایش save popup dialog می‌شود و مقدار inline، فایل را در مرورگر نمایش خواهد داد.

اینبار اگر برنامه را اجرا کنیم، iframe ایی که به PdfHandler.ashx اشاره می‌کند، فایل PDF را در صفحه نمایش می‌دهد.

نظرات خوانندگان

نویسنده: آرمان
تاریخ: ۱۳۹۱/۱۲/۲۸ ۱۱:۴۸

عالی بود مهندس. در صورتی که inline باشه ولی برنامه ای برای استفاده از پی دی اف نباشه یا مرورگر مجاز به استفاده نباشه باز صفحه دانلود دیده میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۲/۲۸ ۱۲:۱۲

بله. صفحه download file فوق باز می‌شود.

نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۳/۰۵/۲۰ ۲۲:۱۴

آیا امکان عدم دریافت فایل مذکور توسط دانلود منیجرها وجود دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۰ ۲۲:۱۹

با یک پسوند دیگر که نمی‌شناسند تست کنید؛ مثلا filename=test.abc

نویسنده: افتابی
تاریخ: ۱۳۹۳/۰۵/۳۱ ۰:۱۶

متشکر؛ خوب اگه من بخوام با استفاده از PDFReport گزارش درست کنم و توی خود view نمایش بدم ، این کدها رو توی کدوم کلاس بگذارم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۳۱ ۰:۲۴

به صورت توکار لحاظ شده:

```
.Generate(data =>
                {
                    fileName = HttpUtility.UrlEncode(fileName, Encoding.UTF8);
                    data.FlushInBrowser(fileName, FlushType.Inline);
                }); // creating an in-memory PDF file
```

Inline. FlushType آن همان مطلب جاری است.

عنوان: **معماری لایه بندی نرم افزار #1**
نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۲۸ ۱۸:۵
آدرس: www.dotnettips.info
گروه‌ها: Design patterns, SoC, Separation of Concerns, ASP.Net, Domain Driven Design, DDD, SOLID Principals, C#, MVC, WPF, N-Layer Architecture

طراحی یک معماری خوب و مناسب یکی از عوامل مهم تولید یک برنامه کاربردی موفق می‌باشد. بنابراین انتخاب یک ساختار مناسب به منظور تولید برنامه کاربردی بسیار مهم و تا حدودی نیز سخت است. در اینجا یاد خواهیم گرفت که چگونه یک طراحی مناسب را انتخاب نماییم. همچنین روش‌های مختلف تولید برنامه‌های کاربردی را که مطمئنا شما هم از برخی از این روشها استفاده نمودید را بررسی می‌نماییم و مزایا و معایب آن را نیز به چالش می‌کشیم.

ضد الگو (Antipattern) – رابط کاربری هوشمند (Smart UI)

با استفاده از Visual Studio یا به اختصار VS ، می‌توانید برنامه‌های کاربردی را به راحتی تولید نمایید. طراحی رابط کاربری به آسانی عمل کشیدن و رها کردن (Drag & Drop) کنترل‌ها بر روی رابط کاربری قابل انجام است. همچنین در پشت رابط کاربری (Code Behind) تمامی عملیات مربوط به مدیریت رویدادها، دسترسی به داده ها، منطق تجاری و سایر نیازهای برنامه کاربردی، کد نویسی خواهند شد. مشکل این نوع کدنویسی بدین شرح است که تمامی نیازهای برنامه در پشت رابط کاربری قرار می‌گیرند و موجب تولید کدهای تکراری، غیر قابل تست، پیچیدگی کدنویسی و کاهش قابلیت استفاده مجدد از کد می‌گردد.

به این روش کد نویسی Smart UI می‌گویند که موجب تسهیل تولید برنامه‌های کاربردی می‌گردد. اما یکی از مشکلات عمده‌ی این روش، کاهش قابلیت نگهداری و پشتیبانی و عمر کوتاه برنامه‌های کاربردی می‌باشد که در برنامه‌های بزرگ به خوبی این مشکلات را حس خواهید کرد.

از آنجایی که تمامی برنامه نویسان مبتدی و تازه کار، از جمله من و شما در روزهای اول برنامه نویسی، به همین روش کدنویسی می‌کردیم، لزومی به ارائه مثال در رابطه با این نوع کدنویسی نمی‌بینم.

تفکیک و جدا سازی اجزای برنامه کاربردی (Separating Your Concern)

راه حل رفع مشکل Smart UI ، لایه بندی یا تفکیک اجزای برنامه از یکدیگر می‌باشد. لایه بندی برنامه می‌تواند به شکل‌های مختلفی صورت بگیرد. این کار می‌تواند توسط تفکیک کدها از طریق فضای نام (Namespace) ، پوشه بندی فایلهای حاوی کد و یا جداسازی کدها در پروژه‌های متفاوت انجام شود. در شکل زیر نمونه ای از معماری لایه بندی را برای یک برنامه کاربردی بزرگ می‌بینید.

به منظور پیاده سازی یک برنامه کاربردی لایه بندی شده و تفکیک اجزای برنامه از یکدیگر، مثالی را پیاده سازی خواهیم کرد. ممکن است در این مثال با مسائل جدید و شیوه‌های پیاده سازی جدیدی مواجه شوید که این نوع پیاده سازی برای شما قابل درک نباشد. اگر کمی صبر پیشه نمایید و این مجموعه‌ی آموزشی را پیگیری کنید، تمامی مسائل نامانوس با جزئیات بیان خواهند شد و درک آن برای شما ساده خواهد گشت. قبل از شروع این موضوع را هم به عرض برسانم که علت اصلی این نوع پیاده سازی، انعطاف پذیری بالای برنامه کاربردی، پشتیبانی و نگهداری آسان، قابلیت تست پذیری با استفاده از ابزارهای تست، پیاده سازی پروژه بصورت تیمی و تقسیم بخشهای مختلف برنامه بین اعضای تیم و سایر مزایای فوق العاده آن می‌باشد.

1- Visual Studio را باز کنید و یک Solution خالی با نام SoCPatterns.Layered ایجاد نمایید.

- جهت ایجاد Solution خالی، پس از انتخاب New Project ، از سمت چپ گزینه Other Project Types و سپس Visual Studio Solutions را انتخاب نمایید. از سمت راست گزینه Blank Solution را انتخاب کنید.

2- بر روی Solution کلیک راست نموده و از گزینه Add > New Project یک پروژه Class Library با نام SoCPatterns.Layered.Repository ایجاد کنید.

-3 با استفاده از روش فوق سه پروژه Class Library دیگر با نامهای زیر را به Solution اضافه کنید:

SoCPatterns.Layered.Model

SoCPatterns.Layered.Service

SoCPatterns.Layered.Presentation

-4 با توجه به نیاز خود یک پروژه دیگر را باید به Solution اضافه نمایید. نوع و نام پروژه در زیر لیست شده است که شما باید با توجه به نیاز خود یکی از پروژه‌های موجود در لیست را به Solution اضافه کنید.

(Windows Forms Application (SoCPatterns.Layered.WinUI

(WPF Application (SoCPatterns.Layered.WpfUI

(ASP.NET Empty Web Application (SoCPatterns.Layered.WebUI

(ASP.NET MVC 4 Web Application (SoCPatterns.Layered.MvcUI

-5 بر روی پروژه SoCPatterns.Layered.Repository کلیک راست نمایید و با انتخاب گزینه Add Reference به پروژه‌ی SoCPatterns.Layered.Model ارجاع دهید.

-6 بر روی پروژه SoCPatterns.Layered.Service کلیک راست نمایید و با انتخاب گزینه Add Reference به پروژه‌های SoCPatterns.Layered.Model و SoCPatterns.Layered.Repository ارجاع دهید.

-7 بر روی پروژه SoCPatterns.Layered.Presentation کلیک راست نمایید و با انتخاب گزینه Add Reference به پروژه‌های SoCPatterns.Layered.Model و SoCPatterns.Layered.Service ارجاع دهید.

-8 بر روی پروژه‌ی UI خود به عنوان مثال SoCPatterns.Layered.WebUI کلیک راست نمایید و با انتخاب گزینه Add Reference به پروژه‌های SoCPatterns.Layered.Service ، SoCPatterns.Layered.Repository ، SoCPatterns.Layered.Model و SoCPatterns.Layered.Presentation ارجاع دهید.

-9 بر روی پروژه‌ی UI خود به عنوان مثال SoCPatterns.Layered.WebUI کلیک راست نمایید و با انتخاب گزینه Set as StartUp Project پروژه‌ی اجرایی را مشخص کنید.

-10 بر روی Solution کلیک راست نمایید و با انتخاب گزینه Add > New Solution Folder پوشه‌های زیر را اضافه نموده و پروژه‌های مرتبط را با عمل Drag & Drop در داخل پوشه‌ی مورد نظر قرار دهید.

1. UI
§ SoCPatterns.Layered.WebUI

2. Presentation Layer
§ SoCPatterns.Layered.Presentation

3. Service Layer
§ SoCPatterns.Layered.Service

4. Domain Layer
§ SoCPatterns.Layered.Model

5. Data Layer
§ SoCPatterns.Layered.Repository

توجه داشته باشید که پوشه بندی برای مرتب سازی لایه‌ها و دسترسی راحت‌تر به آنها می‌باشد.

پیاده سازی ساختار لایه بندی برنامه به صورت کامل انجام شد. حال به پیاده سازی کدهای مربوط به هر یک از لایه‌ها و بخش‌ها می‌پردازیم و از لایه Domain شروع خواهیم کرد.

نظرات خوانندگان

نویسنده: آرمان فرقانی
تاریخ: ۱۳۹۱/۱۲/۲۸ ۲۰:۲

مباحثی از این دست بسیار مفید و ضروری است و به شدت استقبال می‌کنم از شروع این سری مقالات. البته پیش‌تر هم مطالبی از این دست در سایت ارائه شده است که امیدوارم این سری مقالات بتونه تا حدی پراکندگی مطالب مربوطه را از بین ببرد. فقط لطف بفرمایید در این سری مقالات مرز بندی مشخصی برای برخی مفاهیم در نظر داشته باشید. به عنوان مثال گاهی در یک مقاله مفهوم Repository معادل مفهوم لایه سرویس در مقاله دیگر است. یا Domain Model مرز مشخصی با View Model داشته باشد. همچنین بحث‌های خوبی مهندس نصیری عزیز در مورد عدم نیاز به ایجاد Repository در مفهوم متداول در هنگام استفاده از EF داشتند که در رفرنس‌های معتبر دیگری هم مشاهده می‌شود. لطفاً در این مورد نیز بحث بیشتری با مرز بندی مشخص داشته باشید.

نویسنده: حسن
تاریخ: ۱۳۹۱/۱۲/۲۸ ۲۲:۵

آیا صرفا تعریف چند ماژول مختلف برنامه را لایه بندی می‌کند و ضمانتی است بر لایه بندی صحیح، یا اینکه استفاده از الگوهای MVC و MVVM می‌توانند ضمانتی باشند بر جدا سازی حداقل لایه نمایشی برنامه از لایه‌های دیگر، حتی اگر تمام اجزای یک برنامه داخل یک اسمبلی اصلی قرار گرفته باشند؟

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۲۹ ۰:۵۳

این سری مقالات جمع بندی کامل معماری لایه بندی نرم افزار است. پس از پایان مقالات یک پروژه کامل رو با معماری منتخب پیاده سازی میکنم تا تمامی شک و شبهات برطرف بشه. در مورد مرزبندی لایه‌ها هم صحبت می‌کنم و مفهوم هر کدام را دقیقا توضیح میدم.

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۲۹ ۰:۵۹

اگر مقاله فوق رو با دقت بخونید متوجه میشید که MVC و MVVM در لایه UI پیاده سازی میشن. البته در MVC لایه Model رو به Domain Model و Repository در برخی مواقع لایه Controller رو در لایه Presentation قرار میدن. در MVVM نیز لایه Model در Domain Model و Repository و لایه View Model نیز در لایه Presentation قرار میگیره. همچنین View Modelها نیز در لایه Service قرار میگیرن.
در مورد ماژول بندی هم اگر در مقاله خونده باشید میتونید لایه‌ها رو از طریق پوشه ها، فضای نام و یا پروژه‌ها از هم جدا کنید

نویسنده: حسن
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۰:۱۴

شما در مطلبتون با ضدالگو شروع کردید و عنوان کردید که روش code behind یک سری مشکلاتی رو داره. سؤال من هم این بود که آیا صرفا تعریف چند ماژول جدید می‌تواند ضمانتی باشد بر رفع مشکل code behind یا اینکه با این ماژول‌ها هم نهایتا همان مشکل قبل پابرجا است یا می‌تواند پابرجا باشد.

ضمن اینکه تعریف شما از لایه دقیقا چی هست؟ به نظر فقط تعریف یک اسمبلی در اینجا لایه نام گرفته.

نویسنده: آرمان فرقانی
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۱:۵۸

صحبت شما کاملاً صحیح است و صرفاً با ماژولار کردن به معماری چند لایه نمی‌رسیم. اما نویسنده مقاله نیز چنین نگفته و در پایان مقاله بحث پایان **ساختار** چند لایه است و نه پایان پروژه. این قسمت اول این سری مقالات است و قطعاً در هنگام پیاده سازی کدهای هر لایه مباحثی مطرح خواهد شد تا تضمین مفهوم مورد نظر شما باشد.

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۲:۳۸

با تشکر از دوست عزیزم جناب آقای آرمان فرقانی با توضیحی که دادند.
یکی از دلایل این شیوه کد نویسی امکان تست نویسی برای هر یک از لایه‌ها و همچنین استقلال لایه‌ها از هم دیگه هست که هر لایه بتونه بدون وجود لایه‌ی دیگه تست بشه. ماژولار کردنه ممکنه مشکل Smart UI رو حل کنه و ممکنه حل نکنه. بستگی به شیوه کد نویسی داره.

نویسنده: بهروز
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۳:۱۱

وقتی نظرات زیر مطلب شما رو میخونم میفهمم که نیاز به این سری آموزشی که دارید ارائه میدید چقدر زیاد احساس میشه فقط می‌خواستم بگم بر سر این مبحثی که دارید ارائه می‌دید اختلاف بین علما زیاد است!(حتی در عمل و در شرکت‌های نرم افزاری که تا به حال دیدم چه برسد در سطح آموزش...)
امیدوارم این حساسیت رو در نظر بگیرید و همه ما پس از مطالعه این سری آموزشی به فهم مشترک و یکسانی در مورد مفاهیم موجود برسیم
فکر می‌کنم وجود یک پروژه برای دست یافتن به این هدف هم ضروری باشد
باز هم تشکر

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۲۹ ۱۳:۵۹

من هم وقتی کار بر روی این معماری رو شروع کردم با مشکلات زیادی روبرو بودم و خیلی از مسائل برای من هم نامانوس و غیر قابل هضم بود. ولی بعد از اینکه چند پروژه نرم افزاری رو با این معماری پیاده سازی کردم فهم بیشتری نسبت به اون پیدا کردم و خیلی از مشکلات موجود رو با دقت بالا و با در نظر گرفتن تمامی الگوها رفع کردم. امیدوارم این حس مشترک بوجود بیاد. ولی دلیل اصلی ایجاد تکنولوژی‌ها و معماری‌های جدید اختلاف نظر بین علماست. این اختلاف نظر در اکثر مواقع میتونه مفید باشه.
ممنون دوست عزیز

نویسنده: مسعود مشهدی
تاریخ: ۱۳۹۲/۰۱/۰۴ ۱۸:۳۳

با سلام
بابت مطالبتون سپاسگذارم
همون طور که خودتون گفتید نظرات و شیوه‌های متفاوتی در نوع لایه بندی‌ها وجود داره.
در مقام مقایسه لایه بندی زیر چه وجه اشتراک و تفاوتی با لایه بندی شما داره.

```
Application.Web
Application.Manager

Application.DAL

Application.DTO

Application.Core
```

Application.Common

با تشکر

عنوان: معماری لایه بندی نرم افزار #2
نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۳۰ ۱:۴۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, C#, Design patterns, MVC, WPF, SoC, Separation of Concerns, Domain Driven Design, DDD, SOLID Principals, N-Layer Architecture

Business Layer یا Domain Model

پیاده سازی را از منطق تجاری یا Business Logic آغاز می‌کنیم. در روش کد نویسی Smart UI ، منطق تجاری در Code Behind قرار می‌گرفت اما در روش لایه بندی، منطق تجاری و روابط بین داده‌ها در Domain Model طراحی و پیاده سازی می‌شوند. در مطالب بعدی راجع به Domain Model و الگوهای پیاده سازی آن بیشتر صحبت خواهم کرد اما بصورت خلاصه این لایه یک مدل مفهومی از سیستم می‌باشد که شامل تمامی موجودیت‌ها و روابط بین آنهاست.

الگوی Domain Model جهت سازماندهی پیچیدگی‌های موجود در منطق تجاری و روابط بین موجودیت‌ها طراحی شده است.

شکل زیر مدلی را نشان می‌دهد که می‌خواهیم آن را پیاده سازی نماییم. کلاس Product موجودیتی برای ارائه محصولات یک فروشگاه می‌باشد. کلاس Price جهت تشخیص قیمت محصول، میزان سود و تخفیف محصول و همچنین استراتژی‌های تخفیف با توجه به منطق تجاری سیستم می‌باشد. در این استراتژی همکاران تجاری از مشتریان عادی تفکیک شده اند.

Domain Model را در پروژه SoCPatterns.Layered.Model پیاده سازی می‌کنیم. بنابراین به این پروژه یک Interface به نام IDiscountStrategy را با کد زیر اضافه نمایید:

```
public interface IDiscountStrategy
{
    decimal ApplyExtraDiscountsTo(decimal originalSalePrice);
}
```

علت این نوع نامگذاری Interface فوق، انطباق آن با الگوی Strategy Design Pattern می‌باشد که در مطالب بعدی در مورد این الگو بیشتر صحبت خواهم کرد. استفاده از این الگو نیز به این دلیل بود که این الگو مختص الگوریتم هایی است که در زمان اجرا قابل انتخاب و تغییر خواهند بود.

توجه داشته باشید که معمولا نام Design Pattern انتخاب شده برای پیاده سازی کلاس را بصورت پسوند در انتهای نام کلاس ذکر می‌کنند تا با یک نگاه، برنامه نویس بتواند الگوی مورد نظر را تشخیص دهد و مجبور به بررسی کد نباشد. البته به دلیل تشابه برخی از الگوها، امکان تشخیص الگو، در پاره ای از موارد وجود ندارد و یا به سختی امکان پذیر است.

الگوی Strategy یک الگوریتم را قادر می‌سازد تا در داخل یک کلاس کپسوله شود و در زمان اجرا به منظور تغییر رفتار شی، بین رفتارهای مختلف سوئیچ شود.

حال باید دو کلاس به منظور پیاده سازی روال تخفیف ایجاد کنیم. ابتدا کلاسی با نام TradeDiscountStrategy را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public class TradeDiscountStrategy : IDiscountStrategy
{
    public decimal ApplyExtraDiscountsTo(decimal originalSalePrice)
    {
        return originalSalePrice * 0.95M;
    }
}
```

سپس با توجه به الگوی Null Object کلاسی با نام NullDiscountStrategy را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public class NullDiscountStrategy : IDiscountStrategy
{
    public decimal ApplyExtraDiscountsTo(decimal originalSalePrice)
    {
        return originalSalePrice;
    }
}
```

از الگوی Null Object زمانی استفاده می‌شود که نمی‌خواهید و یا در برخی مواقع نمی‌توانید یک نمونه (Instance) معتبر را برای یک کلاس ایجاد نمایید و همچنین مایل نیستید که مقدار Null را برای یک نمونه از کلاس برگردانید. در مباحث بعدی با جزئیات بیشتری در مورد الگوها صحبت خواهم کرد.

با توجه به استراتژی‌های تخفیف کلاس Price را ایجاد کنید. کلاسی با نام Price را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public class Price
{
    private IDiscountStrategy _discountStrategy = new NullDiscountStrategy();
    private decimal _rrp;
    private decimal _sellingPrice;
    public Price(decimal rrp, decimal sellingPrice)
    {
        _rrp = rrp;
        _sellingPrice = sellingPrice;
    }
    public void SetDiscountStrategyTo(IDiscountStrategy discountStrategy)
    {
        _discountStrategy = discountStrategy;
    }
    public decimal SellingPrice
    {
        get { return _discountStrategy.ApplyExtraDiscountsTo(_sellingPrice); }
```

```
    }
    public decimal Rrp
    {
        get { return _rrp; }
    }
    public decimal Discount
    {
        get {
            if (Rrp > SellingPrice)
                return (Rrp - SellingPrice);
            else
                return 0;
        }
    }
    public decimal Savings
    {
        get{
            if (Rrp > SellingPrice)
                return 1 - (SellingPrice / Rrp);
            else
                return 0;
        }
    }
}
```

کلاس Price از نوعی Dependency Injection به نام Setter Injection در متد SetDiscountStrategyTo استفاده نموده است که استراتژی تخفیف را برای کالا مشخص می‌نماید. نوع دیگری از Dependency Injection با نام Constructor Injection وجود دارد که در مباحث بعدی در مورد آن بیشتر صحبت خواهم کرد.

جهت تکمیل لایه Model ، کلاس Product را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

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

موجودیت‌های تجاری ایجاد شدند اما باید روشی اتخاذ نمایید تا لایه Model نسبت به منبع داده ای بصورت مستقل عمل نماید. به سرویسی نیاز دارید که به کلاینت‌ها اجازه بدهد تا با لایه مدل در اتباط باشند و محصولات مورد نظر خود را با توجه به تخفیف اعمال شده برای رابط کاربری برگردانند. برای اینکه کلاینت‌ها قادر باشند تا نوع تخفیف را مشخص نمایند، باید یک نوع شمارشی ایجاد کنید که به عنوان پارامتر ورودی متد سرویس استفاده شود. بنابراین نوع شمارشی CustomerType را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public enum CustomerType
{
    Standard = 0,
    Trade = 1
}
```

برای اینکه تشخیص دهیم کدام یک از استراتژی‌های تخفیف باید بر روی قیمت محصول اعمال گردد، نیاز داریم کلاسی را ایجاد کنیم تا با توجه به CustomerType تخفیف مورد نظر را اعمال نماید. کلاسی با نام DiscountFactory را با کد زیر ایجاد نمایید:

```
public static class DiscountFactory
{
```

```
    public static IDiscountStrategy GetDiscountStrategyFor
        (CustomerType customerType)
    {
        switch (customerType)
        {
            case CustomerType.Trade:
                return new TradeDiscountStrategy();
            default:
                return new NullDiscountStrategy();
        }
    }
}
```

در طراحی کلاس فوق از الگوی Factory استفاده شده است. این الگو یک کلاس را قادر می‌سازد تا با توجه به شرایط، یک شی معتبر را از یک کلاس ایجاد نماید. همانند الگوهای قبلی، در مورد این الگو نیز در مباحث بعدی بیشتر صحبت خواهم کرد.

لایه‌ی سرویس با برقراری ارتباط با منبع داده ای، داده‌های مورد نیاز خود را بر می‌گرداند. برای این منظور از الگوی Repository استفاده می‌کنیم. از آنجایی که لایه Model باید مستقل از منبع داده ای عمل کند و نیازی به شناسایی نوع منبع داده ای ندارد، جهت پیاده سازی الگوی Repository از Interface استفاده می‌شود. یک Interface به نام IProductRepository را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public interface IProductRepository
{
    IList<Product> FindAll();
}
```

الگوی Repository به عنوان یک مجموعه‌ی در حافظه (In-Memory Collection) یا انباره ای از موجودیت‌های تجاری عمل می‌کند که نسبت به زیر بنای ساختاری منبع داده ای کاملا مستقل می‌باشد.

کلاس سرویس باید بتواند استراتژی تخفیف را بر روی مجموعه ای از محصولات اعمال نماید. برای این منظور باید یک Collection سفارشی ایجاد نماییم. اما من ترجیح می‌دهم از Extension Methods برای اعمال تخفیف بر روی محصولات استفاده کنم. بنابراین کلاسی به نام ProductListExtensionMethods را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public static class ProductListExtensionMethods
{
    public static void Apply(this IList<Product> products,
                                     IDiscountStrategy discountStrategy)
    {
        foreach (Product p in products)
        {
            p.Price.SetDiscountStrategyTo(discountStrategy);
        }
    }
}
```

الگوی Separated Interface تضمین می‌کند که کلاینت از پیاده سازی واقعی کاملا نامطلع می‌باشد و می‌تواند برنامه نویس را به سمت Abstraction و Dependency Inversion به جای پیاده سازی واقعی سوق دهد.

حال باید کلاس Service را ایجاد کنیم تا از طریق این کلاس، کلاینت با لایه Model در ارتباط باشد. کلاسی به نام ProductService را با کد زیر به پروژه SoCPatterns.Layered.Model اضافه کنید:

```
public class ProductService
{
    private IProductRepository _productRepository;
    public ProductService(IProductRepository productRepository)
    {
        _productRepository = productRepository;
    }
    public IList<Product> GetAllProductsFor(CustomerType customerType)
    {
        IDiscountStrategy discountStrategy =
                                DiscountFactory.GetDiscountStrategyFor(customerType);
        IList<Product> products = _productRepository.FindAll();
        products.Apply(discountStrategy);
        return products;
    }
}
```

در اینجا کدنویسی منطق تجاری در Domain Model به پایان رسیده است. همانطور که گفته شد، لایه‌ی Business یا همان Domain Model به هیچ منبع داده ای خاصی وابسته نیست و به جای پیاده سازی کدهای منبع داده ای، از Interface ها به منظور برقراری ارتباط با پایگاه داده استفاده شده است. پیاده سازی کدهای منبع داده ای را به لایه‌ی Repository واگذار نمودیم که در بخش‌های بعدی نحوه پیاده سازی آن را مشاهده خواهید کرد. این امر موجب می‌شود تا لایه Model درگیر پیچیدگی‌ها و کد نویسی‌های منبع داده ای نشود و بتواند به صورت مستقل و فارغ از بخشهای مختلف برنامه تست شود. لایه بعدی که می‌خواهیم کد نویسی آن را آغاز کنیم، لایه‌ی Service می‌باشد.

در کد نویسی‌های فوق از الگوهای طراحی (Design Patterns) متعددی استفاده شده است که به صورت مختصر در مورد آنها صحبت کردم. اصلا جای نگرانی نیست، چون در مباحث بعدی به صورت مفصل در مورد آنها صحبت خواهم کرد. در ضمن، ممکن است روال یادگیری و آموزش بسیار نامفهوم باشد که برای فهم بیشتر موضوع، باید کدها را بصورت کامل تست نموده و مثالهایی را پیاده سازی نمایید.

نظرات خوانندگان

نویسنده: سینا کردی
تاریخ: ۱۳۹۱/۱۲/۳۰ ۴:۱۰

سلام
ممنون از شما این بخش هم کامل و زیبا بود
ولی کمی فشرده بود
لطفا اگر ممکن هست در مورد معماری‌ها و الگوها و بهترین‌های آنها کمی توضیح دهید یا منبعی معرفی کنید تا این الگوها و معماری برای ما بیشتر مفهوم بشه
من در این زمینه تازه کارم و از شما میخواهم که من رو راهنمایی کنید که چه مقدماتی در این زمینه‌ها نیاز دارم
باز هم ممنون.

نویسنده: علي
تاریخ: ۱۳۹۱/۱۲/۳۰ ۹:۱۲

در همین سایت مباحث الگوهای طراحی و Refactoring مفید هستند.

و یا الگوهای طراحی Agile رو هم در اینجا می‌تونید پیگیری کنید.

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۳۰ ۱۱:۳۸

فشردگی این مباحث بخاطر این بود که میخواستم فعلا یک نمونه پروژه رو آموزش بودم تا یک شمای کلی از کاری که می‌خواهیم انجام بدیم رو ببینید. در مباحث بعدی این مباحث رو بازتر می‌کنم. خود من برای مطالعه و جمع بندی این مباحث منابع زیادی رو مطالعه کردم. واقعا برای بعضی مباحث نمیشه به یک منبع اکتفا کرد.

نویسنده: محسن.د
تاریخ: ۱۳۹۱/۱۲/۳۰ ۱۷:۱

بسیار عالی

آیا فراخوانی مستقیم تابع SetDiscountStrategyTo کلاس Price در تابع الحاقی Apply از نظر کپسوله سازی مورد اشکال نیست ؟ بهتر نیست که برای خود کلاس Product یک تابع پیاده سازی کنیم که در درون خودش تابع Price.SetDiscountStrategyTo را فراخوانی کند و به این شکل کلاس‌های بیرونی رو از تغییرات درونی کلاس Product مستقل کنیم ؟

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۱/۱۲/۳۰ ۱۸:۱

دوست عزیزم. متد Apply یک Extension Method برای IList<Product> است. اگر این متد تعریف نمی‌شد شما باید در کلاس سرویس حلقه foreach رو قرار می‌دادید. البته با این حال در قسمت هایی از طراحی کلاسها که الگوهای طراحی را زیر سوال نمی‌برد و تست پذیری را دچار مشکل نمی‌کند، طراحی سلیقه ای است. مقاله من هم آیه‌ی نازل شده نیست که دستخوش تغییرات نشود. شما می‌توانید با سلیقه و دید فنی خود تغییرات مورد نظر رو اعمال کنید. ولی اگر نظر من را بخواهید این طراحی مناسب‌تر است.

نویسنده: رضا عرب
تاریخ: ۱۳۹۲/۰۱/۰۹ ۱۴:۴۵

خسته نباشید، واقعا ممنونم آقای خوشبخت، لطفا به نگارش این دست مطالب مرتبط با طراحی ادامه دهید، زمینه بکریه که کمتر عملی به آن پرداخته شده و این نوع نگارش شما فراتر از یک معرفیه که واقعا جای تشکر داره.

نویسنده: f.tahan36
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۷:۱۰

با سلام

تفاوت factory با design factory در چیست؟ (با مثال کد)

و virtual کردن یک تابع معمولی با virtual کردن تابع سازنده چه تفاوتی دارد؟

با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۳۰ ۰:۴۰

از همون رندهایی هستی که تمرین کلاسیت رو آوردی اینجا؟! :)

Service Layer

نقش لایه‌ی سرویس این است که به عنوان یک مدخل ورودی به برنامه کاربردی عمل کند. در برخی مواقع این لایه را به عنوان لایه‌ی Facade نیز می‌شناسند. این لایه، داده‌ها را در قالب یک نوع داده ای قوی (Strongly Typed) به نام View Model ، برای لایه‌ی Presentation فراهم می‌کند. کلاس View Model یک Strongly Typed محسوب می‌شود که نماهای خاصی از داده‌ها را که متفاوت از دید یا نمای تجاری آن است، بصورت بهینه ارائه می‌نماید. در مورد الگوی View Model در مباحث بعدی بیشتر صحبت خواهم کرد.

الگوی Facade یک Interface ساده را به منظور کنترل دسترسی به مجموعه ای از Interface ها و زیر سیستم‌های پیچیده ارائه می‌کند. در مباحث بعدی در مورد آن بیشتر صحبت خواهم کرد.

کلاسی با نام ProductViewModel را با کد زیر به پروژه SoCPatterns.Layered.Service اضافه کنید:

```
public class ProductViewModel
{
    Public int ProductId {get; set;}
    public string Name { get; set; }
    public string Rrp { get; set; }
    public string SellingPrice { get; set; }
    public string Discount { get; set; }
    public string Savings { get; set; }
}
```

برای اینکه کلاینت با لایه‌ی سرویس در تعامل باشد باید از الگوی Request/Response Message استفاده کنیم. بخش Request توسط کلاینت تغذیه می‌شود و پارامترهای مورد نیاز را فراهم می‌کند. کلاسی با نام ProductListRequest را با کد زیر به پروژه SoCPatterns.Layered.Service اضافه کنید:

```
using SoCPatterns.Layered.Model;

namespace SoCPatterns.Layered.Service
{
    public class ProductListRequest
    {
        public CustomerType CustomerType { get; set; }
    }
}
```

در شی Response نیز بررسی می‌کنیم که درخواست به درستی انجام شده باشد، داده‌های مورد نیاز را برای کلاینت فراهم می‌کنیم و همچنین در صورت عدم اجرای صحیح درخواست، پیام مناسب را به کلاینت ارسال می‌نماییم. کلاسی با نام ProductListResponse را با کد زیر به پروژه SoCPatterns.Layered.Service اضافه کنید:

```
public class ProductListResponse
{
    public bool Success { get; set; }
```

```
    public string Message { get; set; }
    public IList<ProductViewModel> Products { get; set; }
}
```

به منظور تبدیل موجودیت Product به ProductViewModel ، به دو متد نیاز داریم، یکی برای تبدیل یک Product و دیگری برای تبدیل لیستی از Product . شما می‌توانید این دو متد را به کلاس Product موجود در Domain Model اضافه نمایید، اما این متدها نیاز واقعی منطق تجاری نمی‌باشند. بنابراین بهترین انتخاب، استفاده از Extension Method ها می‌باشد که باید برای کلاس Product و در لایه‌ی سرویس ایجاد نمایید. کلاسی با نام ProductMapperExtensionMethods را با کد زیر به پروژه SoCPatterns.Layered.Service اضافه کنید:

```
public static class ProductMapperExtensionMethods
{
    public static ProductViewModel ConvertToProductViewModel(this Model.Product product)
    {
        ProductViewModel productViewModel = new ProductViewModel();
        productViewModel.ProductId = product.Id;
        productViewModel.Name = product.Name;
        productViewModel.RRP = String.Format("{0:C}", product.Price.RRP);
        productViewModel.SellingPrice = String.Format("{0:C}", product.Price.SellingPrice);
        if (product.Price.Discount > 0)
            productViewModel.Discount = String.Format("{0:C}", product.Price.Discount);
        if (product.Price.Savings < 1 && product.Price.Savings > 0)
            productViewModel.Savings = product.Price.Savings.ToString("#%");
        return productViewModel;
    }
    public static IList<ProductViewModel> ConvertToProductListViewModel(
        this IList<Model.Product> products)
    {
        IList<ProductViewModel> productViewModels = new List<ProductViewModel>();
        foreach(Model.Product p in products)
        {
            productViewModels.Add(p.ConvertToProductViewModel());
        }
        return productViewModels;
    }
}
```

حال کلاس ProductService را جهت تعامل با کلاس سرویس موجود در Domain Model و به منظور برگرداندن لیستی از محصولات و تبدیل آن به لیستی از ProductViewModel ، ایجاد می‌نماییم. کلاسی با نام ProductService را با کد زیر به پروژه SoCPatterns.Layered.Service اضافه کنید:

```
public class ProductService
{
    private Model.ProductService _productService;
    public ProductService(Model.ProductService ProductService)
    {
        _productService = ProductService;
    }
    public ProductListResponse GetAllProductsFor(
        ProductListRequest productListRequest)
    {
        ProductListResponse productListResponse = new ProductListResponse();
        try
        {
            IList<Model.Product> productEntities =
                _productService.GetAllProductsFor(productListRequest.CustomerType);
            productListResponse.Products = productEntities.ConvertToProductListViewModel();
            productListResponse.Success = true;
        }
        catch (Exception ex)
        {
            // Log the exception…
            productListResponse.Success = false;
            // Return a friendly error message
```

```
            productListResponse.Message = ex.Message;
        }
        return productListResponse;
    }
}
```

کلاس Service تمامی خطاها را دریافت نموده و پس از مدیریت خطا، پیغامی مناسب را به کلاینت ارسال می‌کند. همچنین این لایه محل مناسبی برای Log کردن خطاها می‌باشد. در اینجا کد نویسی لایه سرویس به پایان رسید و در ادامه به کدنویسی Data Layer می‌پردازیم.

Data Layer

برای ذخیره سازی محصولات، یک بانک اطلاعاتی با نام Shop01 ایجاد کنید که شامل جدولی به نام Product با ساختار زیر باشد:

Product

Column Name	Data Type	Allow Nulls
ProductId	int	☐
ProductName	nvarchar(50)	☐
Rrp	smallmoney	☐
SellingPrice	smallmoney	☐
		☐

برای اینکه کدهای بانک اطلاعاتی را سریعتر تولید کنیم از روش Linq to SQL در Data Layer استفاده می‌کنم. برای این منظور یک Data Context برای Linq to SQL به این لایه اضافه می‌کنیم. بر روی پروژه SoCPatterns.Layered.Repository کلیک راست نمایید و گزینه Add > New Item را انتخاب کنید. در پنجره ظاهر شده و از سمت چپ گزینه Data و سپس از سمت راست گزینه Linq to SQL Classes را انتخاب نموده و نام آن را Shop.dbml تعیین نمایید.

از طریق پنجره Server Explorer به پایگاه داده مورد نظر متصل شوید و با عمل Drag & Drop جدول Product را به بخش Design کشیده و رها نمایید.

اگر به یاد داشته باشید، در لایه Model برای برقراری ارتباط با پایگاه داده از یک Interface به نام IProductRepository استفاده نمودیم. حال باید این Interface را پیاده سازی نماییم. کلاسی با نام ProductRepository را با کد زیر به پروژه SoCPatterns.Layered.Repository اضافه کنید:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using SoCPatterns.Layered.Model;

namespace SoCPatterns.Layered.Repository
{
    public class ProductRepository : IProductRepository
    {
        public IList<Model.Product> FindAll()
        {
            var products = from p in new ShopDataContext().Products
                           select new Model.Product
                           {
                               Id = p.ProductId,
                               Name = p.ProductName,
                               Price = new Model.Price(p.Rrp, p.SellingPrice)
                           };
            return products.ToList();
        }
    }
}
```

در متد FindAll ، با استفاده از دستورات Linq to SQL ، لیست تمامی محصولات را برگرداندیم. کدنویسی لایه‌ی Data هم به پایان رسید و در ادامه به کدنویسی لایه‌ی Presentation و UI می‌پردازیم.

Presentation Layer

به منظور جداسازی منطق نمایش (Presentation) از رابط کاربری (User Interface) ، از الگوی Model View Presenter یا همان MVP استفاده می‌کنیم که در مباحث بعدی با جزئیات بیشتری در مورد آن صحبت خواهم کرد. یک Interface با نام IProductListView را با کد زیر به پروژه SoCPatterns.Layered.Presentation اضافه کنید:

```
using SoCPatterns.Layered.Service;

public interface IProductListView
{
    void Display(IList<ProductViewModel> Products);
    Model.CustomerType CustomerType { get; }
    string ErrorMessage { set; }
}
```

این Interface توسط Web Form های ASP.NET و یا Win Form ها باید پیاده سازی شوند. کار با Interface ها موجب می‌شود تا تست View ها به راحتی انجام شوند. کلاسی با نام ProductListPresenter را با کد زیر به پروژه SoCPatterns.Layered.Presentation اضافه کنید:

```
using SoCPatterns.Layered.Service;

namespace SoCPatterns.Layered.Presentation
{
    public class ProductListPresenter
    {
        private IProductListView _productListView;
        private Service.ProductService _productService;
        public ProductListPresenter(IProductListView ProductListView,
            Service.ProductService ProductService)
        {
            _productService = ProductService;
            _productListView = ProductListView;
        }
        public void Display()
        {
            ProductListRequest productListRequest = new ProductListRequest();
            productListRequest.CustomerType = _productListView.CustomerType;
            ProductListResponse productResponse =
                _productService.GetAllProductsFor(productListRequest);
            if (productResponse.Success)
            {
                _productListView.Display(productResponse.Products);
            }
            else
            {
                _productListView.ErrorMessage = productResponse.Message;
            }
        }
    }
}
```

کلاس Presenter وظیفه‌ی واکشی داده ها، مدیریت رویدادها و بروزرسانی UI را دارد. در اینجا کدنویسی لایه‌ی Presentation به پایان رسیده است. از مزایای وجود لایه‌ی Presentation این است که تست نویسی مربوط به نمایش داده‌ها و تعامل بین کاربر و سیستم به سهولت انجام می‌شود بدون آنکه نگران دشواری Unit Test نویسی Web Form ها باشید. حال می‌توانید کد نویسی مربوط به UI را انجام دهید که در ادامه به کد نویسی در Win Forms و Web Forms خواهیم پرداخت.

نظرات خوانندگان

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۰۲ ۱۸:۲۹

ممنون از زحمات شما.
چند سؤال و نظر:
- با تعریف الگوی مخزن به چه مزیتی دست پیدا کردید؟ برای مثال آیا هدف این است که کدهای پیاده سازی آن، با توجه به وجود اینترفیس تعریف شده، شاید روزی با مثلا NHibernate تعویض شود؟ در عمل متاسفانه حتی پیاده سازی LINQ این‌ها هم متفاوت است و من تابحال در عمل ندیدم که ORM یک پروژه بزرگ رو عوض کنند. یعنی تا آخر و تا روزی که پروژه زنده است با همان انتخاب اول سر می‌کنند. یعنی شاید بهتر باشه قسمت مخزن و همچنین سرویس یکی بشن.
- چرا لایه سرویس تعریف شده از یک یا چند اینترفیس مشتق نمی‌شود؟ اینطوری تهیه تست برای اون ساده‌تر میشه. همچنین پیاده سازی‌ها هم وابسته به یک کلاس خاص نمی‌شن چون از اینترفیس دارن استفاده می‌کنند.
- این اشیاء Request و Response هم در عمل به نظر نوعی ViewModel هستند. درسته؟ اگر اینطوره بهتر یک مفهوم کلی دنبال بشه تا سردرگمی‌ها رو کمتر کنه.

یک سری نکته جانبی هم هست که می‌تونه برای تکمیل بحث جالب باشه:
- مثلا الگوی Context per request بجای نوشتن new ShopDataContext بهتر استفاده بشه تا برنامه در طی یک درخواست در یک تراکنش و اتصال کار کنه.
- در مورد try/catch و استفاده از اون بحث زیاد هست. خیلی‌ها توصیه می‌کنن که یا اصلا استفاده نکنید یا استفاده از اون‌ها رو به بالاترین لایه برنامه موکول کنید تا این وسط کرش یک قسمت و بروز استثناء در اون، از ادامه انتشار صدمه به قسمت‌های بعدی جلوگیری کنه.

نویسنده: میثم خوشبخت
تاریخ: ۱۳۹۲/۰۱/۰۲ ۲۳:۳۵

محسن عزیز. از شما ممنونم که به نکته‌های ظریفی اشاره کردید.

در سری مقالات اولیه فقط دارم یک دید کلی به کسایی میدم که تازه دارن با این مفاهیم آشنا میشن. این پروژه اولیه دستخوش تغییرات زیادی میشه. در واقع محصول نهایی این مجموعه مقالات بر پایه همین نوع لایه بندی ولی بادید و طراحی مناسب‌تر خواهد بود.
در مورد ORM هم من با چند Application سروکار داشتم که در روال توسعه بخش‌های جدید رو بنا به دلایلی با ORM یا DB متفاوتی توسعه داده اند. غیر از این موضوع، حتی بخشهایی از مدل، سرویس و یا مخزن رو در پروژه‌های دیگری استفاده کرده اند. همچنین برخی از نکات مربوط به تفکیک لایه‌ها به منظور تست پذیری راحت‌تر رو هم در نظر بگیرید.
در مورد اشیا Request و Response هم باید خدمتتان عرض کنم که برای درخواست و پاسخ به درخواست استفاده می‌شوند که چون پروژه ای که مثال زدم کوچک بوده ممکنه کاملا درکش نکرده باشید. ما کلاسهای Request و Response متعددی در پروژه داریم که ممکنه خیلی از اونها فقط از یک View Model استفاده کنن ولی پارامترهای ارسالی یا دریافتی آنها متفاوت باشد.
در مورد try...catch هم من با شما کاملا موافقم. به دلیل هزینه ای که دارد باید در آخرین سطح قرار بگیرد. در این مورد ما میتونیم اونو به Presentation و یا در MVC به Controller منتقل کنیم.
در مورد DbContext هم هنوز الگویی رو معرفی نکردم. در واقع هنوز وارد جزئیات لایه‌ی Data نشدم. در مورد اون اگه اجازه بدی بعدا صحبت میکنم.

نویسنده: ایلیا
تاریخ: ۱۳۹۲/۰۱/۰۳ ۰:۴۳

آقای خوشبخت خداقوت.

مرسی از مطالب خوبتون.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۱/۰۳ ۰:۴۸

لطفا برای اینکه نظرات حالت فنی‌تر و غنای بیشتری پیدا کنند، از ارسال پیام‌های تشکر خودداری کنید. برای ابراز احساسات و همچنین تشکر، لطفا از گزینه رای دادن به هر مطلب که ذیل آن قرار دارد استفاده کنید.
این مطلب تا این لحظه 76 بار دیده شده، اما فقط 4 رای دارد. لطفا برای ابراز تشکر، امتیاز بدهید. ممنون.

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱:۰

- من در عمل تفاوتی بین لایه مخزن و سرویس شما مشاهده نمی‌کنم. یعنی لایه مخزن داره GetAll می‌کنه، بعد لایه سرویس هم داره همون رو به یک شکل دیگری بر می‌گردونه. این تکرار کد نیست؟ این دو یکی نیستند؟

عموما در منابع لایه مخزن رو به صورت روکشی برای دستورات مثلا EF یا LINQ to SQL معرفی می‌کنند. فرضشون هم این است که این روش ما رو از تماس مستقیم با ORM برحذر می‌داره (شاید فکر می‌کنند ایدز می‌گیرند اگر مستقیم کار کنند!). ولی عرض کردم این روکش در واقعیت فقط شاید با EF یا L2S قابل تعویض باشه نه با ORMهای دیگر با روش‌های مختلف و بیشتر یک تصور واهی هست که جنبه عملی نداره. بیشتر تئوری هست بدون پایه تجربه دنیای واقعی. ضمن اینکه این روکش باعث میشه نتونید از خیلی از امکانات ORM مورد استفاده درست استفاده کنید. مثلا ترکیب کوئری‌ها یا روش‌های به تاخیر افتاده و امثال این.

- پس در عمل شما Request ViewModel و Response ViewModel تعریف کردید.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۲:۲۷

سپاس از سری مطالبی که منتشر می‌کنید.
-پیشنهادی که من دارم اینه که لایه‌ی Repository حذف شود ، همانطور که در مطالب قبلی ذکر شده DbSet در Entity Framework همان پیاده سازی الگوی مخزن هست و ایجاد Repository جدید روی آن یک Abstraction اضافه هست. در نتیجه اگر Repository حذف شود همه‌ی منطق‌ها مانند GetBlaBla به Service منتقل می‌شود.
-یک پیشنهاد دیگر اینکه استفاده از کلمه‌ی New در Presentation Layer را به حداقل رساند و همه جا نیاز مندی‌ها را به صورت وابستگی به کلاس‌های استفاده کننده تزریق شود تا در زمان نوشتن تست‌ها همه‌ی اجزاء قابل تعویض با Mock objects باشند.

نویسنده: افشین
تاریخ: ۱۳۹۲/۰۱/۰۶ ۱۱:۱۵

لطفا دمو یا سورس برنامه رو هم قرار بدید که یادگیری و آموزش سریعتر انجام بشه.

ممنون

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۱۰ ۰:۱۱

با سلام از کار بزرگی که دارین می‌کنین سپاس
یک سوال؟
جای الگوی Unit Of Work در این پروژه کجا میشه؟

در این پست جناب آقای مهندس نصیری در لایه سرویس الگوی واحد کار را پیاده کرده اند، با توجه به وجود الگوی Repository

در پروژه شما ممنون میشم شرح بیشتری بدین که جایگاه پیاده سازی الگو واحد کار با توجه به مزایایی که دارد در کدام لایه است؟

نویسنده: رام
تاریخ: ۱۳۹۲/۰۱/۱۶ ۵:۲۹

محسن جان، چیزی که من از این الگو در مورد واکشی و نمایش داده‌ها برداشت میکنم اینه:

کلاس‌های لایه مخزن با دریافت دستور از لایه سرویس آبجکت مدل مربوطه را پر میکنند و به بالا (لایه سرویس) پاس میدند.

بعد

در لایه سرویس نمونه‌ی مدل مربوطه به ویـــــومدل متناظر باهاش تبدیل میشه و به لایه بالاتر فرستاده میشه

بنابراین
کار در **"لایه مخزن"** روی **"مدل ها"** انجام میگیره
و
کار در **"لایه سرویس"** روی **"ویـــــومدل ها"** انجام میشه

نتیجه: لایه سرویس هدف دیگری را نسبت به لایه مخزن دنبال میکند و این هدف آنقدر بزرگ و مهم هست که برایش یه لایه مجزا در نظر گرفته بشه

نویسنده: رام
تاریخ: ۱۳۹۲/۰۱/۱۶ ۵:۴۹

شاهین جان، من با حذف لایه مخزن مخالف هستم. زیرا:

ما لایه ای به نام "لایه مخزن" را میسازیم تا در نهایت کلیه متدهایی که برای حرف زدن با داده هامون را نیاز داریم داشته باشیم. حالا این اطلاعات ممکنه از پایگاه داده یا جاهای دیگه جمع آوری بشوند (و الزاما توسط EF قابل دسترسی و ارائه نباشند)

همچنین گاهی نیاز هست که بر مبنای چند متد که EF به ما میرسونه (مثلا چند SP) یک متد کلی‌تر را تعریف کنیم (چند فراخوانی را در یک متد مثلا متد X در لایه مخزن انجام دهیم) و در لایه بالاتر آن متد را صدا بزنیم (بجای نوشتن و تکرار پاپی همه کدهای نوشت شده در متد X)

علاوه بر این در لایه مخزن میشه چند ORM را هم کنار هم دید (نه فقط EF) که همونطور که آقای خوشبخت در کامنت‌ها نوشتند گاهی نیاز میشه.

بنابراین:
من وجود لایه مخزن را ضروری میدونم.

(فراموش نکنیم که هدف از این آموزش تعریف یک الگوی معماری مناسب برای پروژه‌های بزرگ هست و الا بدون خیلی از اینها هم میشه برنامه ساخت. همونطور که اکثرا بدون این ساختارها و خیلی ساده‌تر میسازند)

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۱۶ ۹:۳

- بحث آقای شاهین و من در مورد مثال عینی بود که زده شد. در مورد کار با ORM که کدهاش دقیقا ارائه شده. این روش قابل نقد و رد است.

شما الان اومدی یک بحث انتزاعی کلی رو شروع کردید. بله. اگر ORM رو کنار بگذارید مثلا می‌رسید به ADO.NET (یک نمونه که خیلی‌ها در این سایت حداقل یکبار باهاش کار کردن). این افراد پیش از اینکه این مباحث مطرح باشن برای خودشون لایه DAL داشتند و تمام جزئیات ADO.NET رو کپسوله کرده بودن در اون. حالا با اومدن ORMها این لایه DAL کنار رفته چون خود ORM هست که کپسوله کننده ADO.NET است. همین‌ها هم یک لایه دیگر داشتند به نام BLL که از لایه DAL استفاده می‌کرد برای پیاده سازی منطق تجاری برنامه. این لایه الان اسمش شده لایه سرویس.

یعنی تمام مواردی رو که عنوان کردید در مورد ADO.NET صدق می‌کنه. یکی اسمش رو می‌ذاره DAL شما اسمش رو گذشتید Repository. ولی این مباحث ربطی به یک ORM تمام عیار که کپسوله کننده ADO.NET است ندارد.

- ترکیب چند SP در لایه مخزن انجام نمیشه. چیزی رو که عنوان کردید یعنی پیاده سازی منطق تجاری و این مورد باید در لایه سرویس باشه. اگر از ADO.NET استفاده میشه، می‌تونیم با استفاده از DAL جزئیات دسترسی به SP رو مخفی و ساده‌تر کنیم با کدی یک دست‌تر در تمام برنامه. اگر از EF استفاده می‌کنیم، باز همین ساده سازی در طی فراخوانی فقط یک متد انجام شده. بنابراین بهتر است وضعیت و سطح لایه‌ای رو که داریم باهاش کار می‌کنیم خوب بررسی و درک کنیم.

- می‌تونید در عمل در بین پروژه‌های سورس باز و معتبر موجود فقط یک نمونه رو به من ارائه بدید که در اون از 2 مورد ORM مختلف همزمان استفاده شده باشه؟ این مورد یعنی سؤ مدیریت. یعنی پراکندگی و انجام کاری بسیار مشکل مثلا یک نمونه: ORMها لایه‌ای دارند به نام سطح اول کش که مثلا در EF اسمش هست Trackig API. این لایه فقط در حین کار با Context همون ORM کار می‌کنه. اگر دو مورد رو با هم مخلوط کنید، قابل استفاده نیست، ترکیب پذیر نیستند. از این دست باز هم هست مثلا در مورد نحوه تولید پروکسی‌هایی که برای lazy loading تولید می‌کنند و خیلی از مسایل دیگری از این دست. ضمن اینکه مدیریت چند Context فقط در یک لایه خودش یعنی نقض اصل تک مسئولیتی کلاس‌ها.

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۱۶ ۹:۱۵

سعی نکنید انتزاعی بحث کنید. چون در این حالت این حرف می‌تونه درست باشه یا حتی نباشه. اگر از ADO.NET استفاده می‌کنید، درسته. اگر از EF استفاده می‌کنید غلط هست. لازم هست منطق کار با ADO.NET رو یک سطح کپسوله کنیم. چون از تکرار کد جلوگیری می‌کنه و نهایتا به یک کد یک دست خواهیم رسید. لازم نیست اعمال یک ORM رو در لایه‌ای به نام مخزن کپسوله کنیم، چون خودش کپسوله سازی ADO.NET رو به بهترین نحوی انجام داده. برای نمونه در همین مثال عینی بالا به هیچ مزیتی نرسیدیم. فقط یک تکرار کد است. فقط بازی با کدها است.

نویسنده: رام
تاریخ: ۱۳۹۲/۰۱/۱۶ ۱۶:۴۶

من منظور شما را خوب متوجه میشم ولی حرفام یه بحث انتزاعی نیست چون پروژه عملی زیر دستم دارم که توی اون هم با پر کردن View Model کار میکنم.

مشکل از اینجا شروع میشه که شما فکر میکنید همیشه مدل ای که در EF ساختید را باید بدون تغییر در ساختارش به پوسته برنامه برسونید و از پوسته هم دقیقا نمونه ای از همون را بگیرید و به لایه‌های پایین بفرستید ولی یکی از مهمترین کارهای View Model اینه که این قانون را از این سفتی و سختی در بیاره چون خیلی مواقع هست که شما در پوسته برنامه به شکل دیگه ای از داده‌ها (متفاوت با اونچه در Model تعریف کردید و EF باهاش کار میکنه) نیاز دارید. مثلا فیلد تاریخ از نوع DateTime در Model و نوع String در پوسته و یا حتی اضافه و کم کردن فیلدهای یک Model و ایجاد ساختارهای متفاوتی از اون برای عملیات‌های Select, Update و Delete. لذا لایه سرویس قرار نیست فقط همون کار لایه مخزن را تکرار کنه (به قول شما GetAll). بلکه در زمان لزوم تغییرات لازم که نام بردم را هم رووش اعمال میکنه (که به نظر من آقای خوشبخت هم به خوبی از کلمه Convert در لایه سرویس استفاده کردند.)

اما بحث اینکه ما در لایه مخزن روی EF یک سطح کپسوله میسازیم جای گفتگو داره هرچند من در اون مورد هم با وجد لایه مخزن بیشتر موافقم تا گفتگوی مستقیم لایه سرویس با چیزی مثل EF

نتیجه: فرقی نمیکنه شما از Asp.Net استفاده میکنید یا هر ORM مورد نظرتون. کلاس‌های مدل باید در ارتباط با لایه بالاتر خودشون به ویـــو مدل تبدیل بشند و در این الگو این کار در لایه سرویس انجام میشه.

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۱۶ ۱۷:۱۰

- پیاده سازی الگوی مخزن در عمل (بر اساس بحث فعلی که در مورد کار با ORMها است) به صورت کپسوله سازی ORM در همه جا مطرح میشه و اینکار اساسا اشتباه هست. چون هم شما رو محروم می‌کنه از قابلیت‌های پیشرفته ORM و هم ارزش افزوده‌ای رو به همراه نداره. دست آخر می‌بینید در لایه مخزن GetAll دارید در لایه سرویس هم GetAll دارید. این مساله هیچ مزیتی نداره. یک زمانی در ADO.NET برای GetAll کردن باید کلی کد شبیه به کدهای یک ORM نوشته می‌شد. خود ORM الان اومده این‌ها رو کپسوله کرده و لایه‌ای هست روی اون. اینکه ما مجددا یک پوسته روی این بکشیم حاصلی نداره بجز تکرار کد. عده‌ای عنوان می‌کنند که حاصل اینکار امکان تعویض ORM رو ممکن می‌کنه ولی این‌ها هم بعد از یک مدت تجربه با ORMهای مختلف به این نتیجه می‌رسند که ای بابا! حتی پیاده سازی LINQ این ORMها یکی نیست چه برسه به قابلیت‌های پیشرفته‌ای که در یکی هست در دوتای دیگر نیست (واقع بینی، بجای بحث تئوری محض).

- اینکه این تبدیلات (پر کردن ViewModel از روی مدل) هم می‌تونه و بهتره که (نه الزاما) در لایه سرویس انجام بشه، نتیجه مناسبی هست.

UI

در نهایت نوبت به طراحی و کدنویسی UI می‌رسد تا بتوانیم محصولات را به کاربر نمایش دهیم. اما قبل از شروع باید موضوعی را یادآوری کنم. اگر به یاد داشته باشید، در کلاس ProductService موجود در لایه‌ی Domain ، از طریق یکی از روشهای الگوی Dependency Injection به نام Constructor Injection ، فیلدی از نوع IProductRepository را مقداردهی نمودیم. حال زمانی که بخواهیم نمونه ای را از ProductService ایجاد نماییم، باید به عنوان پارامتر ورودی سازنده، شی ایجاد شده از جنس کلاس ProductRepository موجود در لایه Repository را به آن ارسال نماییم. اما از آنجایی که می‌خواهیم تفکیک پذیری لایه‌ها از بین نرود و UI بسته به نیاز خود، نمونه مورد نیاز را ایجاد نموده و به این کلاس ارسال کند، از ابزارهایی برای این منظور استفاده می‌کنیم. یکی از این ابزارها StructureMap می‌باشد که یک Inversion of Control Container یا به اختصار IoC Container نامیده می‌شود. با Inversion of Control در مباحث بعدی بیشتر آشنا خواهید شد. StructureMap ابزاری است که در زمان اجرا، پارامترهای ورودی سازنده‌ی کلاسهایی را که از الگوی Dependency Injection استفاده نموده اند و قطعا پارامتر ورودی آنها از جنس یک Interface می‌باشد را، با ایجاد شی مناسب مقداردهی می‌نماید.

به منظور استفاده از StructureMap در Visual Studio 2012 باید بر روی پروژه UI خود کلیک راست نموده و گزینه‌ی Manage NuGet Packages را انتخاب نمایید. در پنجره ظاهر شده و از سمت چپ گزینه‌ی Online و سپس در کادر جستجوی سمت راست و بالای پنجره واژه‌ی structuremap را جستجو کنید. توجه داشته باشید که باید به اینترنت متصل باشید تا بتوانید Package مورد نظر را نصب نمایید. پس از پایان عمل جستجو، در کادر میانی structuremap ظاهر می‌شود که می‌توانید با انتخاب آن و فشردن کلید Install آن را بر روی پروژه نصب نمایید.

جهت آشنایی بیشتر با NuGet و نصب آن در سایر نسخه‌های Visual Studio می‌توانید به لینک‌های زیر رجوع کنید:

1. آشنایی با NuGet قسمت اول

2. آشنایی با NuGet قسمت دوم

3. Installing NuGet

کلاسی با نام BootStrapper را با کد زیر به پروژه UI خود اضافه کنید:

```
using StructureMap;
using StructureMap.Configuration.DSL;
using SoCPatterns.Layered.Repository;
using SoCPatterns.Layered.Model;

namespace SoCPatterns.Layered.WebUI
{
    public class BootStrapper
    {
        public static void ConfigureStructureMap()
        {
            ObjectFactory.Initialize(x => x.AddRegistry<ProductRegistry>());
        }
    }
```

```
    public class ProductRegistry : Registry
    {
        public ProductRegistry()
        {
            For<IProductRepository>().Use<ProductRepository>();
        }
    }
}
```

ممکن است یک WinUI ایجاد کرده باشید که در این صورت به جای فضای نام SoCPatterns.Layered.WebUI از SoCPatterns.Layered.WinUI استفاده نمایید.

هدف کلاس BootStrapper این است که تمامی وابستگی‌ها را توسط StructureMap در سیستم Register نماید. زمانی که کدهای کلاینت می‌خواهند به یک کلاس از طریق StructureMap دسترسی داشته باشند، Structuremap وابستگی‌های آن کلاس را تشخیص داده و بصورت خودکار پیاده سازی واقعی (Concrete Implementation) آن کلاس را، براساس همان چیزی که ما برایش تعیین کردیم، به کلاس تزریق می‌نماید. متد ConfigureStructureMap باید در همان لحظه ای که Application آغاز به کار می‌کند فراخوانی و اجرا شود. با توجه به نوع UI خود یکی از روالهای زیر را انجام دهید:

در WebUI :

فایل Global.asax را به پروژه اضافه کنید و کد آن را بصورت زیر تغییر دهید:

```
namespace SoCPatterns.Layered.WebUI
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            BootStrapper.ConfigureStructureMap();
        }
    }
}
```

در WinUI :

در فایل Program.cs کد زیر را اضافه کنید:

```
namespace SoCPatterns.Layered.WinUI
{
    static class Program
    {
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
```

BootStrapper.ConfigureStructureMap();

```
Application.Run(new Form1());
        }
    }
}
```

سپس برای طراحی رابط کاربری، با توجه به نوع UI خود یکی از روالهای زیر را انجام دهید:

در WebUI :

صفحه Default.aspx را باز نموده و کد زیر را به آن اضافه کنید:

```
<asp:DropDownList AutoPostBack="true" ID="ddlCustomerType" runat="server">
    <asp:ListItem Value="0">Standard</asp:ListItem>
    <asp:ListItem Value="1">Trade</asp:ListItem>
</asp:DropDownList>
<asp:Label ID="lblErrorMessage" runat="server" ></asp:Label>
<asp:Repeater ID="rptProducts" runat="server" >
    <HeaderTemplate>
        <table>
            <tr>
                <td>Name</td>
                <td>RRP</td>
                <td>Selling Price</td>
                <td>Discount</td>
                <td>Savings</td>
            </tr>
            <tr>
                <td colspan="5"><hr /></td>
            </tr>
    </HeaderTemplate>
    <ItemTemplate>
            <tr>
                <td><%# Eval("Name") %></td>
                <td><%# Eval("RRP")%></td>
                <td><%# Eval("SellingPrice") %></td>
                <td><%# Eval("Discount") %></td>
                <td><%# Eval("Savings") %></td>
            </tr>
    </ItemTemplate>
    <FooterTemplate>
        </table>
    </FooterTemplate>
</asp:Repeater>
```

در WinUI :

فایل Form1.Designer.cs را باز نموده و کد آن را بصورت زیر تغییر دهید:

```
#region Windows Form Designer generated code
/// <summary>
/// Required method for Designer support - do not modify
/// the contents of this method with the code editor.
/// </summary>
private void InitializeComponent()
{
    this.cmbCustomerType = new System.Windows.Forms.ComboBox();
    this.dgvProducts = new System.Windows.Forms.DataGridView();
    this.colName = new System.Windows.Forms.DataGridViewTextBoxColumn();
    this.colRrp = new System.Windows.Forms.DataGridViewTextBoxColumn();
    this.colSellingPrice = new System.Windows.Forms.DataGridViewTextBoxColumn();
    this.colDiscount = new System.Windows.Forms.DataGridViewTextBoxColumn();
    this.colSavings = new System.Windows.Forms.DataGridViewTextBoxColumn();
    ((System.ComponentModel.ISupportInitialize)(this.dgvProducts)).BeginInit();
    this.SuspendLayout();
    //
    // cmbCustomerType
    //
    this.cmbCustomerType.DropDownStyle =
System.Windows.Forms.ComboBoxStyle.DropDownList;
    this.cmbCustomerType.FormattingEnabled = true;
    this.cmbCustomerType.Items.AddRange(new object[] {
        "Standard",
        "Trade"});
    this.cmbCustomerType.Location = new System.Drawing.Point(12, 90);
```

```
    this.cmbCustomerType.Name = "cmbCustomerType";
    this.cmbCustomerType.Size = new System.Drawing.Size(121, 21);
    this.cmbCustomerType.TabIndex = 3;
    //
    // dgvProducts
    //
    this.dgvProducts.ColumnHeadersHeightSizeMode =
System.Windows.Forms.DataGridViewColumnHeadersHeightSizeMode.AutoSize;
    this.dgvProducts.Columns.AddRange(new System.Windows.Forms.DataGridViewColumn[] {
    this.colName,
    this.colRrp,
    this.colSellingPrice,
    this.colDiscount,
    this.colSavings});
    this.dgvProducts.Location = new System.Drawing.Point(12, 117);
    this.dgvProducts.Name = "dgvProducts";
    this.dgvProducts.Size = new System.Drawing.Size(561, 206);
    this.dgvProducts.TabIndex = 2;
    //
    // colName
    //
    this.colName.DataPropertyName = "Name";
    this.colName.HeaderText = "Product Name";
    this.colName.Name = "colName";
    this.colName.ReadOnly = true;
    //
    // colRrp
    //
    this.colRrp.DataPropertyName = "Rrp";
    this.colRrp.HeaderText = "RRP";
    this.colRrp.Name = "colRrp";
    this.colRrp.ReadOnly = true;
    //
    // colSellingPrice
    //
    this.colSellingPrice.DataPropertyName = "SellingPrice";
    this.colSellingPrice.HeaderText = "Selling Price";
    this.colSellingPrice.Name = "colSellingPrice";
    this.colSellingPrice.ReadOnly = true;
    //
    // colDiscount
    //
    this.colDiscount.DataPropertyName = "Discount";
    this.colDiscount.HeaderText = "Discount";
    this.colDiscount.Name = "colDiscount";
    //
    // colSavings
    //
    this.colSavings.DataPropertyName = "Savings";
    this.colSavings.HeaderText = "Savings";
    this.colSavings.Name = "colSavings";
    this.colSavings.ReadOnly = true;
    //
    // Form1
    //
    this.AutoScaleDimensions = new System.Drawing.SizeF(6F, 13F);
    this.AutoScaleMode = System.Windows.Forms.AutoScaleMode.Font;
    this.ClientSize = new System.Drawing.Size(589, 338);
    this.Controls.Add(this.cmbCustomerType);
    this.Controls.Add(this.dgvProducts);
    this.Name = "Form1";
    this.Text = "Form1";
    ((System.ComponentModel.ISupportInitialize)(this.dgvProducts)).EndInit();
    this.ResumeLayout(false);
}
#endregion
private System.Windows.Forms.ComboBox cmbCustomerType;
private System.Windows.Forms.DataGridView dgvProducts;
private System.Windows.Forms.DataGridViewTextBoxColumn colName;
private System.Windows.Forms.DataGridViewTextBoxColumn colRrp;
private System.Windows.Forms.DataGridViewTextBoxColumn colSellingPrice;
private System.Windows.Forms.DataGridViewTextBoxColumn colDiscount;
private System.Windows.Forms.DataGridViewTextBoxColumn colSavings;
```

سپس در Code Behind ، با توجه به نوع UI خود یکی از روالهای زیر را انجام دهید:

در WebUI :

وارد کد نویسی صفحه Default.aspx شده و کد آن را بصورت زیر تغییر دهید:

```
using System;
using System.Collections.Generic;
using SoCPatterns.Layered.Model;
using SoCPatterns.Layered.Presentation;
using SoCPatterns.Layered.Service;
using StructureMap;

namespace SoCPatterns.Layered.WebUI
{
    public partial class Default : System.Web.UI.Page, IProductListView
    {
        private ProductListPresenter _productListPresenter;
        protected void Page_Init(object sender, EventArgs e)
        {
            _productListPresenter = new
ProductListPresenter(this,ObjectFactory.GetInstance<Service.ProductService>());
            this.ddlCustomerType.SelectedIndexChanged +=
                delegate { _productListPresenter.Display(); };
        }
        protected void Page_Load(object sender, EventArgs e)
        {
            if(!Page.IsPostBack)
                _productListPresenter.Display();
        }
        public void Display(IList<ProductViewModel> products)
        {
            rptProducts.DataSource = products;
            rptProducts.DataBind();
        }
        public CustomerType CustomerType
        {
            get { return (CustomerType) int.Parse(ddlCustomerType.SelectedValue); }
        }
        public string ErrorMessage
        {
            set
            {
                lblErrorMessage.Text =
                    String.Format("<p><strong>Error:</strong><br/>{0}</p>", value);
            }
        }
    }
}
```

در WinUI :

وارد کدنویسی Form1 شوید و کد آن را بصورت زیر تغییر دهید:

```
using System;
using System.Collections.Generic;
using System.Windows.Forms;
using SoCPatterns.Layered.Model;
using SoCPatterns.Layered.Presentation;
using SoCPatterns.Layered.Service;
using StructureMap;

namespace SoCPatterns.Layered.WinUI
{
    public partial class Form1 : Form, IProductListView
    {
        private ProductListPresenter _productListPresenter;
        public Form1()
        {
            InitializeComponent();
            _productListPresenter =
                new ProductListPresenter(this, ObjectFactory.GetInstance<Service.ProductService>());
            this.cmbCustomerType.SelectedIndexChanged +=
                delegate { _productListPresenter.Display(); };
            dgvProducts.AutoGenerateColumns = false;
```

```
            cmbCustomerType.SelectedIndex = 0;
        }
        public void Display(IList<ProductViewModel> products)
        {
            dgvProducts.DataSource = products;
        }
        public CustomerType CustomerType
        {
            get { return (CustomerType)cmbCustomerType.SelectedIndex; }
        }
        public string ErrorMessage
        {
            set
            {
                MessageBox.Show(
                    String.Format("Error:{0}{1}", Environment.NewLine, value));
            }
        }
    }
}
```

با توجه به کد فوق، نمونه ای را از کلاس ProductListPresenter ، در لحظه‌ی نمونه سازی اولیه‌ی کلاس UI ، ایجاد نمودیم. با استفاده از متد ObjectFactory.GetInstance مربوط به StructureMap ، نمونه ای از کلاس ProductService ایجاد شده است و به سازنده‌ی کلاس ProductListPresenter ارسال گردیده است. در مورد Structuremap در مباحث بعدی با جزئیات بیشتری صحبت خواهم کرد. پیاده سازی معماری لایه بندی در اینجا به پایان رسید.

اما اصلا نگران نباشید، شما فقط پرواز کوتاه و مختصری را بر فراز کدهای معماری لایه بندی داشته اید که این فقط یک دید کلی را به شما در مورد این معماری داده است. این معماری هنوز جای زیادی برای کار دارد، اما در حال حاضر شما یک Applicaion با پیوند ضعیف (Loosely Coupled) بین لایه‌ها دارید که دارای قابلیت تست پذیری قوی، نگهداری و پشتیبانی آسان و تفکیک پذیری قدرتمند بین اجزای آن می‌باشد. شکل زیر تعامل بین لایه‌ها و وظایف هر یک از آنها را نمایش می‌دهد.

نظرات خوانندگان

نویسنده: حامد
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۴:۲۹

ممنون از مقاله خوبتون
به نظر شما امکانش هست برای این معماری یک generator بسازیم به طوری که فقط ما تمام جداول دیتابیس و رابطه‌ی آنها را بسازیم و بعد این generator از روی اون تمام لایه‌ها را بر اساس آن بسازه و بعد ما صرفا جاهایی که نیاز به جزییات داره را کامل کنیم
آیا نمونه ای از این برنامه‌ها هست که این معماری یا معماری‌های مشابه را بسازه؟

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۵:۳۱

اگر با T4 آشنایی داشته باشید بر اساس هر قالبی می‌توانید کد تولید کنید.

نویسنده: حامد
تاریخ: ۱۳۹۲/۰۱/۰۳ ۱۶:۳۶

متاسفانه آشنایی ندارم میشه یه توضیح مختصر بدین و یا منبع معرفی کنید

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۰۳ ۲۳:۵۹

چون اینجا بحث طراحی مطرح شده یک اصل رو در برنامه‌های وب باید رعایت کرد:

هیچ وقت متن خطای حاصل رو به کاربر نمایش ندید (از لحاظ امنیتی). فقط به ذکر عبارت خطایی رخ داده بسنده کنید. خطا رو مثلا توسط ELMAH لاگ کنید برای بررسی بعدی برنامه نویس.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۰۴ ۱:۲۰

http://codepanic.blogspot.com/2012/03/t4-enum.html

نویسنده: M.Q
تاریخ: ۱۳۹۲/۰۱/۰۴ ۲۲:۱۵

دوست عزیز غیر از ELMAH ابزار دیگری برای لاگ گیری از خطاها وجود دارد که قابل اعتماد باشد؟

همچنین اگر ابزاری جهت لاگ گیری از عملیات کاربران (CRUD => حالا R خیلی مهم نیست) می‌شناسید معرفی نمائید.

با سپاس

نویسنده: محسن
تاریخ: ۱۳۹۲/۰۱/۰۵ ۰:۳۳

متد auditFields مطرح شده در مطلب ردیابی اطلاعات این سایت برای مقصود شما مناسب است.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۴:۲۳

سلام با تشکر از شما
من نفهمیدم که توی ASP.NET MVC شما چگونه از الگوی MVP استفاده کردین؟
ظاهرا مثال این قسمت هم توی پست وجود نداره، اگر اشتباه می‌کنم لطفا تصحیح بفرمایید.

نویسنده: علي
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۶:۳

مثال وب فرم هست. post back و page load داره.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۶:۴

اگر توجه کنید از الگوی MVP در Web Forms استفاده شده و نه در MVC.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۸:۳۰

آقای کیاست و علی آقا
می‌دونم که پروژه چی هست، یکی از UIهای ما قرار بود MVC باشه خواستم بدونم چطور می‌خوان استفاده کنن، اینجا (در این پست) که می‌دونم ASP.NET Web form هست و در MVC می‌دونم که Page_Load و.. وجود نداره سوال من چیز دیگه بود دوستان

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۸:۴۴

شما گفتید:

سلام با تشکر از شما
من نفهمیدم که توی ASP.NET MVC شما چگونه از الگوی MVP استفاده کردین؟
ظاهرا مثال این قسمت هم توی پست وجود نداره، اگر اشتباه می‌کنم لطفا تصحیح بفرمایید.
با خواندن کامنت شما برداشت کردم شما تصور کردید کدهای پست جاری مربوط به تکنولوژی ASP.NET MVC هست.

به نظر نویسنده هنوز برای MVC و WPF مثال‌ها را ایجاد نکرده و توضیح نداده اند.
اما برای استفاده از این نوع معماری در MVC کار خاصی لازم نیست انجام شود. همانطور که قبلا در مثال‌های آقای نصیری دیده ایم کافی است Service Layer در Controller مدل مناسب را تغذیه کند و برای View فراهم کند.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۱۲ ۱۹:۲۶

من هم با توجه به مثال آقای نصیری و استفاده از الگوی کار گیج شدم، این معماری یک لایه Repository دارد، من الگوی کار توی این لایه پیاده کردم، با پیاده سازی در این لایه به نظر میاد لایه سرویس کاربردش از دست میده توی پست‌های قبل هم از آقای خوشبخت سوال کردم اما طاهرا هنوز وقت نکردن پاسخ بدن.

مورد دوم اینکه در این پست الگوی کار شرح داده شده و پیاده سازی شده، و در این پست گفته شده " *حین استفاده از EF code first، الگوی واحد کار، همان DbContext است و الگوی مخزن، همان DbSetها. ضرورتی به ایجاد یک لایه محافظ اضافی بر روی*

این‌ها وجود ندارد. " با توجه به این مسائل کلا مسائل قاطی کردم متاسفانه آقای نصیری هم سرشون شلوغ و درگیر دوره ها است، که بحثی بر سر این معماری بشه.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۱۲ ۲۰:۴۶

روشی که در مثال آقای نصیری گفته شده با روش این سری مقالات کمی متفاوت هست.
در آنجا از روکش اضافه برای Repository استفاده نشده همچنین از الگوی واحد کار استفاده شده.
به علاوه این سری مقالات ممکن است هنوز تکمیل نشده باشند.
به نظر من هر کس با توجه به میزان اطلاعاتی که دارد و درکی که از الگوها دارد با مقایسه‌ی روش‌ها و مقالات می‌تواند تصمیم بگیرد چه معماری به کار بگیرد.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۱/۱۲ ۲۱:۳

حرف شما کاملا متین هست

من قبلا معماری سه لایه کار می‌کردم، که نمونه اون توی همین سایت بخش پروژه ها گذاشتم، اما الان با EF , MVC کمی به مشکل بر خوردم و درست نتونستم تا حالا لایه‌های مورد نظر برای خودم در پروژه‌ها تفکیک کنم، این معماری به نظرم جالب اومد، خواستم که الگوی کار هم توی اون به کار ببرم که به مشکل بر خوردم (چون درک درستی از الگوی کار پیدا نکردم یا شایدم کلا دارم اشتباه می‌کنم). البته به قول شما شاید این معماری هنوز تکمیل نشده پروژه اش، در هر صورت از پاسخ‌های شما متشکرم.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۱/۱۲ ۲۱:۷

خواهش می‌کنم.
فقط جهت یادآوری مثال روش آقای نصیری با پوشش MVC و EF قابل دریافت است.

نویسنده: ابوالفضل روشن ضمیر
تاریخ: ۱۳۹۲/۰۱/۱۷ ۱:۴۵

سلام
با تشکر فروان از شما ...
اگر امکان داره این مثال که در قالی یک پروژه نوشته شده برای دانلود قرار دهید ...تا بهتر بتوانیم برنامه را تجزیه و تحلیل کنیم ....ممنون

نویسنده: فرشید علی اکبری
تاریخ: ۱۳۹۲/۰۱/۱۹ ۱۵:۵۳

با سلام و تشکر از زحمات کلیه دوستان
با زحمتی که آقای خوشبخت تا اینجا کشیدن فکر کنم در صورتیکه خودمون مقاله مربوطه به این پروژه رو قدم به قدم بخونیم وطراحی کینم خیلی بهتر متوجه میشیم تا اینکه اونو آماده دانلود کنیم. من با این روش پیش رفتم و برای ایجاد اون با step by step کردن مراحلش حدود 45 دقیقه وقت گذاشتم ولی درصد یادگیریش خیلی بالاتر بود تا گرفتن فایل آماده...
درضمن لازمه بگم که بخاطر رفع شک وشبهه در سرعت پردازش وبالا اومدن اطلاعات، من تست این روش رو با تعداد 155 هزار رکورد انجام دادم که کمتر از سه ثانیه برام لود شد... باوجودیکه کامپوننت‌های دات نت بار مختلفی رو هم روی فرمم قرار دادم که بیشتر به اهمیت لود اطلاعاتم در پروژه و فرمهای واقعی پی ببرم.
سئوال اینکه :

به نظر شما ما می‌تونیم روی این لایه‌ها الگوی واحد کار رو هم ایجاد کنیم یا نه؟ اصلا ضرورتی داره ؟

نویسنده: علیرضا کیانی مقدم
تاریخ: ۱۳۹۲/۰۱/۲۸ ۱۲:۸

با تشکر از نویسنده مقاله و اهتمام ایشان به بررسی دقیق مفاهیم ،
از آنجا که flexible و reusable بودن برنامه‌ها را نمی‌توان نادیده گرفت تا آنجا که این تفکیک پذیری خود به مسئله ای بغرنج تبدیل نشده و تکرر داده‌ها و پاس دادن غیر ضرور آنها را موجب نشود تلاش در این باره مفید خواهد بود .
امروزه توسعه دهنده گان به سمت کم کردن لایه‌های فرسایشی و حذف پیچیدگی‌های غیر ضرور قدم بر می‌دارند. خلق عبارات لامبادا در دات نت و delegate ها نمونه هایی از تلاش بشر برنامه نویس در این باره است .

نویسنده: مسعود2
تاریخ: ۱۳۹۲/۰۲/۰۹ ۹:۱۲

سلام
Business Rule-1ها و validation-2ها در کجای این معماری اعمال میشوند؟

منظور از DomainService چیست؟

ممکنه منابع بیشتری معرفی نمایید؟
ممنون.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۹ ۱۶:۲۶

منبع برای مطالعه بیشتر

عنوان: **نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۴ ۱۲:۳۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Debugging, MVC, FireBug, jQuery, jQuery Ajax

هر از چندگاهی سؤال «این مثال jQuery رو نمی‌تونم اجرا یا باز سازی کنم» در این سایت یا سایت‌های مشابه تکرار می‌شوند. بنابراین بهتر است نحوه عیب یابی برنامه‌های ASP.NET مبتنی بر jQuery را یکبار با هم مرور کنیم. در اینجا، مثال تهیه یک Image Slider را که پیشتر در سایت مطرح شده است، به نحوی دیگر بررسی خواهیم کرد:

1) فراموش می‌کنیم تا اسکریپت اصلی jQuery را به درستی پیوست و مسیردهی کنیم.
2) مسیر Generic handler دیگری را ذکر می‌کنیم.
3) مسیرهای تصاویری را که Image slider باید نمایش دهد، کاملا بی‌ربط ذکر می‌کنیم.
4) خروجی JSON نامربوطی را بازگشت می‌دهیم.
5) یکبار هم یک استثنای عمدی دستی را در بین کدها قرار خواهیم داد.

و ... بعد سعی می‌کنیم با استفاده از Firebug عیوب فوق را یافته و اصلاح کنیم؛ تا به یک برنامه قابل اجرا برسیم.

معرفی برنامه‌ای که کار نمی‌کند!

یک برنامه ASP.NET Empty web application را آغاز کنید. سپس سه پوشه Content، Scripts و Images را به آن اضافه نمائید. در این پوشه‌ها، اسکریپت‌های نمایش دهنده تصاویر، Css آن و تصاویری که قرار است نمایش داده شوند، قرار می‌گیرند:

سپس یک فایل default.aspx و یک فایل OrbitHandler.ashx را نیز به پروژه با محتویات ذیل اضافه کنید: (در این دو فایل، 5 مورد مشکل ساز یاد شده لحاظ شده‌اند)
محتویات فایل OrbitHandler.ashx.cs مطابق کدهای ذیل است:

```
using System.Collections.Generic;
using System.IO;
using System.Web;
using System.Web.Script.Serialization;

namespace OrbitWebformsTest
{
    public class Picture
    {
        public string Title { set; get; }
        public string Path { set; get; }
    }

    public class OrbitHandler : IHttpHandler
    {
        IList<Picture> PicturesDataSource()
        {
            var results = new List<Picture>();
            var path = HttpContext.Current.Server.MapPath("~/Images");

            foreach (var item in Directory.GetFiles(path, "*.*"))
            {
                var name = Path.GetFileName(item);
                results.Add(new Picture
                {
                    Path = /*"Images/" + name*/ name,
                    Title = name
                });
            }

            return results;
        }

        public void ProcessRequest(HttpContext context)
        {
            var items = PicturesDataSource();
            var json = /*new JavaScriptSerializer().Serialize(items)*/ string.Empty;
            throw new InvalidDataException("همینطوری");
            context.Response.ContentType = "text/plain";
            context.Response.Write(json);
        }

        public bool IsReusable
        {
            get { return false; }
        }
    }
}
```

در اینجا جهت سهولت دموی برنامه (و همچنین امکان باز تولید آن توسط خوانندگان)، از بانک اطلاعاتی استفاده نشده و عمدا از یک لیست جنریک تشکیل شده در حافظه کمک گرفته شده است. تصاویر برنامه در پوشه Images واقع در ریشه سایت، قرار دارند. بنابراین توسط متد PicturesDataSource، فایل‌های این پوشه را یافته و مطابق ساختار کلاس Picture بازگشت می‌دهیم. نهایتا این اطلاعات به ظاهر قرار است با فرمت JSON بازگشت داده شوند تا بتوان نتیجه را توسط افزونه Orbit استفاده کرد.

همچنین کدهای صفحه ASPX ایی که قرار است (به ظاهر البته) از این Generic handler استفاده کند به نحو ذیل است:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="default.aspx.cs"
Inherits="OrbitWebformsTest._default" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-
transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
    <link href="Content/orbit-1.2.3.css" rel="stylesheet" type="text/css" />
    <script src="Script/jquery-1.5.1.min.js" type="text/javascript"></script>
    <script src="Scripts/jquery.orbit-1.2.3.min.js" type="text/javascript"></script>
</head>
<body>
    <form id="form1" runat="server">
    <div id="featured">
    </div>
    </form>
    <script type="text/javascript">
```

```
        $(function () {
            $.ajax({
                url: "Handler.ashx",
                contentType: "application/json; charset=utf-8",
                success: function (data) {
                    $.each(data, function (i, b) {
                        var str = '<img src="' + b.Path + '" alt="' + b.Title + '"/>';
                        $("#featured").append(str);
                    });
                    $('#featured').orbit();
                },
                dataType: "json"
            });
        });
    </script>
</body>
</html>
```

خوب! اگر پروژه را اجرا کنیم، کار نمی‌کند. یک مستطیل مشکی رنگ در کنار صفحه ظاهر شده و همین! حالا چکار باید کرد؟

مراحل عیب یابی برنامه‌ای که کار نمی‌کند!

ابتدا برنامه را در فایرفاکس باز کرده و سپس افزونه Firebug را با کلیک بر روی آیکن آن، بر روی سایت فعال می‌کنیم. سپس یکبار بر روی دکمه F5 کلیک کنید تا مجددا مراحل بارگذاری سایت تحت نظر افزونه Firebug فعال شده، طی شود.

اولین موردی که مشهود است، نمایش عدد 3، کنار آیکن فایرباگ می‌باشد. این عدد به معنای وجود خطاهای اسکریپتی در کدهای ما است.
برای مشاهده این خطاها، بر روی برگه Console آن کلیک کنید:

بله. مشخص است که مسیر دهی فایل jquery-1.5.1.min.js صحیح نبوده و همین مساله سبب بروز خطاهای اسکریپتی گردیده است. برای اصلاح آن سطر زیر را در برنامه تغییر دهید:

```
<script src="Scripts/jquery-1.5.1.min.js" type="text/javascript"></script>
```

پیشتر پوشه Script ذکر شده بود که باید تبدیل به Scripts شود.

مجددا دکمه F5 را فشرده و سایت را با تنظیمات جدید اجرا کنید. اینبار در برگه Console و یا در برگه شبکه فایرباگ، خطای یافت نشدن Generic handler نمایان می‌شوند:

برای رفع آن به فایل default.aspx مراجعه و بجای معرفی Handler.ashx، نام OrbitHandler.ashx را وارد کنید.
مجددا دکمه F5 را فشرده و سایت را با تنظیمات جدید اجرا کنید.

اگر به برگه کنسول دقت کنیم، بروز استثناء در کدها تشخیص داده شده و همچنین در برگه Response پاسخ دریافتی از سرور، جزئیات صفحه خطای بازگشتی از آن نیز قابل بررسی و مشاهده است.
اینبار به فایل OrbitHandler.ashx.cs مراجعه کرده و سطر throw new InvalidDataException را حذف می‌کنیم. در ادامه برنامه را کامپایل و مجددا اجرا خواهیم کرد.

با اجرای مجدد سایت، تبادل اطلاعات صحیحی با فایل OrbitHandler.ashx برقرار شده است، اما خروجی خاصی قابل مشاهده نیست. بنابراین بازهم سایت کار نمی‌کند.

برای رفع این مشکل، متد ProcessRequest را به نحو ذیل تغییر خواهیم داد:

```
public void ProcessRequest(HttpContext context)
        {
            var items = PicturesDataSource();
            var json = new JavaScriptSerializer().Serialize(items);
            context.Response.ContentType = "text/plain";
            context.Response.Write(json);
        }
```

برنامه را کامپایل کرده و اجرا می‌کنیم. برنامه اجرا می‌شود، اما باز هم کار نمی‌کند. مشکل از کجاست؟

بله. تمام تنظیمات به نظر درست هستند، اما در برگه شبکه فایرباگ تعدادی خطای 404 و یا «یافت نشد»، مشاهده می‌شوند.

مشکل اینجا است که مسیرهای بازگشت داده شده توسط متد Directory.GetFiles، مسیرهای مطلقی هستند؛ مانند c:\path\images\01.jpg و جهت نمایش در یک وب سایت مناسب نمی‌باشند. برای تبدیل آن‌ها به مسیرهای نسبی، اینبار کدهای متد تهیه منبع داده را به نحو ذیل ویرایش می‌کنیم:

```
IList<Picture> PicturesDataSource()
        {
            var results = new List<Picture>();
```

```
        var path = HttpContext.Current.Server.MapPath("~/Images");

        foreach (var item in Directory.GetFiles(path, "*.*"))
        {
            var name = Path.GetFileName(item);
            results.Add(new Picture
            {
                Path = "Images/" + name,
                Title = name
            });
        }

        return results;
    }
```

در این کدها فقط قسمت Path ویرایش شده است تا به مسیر پوشه Images واقع در ریشه سایت اشاره کند.
اینبار اگر برنامه را اجرا کنیم، بدون مشکل کار خواهد کرد.

بنابراین در اینجا مشاهده کردیم که اگر «برنامه‌ای مبتنی بر jQuery کار نمی‌کند»، چگونه باید قدم به قدم با استفاده از فایرباگ و امکانات آن، به خطاهایی که گزارش می‌دهد و یا مسیرهایی را که یافت نشد بیان می‌کند، دقت کرد تا بتوان برنامه را عیب یابی نمود.

سؤال مهم: اجرای کدهای jQuery Ajax فوق، چه تغییری را در صفحه سبب می‌شوند؟

اگر به برگه اسکریپت‌ها در کنسول فایرباگ مراجعه کنیم، امکان قرار دادن breakpoint بر روی سطرهای کدهای جاوا اسکریپتی نمایش داده شده نیز وجود دارد:

در اینجا همانند VS.NET می‌توان برنامه را در مرورگر اجرا کرده و تگ‌های تصویر پویای تولید شده را پیش از اضافه شدن به صفحه، مرحله به مرحله بررسی کرد. به این ترتیب بهتر می‌توان دریافت که آیا src بازگشت داده شده از سرور فرمت صحیحی دارد یا خیر و آیا به محل مناسبی اشاره می‌کند یا نه. همچنین در برگه HTML آن، عناصر پویای اضافه شده به صفحه نیز بهتر مشخص هستند:

Console HTML CSS Script DOM Net Cookies YSlow Pixel P... Firefi

Edit | body < html

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1
<html xmlns="http://www.w3.org/1999/xhtml">
  <head>
  <body>
    <form id="form1" action="default.aspx" method="post">
      <div class="aspNetHidden">
      <div class="orbit-wrapper" style="width: 48px; height: 19px;">
        <div id="featured" class="orbit" style="width: 48px; height: 19px;">
          <img alt="01.png" src="01.png" style="z-index: 1; display: block; left:
          -48px;"> jQuery15100924838756054342 2=Object { olddisplay="inline" }
          <img alt="02.jpg" src="02.jpg" style="display: block; left: -48px;
          z-index: 1;">
          <img alt="03.jpg" src="03.jpg" style="display: block; left: 0px; z-index:
          3;">
        </div>
```

نظرات خوانندگان

نویسنده: صادق
تاریخ: ۱۳۹۲/۰۲/۰۴ ۱۲:۵۱

ممنون آقای نصیری - برای کروم چطور، آیا developer tools کفایت میکند یا ابزار بهتری سراغ دارید؟
البته من گاهی که نیاز به دستکاری ریسپانس و ریکوئست باشه از فیدلر استفاده میکنم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۰۴ ۱۲:۵۵

تمام مواردی که در بحث جاری با فایرباگ مطرح شدن به صورت توکار در developer tools مربوط به کروم هم وجود دارند و مباحث آن تقریبا یکی است.

نویسنده: محسن جمشیدی
تاریخ: ۱۳۹۲/۰۲/۰۶ ۱۰:۴۹

یکی از مواردی هم که ممکن است آزار دهنده باشد خطای سینتکسی است که در FireFox با زدن Ctrl+Shift+J قابل مشاهده خواهد بود.

نویسنده: Abolfazl
تاریخ: ۱۳۹۲/۰۲/۰۸ ۲۳:۴۸

سلام آقای نصیری ..ممنونم ..خیلی عالی بود
من قبلا هم این مشکل را داشتم و نمی‌تونستم برطرفش کنم ..الان درست شد..مرسی

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۰۹ ۱۶:۴۲

```
$(document).ready(function () {
    $(':radio').click(function () {
        debugger; //فعال سازی دیباگر ویژوال استودیو
     });
});
```

یک نکته جانبی است برای فعال سازی دیباگر خود ویژوال استودیو در حین کار با جی‌کوئری

نویسنده: ahmad.valipour
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۲:۱۹

با سلام
سوالی برام پیش اومده.اگر ما 10 تا عکس تو پوشه image داشته باشیم ولی فقط 5 تا اونها رو در صفحه ارجاع بهشون داشته باشیم، فقط اون 5 تا به طرف کلاینت میرن.درسته؟
پس چرا وقتی با firebug مسیر عکس رو تغییر میدیم به عکس دیگه، میشناسه اون عکس رو ،و نشون میده؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۴ ۱۲:۲۵

همکاری متقابل موتور مرورگر و فایرباگ. فایرباگ درخواست می‌کنه، مرورگر دریافت و ارائه.
چیزی مثل دیباگر در VS.NET. زمانیکه مثلا در کدهای کار با Entity framework روی سطری break point قرار می‌دید و خروجی

یک کوئری را بررسی می‌کنید، این دیباگر قابلیت دریافت مقادیر بررسی شده و حتی نشده را هم از بانک اطلاعاتی دارا است (حتی اگر این مقادیر، در کوئری اولیه درخواست نشده باشند؛ نوعی lazy loading در اینجا صورت می‌گیرد)

نویسنده: ابوالفضل روشن ضمیر
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۱:۵

سلام
من اولین بار که اجرا می‌کنم تصویر را نشون نمی‌ده به این صورت نمایش میده :

بعد از لود صفحه یک بار که صفحه را Refresh کنم همه چیز درست میشه ؟
در قسمت Console هم خطایی وجود ندارد ....
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۲۹ ۱۱:۲۹

حداقل دو علت می‌تونه داشته باشه:
الف) تصاویر رو نمی‌تونه پیدا کنه، یا صفحه کش شده بیش از حد. قسمت «اجرای کدهای jQuery Ajax فوق، چه تغییری را در صفحه سبب می‌شوند؟» را بررسی کنید که چه آدرسی توسط کدهای جی‌کوئری در حال پردازش است.
همچنین کش شدن نتایج قبلی رو هم می‌شود غیرفعال کرد:

```
$.ajax({
  cache: false /* گاهی از اوقات خصوصا برای آی ای نیاز است */
});
```

ب) چند وقت قبل در یکی از بحث‌های سایت دیدم که مورد زیر رعایت نشده بود و کدهای جی‌کوئری کار نمی‌کردند:

```
<script type="text/javascript">
        $(function () {
            // کدهای جی‌کوئری در اینجا
        });
</script>
```

اجرای کدهای جی‌کوئری نیازی به DOM حاضر و آماده دارند که توسط متد document ready آن مانند کدهای فوق باید تدارک دیده شود. نیازی به این کد نخواهد بود اگر اسکریپت‌ها در آخر صفحه و پیش از بسته شدن تگ body اضافه بشن.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۰۷/۰۷ ۱۰:۵۸

با تشکر از مطلب بسیار کاربردیتون ، در هنگام استفاده از URL Routing همانطور که قبلا راهنماییم کرده بودید برای آدرس فایل‌های جاوا اسکریپت از

```
<%=ResolveUrl("~/App_Themes/MainTheme/jquery.js")%>
```

استفاده کردم و مشکل حل شد ولی برای یو آر ال این تصاویر آیکون که در Jquery تعریف شدند میتونید کمکم کنید

```
<script type="text/javascript">
    $(document).ready(function () {

        $('#exampleMenu').sweetMenu({
            top: 200,
            padding: 8,
            iconSize: 48,
            easing: 'easeOutBounce',
            duration: 500,
            icons: [
'images/home.png',
'images/comments.png',
'images/red_heart.png',
'images/computer.png',
'images/male_user.png',
'images/yellow_mail.png'
]
        });
    });
</script>
```

اگر آدرس آیکون‌ها را به صورت

'http://site.ir/images/home.png'

تعریف کنم مشکل حل میشه ولی فکر کنم راه حل درستی نباشه . بسیار ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۷ ۱۱:۲۸

امکان قرار دادن کدهای سمت سرور داخل اسکریپت‌ها هم هست؛ مثلا:

```
data: "{'username': '" + $('#<%= TextBox1.ClientID %>').val() + "'}",
```

به این شرط که این اسکریپت داخل صفحه runat=server دار باشد یا داخل head ایی با این مشخصات.

عنوان: **نحوه کار با ftp - بخش اول**
نویسنده: بهمن آبادی
تاریخ: ۱۳۹۲/۰۲/۱۱ ۱۹:۵۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, C#, system.net, ftp

امروز می‌خوام نحوه کار با FTP بصورت ساده برای کاربران و برنامه نویسان مبتدی رو آموزش بدم.

برای استفاده از FTP نیاز به یک اکانت FTP در سایت مورد نظر بهمراه دسترسی به پوشه ای مشخص می‌باشد.
برای مثال ما یک اکانت FTP در سایت dotnettip.info داریم که به پوشه upload دسترسی داره.

ابتدا در فایل Web.config و در بین تگ های appSettings مقادیر زیر را برای دسترسی به اکانت و نام کاربری و رمز عبور ذخیره می‌کنیم.

```
<add key="FtpAddress" value="ftp://ftp.dotnetips.info" />
<add key="FtpUser" value="uploadcenter" />
<add key="FtpPass" value="123123" />
<add key="FolderPath" value="~/Upload/" />
```

*نکته : برای امنیت بیشتر و دسترسی به اطلاعات اکانت می‌شود از روش‌های دیگری نیز استفاده کرد.

در ادامه یک کلاس در App_code پروژه خود با نام FTPHelper ایجاد می‌کنیم و کد زیر را در آن قرار می‌دهیم:
تکه کد بالا برای ست کردن مقادیر نام کاربری و رمز عبور و آدرس FTP در کلاس مذکور که بصورت پیش‌فرض از web.config پر می‌شود ایجاد و بکار خواهد رفت.

```
using System.Net;
using System.IO;
using System.Configuration;


public class FtpHelper
{
public FtpHelper()
{
       //Default Value Set From Application
         _hostname = ConfigurationManager.AppSettings.GetValues("FtpAddress")[0];
        _username = ConfigurationManager.AppSettings.GetValues("FtpUser")[0];
        _password = ConfigurationManager.AppSettings.GetValues("FtpPass")[0];
}

    #region "Properties"
    private string _hostname;
    /// <summary>
    /// Hostname
    /// </summary>
    /// <value></value>
    /// <remarks>Hostname can be in either the full URL format
    /// ftp://ftp.myhost.com or just ftp.myhost.com
    /// </remarks>
    public string Hostname
    {
        get
        {
            if (_hostname.StartsWith("ftp://"))
            {
                return _hostname;
            }
            else
            {
                return "ftp://" + _hostname;
            }
        }
        set
        {
            _hostname = value;
        }
    }
    private string _username;
```

```
    /// <summary>
    /// Username property
    /// </summary>
    /// <value></value>
    /// <remarks>Can be left blank, in which case 'anonymous' is returned</remarks>
    public string Username
    {
        get
        {
            return (_username == "" ? "anonymous" : _username);
        }
        set
        {
            _username = value;
        }
    }
    private string _password;
    public string Password
    {
        get
        {
            return _password;
        }
        set
        {
            _password = value;
        }
    }


    #endregion
}
```

سپس فضای نام‌های زیر را در کلاس خود قرار می‌دهیم.

```
using System.Net;
using System.IO;
```

حالا برای بارگذاری فایل می‌توانیم از یک تابع بصورت shared استفاده کنیم که بتوان با دادن آدرس فایل بصورت فیزیکی به تابع و مشخص کردن پوشه مورد نظر آنرا در هاست مقصد (FTP) بارگذاری کرد.توجه داشته باشید که تابع فوق نیازی به قرار گرفتن در کلاس بالا (FtpHelper) ندارد.یعنی می‌توان آنرا در هرجای برنامه پیاده سازی نمود.

```
public static bool Upload(string fileUrl)
        {
            if (File.Exists(fileUrl))
            {
                FtpHelper ftpClient = new FtpHelper();
                string ftpUrl = ftpClient.Hostname + System.IO.Path.GetFileName(fileUrl);


                FtpWebRequest ftp = (FtpWebRequest)FtpWebRequest.Create(ftpUrl);
                ftp.Credentials = new NetworkCredential(ftpClient.Username, ftpClient.Password);

                ftp.KeepAlive = true;
                ftp.UseBinary = true;
                ftp.Timeout = 3600000;
                ftp.KeepAlive = true;
                ftp.Method = WebRequestMethods.Ftp.UploadFile;

                const int bufferLength = 102400;
                byte[] buffer = new byte[bufferLength];
                int readBytes = 0;

                //open file for reading
                using (FileStream fs = File.OpenRead(fileUrl))
                {
                    try
                    {
                        //open request to send
                        using (Stream rs = ftp.GetRequestStream())
```

```
                    {
                        do
                        {
                            readBytes = fs.Read(buffer, 0, bufferLength);
                            fs.Write(buffer, 0, readBytes);
                        } while (!(readBytes < bufferLength));
                        rs.Close();
                    }

                }
                catch (Exception)
                {
                 //Optional Alert for Exeption To Application Layer
                   //throw (new ApplicationException("بارگذاری فایل با خطا  رو به رو شد"));

                }
                finally
                {
                    //ensure file closed
                    //fs.Close();
                }

            }

            ftp = null;
            return true;
        }
        return false;



    }
```

تکه کد بالا فایل مورد نظر را در صورت وجود به صورت تکه‌های 100 کیلوبایتی بر روی ftp بارگذاری می‌کند، که می‌توانید مقدار آنرا نیز تغییر دهید.
اینکار باعث افزایش سرعت بارگذاری در فایل‌های با حجم بالا برای بارگذاری می‌شود.

در بخش‌های بعدی نحوه ایجاد پوشه ، حذف فایل ، حذف پوشه و دانلود فایل از روی FTP را بررسی خواهیم کرد.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۲/۱۱ ۲۲:۳۸

ممنون از مطلب شما.
چند نکته جزئی در مورد کدهای تهیه شده:
- وجود این try/catch در اینجا هیچ هدفی رو برآورده نکرده. از قسمت throw ex هم توصیه می‌شود که استفاده نکنید. از thow خالی استفاده کنید تا stack trace پاک نشه. به علاوه زمانیکه مشغول به طراحی یک کتابخانه هستید تا حد ممکن از ذکر try/catch خودداری کنید. وظیفه بررسی این مسایل مرتبط است به لایه‌های بالاتر استفاده کننده و نه کتابخانه پایه.
- if ابتدای متد هم ضرورتی ندارد. اگر قرار است باشد، باید به استفاده کننده در طی یک استثناء اعلام شود که چرا فایل درخواستی او آپلود نشده. در کل استفاده از متد File.Exists به همراه صدور استثناء در صورت عدم وجود فایل، در اینجا مناسب‌تر است.
- نامگذاری‌هایی مانند obj_ftp مربوط به دوران C است. در سی‌شارپ روش دیگری رو باید استفاده کنید که در مطلب اصول نامگذاری در دات نت به تفصیل بررسی شده.
- بررسی صفر بودن readBytes بهتر است پیش از فراخوانی متد Write انجام شود.
- یک سری از اشیاء در دات نت پیاده سازی کننده IDispoable هستند. به این معنا که بهتر است از using برای استفاده از آن‌ها کمک گرفته شود تا کامپایلر قسمت finally به همراه آزاد سازی منابع را به صورت خودکار اضافه کند. این نکته برای مواردی که در بین کار استثنایی رخ می‌دهد جهت آزاد سازی منابع لازم است. یعنی بهتر بود بجای try/catch از try/finally و یا using در مکان‌های مناسب استفاده می‌شد.
- علت استفاده از شیء Application دراینجا چه چیزی بوده؟ AppSettings خوانده شده از وب کانفیگ برنامه و کل اطلاعات آن در آغاز به کار یک برنامه ASP.NET به صورت خودکار کش می‌شوند. به همین جهت است که اگر حتی یک نقطه در فایل وب کانفیگ تغییر کند برنامه ASP.NET ری استارت می‌شود (تا دوباره تنظیمات را بخواند). بنابراین مستقیما از همان امکانات ConfigurationManager بدون انتساب آن به شیء سراسری Application استفاده کنید. اینکار سربارى آنچنانی هم ندارد؛ چون از حافظه خوانده می‌شود و نه از فایل. هر بار فراخوانی ConfigurationManager.AppSettings به معنای مراجعه به فایل web.config نیست. فقط بار اول اینکار انجام می‌شود؛ تا زمانیکه این فایل تغییر کند یا برنامه ری استارت شود.

نویسنده: بهمن آبادی
تاریخ: ۱۳۹۲/۰۲/۱۲ ۳:۳۰

با تشکر از نظر شما. تمام صحبت‌های شما منطقی است. اولا من این بخش رو می‌خواستم بعدا تکمیل کنم ولی بدلایلی زودتر ارسال کردم.

مواردی هم که اعلام کزدید کاملا صحیح می‌باشد.فقط بخش تابعی که درون آن از try/catch استفاده شده رو میخواستم ابتدا در تیکه کد دیگری از یک صفحه aspx بنویسم که وقتی برای بررسی مجدد تابع پیدا نکردم وگرنه اونجا try/catch بلا استفاده است.

موارد متذکر شده برای سهولت بیشتر کاربران در کدهای فوق لحاظ و ویرایش گردید.
با تشکر.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۱:۴

```
.locals init ([0] class [mscorlib]System.IO.TextWriter w)
  IL_0000:  ldstr      "log.txt"
  IL_0005:  call       class [mscorlib]System.IO.StreamWriter
      [mscorlib]System.IO.File::CreateText(string)
  IL_000a:  stloc.0

 .try
  {
    IL_000b:  ldloc.0
```

```
    IL_000c:  ldstr      "This is line one"
    IL_0011:  callvirt   instance void [mscorlib]
      System.IO.TextWriter::WriteLine(string)
    IL_0016:  leave.s    IL_0022
  }  // end .try
  finally
  {
    IL_0018:  ldloc.0
    IL_0019:  brfalse.s  IL_0021
    IL_001b:  ldloc.0
    IL_001c:  callvirt   instance void [mscorlib]
      System.IDisposable::Dispose()
    IL_0021:  endfinally
  }  // end handler
```

ماخذ

زمانیکه از using statement استفاده می‌کنید، خود کامپایلر try/finally رو اضافه می‌کنه. یعنی قسمت close الان بهش نیازی نیست چون استریم مورد استفاده حتما در اینجا dispose میشه.

نویسنده: بهمن آبادی
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۲:۵۰

از دستم در رفته ...شما ببخشید.

بیشتر هدفم روند بارگذاری و تکنیک بارگذاری اون بود ولی خوب درست کد نوشتن کاملاً صحیحه و من می‌خواستم برای کاربرای آماتور بنویسم ولی مجبورم کردین که کاربرای حرفه ای هم مثل شما به کد ایرادی نباشه.

بازم ممنون از دقت و راهنماییتون .

حالا بهتر شد نسبت به قبلیه ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۲/۱۲ ۱۷:۲۸

من فکر می‌کنم اگر ReSharper رو نصب کنید، پیش از ارائه یک مطلب یا پروژه خیلی از ایرادات رو با خط کشیدن زیر اون یا نمایش یک علامت زرد کنار صفحه گوشزد می‌کنه. مثلا می‌گه که این نوع نامگذاری درست نیست یا این شیء رو میشه با using محصور کرد. خلاصه از دستش ندید، حیفه!

نویسنده: سمیرا قادری
تاریخ: ۱۳۹۲/۰۶/۱۳ ۱۲:۳۷

با تشکر از مطلب خوبتان
ببخشید این روش کار کردن با Ftp چه مزایایی دارد و چه کاربردی دارد؟ چون من تا حالا فقط از Cute Ftp استفاده کردم آن هم برای Upload سایت بعد از Publish

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۱۳ ۱۲:۵۱

یه خورده وقت بذاری، باهاش می‌تونی یک Cute Ftp بنویسی.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۲/۲۰ ۱۵:۵۰

با تشکر از مطلب خوبتون ،
برای اینکه به طور مستقیم از file.SaveAs استفاده کنیم و فایل را مستقیم به Ftp آپلود کنیم (نه اینکه از فضای ذخیره شده ای کپی کنیم) چه پیشنهادی می‌دهید ؟ مثلا ما هاست دانلود داریم و میخواهیم فایل‌های ارسالی را در هاست دانلود ذخیره کنیم.

عنوان: **Image Annotations**
نویسنده: سجـــــاد ف
تاریخ: ۱۳۹۲/۰۳/۱۷ ۰:۱۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, jQuery, jQuery Ajax, Data Annotation, jQuery-Plugins

Add Note

می‌خواهیم با تغییر jQuery Image Annotation این پلاگین و برای asp.net استفاده کنیم

ایجاد دیتابیس

ابتدا یک دیتابیس به نام Coordinates ایجاد کنید و سپس جدول زیر رو ایجاد کنید

```
USE [Coordinates]
GO
CREATE TABLE [dbo].[Coords2](
[top] [int] NULL,
[left] [int] NULL,
[width] [int] NULL,
[height] [int] NULL,
[text] [nvarchar](50) NULL,
[id] [uniqueidentifier] NULL,
[editable] [bit] NULL
) ON [PRIMARY]
GO
```

ایجاد کلاس Coords برای خواندن و ذخیره اطلاعات

```
public class Coords
{
    public string top;
    public string left;
    public string width;
    public string height;
    public string text;
    public string id;
    public string editable;

    public Coords(string top, string left, string width, string height, string text, string id, string
editable)
    {
        this.top = top;
        this.left = left;
        this.width = width;
        this.height = height;
        this.text = text;
        this.id = id;
        this.editable = editable;


    }
}
```

فرم اصلی برنامه شامل 3 وب سرویس به شرح زیر می‌باشد

1-GetDynamicContext

این متد در زمان لود اطلاعات از دیتابیس استفاده می‌شود (وقتی که postback صورت می‌گیرد)

```
[WebMethod]
    public static List<Coords> GetDynamicContext(string entryId, string entryName)
    {
        List<Coords> CoordsList = new List<Coords>();

        string connect = "Connection String";
        using (SqlConnection conn = new SqlConnection(connect))
        {
            string query = "SELECT [top], [left], width, height, text, id, editable FROM  Coords2";
            using (SqlCommand cmd = new SqlCommand(query, conn))
            {
                conn.Open();
                using (SqlDataReader reader=cmd.ExecuteReader())
                {
                    while (reader.Read())
                    {
                        CoordsList.Add(new Coords(reader["top"].ToString(), reader["left"].ToString(),
                                                  reader["width"].ToString(),
reader["height"].ToString(),
                                                  reader["text"].ToString(), reader["id"].ToString(),
                                                  reader["editable"].ToString()));
                    }
                }

               conn.Close();
            }
        }

         return CoordsList;


    }
```

2,3 -SaveCoordsو DeleteCoords

این دو متد هم واسه ذخیره و حذف می‌باشند که نکته خاصی ندارند و خودتون بهینه اش کنید(در فایل ضمیمه موجودند)

تغییر فایل jquery.annotate.js جهت فراخوانی وب سرویس ها

فقط لازمه که سه قسمت زیر رو در فایل اصلی تغییر بدید

```
$.fn.annotateImage.ajaxLoad = function (image) {
        ///<summary>
        ///Loads the annotations from the "getUrl" property passed in on the
        ///     options object.
        ///</summary>

        $.ajax({
            type: "POST",
            contentType: "application/json; charset=utf-8",
            url: "Default.aspx/GetDynamicContext",
            data: "{'entryId': '" + 1 + "','entryName': '" + 2 + "'}",
            dataType: "json",
            success: function (msg) {
                image.notes = msg.d;
                $.fn.annotateImage.load(image);
            }
        });



 };
```

```
$.fn.SaveCoords = function (note) {
        $.ajax({
            type: "POST",
            contentType: "application/json; charset=utf-8",
            url: "Default.aspx/SaveCoords",
            data: "{'top': '" + note.top + "','left': '" + note.left + "','width': '" + note.width +
"','height': '" + note.height + "','text': '" + note.text + "','id': '" + note.id + "','editable': '" +
note.editable + "'}",
            dataType: "json",
            success: function (msg) {
                note.id = msg.d;
             }
        });



 };



 $.fn.annotateView.prototype.edit = function () {

        ///<summary>

        ///Edits the annotation.

        ///</summary>

        if (this.image.mode == 'view') {

            this.image.mode = 'edit';

            var annotation = this;

            // Create/prepare the editable note elements

            var editable = new $.fn.annotateEdit(this.image, this.note);

            $.fn.annotateImage.createSaveButton(editable, this.image, annotation);

            // Add the delete button

            var del = $('<a>حذف</a>');

            del.click(function () {

                var form = $('#image-annotate-edit-form form');

                $.fn.annotateImage.appendPosition(form, editable)

                if (annotation.image.useAjax) {
```

```
                $.ajax({
                    type: "POST",
                    contentType: "application/json; charset=utf-8",
                    url: "Default.aspx/DeleteCoords",
                    //                        url: annotation.image.deleteUrl,
                    //                        data: form.serialize(),
                    data: "{'id': '" + editable.note.id + "'}",
                    dataType: "json",
                    success: function (msg) {
                        //                image.notes = msg.d;
                        //                $.fn.annotateImage.load(image);
                    },
                    error: function (e) { alert("An error occured deleting that note.") }
                });
            }
            annotation.image.mode = 'view';
            editable.destroy();
            annotation.destroy();
        });
        editable.form.append(del);
        $.fn.annotateImage.createCancelButton(editable, this.image);
    }
};
```

این پروژه شامل یه سری فایل css هم هست که می‌تونید کل پروژه رو از اینجا دانلود کنید

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۷ ۰:۲۲

ضمن تشکر، فقط نکته استفاده از JSON.stringify در حین کار با jQuery Ajax رو بهتره اعمال کنید تا در دراز مدت و حالت‌های مختلف ورودی به مشکل برنخورید. به صورت خلاصه اطلاعات ارسالی رو جمع نزنید و تبدیل به رشته نکنید. یک شیء کامل درست کنید و اجازه بدید JSON.stringify اون رو تبدیل کنه.

نویسنده: یزدان
تاریخ: ۱۳۹۲/۰۳/۱۸ ۱۰:۳۵

اگر متدهای وب سرویس رو درون صفحاتی قرار بدیم که نیاز به لاگین کردن و احراز هویت جهت دسترسی به آنها باشد ، آیا میتوان متدهای وب سرویس رو خارج از اون صفحات فراخوانی نمود ؟
چه تضمینی وجود دارد که در خارج از صفحات با سطح دسترسی (حداقل لاگین) وب سرویسها فراخوانی نشوند ؟
به نظر شما MS AJAX کارش به پایان رسیده ؟
تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۳/۱۸ ۱۰:۵۴

- اگر دقت کرده باشید در کدهای فوق این متدها استاتیک تعریف شدن، یعنی مراحل چرخه طول عمر یک صفحه به آن‌ها اعمال نشده و اصلا جزئی از مباحث اعتبارسنجی صفحه جاری لحاظ نخواهند شد.
- در وب فرم‌ها استفاده از وب متدها یک روش برای کار با jQuery Ajax است. روش دوم استفاده از Generic handlerها و فایل‌های ashx است. در این موارد به علت استاتیک نبودن handlerهای تولیدی، می‌شود همه نوع اعتبارسنجی رو اعمال کرد اعم از روش Forms Authentication مثلا توسط context.Request.IsAuthenticated یا حتی روش منسوخ شده استفاده از سشن‌ها برای اعتبارسنجی با پیاده سازی IRequiresSessionState.
- در مطلب فوق اصلا از MS Ajax استفاده نشده. اون هم جایگاه خودش رو در کاربردهای خاص خودش دارد.

عنوان: **Cookie - قسمت سوم**
نویسنده: یوسف نژاد
تاریخ: ۱۳۹۲/۰۳/۳۱ ۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Cookie, Http Cookie

Cookie - قسمت اول : مقدمه، تاریخچه، معرفی، و شرح کامل

Cookie - قسمت دوم : کوکی در جاوا اسکریپت

نکته مهم: خواندن قسمت‌های قبلی این سری (مخصوصا قسمت اول) برای درک بهتر مطالب پیشنهاد می‌شود.

.

کوکی در ASP.NET - بخش اول

در قسمت‌های قبلی مقدمات و مباحث کلی راجع به کوکی‌ها و انواع آن، شرح کامل خواص، نحوه رفتار مرورگرها با انواع کوکی‌ها و درنهایت نحوه کار کردن با کوکی‌ها در سمت کلاینت با استفاده از زبان محبوب جاوا اسکریپت پرداخته شد.

در ادامه این سری مطالب به نحوه برخورد ASP.NET با کوکی‌ها و چگونگی کار کردن با کوکی در سمت سرور آشنا خواهیم شد. در بخش اول این قسمت مباحث ابتدایی و اولیه برای کار با کوکی‌ها در ASP.NET ارائه می‌شود. در بخش دوم مباحث پیشرفته‌تر همچون SubCookieها در ASP.NET و نیز سایر نکات ریز کار با کوکی‌ها در ASP.NET بحث خواهد شد.

.

Response و Request در ASP.NET

در قسمت اول این سری به مفاهیم Http Response و Http Request اشاره کوتاهی شده بود. به‌صورت خلاصه، درخواستی که از

سمت یک کلاینت به یک وب سرور ارسال می‌شود **Request** و پاسخی که وب سرور به آن درخواست می‌دهد **Response** نامیده می‌شود.

در ASP.NET، کلیه اطلاعات مرتبط با **درخواست** رسیده از سمت یک کلاینت در نمونه‌ای منحصر به فرد از کلاس HttpRequest نگه‌داری می‌شود. محل اصلی نگه‌داری این نمونه در پراپرتی Request از نمونه جاری کلاس System.Web.HttpContext (قابل دسترسی ازطریق HttpContext.Current) است. البته کلاس Page هم یک پراپرتی با نام Request دارد که دقیقا از همین پراپرتی کلاس HttpContext استفاده می‌کند.

هم‌چنین کلیه اطلاعات مرتبط با **پاسخ** ارسالی وب سرور به سمت کلاینت مربوطه در نمونه‌ای از کلاس HttpResponse ذخیره می‌شود. محل اصلی نگه‌داری این نمونه نیز در پراپرتی Response از نمونه جاری کلاس HttpContext است. همانند Request، کلاس Page یک پراپرتی با نام Response برای نگه‌داری این نمونه دارد که این هم دقیقا از پراپرتی متناظر در کلاس HttpContext استفاده می‌کند.

.

.

کوکی‌ها در Response و Request

هر دو کلاس HttpResponse و HttpRequest یک پراپرتی با عنوان Cookies (^ و ^) دارند که مخصوص نگهداری کوکی‌های مربوطه هستند. این پراپرتی از نوع System.Web.HttpCookieCollection است که یک کالکشن مخصوص برای ذخیره کوکی‌هاست.

- این پراپرتی (Cookies) در کلاس HttpRequest محل نگه‌داری کوکی‌های ارسالی توسط مرورگر در درخواست متناظر آن است. کوکی‌هایی که مرورگر با توجه به شرایط جاری و تنظیمات کوکی‌ها اجازه ارسال به سمت سرور را به آن‌ها داده و در درخواست ارسالی ضمیمه کرده است (با استفاده از هدر :Cookie که در قسمت اول شرح داده شد) و ASP.NET پس از پردازش و Parse داده‌ها، درون این پراپرتی اضافه کرده است.

- این پراپرتی (Cookies) در کلاس HttpResponse محل ذخیره کوکی‌های ارسالی از وب سرور به سمت مرورگر کلاینت در پاسخ به درخواست متناظر است. کوکی‌های درون این پراپرتی پس از بررسی و استخراج داده‌های موردنیاز توسط ASP.NET در هدر پاسخ ارسالی ضمیمه خواهند شد (با استفاده از هدر :Set-Cookie که در قسمت اول توضیح داده شد).

.

.

ایجاد و به‌روزرسانی کوکی در ASP.NET

برای ایجاد یک کوکی و ارسال آن به سمت کلاینت همان‌طور که در بالا نیز اشاره شد، باید از پراپرتی Cookies **Response.** از کلاس HttpContext استفاده کرد. برای ایجاد یک کوکی روش‌های مختلفی وجود دارد.
در روش اول با استفاده از ویژگی مخصوص **ایندکسر** کلاس HttpCookieCollection عملیات تولید کوکی انجام می‌شود. در این روش، ابتدا بررسی می‌شود که کوکی موردنظر در لیست کوکی‌های جاری وجود دارد یا خیر. درصورتی‌که با این نام قبلا یک کوکی ثبت شده باشد، مقدار کوکی موجود بروزرسانی خواهد شد. اما اگر این نام وجود نداشته باشد یک کوکی جدید با این نام به لیست افزوده شده و مقدار آن ثبت می‌شود. مثال:

```
HttpContext.Current.Response.Cookies["myCookie"].Value = "myCookieValue";
```

روش بعدی استفاده از متد Add در کلاس HttpCookieCollection است. در این روش ابتدا یک نمونه از کلاس HttpCookie ایجاد شده و سپس این نمونه به لیست کوکی‌ها اضافه می‌شود. کد زیر چگونگی استفاده از این روش را نشان می‌دهد:

```
var myCookie = new HttpCookie("myCookie", "myCookieValue");
HttpContext.Current.Response.Cookies.Add(myCookie);
```

روش دیگر استفاده از متد Set کلاس HttpCookieCollection است. تفاوت این متد با متد Add در این است که متد Set ابتدا سعی می‌کند عملیات update انجام دهد. یعنی عملیات افزودن تنها وقتی‌که نام کوکی موردنظر در لیست کوکی‌ها یافته نشود انجام خواهد شد. برای مثال:

```
HttpContext.Current.Response.Cookies.Set(new HttpCookie("myCookie", "myCookieValue"));
```

نکته: باتوجه به توضیحات بالا، متد Set اجازه افزودن دو کوکی با یک نام را نمی‌دهد. برای اینکار باید از متد Add استفاده کرد. درباره این موضوع در قسمت بعدی بیشتر توضیح داده خواهد شد.
روش دیگری که برای ایجاد یکی کوکی می‌توان از آن استفاده کرد، بکارگیری متد AppnedCookie از کلاس HttpResponse است. در این روش نیز ابتدا باید یک نمونه از کلاس HttpCookie تولید شود. این روش همانند استفاده از متد Add از کلاس HttpCookieCollection است. کد زیر مثالی از این روش را نشان می‌دهد:

```
HttpContext.Current.Response.AppendCookie(new HttpCookie("myCookie", "myCookieValue"));
```

روش بعدی استفاده از متد SetCookie از کلاس HttpResponse است. فرق این متد با متد AppendCookie در این است که در متد SetCookie ابتدا وجود یک کوکی با نام ارائه شده بررسی می‌شود و درصورت وجود، مقدار این کوکی بروزرسانی می‌شود. درصورتی‌که قبلا یک کوکی با این نام وجود نداشته باشد، یک کوکی جدید به لیست کوکی‌ها اضافه می‌شود. این روش همانند استفاده از متد Set از کلاس HttpCookieCollection است. نمونه‌ای از نحوه استفاده از این متد در زیر آورده شده است:

```
HttpContext.Current.Response.SetCookie(new HttpCookie("myCookie", "myCookieValue"));
```

نکته: تمامی فرایندهای نشان داده شده در بالا تنها موجب تغییر محتویات کالکشن کوکی‌ها درون HttpContext می‌شود و تا زمانی‌که توسط وب سرور با استفاده از دستور Set-Cookie به سمت مرورگر ارسال نشوند تغییری در کلاینت بوجود نخواهند آورد.

برای آشنایی بیشتر با این روند کد زیر را برای تعریف یک کوکی جدید درنظر بگیرید:

```
HttpContext.Current.Response.Cookies["myCookie"].Value = "myValue";
```

برای مشاهده هدر تولیدی توسط وب سرور می‌توان از نرم افزار محبوب Fiddler استفاده کرد (از اواخر سال 2012 که نویسنده این ابزار به Telerik پیوسته، توسعه آن بسیار فعال‌تر شده و نسخه‌های جدید با لوگوی جدید! ارائه شده است).
تصویر زیر مربوط به مثال بالاست:

همانطور که مشاهده می‌کنید دستور ایجاد یک کوکی با نام و مقدار وارده در هدر پاسخ تولیدی توسط وب سرور گنجانیده شده است.

نکته: در ASP.NET به صورت پیش فرض از مقدار "/" برای پراپرتی Path استفاده می‌شود.

.

خواص کوکی در ASP.NET

برای تعیین یا تغییر خواص یک کوکی در ASP.NET باید به نمونه HttpCookie مربوطه دست یافت. سپس با استفاده از پراپرتی‌های این کلاس می‌توان خواص موردنظر را تعیین کرد. برای مثال:

```
var myCookie = new HttpCookie(string.Empty);
myCookie.Name = "myCookie";
myCookie.Value = "myCookieValue";
myCookie.Domain = "dotnettip.info";
myCookie.Path = "/post";
myCookie.Expires = new DateTime(2015, 1, 1);
myCookie.Secure = true;
myCookie.HttpOnly = true;
```

نکته مهم: امکان تغییر خواص یک کوکی به صورت مستقیم در سمت سرور وجود ندارد. درواقع برای اعمال این تغییرات در سمت کلاینت باید به ازای هر کوکی موردنظر یک کوکی جدید با مقادیر جدید ایجاد و به کالکشن کوکی‌ها در Http Response مربوطه اضافه شود تا پس از قرار دادن دستور Set-Cookie متناظر در هدر پاسخ ارسالی به سمت کلاینت و اجرای آن توسط مرورگر، مقادیر خواص مورنظر در سمت کلاینت بروزرسانی شوند. دقت کنید که تمامی نکات مرتبط با هویت یک کوکی که در قسمت اول شرح داده شد در اینجا نیز کاملا صادق است.

روش دیگری نیز برای تعیین **برخی** خواص کوکی‌ها به صورت کلی در فایل وب کانفیگ وجود دارد. برای اینکار از تگ [httpCookies]

در قسمت system.web استفاده می‌شود. برای مثال:

```
<httpCookies domain="www.example.com" httpOnlyCookies="true" requireSSL="true" />
```

این امکان از ASP.NET 2.0 به بعد اضافه شده است. با استفاده از این تگ، تنظیمات اعمال شده برای تمامی کوکی‌ها درنظر گرفته می‌شود. البته درصورتی‌که تنظیم موردنظر برای کوکی به صورت صریح آورده نشده باشد. برای نمونه به کد زیر دقت کنید:

```
var myCookie = new HttpCookie("myCookie", "myCookieValue");
myCookie.Domain = "test.com";
HttpContext.Current.Response.Cookies.Add(myCookie);
var myCookie2 = new HttpCookie("myCookie2", "myCookieValue2");
myCookie2.HttpOnly = false;
myCookie2.Secure = false;
HttpContext.Current.Response.Cookies.Add(myCookie2);
```

با استفاده از تنظیمات تگ httpCookies که در بالا نشان داده شده است، هدر پاسخ تولیدی توسط وب سرور به صورت زیر خواهد بود:

همان‌طور که می‌بینید تنها مقادیر پراپرتی‌هایی که صراحتا برای کوکی آورده نشده است از تنظیمات وب کانفیگ خوانده می‌شود.

.

.

حذف کوکی در ASP.NET

برای حذف یک کوکی در ASP.NET یک روش کلی وجود دارد که در قسمت‌های قبلی نیز شرح داده شده است، یعنی تغییر خاصیت Expires کوکی به تاریخی در گذشته. برای نمونه داریم:

```
var myCookie = new HttpCookie("myCookie", "myCookieValue");
myCookie.Expires = DateTime.Now.AddYears(-1);
```

نکته مهم: در کلاس HttpCookieCollection یک متد با نام Remove وجود دارد. از این متد برای حذف یک کوکی از لیست موجود

در این کلاس استفاده می‌شود. دقت کنید که حذف یک کوکی از لیست کوکی‌ها با استفاده از این متد تاثیری بر موجودیت آن کوکی در سمت کلاینت نخواهد گذاشت و تنها روش موجود برای حذف یک کوکی در سمت کلاینت همان تنظیم مقدار خاصیت Expires است.

.

خواندن کوکی در ASP.NET

برای خواندن مقدار یک کوکی ارسالی از مرورگر کلاینت در ASP.NET، باتوجه به توضیحات ابتدای این مطلب، طبیعی است که باید از پراپرتی **Request.** Cookies در نمونه جاری از کلاس HttpContext استفاده کرد. برای این کار نیز چند روش وجود دارد.
روش اول استفاده از **ایندکسر** کلاس HttpCookieCollection است. برای اینکار نیاز به نام یا ایندکس کوکی موردنظر در لیست مربوطه داریم. برای مثال:

```
var myCookie = HttpContext.Current.Request.Cookies["myCookie"];
```

یا این نمونه با استفاده از **ایندکسر عددی** :

```
var myCookie = HttpContext.Current.Request.Cookies[0];
```

روش دیگری که برای خواند مقدار یک کوکی می‌توان بکار برد، استفاده از متد Get از کلاس HttpCookieCollection است. این متد همانند ایندکسر این کلاس نیاز به نام یا ایندکس کوکی موردنظر دارد. برای نمونه:

```
var myCookie = HttpContext.Current.Request.Cookies.Get("myCookie");
```

یا استفاده از ایندکس کوکی :

```
var myCookie = HttpContext.Current.Request.Cookies.Get(0);
```

.

.

بحث و نتیجه گیری

تا اینجا با مفاهیم اولیه درباره نحوه برخورد ASP.NET با کوکی‌ها آشنا شدیم. روش‌های مختلف ایجاد و یا به‌روزرسانی کوکی‌ها نشان داده شد. با تعیین انواع خواص کوکی‌ها آشنا شدیم. نحوه حذف یک کوکی در ASP.NET را دیدیم. روش‌های خواندن مقادیر کوکی‌ها را نیز مشاهده کردیم.

باز هم تاکید می‌کنم که تمامی تغییرات اعمالی در سمت سرور تا زمانی‌که به‌صورت دستورات Set-Cookie در هدر پاسخ وب سرور قرار نگیرند هیچ کاری در سمت کلاینت انجام نمی‌دهند.

در قسمت بعدی این سری مطالب به مباحث پیشرفته‌تری چون SubCookieها در ASP.NET و هویت منحصر به فرد کوکی‌ها در سمت سرور پرداخته می‌شود.

.

.

منابع

http://msdn.microsoft.com/en-us/library/ms178194(v=vs.100).aspx

http://msdn.microsoft.com/en-us/library/aa289495(v=vs.71).aspx

http://www.codeproject.com/Articles/31914/Beginner-s-Guide-To-ASP-NET-Cookies

http://www.codeproject.com/Articles/244904/Cookies-in-ASP-NET

نظرات خوانندگان

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۸/۰۶ ۱۸:۱۸

یک کوکی ایجاد کردم و یه سری دیتا درون اون قرار دادم. در جایی از سایت برای پرداخت اینترنتی به درگاه بانک متصل میشم اما بعد از برگشت از بانک ، وقتی میخوام مقدار کوکی رو بخونم ، کوکی مقدارش خالیه. حتی دستور Page.User.Identity.IsAuthenticate هم مقدار false بر می‌گردونه.در صورتی که قبل از ارسال به درگاه، کاربر لاگین بوده. ممنون میشم راهنمایی کنید.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۰:۹

در دات نت 4.5 ، مشکل طولانی بودن حاصل BinaryFormatter serialization برطرف شده (January 2013). این مشکل سبب می‌شده تا حاصل RolePrincipal.ToEncryptedTicket بسیار طولانی شده و بیشتر از حد مجاز اندازه قابل ذخیره سازی در یک کوکی شود.
- وصله‌ی نسخه‌ی ویندوز 8 و ویندوز سرور 2012 آن از اینجا قابل دریافت است؛ نسخه‌ی ویندوز 7 و ویندوز سرور 2008 از اینجا .
+ آپدیت ویندوز را روشن کنید تا آخرین به روز رسانی‌ها و نگارش‌های دات نت نصب شده را به صورت خودکار دریافت کنید.

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۲:۳۰

با تشکر؛ فقط یک نکته تکمیلی که فراموش کردم اینکه مشکل مورد نظر مربوط به asp.net web form هست و من از simple membership برای فرآیند احراز هویت استفاده میکنم.با توجه به این مسائل ، انجام مواردی که شما فرمودید برای حل مشکل کفایت میکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۲:۳۵

فرقی نمی‌کند. مباحث پایه Forms Authentication برای تمام فناوری‌هایی که از آن استفاده می‌کنند یکسان است.

عنوان: **ASP.NET FriendlyUrls و دریافت خطای 404**
نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۲/۰۴/۱۰ ۱۴:۱۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, FriendlyUrls

چند روز پیش تصمیم گرفتم از ASP.NET FriendlyUrls برای سفارشی کردن Urlهای سایت استفاده کنم

وقتی در لوکال تست میکردم همه چیز درست بود و بدون مشکل کار میکرد اما به محض اینکه سایت رو روی سرور Publish کردم و قصد پیمایش بین صفحات رو داشتم، با کلیک بر روی منوها با خطا 404 not found مواجه میشدم!

پس از کمی بررسی متوجه شدم که باید اسمبلی System.Web.Optimization.dll رو به وب سایت اد کنم و چند خط کد به web.config سایت اضافه کنم.

1- برای اضافه کردن System.Web.Optimization.dll به سایت روی References کلیک راست کنید و گزینه Manage Nuget Packages رو انتخاب کنید و سپس با جستجوی عبارت optimize از آیتم‌های پیدا شده Microsoft ASP.NET Web Optimization Framework رو نصب کنید.

2- خطوط زیر رو به web.config سایت اضافه کنید

```
<system.webServer>
<modules runAllManagedModulesForAllRequests="true">
<remove name="BundleModule" />
<add name="BundleModule" type="System.Web.Optimization.BundleModule" />
</modules>
</system.webServer>
```

با این تغییرات ایراد ASP.NET FriendlyUrls برطرف شد

نظرات خوانندگان

نویسنده: محسن تقی پور
تاریخ: ۱۳۹۲/۱۱/۰۴ ۱۸:۵۷

سلام
من همین کاریو که شما فرمودینو انجام دادم ولی بازم 404 میده
امکان داره دلیل دیگه ای داشته باشه ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۰۴ ۱۹:۲۴

- شخصی در نظرات بحث معرفی Friendly URLs عنوان کرده با نصب یک hot fix روی سرور مشکل خطای 404 آن حل شده.
- همچنین نصب URL Rewrite extension روی IIS هم ضروری است.

نویسنده: مجتبی صحرائی
تاریخ: ۱۳۹۲/۱۱/۰۴ ۲۰:۲۹

سلام
با توجه به بررسی هایی که انجام داده بودم، با استفاده از روش ذکر شده مشکلم برطرف شد و بیشتر از اون هم جستجو نکردم
در حال حاضر نظری ندارم و پیشنهاد میکنم توصیه آقای نصیری رو انجام بدید

نویسنده: محسن تقی پور
تاریخ: ۱۳۹۲/۱۱/۰۴ ۲۱:۴۳

ممنون از پاسخ دوستان
بله طبق بررسی بنده هاست من URL Rewrite extension نداره

عنوان: **ASP.NET FriendlyUrls و دریافت خطای Unauthorized 401 در حین عملیات Ajax**
نویسنده: حسین صفدری
تاریخ: ۱۳۹۲/۰۵/۰۶ ۲۳:۱۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Web Service, jQuery Ajax, SEO, FriendlyUrls

امروز از ASP.NET FriendlyUrls که دوست عزیزمون به اشتراک گذاشتن استفاده می‌کردم (اینجا) و در صفحه اول سایتم مجبور بودم با استفاده از JSON یک متد را از صفحه Defaul.aspx صدا بزنم که کد زیر را نوشتم:

```
$.ajax({
                    type: "POST",
                    url: "/Default.aspx/MyMethod",
                    data: "{}",
                    contentType: "application/json; charset=utf-8",
                    dataType: "json",
                    async: true,
                    cache: true,
                    success: function (data) {
                        $("#plandata").html(data.d);
                    },
                    error: function (x, e) {
                        alert("The call to the server side failed. " + x.responseText);
                    }
                });
```

هنگام کار، alert ایی بر روی صفحه ظاهر می‌شد و خطای Unauthorized 401 را نمایش می‌داد. حتی آدرس را به صورت Default/MyMethod/ امتحان کردم باز هم خطا داد و متد را شناسایی نمی‌کرد. به جای این کار کد زیر را در web.config اضافه کردم:

```
<system.web.extensions>
  <scripting>
    <webServices>
      <authenticationService enabled="true" />
    </webServices>
  </scripting>
</system.web.extensions>
```

با استفاده از راه حل مطلب اینجا ، به جای صفحه aspx یک صفحه WebService به پروژه اضافه و متدها را به آن جا منتقل کردم. نتیجه کد به شرح زیر شد:

```
$.ajax({
                    type: "POST",
                    url: "/websrv.asmx/MyMethod",
                    data: "{}",
                    contentType: "application/json; charset=utf-8",
                    dataType: "json",
                    async: true,
                    cache: true,
                    success: function (data) {
                        $("#plandata").html(data.d);
                    },
                    error: function (x, e) {
                        alert("The call to the server side failed. " + x.responseText);
                    }
                });
```

و خدا را شکر مشکل برطرف شد.

عنوان: رفع مشکل postBack در Master Page و دریافت خطای Operation is not valid due to the current state of the object

نویسنده: حسین صفدری

تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۳۵

آدرس: www.dotnettips.info

برچسب‌ها: ASP.Net, Tips, Handle Errors

با این خطا هنگامی مواجه شدم که در Master Page یک TextBox و یک دکمه قرار داده بودم و در رویداد دکمه به یک صفحه دیگر انتقال پیدا می‌کردم. کد زیر:

```
protected void imgBtnSearch_Click(object sender, ImageClickEventArgs e)
        {
            if (TxtbSearch.Text != string.Empty)
            {
                Response.Redirect("/Exhibition/ShowSearch.aspx?title=" +
                      Sanitizer.GetSafeHtmlFragment(TxtbSearch.Text) + "&b=true", false);
            }
        }
```

این کد در ماشین خودم کار می‌کرد و مشکلی نداشت. وقتی پروژه را به سرور انتقال می‌دادم و بر روی IIS 6 تست می‌کردم خطای 404 دریافت می‌کردم. ضمن اینکه به صفحه خواسته شده انتقال پیدا نمی‌کردم، و همه چیز کاملا گیج کننده بود. وقتی صفحه را از خود سرور باز و رویداد فوق را اجرا می‌کردم خطای Operation is not valid due to the current state of the object دریافت می‌کردم به همراه کد زیر در Track trace به شرح زیر:

```
at System.Web.HttpValueCollection.ThrowIfMaxHttpCollectionKeysExceeded()
at System.Web.HttpValueCollection.FillFromEncodedBytes(Byte[] bytes, Encoding encoding)
at System.Web.HttpRequest.FillInFormCollection()
```

با افزودن کد زیر در Web.config مشکل برطرف گردید.

```
<appSettings>
    <add key="aspnet:MaxHttpCollectionKeys" value="2001" />
  </appSettings>
```

منابع:

Code Project

stack overflow

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۸:۴۷

این مشکل زمانی به وجود میاد که تعداد انبوهی کنترل روی صفحه داشته باشید. ساختار صفحه شما به چه صورتی است؟ (کابین کنترل خلبان؟!)

نویسنده: حسین صفدری
تاریخ: ۱۳۹۲/۰۵/۰۸ ۱۹:۲

D: نه بابا!

گفتم در master Page و صفحه اول سایت.شامل منو، 3 تا repatear (تبلیغات، اخبار) و چندین باکس دیگه...

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۲/۰۵/۰۸ ۲۰:۳۸

خیلی ممنون بابت این نکته.

ولی دیگه سعی کنید که کوچ کنید به MVC. تو وب فرم آدم نمیدونه تو وب هست یا ویندوز فرم. یک وب مصنوعی.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۰۸ ۲۰:۴۶

مشکل وب فرم‌ها اینه که کل صفحه رو یک فرم درنظر می‌گیره. بنابراین موقع postback به سرور تمام صفحه رو ارسال می‌کنه که ممکنه سنگین باشه. در MVC می‌تونید در یک صفحه چندتا فرم داشته باشید با محدوده‌های عملکرد کوچک‌تر.

نویسنده: حسین صفدری
تاریخ: ۱۳۹۲/۰۵/۰۸ ۲۲:۷

چشم، مرسی نکات زیبایی دوستان گفتند، ولی راستش این پروژه برای 2 سال پیش بود، امروز به دلایلی یه گزینه جستجو اضافه کردم و باقی ماجرا ;-)

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت اول**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۰۹ ۱۸:۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Performance

در این سلسله مقالات قصد دارم چندین مطلب راجع به افزایش سرعت نرم افزارهای تحت وب مطرح نمایم. این مطالب هرچند بسیار مختصر می‌باشند ولی در کارایی و سرعت برنامه‌های شما در آینده تاثیر خواهند داشت.

1.کش کردن همیشه آخرین حربه می‌باشد

این مهم است که بخش‌های مختلف سایت شما در سطوح مختلف کش شوند (ASP.NET, Kernel, Server, Proxy Server, Browser ,...) ولی این موضوع باید همیشه آخرین حربه و نکته ای باشد که آن را در مورد سایت خود اعمال می‌کنید.
یعنی همیشه مطمئن شوید ابتدا تمامی نکات مربوط به افزایش کارایی در برنامه خود را رعایت کرده اید، سپس اقدام به کش داده‌ها در سطوح مختلف نمایید. توجه کنید کش کردن داده‌ها و صفحات می‌تواند مشکلات را برای شما به عنوان یک برنامه نویس یا تست کننده برنامه پنهان کند و به شما اطمینان دهد که همه چیز خوب کار می‌کند در حالی که این چنین نیست!
البته ذکر این نکته هم بی فایده نیست که کش کردن همه چیز بعضی مواقع دشمن برنامه شما محسوب می‌شود! هیچ وقت یادم نمی‌رود، در پورتال داخلی یک شرکت که در وقت استراحت به کارکنان اجازه مطالعه روزنامه‌های روز را می‌داد (به صورت آفلاین)، این نکته در بالای صفحه آورده شده بود: «لطفا برای به روز رساندن صفحات روزنامه‌ها از کلید Ctrl+F5 استفاده نمایید». این موضوع یعنی بحث کشینگ در برنامه آن پرتال در سطح فاجعه می‌باشد! حالا فرض کنید این مشکل در فرم ورود و یا مرور اطلاعات یک برنامه به وجود آید...

2.حذف View Engineهای غیر ضروری

به عنوان یک برنامه نویس ASP.NET MVC، یابد اطلاع داشته باشید که CLR به صورت خودکار View Engineهای Razor و Web Forms را لود می‌کند. این موضوع به این دلیل است که اطلاعی از نحوه برنامه نویسی شما ندارد. اگر شما فقط از یکی از این دو View Engine استفاده می‌کنید،لطفا دیگری را غیر فعال کنید! فعال بودن هر دوی آنها یعنی اتلاف وقت گرانبهای CPU سرور شما برای رندر کردن تمامی صفحات شما توسط دو انجین! ایتدا viewهای شما با Web Forms Engine رندر شده سپس نتیجه به Razor Engine منتقل شده و مجدد توسط این انجین رندر می‌شود. این موضوع در سایت‌های با تعداد کاربر بالا یعنی فاجعه!
برای حل این مشکل کافی است خطوط زیر را در فایل Global.asax و در رویداد بخش Application_Start وارد نمایید:

```
ViewEngines.Engines.Clear();
ViewEngines.Engines.Add(new RazorViewEngine());
```

این دو خط یعنی خداحافظ Web Forms Engine...
قبل از استفاده از این کد، اطمینان حاصل کنید کل برنامه شما توسط Razor تهیه شده است وگرنه بنده هیچ مسئولیتی در رابطه با فریادهای کارفرمای شما متقبل نمی‌شوم!
صد البته برای حذف Razor Engine و استفاده از Web Form Engine می‌توان از کد زیر در همان موقعیت فوق استفاده کرد:

```
ViewEngines.Engines.Clear();
ViewEngines.Engines.Add(new WebFormViewEngine());
```

البته همانطور که حتما دوستان مطلع هستند امکان گسترش Engineهای فوق توسط ارث بری از کلاس BuildManagerViewEngine جهت ایجاد Engineهای دیگر همیشه محیا است. در این صورت می‌توانید تنها انجین سفارشی مورد نظر خود را لود کرده و از لود دیگر انجین‌ها پرهیز کنید.

3. استفاده از فایل‌های PDB مایکروسافت برای دیباگ و یا پروفایل کردن DLLهای دیگران

دیباگ یا پروفایل کردن برنامه ها، DLL ها، اسمبلی‌ها و منابعی از برنامه که شما آن را خود ننوشته اید (سورس آنها در دسترس شما نمی‌باشد) همیشه یکی از سخت‌ترین مراحل کار می‌باشد. جهت کمک به دیباگرها یا پروفایلرها، نیاز است فایل‌های PDB مرتبط با DLLها را در اختیار آنها قرار دهید تا به بهترین نتیجه دسترسی پیدا کنید. این فایل‌ها محتوی نام توابع، شماره خطوط برنامه و metadataهای دیگر برنامه اصلی قبل از optimize شدن توسط کامپایلر یا JIT می‌باشد. خوب حالا اگر نیاز شد این کار را در رابطه با DLLها و کلاس‌های پایه Microsoft.NET انجام دهیم چه کار کنیم؟

خیلی ساده! خود Microsoft سروری جهت این موضوع تدارک دیده که فایل‌های PDB را جهت دیباگ کردن در اختیار تیم‌های برنامه نویسی قرار می‌دهد.کافی است از منوی Tools گزینه Options را انتخاب، سپس به بخش Debugging و به بخش Symbols بروید و گزینه Microsoft Symbol Servers as your source for Symbols را انتخاب کنید. برای اطمینان از اینکه هر مرتبه که برنامه را دیباگ می‌کنید مجبور به دانلود این فایل‌ها نشوید، فراموش نکنید پوشه ای را جهت کش این فایل‌ها ایجاد و آدرس آن را در بخش Cache symbols in this directory همین صفحه وارد نمایید.

این امکان در Visual Studio 2010, 2012 در دسترس می‌باشد.

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۰ ۱۷:۱۴

اثر View Engineهای اضافی رو با Glimpse بهتر میشه دید.

عنوان: چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت دوم
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۰ ۱۸:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Performance

قسمت اول

4. فشرده سازی HTTP را فعال کنید

اطمینان حاصل کنید که HTTP Compression در تمامی بخش‌های اصلی برنامه شما فعال است. حداقل کاری که می‌توانید در این رابطه بکنید این است که خروجی HTML که توسط برنامه شما تولید می‌شود را فشرده سازی کنید. جهت فعال سازی فشرده سازی در برنامه خود بهتر است در اولویت اول از ماژول ویژه ای که جهت این کار در IIS در نظر گرفته شده استفاده کنید. این ماژول تمامی کارها را به صورت خودکار برای شما انجام می‌دهد. اگر دسترسی به IIS جهت فعال سازی آن را ندارید، می‌توانید از ماژول‌های ASP.NET که جهت این کار تهیه شده استفاده کنید. می‌توانید کمی جستجو کنید و یا خودتان یکی تهیه کنید!

5.تنظیم CacheControlMaxAge

مقدار CacheControlMaxAge را در فایل web.config را طوری تنظیم کنید تا هیچ کاربری هیچ فایل static را دیگر درخواست نکند. مثلا می‌توانید این مقدار را بر روی چند ماه تنظیم کنید و البته فراموش نکنید این مقدار را در صفحات پویای خود بازنویسی (override) کنید تا مشکلی در رابطه با کش شدن فرم‌های اصلی برنامه (همانطور که در نکته اول بخش اول ذکر شد) پدید نیاید. البته کش کردن فایل‌های استاتیک برنامه بار مالی نیز برای شما و کاربرانتان خواهد داشت. دیگر هزینه پهنای باند اضافی جهت دانلود این فایل‌ها در هر درخواست برای شما (در سمت سرور) و کاربرانتان (در سمت کاربر) پرداخت نخواهد شد!

6. استفاده از OutputCache

اگر از MVC استفاده می‌کنید، فراموش نکنید که از OutputCache در کنترل‌های MVC استفاده نمایید. اگر سرور شما بتواند اطلاعات را از رم خود بازیابی کند بهتر از آن است که آن را مجدد از دیتابیس واکشی نماید و عملیاتی نیز بر روی آن انجام دهد. البته مدیریت حافظه .NET به صورت خودکار کمبود حافظه را مدیریت کرده و از نشت حافظه جلوگیری خواهد کرد. برای توضیحات بیشتر در این رابطه می‌توانید از این مقاله کمک بگیرید.

7. بهره برداری از ORM Profiler

ORM Profiler ها تمامی فعالیت‌های ORM تحت نظر گرفته، دستورات T-SQL ارسالی به بانک اطلاعاتی را واکشی کرده و برای شما نمایش می‌دهند. تعدادی از آنها نیز این دستورات را آنالیز کرده پیشنهاداتی در رابطه با بهبود کارایی به شما ارائه می‌دهند. برای مثال به جای اینکه شما 2000 رکورد را یکی یکی از بانک بازیابی کنید، می‌توانید آن را به صورت یک query به بانک ارسال کنید. این موضوع به سادگی توسط ORM Profilerها قابل بررسی است. نمونه ای از این نرم افزارها را می‌توانید در این سایت یا این سایت پیدا کنید. البته در صورتی که نمی‌خواهید از نرم افزارهای جانبی استفاده کنید، می‌توانید از ابزارهای توکار بانک‌های اطلاعاتی مانند SQL Profiler نیز استفاده کنید (راهنمایی).

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت سوم**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۱ ۱۲:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Performance

قسمت دوم

8.ORM Lazy Load

در هنگام استفاده از ORMها دقت کنید کجا از Lazy Load استفاده می‌کنید. Lazy Load باعث می‌شود وقتی شما اطلاعات مرتبط را از بانک اطلاعات واکشی می‌کنید، این واکشی اطلاعات در چند query از بانک انجام شود. درعوض عدم استفاده از Lazy Load باعث می‌شود تمامی اطلاعات مورد نیاز شما در یک query از بانک اطلاعاتی دریافت شود. این موضوع یعنی سربار کمتر در شبکه، در بانک اطلاعاتی، در منابع حافظه و منابع پر ارزش cpu در سرورها. البته استفاده از include در حالت فعال بودن یا نبودن lazy هم داستان مجزایی دارد که اگر عمری باقی باشد راجع به آن مقاله ای خواهم نوشت.
به این نمونه دقت کنید:

```
List<Customer> customers = context.Customers.ToList();
foreach (Customer cust in context.Customers){
   Console.WriteLine("Customer {0}, Account {1}", cust.Person.LastName.Trim() + ", " +
cust.Person.FirstName, cust.AccountNumber);
}
```

همچین کدی (در صورت فعال بودن Lazy Load در ORM) در صورتی که جدول Customers دارای 1000 رکورد باشد، باعث می‌شود برنامه 1001 دستور sql تولید و در بانک اجرا گردد.
برای اطلاع بیشتر می‌توانید به این مقاله مراجعه نمایید.

9.استفاده از MiniProfiler

سعی کنید از MiniProfiler در تمامی پروژه‌ها استفاده کنید. البته وقتی نرم افزار را در اختیار مصرف کننده قرار می‌دهید، آن را غیر فعال کنید. می‌توانید از متغیرهای compiler برای مجزا کردن buildهای متفاوت در برنامه خود استفاده کنید:

```
#if DEBUG then
// فعال سازی MiniProfiler
#endif
```

ایده دیگری هم وجود دارد. شما می‌توانید MiniProfiler را برای کاربر Admin یا کاربر Debugger فعال و برای بقیه غیر فعال کنید.
در باب MiniProfiler مسائل زیادی وجود دارد که چند نمونه از آن در همین سایت در این مقاله و این مقاله در دسترس است.
البته می‌توانید از ابزارهای دیگری مانند Glimpse که در این زمینه وجود دارد نیز استفاده کنید. لب کلام این نکته استفاده از profiler برای نرم افزار خود می‌باشد.

10. Data Paging در بانک اطلاعاتی

هنگامیکه از کامپوننت‌های شرکت‌های دیگر (Third party) استفاده می‌کنید، اطمینان حاصل کنید که صفحه بندی اطلاعات در بانک اطلاعاتی انجام می‌شود. برای نمونه کاپوننت گرید شرکت Telerik چند نوع صفحه بندی را پشتیبانی می‌کند. صفحه بندی سمت کاربر (توسط JavaScript)، صفحه بندی سمت سرور توسط کامپوننت و صفحه بندی مجازی. صفحه بندی سمت کاربر یعنی تمامی اطلاعات از سرور به کاربر فرستاده شده و در سمت کاربر عمل صفحه بندی انجام می‌شود. این یعنی واکشی تمامی اطلاعات از بانک و در مورد نرم افزارهای پرکاربر با حجم اطلاعات زیاد یعنی فاجعه. صفحه بندی سمت سرور ASP.NET هم یعنی واکشی اطلاعات از سرور بانک به سرور برنامه و سپس صفحه بندی توسط برنامه. این موضوع هم ممکن است مشکلات زیادی را ایجاد نماید چون باید حداقل تمامی رکوردها از اولین رکورد تا آخرین رکورد صفحه جاری از بانک واکشی شود که این عمل علاوه بر ایجاد سربار شبکه، سربار IO در بانک اطلاعاتی و سربار cpu در سرور ASP.NET ایجاد می‌کند. استفاده از صفحه بندی مجازی، شما را قادر می‌کند بتوانیم اطلاعات را در بانک صفحه بندی کرده و فقط صفحه مورد نظر خود را از بانک واکشی کنیم.

این حالت مجازی در اکثر componentها که توسط شرکت‌های مختلف ایجاد شده وجود دارد ولی ممکن است نام‌های متفاوتی داشته باشد. برای این موضوع باید به راهنمای component خریداری شده مراجعه کنید و یا به فروم‌ها و... مراجعه نمایید.

11. بررسی تعداد کوئری‌های صادر شده در یک صفحه و تعداد رکوردهای بازگشت داده شده توسط آن‌ها

این به این معنا نیست که برای هر query یک context مجزا ایجاد کنید، منظور این است که به بهانه اینکه اطلاعات مختلفی از جداول مختلف مورد نیاز است، query خود را آن قدر پیچیده یا گسترده ننویسیم که یا process آن در بانک زمان و سربار زیادی ایجاد کند و یا حجم اطلاعات بلا استفاده ای را از بانک به سرور برنامه لود نماید. به جای این موضوع می‌توانید در یک یا چند context دستورات مجزای واکشی اطلاعات صادر کنید تا تنها اطلاعات مورد نیاز خود را واکشی نمایید. البته این موضوع باعث نشود که تعداد queryها مثلا به 1000 عدد برسد! یعنی باید فیمابین queryهای پیچیده و queryهای ساده ولی با تعداد یکی را که مناسبتر با پروژه است انتخاب کنید که این موضوع با تجربه و تست حاصل می‌شود.

نظرات خوانندگان

نویسنده: جواد
تاریخ: ۱۳۹۲/۰۵/۱۲ ۳:۱۵

"استفاده از صفحه بندی مجازی، شما را قادر می‌کند بتوانیم اطلاعات را در بانک صفحه بندی کرده و فقط صفحه مورد نظر خود را از بانک واکشی کنیم. این حالت مجازی در اکثر componentها که توسط شرکت‌های مختلف ایجاد شده وجود دارد "

میشه این رو بیشتر توضیح بدید که منظورتون چیه .یا باید تمامی اطلاعات رو بفرستیم بعد صفحه بندی کنه یا اینکه به ازای هر صفحه یک کوئری به بانک بفرسته و اطلاعات رو نشون بده . حالا این صفحه بندی مجازی کجا کاربرد داره .

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۲ ۱۵:۱۷

هیچ کامپوننتی وجود خارجی نداره که قسمت مدیریت سمت بانک اطلاعاتی رو هم خودش به تنهایی انجام بده. همین کنترل‌های پیش فرض ASP.NET رو هم اگر ازشون درست استفاده کنیم، مشکلات کارآیی ندارند. مثلا: (نکته مهمش Skip.Take.ToList استفاده شده هست)

واکشی اطلاعات به صورت chunk chunk (تکه تکه) و نمایش در ListView

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۵/۱۲ ۲۱:۲۴

چرا وجود نداره!
از گرید Kendo استفاده کنید اگر paging رو فعال کنید خودش مدیریت میکنه

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۲ ۲۲:۲۷

بله. مدیریت می‌کنه، نه به تنهایی. اینجا هم باید اطلاعات Skip و Take رو بهش بدی تا صفحه بندی کم هزینه‌ای رو اعمال کنه . خود GridView در وب فرم‌ها هم paging داره. مشکلش اینه که در حالت پیش فرض کل اطلاعات رو از سرور واکشی می‌کنه و بعد یک صفحه رو نمایش می‌ده. بحث اینه که گرید اگر قراره یک صفحه رو نمایش بده که 20 ردیف داره، بتونه فقط 20 رکورد رو واکشی کنه و نه کل اطلاعات رو و این نیاز به کوئری‌های خاصی در سمت سرور داره. یک نمونه‌اش رو در واکشی اطلاعات به صورت تکه تکه لینک دادم.

نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۲ ۲۲:۴۶

1. اکثر کنترل‌های ASP.NET WebForm قابلیت bind به DataSet را دارد. اگر از آنها در این مدل استفاده نمایید کار صفحه بندی به عهده کنترل+DataSet می‌افتد.
2. صفحه بندی مجازی یعنی شما به کنترل می‌گویید کل اطلاعات شما مثلا 51 صفحه است، الان صفحه 4 را نمایش می‌دهی این هم اطلاعات صفحه 4. بعد وقتی کاربر بر روی گزینه صفحه بعد کلیک کرد، به کنترل می‌گویید کل اطلاعات 51 صفحه است، الان صفحه 5 را نمایش می‌دهی و این هم اطلاعات آن.
برای مثال می‌توانی به این مثال در DataTables مراجعه کنید

نویسنده: مرتضی

تاریخ: ۱۳۹۲/۰۵/۱۳ ۰:۱۳

گفتم که خودش مدیریت می‌کنه **یعنی اینکه اطلاعات Skip , Take رو نمی‌خواد بهش بدی و خودش اینکار رو انجام میده** - من دارم ازش تو پروژه استفاده میکنم - **کل اطلاعات رو واکشی نمی‌کنه و همون 20رکورد فرضا صفحه 3 رو واکشی میکنه**

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۳ ۰:۳۱

می‌تونی یک مثال با پروفایل SQL نهایی آن ارائه بدی. مطابق بررسی که کردم و حداقل دو تا لینکی که دادم در مورد این کتابخانه، موارد پردازش Take و Skip سمت سرور اون خودکار نیست.

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۵/۱۳ ۴:۴۹

کد Razor - کد VB - خروجی SQL

```
@(Html.Kendo.Grid(Of Models.vProject).Name("ProjectsGrid") _
        .Columns(Sub(column)
                        With column
                            .Bound(Function(p) p.ProjectId).Hidden()
                            .Bound(Function(p) p.Supervisor).Title("ناظر")
                            .Bound(Function(p) p.MapNumber).Title("شماره نقشه")
                            .Bound(Function(p) p.MapCode).Title("کد نقشه")
                            .Bound(Function(p) p.NewStructureArea).Title("متراژ")
                            .Bound(Function(p) p.NumberOfFloors).Title("تعداد طبقات")
                            .Bound(Function(p) p.InsuranceName).Title("بیمه")
                        End With
                End Sub).Pageable(Sub(p)
                                        p.Enabled(True)
                                        p.Info(True)
                                        p.PageSizes(True)
                                        p.Messages(Sub(m)
                                                        m.Display("{0} - {1} رکورد {2} از")
                                                        m.Empty("رکوردی برای نمایش وجود ندارد")
                                                        m.Of("از")
                                                        m.Page("صفحه")
                                                        m.ItemsPerPage("رکورد در هر صفحه")
                                                        m.Refresh("بروزرسانی")
                                                End Sub)
                                    End Sub).Selectable(Sub(s)
s.Enabled(True).Mode(GridSelectionMode.Single).Type(GridSelectionType.Row)) _
                                                .DataSource(Sub(ds)
ds.Ajax.ServerOperation(True).Read("GetProjects", "Home") _
                                                        .Model(Sub(m) m.Id(Function(modelId)
modelId.ProjectId)))
    )
```

```
<HttpPost>
        Function GetProjects(<DataSourceRequest> request As DataSourceRequest) As JsonResult
            Return Json(db.vProjects.ToDataSourceResult(request))
        End Function
```

```
SELECT TOP (5) [Extent1].[ProjectId]        AS [ProjectId],
               [Extent1].[Supervisor]       AS [Supervisor],
               [Extent1].[MapCode]          AS [MapCode],
               [Extent1].[MapNumber]        AS [MapNumber],
               [Extent1].[EmployerName]     AS [EmployerName],
               [Extent1].[InsuranceName]    AS [InsuranceName],
               [Extent1].[NewStructureArea] AS [NewStructureArea],
               [Extent1].[NumberOfFloors]   AS [NumberOfFloors]
FROM   (SELECT [Extent1].[ProjectId]                                AS [ProjectId],
               [Extent1].[Supervisor]                               AS [Supervisor],
               [Extent1].[MapCode]                                  AS [MapCode],
               [Extent1].[MapNumber]                                AS [MapNumber],
               [Extent1].[EmployerName]                             AS [EmployerName],
               [Extent1].[InsuranceName]                            AS [InsuranceName],
               [Extent1].[NewStructureArea]                         AS [NewStructureArea],
               [Extent1].[NumberOfFloors]                           AS [NumberOfFloors],
               row_number() OVER (ORDER BY [Extent1].[ProjectId] ASC) AS [row_number]
        FROM   (SELECT [vProject].[ProjectId]        AS [ProjectId],
                       [vProject].[Supervisor]       AS [Supervisor],
```

```
                        [vProject].[MapCode]          AS [MapCode],
                        [vProject].[MapNumber]        AS [MapNumber],
                        [vProject].[EmployerName]     AS [EmployerName],
                        [vProject].[InsuranceName]    AS [InsuranceName],
                        [vProject].[NewStructureArea] AS [NewStructureArea],
                        [vProject].[NumberOfFloors]   AS [NumberOfFloors]
                FROM    [dbo].[vProject] AS [vProject]) AS [Extent1]) AS [Extent1]
WHERE  [Extent1].[row_number] > 5
ORDER  BY [Extent1].[ProjectId] ASC
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۳ ۸:۲۰

کد سمت سرور db.vProjects.ToDataSourceResult چطور تهیه شده؟ در موردش اینجا بحث شده . شما یک IQueryable باید در اختیارش قرار بدی (که از لحاظ لایه بندی کار مشکل داره) تا بر اساس اطلاعات شماره صفحه و غیره‌ای که از کلاینت می‌رسه خودش مباحث Take و Skip رو پیاده سازی کنه. در حقیقت این کتابخانه فقط یک متد الحاقی اضافه‌تر برای اینکار جهت مدیریت مباحث سمت سرور داره.

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۵/۱۳ ۱۳:۴۵

درسته محسن خان- متد الحاقی ToDataSourceResult در خواست رو می‌گیره و....
بحث سر این بود که هیچ کامپوننتی وجود خارجی نداره که قسمت مدیریت سمت بانک اطلاعاتی رو هم خودش به تنهایی انجام بده
آره شیء DataSourceRequest شامل PageSize , PageNumber , ... میشه
ولی دیگه ما کاری بهش نداریم و خودش صفحه بندی رو انجام می‌ده حالا به طریقی
نمیشود گفت اگر ما از IQueryable استفاده کردیم **حتما لایه بندی ما مشکل داره**

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۵/۱۳ ۱۴:۱۷

داره. بهش می‌گن leaky abstraction .

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۵/۱۳ ۱۴:۲۴

درسته بهش میگن leaky abstraction اما نمیگن 100% ایراد

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت چهارم**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۲ ۱۵:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Performance

قسمت سوم

12.استفاده از validation سمت کاربر

برای جلوگیری از ارسال و دریافت‌های متناوب اطلاعات به سرور، از validation سمت کاربر استفاده نمایید. فرم‌های html 5 قابلیت‌های چک کردن نوع ورودی‌ها را به صورت خودکار دارد ولی ازاتکای به آن پرهیز کنید چون ممکن است یا کاربران برنامه شما از مرورگری استفاده کنند که از html5 پشتیبانی نکند و یا پشتیبانی کاملی از آن نداشته باشند. برای حل این مشکل می‌توانید از کتابخانه هایی مانند JQuery و ابزارهایی مانند JQuery Validation استفاده کنید. البته در MVC استفاده وسیعی از JQuery Validation شده که می‌تواند مورد استفاده قرار گیرد.

فراموش نکنید می‌توانید از ابزارهایی مانند Regex برای چک کردن سختی کلمات عبور و... نیز در JavaScript بهره برداری نمایید. البته دقت کنید که حتما پیامی مرتبط با خطای به وقوع پیوسته در اختیار کاربر قراردهید تا بتواند آن را بر طرف کند در غیر این صورت بنده مسئولیتی راجع به از دست دادن کاربرانتان و یا عصبانیت کارفرما بر عهده نمی‌گیرم!

13.استفاده از validation سمت سرور

حتما به خود می‌گویید نویسنده دچار چندگانگی شخصیت شده است! ولی چنین نیست. این مطلب بیشتر از اینکه در رابطه با ایجاد سرعت بیشتر باشد مربوط به امنیت است. چون validation سمت کاربر به سادگی قابل دور زدن می‌باشد. اگر شما تنها validation را سمت کاربر انجام دهید و سمت سرور از آن چشم پوشی کنید، به سرعت تمام برنامه شما هک می‌شود. لطفا دقت کنید که **امنیت را فدای هیچ چیز نکنید.** این یک نکته کلیدی است. البته سوای اینکه این یک نکته امنیتی است، validation سمت سرور باعث می‌شود شما بخشی از درخواست‌ها را قبل از انجام process زیاد از گردونه خارج کنید و از ارسال اطلاعات اضافی به بانک و ایجاد سربار اضافی جلوگیری کنید.

14.چک کردن scriptهای مورد استفاده سمت کاربر

استفاده از master pageها بسیار سرعت کار را زیاد می‌کنند. بیشتر دوستان scriptهای سمت کاربر خود را در master page قرار می‌دهند تا در تمامی صفحات لود شوند. این موضوع از طرفی سرعت برنامه نویسی را زیاد می‌کند ولی از طرف دیگر به دلیل اینکه باعث می‌شود فایل‌های script در تمامی صفحات بارگذاری شوند، باعث هدر رفت منابع شبکه شما (و کاربرانتان)، ایجاد سربار حافظه و cpu در سمت کاربر و در نتیجه سرعت پایین‌تر برنامه شما خواهد شد. سخت گیری در این موضوع می‌تواند این باشد که حتی شما function اضافی هم در سمت کاربر نداشته باشید.

برخی ناظران پروژه به این موضوعات دقت زیادی می‌کنند. در پروژه ای که به عنوان ناظر بودم مجری همین کار را انجام داده بود و به دلیل نیاز مبرم کارفرما به سرعت برنامه، این بخش از نظر اینجانب مردود اعلام شده و مجری مجبور به نوشتن دوباره کدهای آن گردید.

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت پنجم**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۳ ۲۳:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Performance

قسمت چهارم

15.استفاده از using

اگر از objectهایی استفاده می‌کنید که interface مربوط به IDisposable را پیاده سازی کرده اند، حتما از عبارت using استفاده کنید. استفاده از دستور using باعث می‌شود زمانی که دیگر نیازی به object شما نباشد، به صورت خودکار از حافظه حذف شود و در روال جمع آوری زباله (GC) قرار گیرد. این عمل باعث حداقل رسیدن احتمال نشت حافظه در نرم افزار شما می‌شود. برای مثال:

```
using System;
using System.Text;

class Program
{
    static void Main()
    {
// Use using statement with class that implements Dispose.
using (SystemResource resource = new SystemResource())
{
    Console.WriteLine(1);
}
Console.WriteLine(2);
    }
}

class SystemResource : IDisposable
{
    public void Dispose()
    {
// The implementation of this method not described here.
// ... For now, just report the call.
Console.WriteLine(0);
    }
}
```

برای اطلاعات بیشتر می‌توانید از این مقاله استفاده کنید.

16.اطلاعات ارسالی توسط شبکه را به حداقل برسانید

حجم اطلاعات ارسالی به شبکه را به حداقل برسانید. ارسال اطلاعات در شبکه به معنی گذر اطلاعات شما از 7 لایه مختلف شبکه در رایانه شما، گذر از media شبکه، گذر مجدد از 7 لایه شبکه در رایانه مقصد می‌باشد. به این معنی که هرچه اطلاعات بیشتری در شبکه ارسال کنید، سربار بیشتری متوجه سیستم شما خواهد بود. برای رفع این مشکل از فشرده سازهای css و javascript استفاده کنید. این فشرده سازها فواصل خالی، دستورات اضافی و... را از کد شما حذف و حجم آن را به حداقل می‌رسانند.
کم کردن تعداد درخواست‌ها و در نتیجه کم کردن تعداد فایل‌های ارسالی از سرور به کاربر نیز حربه ای در این زمینه می‌باشد.
برای مقایسه فشرده سازها به صورت آنلاین و استفاده از بهترین آنها (متناسب کد شما) می‌توانید از این سایت استفاده کنید.
امکانات توکاری هم وجود دارد که در زمان اجرای برنامه css و javascript شما را فشرده و تلفیق می‌کند ولی با توجه به اینکه برای سرور در هر مرتبه فراخوانی سربار دارد (حتی در صورت کش کردن) اکیدا توصیه می‌شود از فشرده سازها قبل از اجرای برنامه (Pre-Compressed) به جای فشرده سازهای زمان اجرا (Run-time Compression)استفاده کنید.

نظرات خوانندگان

نویسنده: حسین حقیان
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۲:۳۸

با سلام و تشکر از مجموعه مطالب مرتبط که ارائه کردید
میشه برای این قسمت مثالی رو ذکر بفرمایید
اکیدا توصیه می‌شود از فشرده سازها قبل از اجرای برنامه (Pre-Compressed) به جای فشرده سازهای زمان اجرا (Run-time Compression)استفاده کنید.

با تشکر

نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۶ ۱۹:۵۱

فشرده سازی قبل از اجرای برنامه (Pre-Compression) یعنی که شما قبل از اینکه برنامه خود را در محیط اصلی نصب و اجرا کنید، فایل‌های اسکریپت آن را فشرده کنید. یعنی کاربران فایل اسکریپت فشرده شده را درخواست و دانلود می‌کنند و عملیات اضافی در سمت سرور انجام نمی‌شود. به عنوان مثال شما از فایل JQuery.min.js به جای jquery.js استفاده کنید. یعنی استفاده از نسخه فشرده شده اسکریپت ها.
فشرده سازی زمان اجرا (Run-time Compression) یعنی فشرده سازی اسکریپت‌های مورد نیاز کاربر توسط خود برنامه وب (به صورت خودکار و یا توسط یک ماژول اضافی). این عمل باعث می‌شود که در هر بار درخواست هر کاربر برای یک فایل، برنامه آن را مجدد فشرده سازی کند (و یا از cache استفاده کند). این عمل به معنی استفاده بیشتر از منابع پر ارزش سرور شما می‌باشد. به عنوان مثال شما بخواهید در هر مرحله درخواست هر کاربر jquery.js را فشرده کنید!

از مقایسه دو حالت بالا مشخص است وقتی شما فقط یکبار اسکریپت‌های خود را فشرده می‌کنید بسیار از حالتی که در هر مرتبه از درخواست کاربران آن را فشرده کنید بهتر است و کمتر منابع سرور را هدر می‌دهد.

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت ششم**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۵ ۱۱:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Performance

قسمت پنجم

17. پرهیز از استفاده نسخه debug

وقتی به ASP.NET مراجعه می‌کنید، توجه فرمایید که از چه نوع build برای محصول نهایی استفاده می‌کنید. وقتی از نسخه debug برنامه استفاده می‌کنید، بهبود دهنده‌های سطح کامپایلر عمل نکرده و کدشما در حالت بهینه اجرا نخواهد شد (کد شما همانگونه که هست اجرا می‌شود!).

برای مثال هنگامی که از نسخه release استفاده می‌کنید، کامپایلر c# به صورت خودکار از StringBuilderها به جای تلفیق عادی رشته ها، از آرایه‌ها به جای لیست ها، از دستور switch/case به جای دستورات if/then/else، تلفیق شروط با یکدیگر و...

استفاده کرده و کد شما را در حالت بهینه‌تری اجرا می‌کند. عدم استفاده از این نسخه شما را از این مزایا محروم می‌سازد و نرم افزار شما به کندی اجرا خواهد شد. البته ناگفته نماند این موضوع فقط باید برای محصول نهایی استفاده شود و جهت دیباگ کردن برنامه همچنان باید از نسخه debug استفاده نمایید.

توجه نمایید می‌توانید با استفاده از متغیرهای کامپایلر در کد خود بخشی از کد را مختص build خاصی از برنامه کنید. مثلا اگر برنامه در حال debug کامپایل شد، MiniProfiler را فعال کن در غیر این صورت غیر فعال باشد.

```
#if DEBUG
    //فعال کردن MiniProfiler
#endif
```

18.تنظیم دقیق لاگ‌های سیستم در محیط اجرا

وقتی محصول نهایی را آماده می‌کنید، فراموش نکنید که سطح لاگ گیری را در سطح مطلوبی قرار دهید تا بتوانید در صورت نیاز برنامه را اشکال زدایی کنید. البته زیاده روی در این مورد نیز می‌تواند مشکل زا باشد.

اکثر برنامه نویسان هنگامی که محصول نهایی را برای مشتری آماده می‌کنند، لاگ را غیر فعال می‌کنند تا کاربر سرعت بیشتری را تجربه کند. این سیاست غلط شما را از امکانات بی نظیر لاگ کردن (مانند وقایع نگاری امنیتی، رفع سریع مشکلات و...) محروم می‌سازد. بنابر این حتما سیستم لاگ خود را در زمان تولید محصول اصلی (و نصب بر روی سرور اصلی) در حالت متعادلی تنظیم نمایید. کمی تست و تجربه شما را در این امر یاری می‌کند.

19.مشخص کردن اندازه عکس

مشخص کردن اندازه عکس در تک img به صورت css یا attribute باعث می‌شود که همان اولین بار که صفحه رندر می‌شود، اندازه مورد نیاز عکس به آن اختصاص یابد تا در صورت دانلود سریعا جایگزین آن گردد. عدم مشخص کردن سایز عکس (طول و عرض) باعث رندر شدن مجدد تمامی المان‌های صفحه بعد از دانلود هر عکس از سرور می‌شود و منابع با ارزش cpu کاربر شما را به سادگی از بین می‌برد.

```
<img src="smiley.gif" alt="Smiley face" height="42" width="42">
```

نظرات خوانندگان

نویسنده: مرتضی
تاریخ: ۱۳۹۲/۰۷/۰۷ ۱۶:۵۴

سلام. ممنون از مطلب خوبتون. بنده وقتی سایتم رو با ویژوال استادیو باز میکنم فقط مود دیباگ داره و مود release رو نداره. فقط برای برنامه‌های ویندوز فرم و wpf مود release داره. پس با این اوصاف بنده چطوری سایتم رو تویه مود release پاپلیش کنم؟ فقط کد زیر رو تویه وب کانفیگ قرار میدم

```
<compilation debug="false" targetFramework="4.0">
```

آیا قطعه کد بالا همون کار release رو انجام میده. با تشکر

نویسنده: م منفرد
تاریخ: ۱۳۹۲/۱۰/۱۴ ۰:۱۰

در بخش publish از منوی debug می‌توانید نوع خروجی را مشخص کنید که برنامه با debug پابلیش شود یا realease

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت هفتم**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۱۶ ۲۰:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Performance

قسمت ششم

20.اسکریپت در پایین صفحه

لینک‌های مربوطه به javascriptهای خود را تا جای ممکن در پایین صفحه قرار دهید. وقتی parser مرورگر به فایل‌های javascript می‌رسد، تمامی فعالیت‌ها را متوقف کرده و سعی در دانلود و سپس اجرای آن دارد. برخلاف اینکه مرورگرها امکان دانلود چند فایل را به صورت همزمان از سرور دارند، هنگامی که به اسکریپت‌ها می‌رسند، تنها یک فایل را دانلود می‌کنند. یعنی اجرای برنامه و دانلودهای مرتبط با صفحه شما متوقف شده و پس از دانلود و اجرای اسکریپت اجرای آنها ادامه پیدا می‌کند. این مسئله وقتی نمود بیشتری پیدا می‌کند که شما فایل هغای اسکریپت با حجم و تعداد بالا در برنامه خود استفاده می‌کنید.
برای فرار از این مشکل می‌توانید تگهای مربوط به اسکریپت را در آخر صفحات خود بگذارید. فقط دقت کنید اگر نیاز است که قبل از نمایش صفحه تغییری ذر DOM ایجاد کنید، باید حتما اسکریپت‌های مربوطه را بالای صفحه قرار دهید.
روش دیگر دانلود فایل‌های اسکریپت به وسیله AJAX است که انشاء الله در آینده مقاله ای در این رابطه خواهم نوشت.

CDN.21

CDN یا Content Delivery Network سرورهای توزیع شده ای در سطح دنیا هستند که یک نسخه از برنامه شما برای اجرا بر روی آن قرار دارد. هنگامی که کاربر می‌خواهد به سایت شما دسترسی پیدا کند به صورت خودکار به نزدیکترین سرور منتقل می‌شود تا بتواند سرعت بیشتری را تجربه کند. علاوه بر این CDN باعث بالانس شدن بار ترافیک شبکه شما شده خط حملات D.O.S و D.D.O.S را به حداقل می‌رساند.

زمانیکه شما یک سیستم CDN را فعال میکنید تاثیر آن بصورت زیر خواهد بود:
۱- شبکه توزیع محتوا یا همان CDN تمامی سرورهای شبکه جهانی اینترنت را پوشش میدهد. بنابراین زمانیکه شما این سیستم را برای سایت خود فعال میکنید اطلاعات شما بر روی تمامی این سرورها کپی و ذخیره میشود و زمانیکه یک بازدیدکننده به سایت یا وبلاگ شما وارد میشود محتوای سایت شامل تصاویر و متون را از نزدیک‌ترین سرور نزدیک به خود دریافت میکند و مستقیما به هاست یا سرور شما متصل نمیشود. این کار موجب بهبودی چشمگیر در عملکرد سایت شما خواهد شد.
۲- CDN تمام اطلاعات ثابت شما مانند تصاویر، کدهای CSS و javascript، mp3، pdf و فایلهای ویدئویی شما را پشتیبانی میکند و تنها اطلاعاتی که قابل تغییر و بروزرسانی هستند مانند متون و کدهای HTML از سرور اصلی شما فراخوان میشوند. با این کار مصرف پهنای باند هاست شما کاهش یافته و هزینه ای که سالانه برای آن میپردازید کاهش چشمگیری خواهد داشت.
۳- تفاوت سرعت و عملکرد برای خودتان یا افرادی که در نزدیکی سرور اصلی شما هستند تفاوت زیادی نخواهد داشت ولی برای کسانی که ار نقاط مختلف جهان به سایت شما وارد میشوند این افزایش سرعت ناشی از CDN کاملا محسوس خواهد بود، با توجه به اینکه سایتهای ایرانی معمولا سرور و هاست خود را از خارج و کشورهایی مانند آلمان و آمریکا تهیه میکنند و عموم بازدیدکنندگان

از داخل کشور هستند استفاده از CDN میتواند بسیار موثر باشد. برای تعیین تاثیر CDN بر سرعت سایت میتوانید عملکرد خود را با ابزارهایی مانند Pingdom و GTmetrix بعد و قبل از فعال سازی CDN بررسی و مقایسه کنید.

ابزارها، تکنیک‌ها و روش‌های متفاوتی برای راه اندازی CDN وجود دارد که نیازمند مقاله ای مجزا می‌باشد.

عنوان: **چگونه نرم افزارهای تحت وب سریعتری داشته باشیم؟ قسمت هشتم**
نویسنده: م منفرد
تاریخ: ۱۳۹۲/۰۵/۲۸ ۲۱:۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, Performance

قسمت هفتم

22. استفاده از CSS Sprites

ایده اصلی این تکنیک به این صورت است که تمامی عکس‌های کوچک (دراینجا همه 100 عکس) در قالب یک تصویر بزرگ قرار خواهد گرفت و با استفاده از CSS مختصات هر عکس کوچک را در تصویر بزرگ پیدا کرده و نمایش می‌دهیم. یکی شدن 100 عکس کوچک به یک عکس بزرگ، تاثیر زیادی در پایین آمدن حجم عکس جدید خواهد داشت و مرورگر شما به جای درخواست 100 عکس از سرور، تنها یکی دانلود می‌کند و از این به بعد از کش مرورگر برای بازیابی آن استفاده می‌کند. این موضوع به معنی ترافیک کمتر شبکه و آزاد شدن منابع پر ارزش حافظه، cpu و پهنای باند در سمت سرور و کاربران. برای اطلاع بیشتر از این تکنیک می‌توانید به این مقاله مراجعه نمایید.

23. استفاده مطلوب از AJAX

شما می‌توانید برای لود کردن بخش‌های مخفی در صفحه خود از AJAX کمک بگیرید. به جای دانلود کردن تمامی بخش‌های صفحه در مرورگر کاربر، بخش هایی که در دید کاربر قرار ندارد را به صورت AJAX بارگیری کنید. نمونه ای از این تکنیک را در این صفحه مشاهده نموده و البته از کد آن استفاده نمایید.

24. حذف HTTP modules های اضافی

HTTP modules هایی را که در برنامه خود استفاده نمی‌کنید را حذف کنید. این کار یعنی سربار مدیریتی کمتر در ماژول ASP.NET سرور شما. برای اجرای این مورد می‌توانید از کدی مشابه این کد در web.config خود استفاده کنید:

```
<httpModules>
    <remove name="OutputCache"/>
    <remove name="Session"/>
    <remove name="WindowsAuthentication"/>
    <remove name="FormsAuthentication"/>
    <remove name="PassportAuthentication"/>
    <remove name="RoleManager"/>
    <remove name="UrlAuthorization"/>
    <remove name="FileAuthorization"/>
    <remove name="AnonymousIdentification"/>
    <remove name="Profile"/>
    <remove name="ErrorHandlerModule"/>
    <remove name="ServiceModel"/>
</httpModules>
```

البته حواستان به این موضوع باشد ماژول‌های مورد استفاده در برنامه خود را حذف نکنید که در این صورت ممکن است این آخرین پروژه شما با صاحب کارتان باشد!

عنوان: UrlRewriter توسط Intelligencia.UrlRewriter
نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۰۵/۳۰ ۱۹:۰
آدرس: www.dotnettips.info
گروه‌ها: SEO, FriendlyUrls, UrlRewrite, routing, ASP.Net

Intelligencia.UrlRewriter یک DLL رایگان برای انجام بازنویسی URLها در ASP.Net می‌باشد. به این صورت که بر اساس تنظیماتی که در فایل Web.Config انجام می‌شود، عمل تغییر URL را انجام میدهد و با زبان فارسی نیز سازگار است؛ مثل استفاده از عنوان فارسی در URL برای SEO بهتر:

برای استفاده، ابتدا DLL مورد نظر را دانلود و به Referencesها اضافه می‌کنیم؛ دانلود فایل Intelligencia.UrlRewriter.dll

دانلود آخرین نسخه‌ی فایل از سایت مرجع: urlrewriter.net

مرحله‌ی بعد تنظیمات Web.Config است که به صورت زیر می‌باشد:

```
<configuration>
  <configSections>
    <section name="rewriter" requirePermission="false"
type="Intelligencia.UrlRewriter.Configuration.RewriterConfigurationSectionHandler,
Intelligencia.UrlRewriter" />
  </configSections>
  <system.web>
    <httpModules>
      <add type="Intelligencia.UrlRewriter.RewriterHttpModule,Intelligencia.UrlRewriter"
name="UrlRewriter" />
    </httpModules>
  </system.web>
  <rewriter>
    <rewrite url="^.*-T([0-9]+)/?$*" to="~/ShowTerol.aspx?ID=$1" processing="stop" />
  </rewriter>
</configuration>
```

مرحله بعد تنظیم این است که کدام لینک‌ها بازنویسی شوند که در کدهای بالا فقط یک مدل بازنویسی وجود دارد:

```
<rewrite url="^.*-T([0-9]+)/?$*" to="~/ShowTerol.aspx?ID=$1" processing="stop" />
```

و می‌گوید تمام لینک‌هایی را که در آخر آنها T- و عدد وجود دارد، به صفحه‌ی ShowTerol.aspx با یک کوئری استرینگ به نام ID که مقدار آن، همان عدد بعد حرف T میباشد بفرست.

یعنی این لینک را

به این لینک تبدیل میکند

```
ShowTerol.aspx?ID=87
```

مثال‌های بیشتر

```
<rewrite url="^.*-p([0-9]+)/?$*" to="~/Products.aspx?p=$1" processing="stop"/>
<rewrite url="^.*-c([0-9]+)pa([0-9]+)/?$*" to="~/Default.aspx?c=$1&amp;pa=$2" processing="stop"/>
<rewrite url="^.*-c([0-9]+)/?$*" to="~/Default.aspx?c=$1" processing="stop"/>
```

الگوی تبدیلات توسط Regular Expansion

نظرات خوانندگان

نویسنده: رضا
تاریخ: ۱۳۹۲/۰۶/۲۹ ۲۳:۷

سلام , با تشکر از پست خوبتون .
بعد از اجرا سایت ارور 500 میده چیکار کنم ؟
هم لوکال و هم روی هاست .
خطا مربوط به این خط برنامه در Web.Config است :

```
<httpModules>
      <add type="Intelligencia.UrlRewriter.RewriterHttpModule,Intelligencia.UrlRewriter"
name="UrlRewriter" />
    </httpModules>
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۹ ۲۳:۲۵

خطای 500 یک خطای عمومی هست. اصل خطا چی بوده؟ چه stack trace ایی در صفحه لاگ میشه؟

نویسنده: رضا
تاریخ: ۱۳۹۲/۰۶/۲۹ ۲۳:۴۶

با تشکر از شما،
مشکل را پیدا کردم. فکر کنم برای IISهای از 6 به بالا باید به سبک زیر تنظیم بشه .

```
<configuration>
<configSections>
    <section name="rewriter" requirePermission="false"
type="Intelligencia.UrlRewriter.Configuration.RewriterConfigurationSectionHandler,
Intelligencia.UrlRewriter" />
  </configSections>

 <system.webServer>
    <modules>
      <add type="Intelligencia.UrlRewriter.RewriterHttpModule,Intelligencia.UrlRewriter"
name="UrlRewriter" />
    </modules>
  </system.webServer>

  <rewriter>
    <rewrite url="^.*-T([0-9]+)/?$*" to="~/ShowTerol.aspx?ID=$1" processing="stop" />
  </rewriter>
</configuration>
```

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۰۶/۳۰ ۱۶:۴۱

خوشحالم مشکلتون رفع شد ولی من بارها استفاده کردم و مشکلی نداشتم

نویسنده: heaven
تاریخ: ۱۳۹۲/۱۱/۰۳ ۲۲:۴۱

سلام
من این کارهایی که گفتید رو انجام دادم ولی تغییری حاصل نشد. به علاوه چند تا مسیج میده:

```
Could not find schema information for the element 'rewrite'
Could not find schema information for the attribute 'url'
Could not find schema information for the attribute 'to'
 Could not find schema information for the attribute 'processing'
```

چی کار کنم؟

نویسنده: heaven
تاریخ: ۱۳۹۲/۱۱/۰۳ ۲۳:۵۲

برای تبدیل Details.aspx?code=1 به Details/1 این کد درسته؟

```
<rewrite url="~/Details/(.+)" to="~/Details.aspx?code=$1" processing="stop" />
```

راستی میشه بپرسم چطور می‌تونم توی آدرس اسم محصولی رو که روش کلیک شده نشون بدم؟ یعنی به جای اون code=1 اسم محصولم رو نشون بده

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۰۴ ۱۱:۶

سلام
برای IISهای 6 به بالا همانطور که دوستمون اشاره کرد باید به جای

```
<system.web>
    <httpModules>
      <add type="Intelligencia.UrlRewriter.RewriterHttpModule,Intelligencia.UrlRewriter"
name="UrlRewriter" />
    </httpModules>
  </system.web>
```

از این استفاده کنیم

```
<system.webServer>
    <modules>
      <add type="Intelligencia.UrlRewriter.RewriterHttpModule,Intelligencia.UrlRewriter"
name="UrlRewriter" />
    </modules>
  </system.webServer>
```

دیگه نباید مشکلی وجود داشته باشه

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۰۴ ۱۱:۱۵

یه نمونه برنامه براتون ضمیمه کردم که همین کاری که گفتیدو انجام دادم

Test.zip

موفق باشید

نویسنده: heaven
تاریخ: ۱۳۹۲/۱۱/۰۴ ۲۰:۳۸

واقعا ممنونم. من دقیقا دنبال این یه خط می‌گشتم:

```
string.Format("Details/{0}-{1}", Eval("Name").ToString().Replace(" ", "-"),Eval("id")
```

که البته من اون P رو حذف کردم. اگه بخوام id رو هم نشون نده فقط همون اسم محصول رو نشون بده باید چی کار کنم؟ یا مثلا بخوام فیلدهای دیگه محصول رو هم نشون بده.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۰۴ ۲۳:۴۴

خواهش میکنم
شما باید بر اساس کلید در صفحه‌ی مقصد محصولی که فرستادیدو پیدا کنید منظورم کار این خطه

```
Label1.Text =
                GetProducts.Get()
                    .Where(x => x.Id == int.Parse(Request.QueryString["PID"].ToString()))
                    .Select(x => x.Name)
                    .FirstOrDefault();
```

اگه کد محصولو به صفحه‌ی که میخواهید جزییاتو تو اون نشون بدید نفرستید چجوری میخواید محصول مورد نظر رو شناسایی کنید تا بتونید جزییات اونو نشون بدید ؟
البته راه‌های دیگه هم هست که مثلا نام محصولتون به عنوان کلید باشه و بر اساس اون محصولو تو صفحه‌ی مقصد شناسایی کنید که به نظر من اصلا کار منطقی نیست.
یا کد محصولو توسط راه دیگری بجز URL بفرستید مثل session که غیر منطقی تره .
در کل میتونید فقط نام محصولو بفرستید که اینطوری میشه

```
<a href='<%# "Details/" + Eval("Name")%>'><%# Eval("Name") %></a>
```

ولی چجوری محصولو تو صفحه‌ی مقصد میخواید شناسایی کنید تا جزئیاتشو نشون بدید؟
بهترین راه اینه که کد محصول هم بفرستید.

نویسنده: heaven
تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۲:۱۰

بله در این که باید کد محصول رو بفرستیم که شکی نیست ولی منظورم این بود راهی نیست که توی آدرس این کد نشون داده نشه؟ در واقع کد توی آدرس اصلی هست اما توی آدرس rewrite شده نشون داده نمیشه. مثلا این نمونه رو ببینید. یه چیزی مثل این مد نظرم بود که البته زیاد مهم نیست.

```
http://www.site.com/Product/DKP-14997/Samsung-Galaxy-S4-I9500-16GB/name
```

یه سوال دیگه هم داشتم. اگه بخوام پسوند .aspx رو از آخر آدرس همه صفحات سایت حذف کنم باید چی کار کنم؟ ممنون.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۲:۱۸

- اون DKP-14997 کد محصول است که یا منحصر بفرد تعریف شده یا کلید اصلی.

- برای حذف از تمام سایت روش friendly urls هم هست علاوه بر روش مطلب جاری.

نویسنده: رضا منصوری

تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۵:۱۶

همونطور که دوستمون اشاره کرد DKP-14997 تو سایت دیجی کالا کلید محصولاته.
روشی که در این مطلب برای انجام UrlReWrite خوندین **به نظرم** روش کامل و اصولی برای انجام باز نویسی یو آر ال‌ها نیست .
اگه بخواید کاملا بر روی یو آر ال‌های سایت مدیریت داشته باشید باید از امکان مسیر یابی خود ASP.Net استفاده کنید.
در وب فرم این مقاله کمکتون میکنه. در ام وی سی هم این مطلب . سایتهایی که اصولی هستند از قابلیت مسیر یابی خود ASP.Net استفاده میکنند. مثل همون دی جی کالا یا سایت جاری.

نویسنده: afsoon
تاریخ: ۱۳۹۲/۱۱/۰۷ ۱۱:۴۰

من می‌خوام علاوه بر اینکه وقتی به یه صفحه جدید میرم آدرس عوض بشه، توی همون صفحه هم با ایجاد تغییراتی آدرس صفحه ام عوض بشه. مثلا وقتی کاربر سرچ می‌کنه صفحه تغییر نمی‌کنه و نتایج سرچ توی همون صفحه نشون داده میشه اما می‌خوام آدرس عوض بشه. به این صورت که عبارت سرچ شده توی آدرس بیاد. یا مثلا وقتی از داخل دراپ دان لیست یک ویژگی خاص محصول انتخاب میشه (که اینم به نوعی سرچ هست) اون ویژگی توی آدرس همون صفحه نشون داده بشه.
برای این کار از کدوم روش استفاده کنم بهتر و راحت تره؟ url rewriting, url routing, url friendly ؟
اگه یک راهنمایی هم در مورد اینکه چطور باید این کار رو انجام بدم بکنید ممنون میشم. یعنی وقتی صفحه عوض نمیشه من این تغییر رو دقیقا کجا باید اعمال کنم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۷ ۱۴:۳۱

- اطلاعات ارسالی توسط کاربر فقط زمانی در URL صفحه ظاهر می‌شوند که متد ارسال اطلاعات فرم ، بجای Post حالت Get باشد.
- یا اینکه در حین Redirect خودتان دستی این کوئری استرینگ را تنظیم کنید.

```
Response.Redirect("tst.aspx?id=" + txtSrch.Text)
```

الان که این کوئری استرینگ رو دارید، مابقی آن مباحثی هست که در این مطلب و پرسش و پاسخ‌های آن مطرح شده.

نویسنده: afsoon
تاریخ: ۱۳۹۲/۱۱/۰۸ ۱۷:۲۷

اگه بخوام آدرسم به این صورت دربیاد tst/Keyword=txtSrch.Text باید به جای عبارت داخل " " چی بنویسم؟ الگوی تبدیلش رو هم میشه بگید؟ این الگوی تبدیل‌ها و نحوه نوشتن آدرس جدید خیلی سخته چطور باید اینا رو فهمید؟

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۰۸ ۲۰:۱۶

برای ساختن لینکش همونطور کو دوستمون توضیح داد .

```
Response.Redirect("tst/" + txtSrch.Text)
```

برای باز نویسی آدرس اون هم

```
<rewrite url="~/tst/(.+)" to="~/mypage.aspx?searchtxt=$1" />
```

که در صفحه‌ی مقصد با کوئری استرینگ searchtxt میتونید اطلاعاتو بدست بیارید.
همونطور که تو مطلب گفتم الگوی تبدیلات توسط Regular Expansion انجام میشه .
اینم یه مطلب که کمکتون میکنه.

نویسنده: مائده
تاریخ: ۱۳۹۲/۱۱/۰۹ ۲۳:۲۶

اگه وب فرم‌ها توی فولدر جدا باشند، الگوی تبدیل چجوری میشه؟
اگه اسم فولدر رو قبل از اسم پیج بیاریم صفحه بهم میریزه! اگه نیاریم هم جواب نمیده!

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۰:۳۶

من از این روش برای تبدیلات ساده و کوچک استفاده کردم ، همونطور که گفتم این روش **به نظرم** روش کامل و اصولی برای بازنویسی Urlها نیست و روش درستش استفاده از امکان Url Routing خود Asp.Net هستش.
این مطلب کمکتون میکنه .

نویسنده: مائده
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۱:۴

من دات نت فریم ورکم 3.5 هست برای همین نمی‌تونم از url routing و url friendly استفاده کنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۱:۱۱

url routing از دات نت 3 و نیم، سرویس پک یک به بعد به صورت توکار و استاندارد، اضافه شده. فقط برای حالت دات نت سه و نیم باید یک سری تنظیمات اضافه‌تر به وب کانفیگ اضافه شوند (تعریف System.Web.Routing.UrlRoutingModule باید در قسمت httpModules ذکر شود).

نویسنده: زیلایی
تاریخ: ۱۳۹۳/۰۵/۲۰ ۱۰:۳

سلام.من کل مراحل رو انجام دادم ولی یک مشکل دارم که فکر میکنم به عبارت ریگولار رو اشتباه نوشتم.
میخوام به طور مثال این آدرس:
www.sample.com/page2/page2.aspx?MID=1251
به این تبدیل بشه :
www.sample.com/1251
سوال دیگه ای که داشتم اینه که آیا توی صفحاتی که میخوام کوری استرینگ رو میخوام بگیرم باید کاری کنم؟
و اینکه آیا با این روش فایل‌های javascript, CSS مشکل پیدا نمیکنن؟

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۵/۲۰ ۱۶:۴۳

سلام . همونطور که گفتم به نظرم استفاده از امکان مسیر یابی خود دات نت بهترین گزینه است و نمیخواد خودتونو با ریگولار و ... درگیر کنید .
آموزششو از اینجا میتونید بخونید.

یه نمونه برنامه ساختم و کاری که میخواید رو از روش روتینگ خود دات نت انجام دادم . یعنی این آدرس رو www.sample.com/1251 به صفحه‌ی News.aspx میفرسته و اونجا میتونید کد 1251 رو بخونید و کارتون را ادامه بدید. Test-52676fa0948d4909be628f7a6a24bc5d.zip
در مورد روش درست لینک کردن فایل‌های جاوا اسکرسپت و css میتونید اینجا متوجه بشید.

عنوان: **PHP سریعتر از ASP.NET! افسانه یا واقعیت؟**
نویسنده: مهرداد اشکانی
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۱:۵۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, PHP, Performance

چرا افسانه‌ای که می‌گوید PHP از ASP.NET سریعتر است اینقدر شایع است؟ در این مقاله به بیان حقایقی می‌پردازیم که این افسانه را زیر سوال می‌برد؟

خیلی وقتها در بسیاری از نوشته‌ها و اظهارنظرها می‌بینیم ادعا می‌شود که PHP بسیار سریعتر از ASP.net است و اینکه ASP.net از لحاظ سرعت کند است. آزار دهنده‌ترین بخش این ادعاها، آن است که هر یک از آنها را که نگاه می‌کنی بصورت کاملا غیر واقع بینانه به موضوع نگاه می‌کنند و فقط بدون دلیل این موضوع را ادعا می‌کنند. زیرا به این موضوع بصورتی کاملا متعصبانه و بدور از واقعیتها نگاه می‌شود. به همین دلیل بصورت گسترده ای این افسانه در میان اهالی وب پذیرفته شده است.

حال بجای اینکه این موضوع را بارها و بارها در جاهای مختلف بیان کنیم، این مقاله را نوشته و در هر کجا که لازم باشد به آن ارجاع خواهیم داد. باید توجه کنید این حقیقت که زبان PHP یک زبان اصیل و قدرتمند است هیچ شکی در آن نیست اما اینکه بخواهیم بصورت مغرضانه و به این دلیل که ما از این زبان استفاده می‌کنیم، آنرا از هر لحاظ برتر از سایر زبانها بدانیم (کمی که نه) بسیار اغراق آمیز است.

این مقاله برای این نیست که ما هریک از این زبانها را زیر سوال ببریم. بلکه برای آن است که این موضوع را با دلایل منطقی و حقیقی بررسی کنیم که آیا اینکه می‌گویند PHP از ASP.net سریعتر است واقعیت دارد یا نه؟

Compiled در مقابل Interpreted Languages:
قبل از هرچیز ذکر این نکته الزامی است که این دو زبان تفاوتهای اساسی در base دارند. ASP.net یک زبان بهینه سازی و کامپایل شده است، به این معنی که کدهای نوشته شده در این زبان قبل از اینکه قابل اجرا شوند، به مجموعه ای از دستورالعمل‌های خاص ماشین تبدیل می‌شوند. از سوی دیگر PHP یک زبان تفسیر شده است، به این معنی که کدهای نوشته شده به همان شکل ذخیره شده و در زمان اجرا این کدها تفسیر می‌شوند. این موضوع بطور گسترده‌ای پذیرفته شده و ثابت شده است که برنامه‌های کامپایل شده به مراتب سریعتر از برنامه‌های تفسیر شده اجرا می‌شوند، به این دلیل که برنامه‌های تفسیر شده نیاز دارند تا در زمان اجرا به دستورالعملهای ماشین تبدیل شوند.

در اینجا به یک نقل قول از دانشنامه آزاد ویکی پدیا اشاره می‌کنم که میزان سریعتر بودن برنامه‌های کامپایل شده را نشان می‌دهد:

"A program translated by a compiler tends to be much faster than an interpreter executing the same program: even a 10:1 ratio is not uncommon. The mixed solution's efficiency is typically somewhere in between."

به این معنا که یک برنامه بصورت کامپایل شده بسیار سریعتر از همان برنامه بصورت تفسیر شده، اجرا می‌شود.

اعداد و ارقام:
حال که تئوری خود را مبنی بر دلیل سریعتر بودن ASP.net بیان کردیم بیایید با هم نگاهی به برخی آمارها بیاندازیم تا این تئوری را در عمل هم نشان داده باشیم.

آمارهای زیر توسط شرکت WrenSoft جهت مقایسه زمان اجرای یک کد مشابه در زبانهای مختلف تهیه شده است. اگر می‌خواهید توصیف عمیق‌تری از آمارها داشته باشید لطفا لینک را دنبال کنید.

نمودار اول: زمان صرف شده برای تولید ونمایش نتایج برای جستجوی وب سایت‌های کوچک

همانطور که میبینید زمان متوسط برای سایت 0.1500، PHP ثانیه و برای سایت 0.0150، ASP.net ثانیه است. یک تفاوت بزرگ: PHP ده برابر نسبت به ASP.net کندتر است!

نمودار دوم: زمان صرف شده برای تولید و نمایش نتایج برای جستجوی وب سایت‌های متوسط

PHP، 1.0097 ثانیه طول می‌کشد در حالی که ASP.net، 0.0810 ثانیه زمان نیاز دارد. می‌بینیم که PHP دوازده بار بیشتر از ASP.net زمان می‌برد.

درحال حاضر این آزمون با یک کد مشابه در زبانهای برنامه نویسی مختلف پیاده سازی و اجرا شد و نتیجه را مشاهده نمودید. حال این موضوع پیش میاید که این اجرای کدها بر روی سیستم عامل ویندوزی بوده است و این می‌تواند به نفع ASP.net باشد، پس همین آزمون را بر روی سیستم عامل لینوکسی مشاهده می‌کنیم.

آمارهای زیر از سایت معتبر shootout.alioth.debian.org گرفته شده است. این آمارها نحوه اجرای همان کد را بر روی سیستم عامل لینوکسی برای هردو زبان نشان می‌دهد:

همانطور که مشاهده می‌کنید در سیستم لینوکسی نیز همچنان ASP.net سریعتر از PHP عمل می‌کند.

نتیجه گیری:

همین حالا جمله‌ی "asp.net vs php speed" را در google جستجو کنید. خواهید دید که در اکثر پست‌ها گفته شده که PHP از ASP.net سریعتر است اما دلیلی بر این ادعا نخواهید یافت و فقط در حد حرف است. مشکل این است که اکثر مردم وقتی چیزی را زیاد می‌بینند یا زیاد می‌شنوند بدون آنکه دلیل بخواهند آنرا می‌پذیرند و حتی بعضی اوقات از آن نیز دفاع می‌کنند که واقعا جای تاسف دارد.

توسعه وب بوسیله PHP کار خوبی است، بسیاری از اپلیکیشن‌ها و وبسایتهای شگفت انگیز توسط این زبان نوشته شده اند. اگر احساس می‌کنید PHP یک زبان برتر است از آن استفاده کنید اما این دلیل نمی‌شود که اطلاعات غلط را به دیگران القاء کنید و بدون دلیل و مدرک این زبان را از هر لحاظ برتر بدانید حال آنکه در این مقاله دیدیم که براساس چیزی که ارائه شد، **ASP.net سرعت بیشتری نسبت به PHP دارد** .

اگر با من در این امر موافق نیستید می‌توانید با نظرهای مستدل خود ما را راهنمایی کنید.

نظرات خوانندگان

نویسنده: سیروس
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۱

به دلیل وسعت استفاده بیشتر از php و نیز استفاده سایت‌ها و شرکت‌های بزرگ از php خیلی‌ها فکر می‌کنن php بهتر و سریعتر از asp.net هست در حالیکه این وسعت استفاده بخاطر اوپن سورس و رایگان بودنه php هست و چون وب سرور apache هم معمولا رو لینوکس نصب میشه و خود لینوکس هم اپن سورسه، تمام این دلایل دست به دست هم داده تا php بهتر به نظر بیاد. جدا از بحث سرعت اگر از لحاظ ساختاری بررسی کنیم php بیشتر یک زبان اسکریپتی است تا برنامه نویسی و ویژگی‌های زبان‌های خوب و شی گرا رو نداره.

نویسنده: مسلم
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۴

نمی دونم چرا ولی توی عمل واسه استارت زدن دات نت خیلی دیر میجنبه . حتی بعد از کامپایل و پابلیش یه خورده تاخیر داره ولی یکم که باهاش کار کنی میفته رو دور و خوب میشه !
ولی پی اچ پی همون اول سریعه . شاید بخاط پیچیدگی فریم ورک هست . چرا که linq , entity هم در پرژه استفاده شده است .

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۲۰

چندین علت داره:

- پروسه کامپایل کدهای دات نت یک مرحله‌ای نیست. کلا در دات نت کدها به یک زبان میانی به نام IL ترجمه میشن. بعد این IL توسط JIT compiler تبدیل به کدهای ماشین میشه. در ASP.NET این مساله هم برای کدهای پشت صحنه برنامه و هم برای خود صفحات رخ می‌ده. بنابراین برای بار اول مشاهده، روند این پروسه الزامی هست. ولی برای دفعات بعدی مشاهده خیر. البته روش برای پیش کامپایل کردن کامل صفحات هم وجود داره و یا در IISهای جدید یک سری مبحث warmup توکار پیش بینی شده.

- همچنین IIS برای مدیریت منابع سرور، یک سایت رو مدام در حافظه نگه نمی‌داره. فقط زمانیکه اولین درخواست به سرور میرسه سایت رو بارگذاری می‌کنه و application pool اون رو استارت. این هم یک زمان اولیه اندکی رو ممکنه به خودش اختصاص بده. بعلاوه پس از مدتی، یک سایت بیکار رو از حافظه خارج می‌کنه. بهش می‌گن ریسایکل کردن. در ASP.NET 4.0 به بعد امکان تنظیم auto-start برای سایت‌ها هست.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۲:۳۰

به نظر من برای بحث در مورد PHP مقایسه سرعت در رده آخر اهمیت هست. مسایل بهتری برای بحث وجود دارند. مثلا:

- بدترین زبانی که تابحال با آن کار کردید، کدوم بوده؟

- The PHP Singularity

- PHP: a fractal of bad design

- PHP Sadness

- PHP is meant to die

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۵۰

هر کسی به من گفت php سریع‌تر هست و یا ASP.NET سریع‌تر هست؛ من هم در جواب گفتم شما درست می‌فرمایید و هیچ گاه با آن‌ها بحث نکردم. هنوز که هنوزه من نمی‌فهمم که واقعا دارید چی را با چی مقایسه کنید. ASP.NET و php کاملا دو مقوله‌ی متفاوت هستند. اگر قرار است مقایسه ای در سرعت عمومی انجام شود، معقول‌تر است که در سطح فریم ورک هایی با قدرت برابر انجام شود؛ برای مثال سرعت عمومی Zend بالاتر است یا ASP.NET MVC.
اصلا بهتر است یه مقایسه با مستندات کافی برای شما مطرح کنم تا به کندی ASP.NET MVC پی ببرید:
هدف از این برنامه نمایش عبارت Hello,World در مرورگر است.
در php خام نوشتن کد زیر کفایت می‌کند:

```
echo 'Hello, World';
```

اما در ASP.NET MVC شما باید ابتدا یک کنترلر تعریف کرده سپس در یک Actionعبارت Hello,World را بازگشت دهید. اگر این دو برنامه را اجرا کنید از سرعت فوق العاده‌ی php متحیر خواهید شد.
البته بگذریم از اینکه در ASP.NET سربارهای به نام Routing و ... در ابتدای کار وجود داره. نتیجه این هست که ASP.NET خیلی کنده و حرفی برای گفتن نداره در مقابل php.
از این دست مقایسه‌ها من هم زیاد دیدم. واقعا خودشان هم نمی‌فهمند چی را با چی مقایسه می‌کنند.
این نوع مقایسه‌ها بیشتر منو یاد کسی می‌اندازه که گوشی موبایل خریده بود چهار هسته ای و می‌گفت چقدر تکنولوژی پیشرفت کرده که از لپتاپم دو هسته بیشتر داره و سریعتره!
من هم با خواندن این مقاله به جمله‌ی شما درست می‌فرمایید بسنده می‌کنم.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۲:۵۷

مثالت بی‌ربطه دوست عزیز. echo خام در PHP معادل هست با Response.Write خام در ASP.NET در حالتیکه در یک فایل ashx اجرا می‌شود. احتمالا می‌دونی که این نوع فایل‌ها در حالت پیش فرض حتی ماژول سشن هم براشون فعال نیست چه برسد به مسیریابی و در حداقل سربار کار می‌کنند.

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۳

خب منظور من دقیقا همین بود. یک مقایسه کاملا بی ربط. یک فریم ورک با کلی امکانات را با یک زبان خام مقایسه کردم! پی نوشت: *من در آینده نظر قبلی را دادم. ساعت 13:50!*

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۹

خوب، مثال‌های بحث جاری در ساده‌ترین حالت ممکن تهیه شدند. یعنی یک فریم ورک ASP.NET با کلی دم و دستگاه (ماژول سشن، ماژول مسیریابی، ماژول امنیت، ماژول اعتبارسنجی درخواست‌ها، ماژول فشرده سازی و ...) از یک سیستم ساده PHP سریعتر عمل می‌کنه. این چطور بی‌ربط به عنوان مقاله هست؟

نویسنده: مهدی سعیدی فر
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۳:۲۲

آخه من از منبع این مقاله چیزی نفهمیدم.
کلا منبع داره در مورد یه چیز دیگه صحبت می‌کنه. http://www.wrensoft.com/zoom/benchmarks.html

نویسنده: احسان
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۴:۱۵

فکر کنم خیلی مشخصه که یک زبان کامپیل شده چقدر میتونه از یک واسط سریعتر باشه کاش با جاوا مقایسه میکردی

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۰۶/۲۶ ۱۸:۲۱

توسعه وب بوسیله PHP کار خوبی است، بسیاری از اپلیکیشن‌ها و وبسایتهای شگفت انگیز توسط این زبان نوشته شده اند. اگر احساس می‌کنید PHP یک زبان برتر است از آن استفاده کنید اما این دلیل نمی‌شود که اطلاعات غلط را به دیگران القاء کنید و بدون دلیل و مدرک این زبان را از هر لحاظ برتر بدانید حال آنکه در این مقاله دیدیم که براساس چیزی که ارائه شد، **ASP.net سرعت بیشتری نسبت به PHP دارد** .

نویسنده: ناصر فرجی
تاریخ: ۱۳۹۲/۰۶/۲۶ ۲۲:۵۰

مطلبی مشابه که چند روز پیش خوندم . لحظاتی یاد بحث‌های فروم برنامه نویس افتادم :)
http://www.rezashirazi.com/post261.aspx

نویسنده: آرایه
تاریخ: ۱۳۹۲/۰۶/۲۷ ۲۱:۳۷

به نظرم این بحث سود چندانی برای خوانندگان ندارد، benchmarkها بر مبنای کد مشابه هستند اما عملاً کد مشابه روی دو پلتفرم مختلف رو نمی‌شه مقایسه کرد.
مطلبی در این خصوص نوشتم: اینجا

نویسنده: آرمان فرقانی
تاریخ: ۱۳۹۲/۰۶/۲۸ ۸:۹

فناوری-زبان PHP بسیار محترم است و بسیار قابل توصیه اما برای آن زمانی که در رقابت با ASP کلاسیک بود و پدیده ای به نام دات نت ظهور نکرده بود حال آنکه پدیده یاد شده در زمان حال به پختگی و پیشرفت چشمگیری دست یافته است.
دنیای دات نت و مباحث مربوط به آن گسترده‌تر و پیچیده‌تر از آن است که برای توجیه استفاده از دات نت سرعت مقایسه شود. ده‌ها ویژگی منحصر به دات نت وجود دارد که برای انتخاب فناوری سمت سرور مجالی برای تفکر درباره سرعت باقی نمی‌گذارد و آنان که اهل تفکر باشند را جذب خود می‌کند و نه گمراهان (کسانی که یا تعصب دارند یا خسته‌تر از آن هستند که دنیای جدیدی را تجربه نمایند.)
در مورد سرعت کافی است نکات ساده ای که چند دقیقه بیشتر زمان نمی‌برند رعایت شود تا وب سایت‌های دات نتی چندین برابر سریعتر عمل کنند.
فراموش نکنید دنیای دات نت تا حد بیشتری با اصول مهندسی نرم افزار گره خورده است و باب میل همگان نیست.
عموماً سرویس دهنده بر حسب فناوری انتخاب می‌شود و نه برعکس! در مقالاتی که مقایسه انجام می‌دهند صحبت از رایگان بودن لینوکس و آپاچی و ... چندان ضرورتی ندارد.
بسیاری از آن‌ها که Open Source بودن PHP را با آب و تاب و به عنوان برتری مطرح می‌کردند هرگز در عمر خود به کدهای سورس آن نگاهی نکرده اند. البته اگر بداند از کجا باید دانلود کنند. توصیه می‌کنم در مورد رویکرد و سیاست‌های چند سال اخیر مایکروسافت در رابطه با Open Source تحقیق بیشتری صورت گیرد.
مثال هایی از سایت‌های موفق و یا تعداد سایت‌ها در یک فناوری هرگز دلیلی برای انتخاب فناوری نیست. ضمناً زمان ظهور این دو فناوری یکسان نبوده است که تعداد وب سایت‌ها معیار مقایسه باشد.
فناوری-زبان PHP هنوز وجود دارد. هنوز قدرت خود را دارد. و هنوز هر کجا به هر دلیلی امکان استفاده از دات نت نبود، با کمال

افتخار و خوشحالی از این که چند سال عمری که برای آن گذاشته ام هدر نرفته است به آن باز می‌گردم و آن را مورد استفاده قرار می‌دهم.

نویسنده: نیما
تاریخ: ۱۳۹۲/۰۶/۳۰ ۷:۱۸

با سلام خدمت دوستان گرامی

نکته ای که باید به آن دقت داشت، فرق بین مجانی و سورس باز است. لطفا این موارد را از یکدیگر جدا نمایید.

نکته دوم هم اینست که چند نفر در کشور خودمون داریم که میتوانند سورس PHP، لینوکس و ... را سفارشی کنند؟

نویسنده: حامد وهابی املشی
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۴:۸

چرا در نمودار جاوا اسکریپت توی نمودار قرار گرفته ؟ اونها زبانهایی سمت سرورند ، جاوا اسکریپت سمت کلاینت . البته ممکنه منظورش جاوا اسکریپت سمت سرور باشه ! در این صورت هم باز اشتباهه ، چون جاوا اسکریپ سمت سرور مربوط به تکنولوژی ASP کلاسیک بوده که توی نمودار اون رو هم آوردن . ASP کلاسیک هم وبی اسکریپت داشت و هم جاوا اسکریپت.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۴:۱۷

منظورش پیاده سازی خاص خودشون بوده.

نویسنده: حامد وهابی املشی
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۵:۱۹

بلاخره پیاده سازی خودش ، سمت سرور بوده ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۸:۸

بله. ضمنا node.js یک فناوری دیگر سمت سرور مبتنی بر جاوا اسکریپت است (و از این دست دارن زیاد میشن)

نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۰۹/۱۴ ۵:۱۰

البته این مورد هم هست که بسیاری از شرکت‌ها یا افراد که از php استفاده می‌کنن به خاطر عدم وابستگیشون به یک شرکت خاص چون مایکروسافت هم هست مثل گوگل که کلا استفاده از برنامه‌های شرکت مایکروسافت رو در شرکتش ممنوع اعلام کرده، کلا عدم وابستگی پی اچ پی و همچنین متن باز بودنش برای به روزآوری و تغییراتش توسط برنامه نویسان سراسر جهان بیشتر مورد توجه قرار گرفته.
البته این نکته هم هست که اکثر مردم با دیدن آدرس‌های بدون پسوند فکر می‌کنن که php هست مثلا همین stack overflow رو خیلی‌ها فکر می‌کنن با php نوشتن.
البته بودن بیشتر سیستم‌های آماده از نوع php و همچنین ارزونتر بودن هاست‌های لینوکس به خصوص در ایران هم سبب این اتفاق بوده.

عنوان: اجرای برنامه‌های ASP.NET توسط Mono در Ubuntu
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۰:۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, Mono, Linux

در ادامه مباحث بررسی اجرای برنامه‌های دات نت بر روی لینوکس، قصد داریم برنامه‌های ASP.NET را به کمک Mono 3.0 و یک وب سرور لینوکسی، بر روی Ubuntu اجرا کنیم .

پیشنیازها

دو پروژه خالی ASP.NET Web forms و ASP.NET MVC را در VS.NET تحت ویندوز ایجاد نمائید. آن‌ها را یکبار کامپایل کرده و اجرا کنید. سپس فایل‌های آن‌ها را به ubuntu منتقل کنید (پوشه‌های bin پروژه‌ها فراموش نشوند؛ خصوصا نگارش MVC که به همراه یک سری کتابخانه جانبی است).

برای انتقال فایل‌ها به لینوکس، اگر از VMWare workstation برای اجرا و آزمایش Ubuntu استفاده می‌کنید، کپی و paste مستقیم فایل‌ها از ویندوز به درون ماشین مجازی لینوکس پشتیبانی می‌شود.

نصب وب سرور آزمایشی مونو یا XSP

اگر نیاز به یک وب سرور آزمایشی، چیزی شبیه به وب سرور توکار VS.NET داشتید، پروژه XSP جهت این نوع آزمایشات ایجاد شده است.

پس از نصب آن (که به همراه همان بسته PPA قسمت قبل ، هم اکنون بر روی سیستم شما نصب است)، در ترمینال لینوکس، با استفاده از دستور cd به ریشه وب سایت خود وارد شوید، سپس دستور xsp4 را اجرا کنید تا وب سرور xsp4 مشغول هاست سایت شما شود (برای اجرا در مسیر /opt/bin/xsp4 نصب شده است).

اجرای برنامه ASP.NET Web forms 4 توسط XSP

بدون هیچ مشکل خاصی در همان ابتدای کار اجرا شد (البته باید دقت داشت که لینوکس به کوچکی و بزرگی حروف حساس است. یعنی حتما باید Default.aspx وارد شود و نه default.aspx):

اجرای برنامه ASP.NET MVC 4 توسط XSP

اجرا نشد! پیام می‌دهد که

```
"Missing method System.Web.Security.FormsAuthentication::get_IsEnabled() in assembly System.Web.dll
```

و یا

```
Compiler Error Message: CS1703: An assembly with the same identity `mscorlib, Version=4.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089'
has already been imported. Consider removing one of the references
```

علت اینجا است که XSP4 همراه با نسخه PPA، قدیمی است. بنابراین باید نسخه اصلی را از مخزن کد آن دریافت و کامپایل کنیم.

پیشنیازهای اینکار مانند Git و Mono، در قسمت قبل دریافت شدند. سپس فرامین ذیل را در خط فرمان لینوکس اجرا کنید:

```
git clone git://github.com/mono/xsp.git
cd xsp
./autogen.sh --prefix=/opt
make
sudo make install
```

پس از کامپایل، اگر این نگارش جدید را اجرا کنید، به خطای ذیل خواهید رسید:

```
System.IO.FileNotFoundException: Could not load file or assembly XSP, Version=3.0.0.0
```

برای رفع این مشکل باید اینبار وب سرور جدید را با فرمان sudo یا دسترسی مدیریتی اجرا کنید تا مشکل برطرف شود.
البته من سورس دریافت شده را در خود monodevelop کامپایل کردم (فایل sln آن‌را در monodevelop باز کرده و پروژه را build کنید). در این حالت دو فایل Mono.WebServer.dll و Mono.WebServer.XSP.exe در پوشه xsp/src/Mono.WebServer.XSP/bin/Debug ظاهر می‌شوند.
یکی دیگر از دلایل ظاهر شدن خطای فوق، نیاز به نصب این دو فایل در GAC است که به نحو زیر قابل انجام می‌باشد:

```
cd xsp/src/Mono.WebServer.XSP/bin/Debug
sudo gacutil -i Mono.WebServer.XSP.exe
sudo gacutil -i Mono.WebServer.dll
```

بعد این دو فایل dll و exe را در پوشه برنامه MVC خود کپی کنید و سپس دستور ذیل را اجرا نمائید:

```
cd myMvcAppPath
sudo mono Mono.WebServer.XSP.exe
```

اینبار وب سرور جدید، روی پورت 9000 شروع به کار می‌کند. اکنون اگر در فایرفاکس آدرس http://localhost:9000 را باز کنید، برنامه اجرا شده اما به خطای ذیل خواهید رسید:

```
CS0234: The type or namespace name `Helpers' does not exist in the namespace `System.Web'.
Are you missing an assembly reference?
```

برای رفع این مشکل باید اندکی فایل web.config برنامه را ویرایش کرد:

```
<system.web>
    <compilation debug="true" targetFramework="4.0">
      <assemblies>
        <add assembly="System.Web.Helpers, Version=2.0.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" />
        <add assembly="System.Web.Mvc, Version=4.0.0.0, Culture=neutral,
PublicKeyToken=31BF3856AD364E35" />
      </assemblies>
    </compilation>
```

سعی بعدی ... اجرا نشد! با هر بار refresh صفحه یک خطای جدید نمایش می‌داد که ... Type خاصی را نمی‌تواند بارگذاری کند (به همراه نام اسمبلی مربوطه). برای رفع این مشکل dllهای ذیل را از پوشه bin پروژه MVC خود که از ویندوز به لینوکس کپی کرده‌اید، حذف کنید:

Microsoft.Web.Infrastructure.dll

System.Net.Http.dll

System.Net.Http.Formatting.dll

System.Web.Http.dll

System.Web.Http.WebHost.dll

این فایل‌ها توسط تیم Mono به صورت مستقل پیاده سازی شده‌اند و نیازی نیست تا از ویندوز به لینوکس کپی شوند.
بعد از حذف این فایل‌های اضافی، برنامه ASP.NET MVC نیز اجرا شد:

چند نکته تکمیلی

- نحوه تشخیص موجود بودن یک DLL خاص، در نگارش جاری Mono نصب شده:

```
$ gacutil -l Microsoft.Web.Infrastructure
The following assemblies are installed into the GAC:
Microsoft.Web.Infrastructure, Version=1.0.0.0, Culture=neutral, PublicKeyToken=31bf3856ad364e35
Number of items = 1
```

- اگر می‌خواهید مطمئن شوید که تمام اسمبلی‌های موجود در GAC درست نصب شده‌اند یا خیر، فرمان ذیل را اجرا کنید:

```
cd /opt/lib/mono/gac # assuming this is your main gac
sudo find . */*/*.dll -exec gacutil -i '{}' \;
```

- در نسخه لینوکسی System.Web ممکن است یک سری از فضاهای نام هنوز موجود نباشند. لیست آن‌ها را در این آدرس می‌توانید مشاهده کنید:

http://go-mono.com/status/status.aspx?reference=4.0&profile=4.0&assembly=System.Web

نظرات خوانندگان

نویسنده: علی صداقت
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۷:۲۲

جناب نصیری، دیتابیس رو چه کنیم؟ واقعا جانشین منبع بازی، همسان با sql server در این طرف داریم؟ MySql کفایت می‌کنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۱ ۱۷:۵۶

- MySQL هم خوبه (البته مجوز استفاده‌اش رو هم باید مدنظر داشته باشید). هرچند از زمانیکه به تملک اوراکل درآمده، خیلی از شرکت‌ها دارند به سمت MariaDB کوچ می‌کنند: ^ و ^
+ Oracle نسخه لینوکسی هم دارد.
- و برای کارهای سبک SQLite نسخه لینوکسی دارد.

نویسنده: محمد جواد تواضعی
تاریخ: ۱۳۹۲/۰۷/۰۲ ۱:۱۲

شما می‌توانید از postgresql استفاده کنید . کاملا متن باز بوده و چیزی از oracle و Ms SqlServer کم ندارد .

عنوان: اجرای برنامه‌های ASP.NET به کمک وب سرور Apache توسط Mono در Ubuntu
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۲ ۸:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Mono, Linux

در قسمت قبل نحوه‌ی اجرای برنامه‌های ASP.NET را به کمک وب سرور آزمایشی Mono بررسی کردیم. بدیهی است این وب سرور برای هاست کردن سایت‌ها و خدمات دهی عمومی، مناسب نیست و صرفا جهت کارهای برنامه نویسی طراحی شده است. در ادامه، ابتدا وب سرور معروف آپاچی را نصب کرده و سپس سعی خواهیم کرد تا همان دو برنامه ASP.NET Web forms و ASP.NET MVC قسمت قبل را بر روی آن اجرا کنیم (دو برنامه خالی ASP.NET کامپایل شده که از ویندوز به لینوکس کپی شده بودند).

نصب وب سرور آپاچی

به اینترنت متصل شده، ترمینال Ubuntu را گشوده (با میانبر ctrl+alt+t) و سپس فرمان‌های ذیل را صادر کنید:

```
sudo apt-get update
sudo apt-get upgrade -y
sudo apt-get install apache2
```

به این ترتیب Apache web server بر روی سیستم نصب خواهد شد.

نصب ماژول‌های ASP.NET مخصوص آپاچی

سپس نیاز است ASP.NET runtime for Apache را نصب کنیم:

```
sudo apt-get install libapache2-mod-mono mono-apache-server4
```

mod_mono درون پروسه آپاچی اجرا شده و تمام درخواست‌های رسیده به یک وب سایت ASP.NET را به mod-mono-server که کار هاست سایت را انجام می‌دهد، انتقال خواهد داد.
اگر علاقمند به مشاهده تنظیمات آن بودید باید به مسیر etc/apache2/mods-enabled مراجعه کرده و فایل mod_mono.conf را بررسی کنید (اختیاری). برای مثال، در آن حالت اجرا، بر روی ASP.NET 4 تنظیم شده‌است.

تنظیمات آپاچی برای کار با ASP.NET

نیاز است فایل تنظیمات پیش فرض وب سرور آپاچی را جهت معرفی ASP.NET به آن، اندکی ویرایش کنیم:

```
sudo gedit /etc/apache2/sites-available/default
```

در ادیتور باز شده، ذیل قسمت ServerAdmin، موارد زیر را اضافه کنید:

```
MonoAutoApplication disabled
AddHandler mono .aspx .ascx .asax .ashx .config .cs .asmx .axd
MonoApplications "/:/var/www"
MonoServerPath default "/usr/bin/mod-mono-server4"
```

در ادامه به المان Directory /var/www سطرهای زیر را اضافه کنید:

```
SetHandler mono
DirectoryIndex index.aspx index.html default.aspx Default.aspx
```

المان جدید ذیل را در انتهای فایل اضافه کنید تا یک سری از پوشه‌های خاص ASP.NET همانند محیطهای ویندوزی در آپاچی نیز محافظت شوند:

```
<DirectoryMatch
"/([bB]in|[Aa]pp_[Cc]ode|[Aa]pp_[Dd]ata|[Aa]pp_[Gg]lobal[Rr]esources|[Aa]pp_[Ll]ocal[Rr]esources)/">
```

```
Order deny,allow
Deny from all
</DirectoryMatch>
```

فایل را ذخیره کرده و ادیتور را ببندید. اکنون وب سرور آپاچی را اجرا و ری استارت کنید:

```
sudo service apache2 restart
```

همچنین سطح دسترسی مسیر var/www که حاوی برنامه‌های ما خواهد بود نیز باید تغییر کند:

```
# r,w,exec for user + group
sudo chgrp -R www-data /var/www
sudo chown -R www-data /var/www
sudo chown -R www-data:www-data /var/www
sudo chmod -R 755 /var/www
```

اکنون اگر آدرس http://127.0.0.1 را در مرورگر وارد کنید، باید پیام «It Works» را مشاهده کنید و اگر خواستید محتوای آن‌را ویرایش کنید، دستور ذیل را صادر نمائید:

```
gedit /var/www/index.html
```

اجرای مثال‌ها

با تنظیمات فوق، برنامه‌های کپی شده در مسیر var/www به کمک مونو و آپاچی اجرا خواهند شد.
دسترسی‌های فعلی کاربر وارد شده به لینوکس اجازه کپی فایل‌ها را به مسیر var/www نمی‌دهد. همچنین می‌خواهیم این کارها را توسط File browser آن انجام دهیم و نه خط فرمان. برای این منظور دستور ذیل را اجرا کنید تا File browser آن با دسترسی مدیریتی اجرا شود:

```
sudo nautilus
```

الان می‌توان بدون مشکل در همان File browser گرافیکی آن، پوشه مثال وب فرم‌های خود را به درون پوشه var/www کپی کرد.
پس از کپی دو برنامه Web forms و MVC ابتدای بحث، نیاز است مجددا فایل تنظیمات آپاچی را ویرایش کنیم:

```
sudo gedit /etc/apache2/sites-available/default
```

اکنون تنظیم برنامه وب فرم را به نحو ذیل به این فایل اضافه کنید:

```
AddMonoApplications default "/webforms_test:/var/www/WebFormsApp/"
<Location /webforms_test>
SetHandler mono
</Location>
```

در تنظیم webforms_test:/var/www/WebFormsApp/، قسمت webforms_test نامی است که در مرورگر وارد خواهد شد و قسمت بعد از : مسیر فایل‌های برنامه و ریشه سایت است.
و تنظیم برنامه MVC به صورت زیر می‌باشد:

```
AddMonoApplications default "/mvc_test:/var/www/Mvc4ProjectApp/"
<Location /mvc_test>
SetHandler mono
</Location>
```

بعد از هر بار تغییر فایل تنظیمات آپاچی باید یکبار وب سرور را توسط دستور sudo service apache2 restart ری استارت کنیم.

اکنون برنامه وب فرم‌ها در مسیر http://127.0.0.1/webforms_test و برنامه MVC در مسیر http://127.0.0.1/mvc_test قابل دسترسی است.

خلاصه بحث

پس از نصب وب سرور آپاچی و ماژول مونوی مخصوص آن، فایل etc/apache2/sites-available/default را به نحو ذیل ویرایش کنید و مسیرهای برنامه‌های خود را در آن تعریف نمائید:

```
<VirtualHost *:80>
ServerAdmin webmaster@localhost
        MonoAutoApplication disabled
        AddHandler mono .aspx .ascx .asax .ashx .config .cs .asmx .axd
        MonoApplications "/:/var/www"
        MonoServerPath default "/usr/bin/mod-mono-server4"

DocumentRoot "/var/www"
<Directory />
Options FollowSymLinks
AllowOverride None
</Directory>
<Directory /var/www/>
                SetHandler mono
                DirectoryIndex index.aspx index.html default.aspx Default.aspx
Options Indexes Includes FollowSymLinks MultiViews
AllowOverride None
Order allow,deny
allow from all
</Directory>

ScriptAlias /cgi-bin/ /usr/lib/cgi-bin/
<Directory "/usr/lib/cgi-bin">
AllowOverride None
Options +ExecCGI -MultiViews +SymLinksIfOwnerMatch
Order allow,deny
Allow from all
</Directory>

        <DirectoryMatch
"/([bB]in|[Aa]pp_[Cc]ode|[Aa]pp_[Dd]ata|[Aa]pp_[Gg]lobal[Rr]esources|[Aa]pp_[Ll]ocal[Rr]esources)/">
          Order deny,allow
          Deny from all
        </DirectoryMatch>

        AddMonoApplications default "/webforms_test:/var/www/WebFormsApp/"
        <Location /webforms_test>
         SetHandler mono
        </Location>

       AddMonoApplications default "/mvc_test:/var/www/Mvc4ProjectApp/"
        <Location /mvc_test>
         SetHandler mono
        </Location>

ErrorLog ${APACHE_LOG_DIR}/error.log

# Possible values include: debug, info, notice, warn, error, crit,
# alert, emerg.
LogLevel warn

CustomLog ${APACHE_LOG_DIR}/access.log combined
</VirtualHost>
```

اجرای ASP.NET MVC بر روی لینوکس

نظرات خوانندگان

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۲/۰۷/۰۳ ۹:۴۷

در مورد هاست کردن سرویس‌های WCF هم اگه ممکنه توضیح بدین.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۷/۰۴ ۱۴:۳

الف) تنظیمات سرور

برای اضافه کردن تنظیمات WCF، ابتدا فایل mod_mono.conf را باز کنید:

```
sudo gedit  /etc/apache2/mods-available/mod_mono.conf
```

بعد در سطر اول آن پسوند svc را هم اضافه کنید:

```
AddType application/x-asp-net .svc .aspx .ashx .asmx .ascx .asax .config .ascx
```

و همچنین اینکار را برای فایل default آپاچی انجام دهید:

```
sudo gedit /etc/apache2/sites-available/default
```

و پسوند svc را در اینجا نیز اضافه نمائید:

```
AddHandler mono .svc .aspx .ascx .asax .ashx .config .cs .asmx .axd
```

بعد از آن یکبار سرور را با دستور sudo service apache2 restart ری استارت کنید.

ب) تنظیمات وب کانفیگ

در اینجا serviceMetadata را طوری تنظیم کنید تا با HTTP GET قابل دریافت باشد

```
<system.serviceModel>
    <behaviors>
      <serviceBehaviors>
        <behavior>
          <serviceMetadata httpGetEnabled="true" httpGetUrl="wsdl"/>
        </behavior>
      </serviceBehaviors>
```

بعد از آن سرویس شما در آدرس http://127.0.0.1/webforms_test/Service1.svc/js یا در آدرس http://127.0.0.1/webforms_test/Service1.svc/jsdebug برای کارهای Ajax ایی قابل استفاده خواهد بود (فرض بر این بود که یک Ajax Enabled WCF Service را در VS.NET ایجاد و به لینوکس منتقل کردید).

نویسنده: ایمان اسلامی
تاریخ: ۱۳۹۲/۱۲/۲۰ ۸:۲۷

میخواستم بدونم در شرایط مساوی نصب یک برنامه asp.net روی iis از نظر performance بهتره یا روی apache؟

عنوان: **بهینه سازی برنامه‌های وب ASP.NET برای موتورهای جستجو (SEO)**
نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۷/۲۴ ۱۴:۵۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, C#, MVC, SEO

می‌دانیم بهینه‌سازی موتورهای جستجو (به انگلیسی: Search engine optimization (SEO)) که گاهی در فارسی به آن سئو نیز گفته می‌شود، عملیاتی است برای بهبود دید یک وب‌گاه یا یک صفحهٔ وب، در صفحه نتایج موتورهای جستجو که می‌تواند طبیعی و یا الگوریتمی باشد. این عملیات برای وبمسترها یکی از عوامل مهم و حیاتی بدست آوردن کاربران جدید از موتورهای جستجو است.

اگر چک لیست‌های SEO وب سایت ها را مشاهده کنیم، می‌توانیم آن‌ها را در دو دسته‌ی کلی بهینه سازی درونی و برونی وب سایت در نظر بگیریم :
Off-Page Optimization یا برونی ، که بیشتر بر دوش مشاوران سئو و خود مدیران وب سایت است . (link building ، فعالیت در شبکه اجتماعی و ...)
و اما در حوزه **On-Page Optimization** یا درونی که بخش‌های مهمی از آن وظیفه‌ی مابرنامه نویس‌ها است . (H1 Tag ، URL Naming ، Meta Tags ، عنوان صفحه و ...)
[البته عامل درونی بهینه سازی محتوا (Content Optimization) که مهمترین عامل در الگوریتم‌های نسل جدید موتورهای جستجو و همچنین الگوریتم جدید گوگل (و +) به حساب می‌آید بر عهده مشاوران سئو و خود مدیران وب سایت می‌باشد]

در ادامه به ارائه چند راهکار جهت بهینه سازی برنامه‌های وب ASP.NET مان برای موتورهای جستجو می‌پردازیم:

1.متدی برای ایجاد عنوان سایت

```
private const string SeparatorTitle = " - ";
    private const int MaxLenghtTitle = 60;
    public static string GeneratePageTitle(params string[] crumbs)
    {
        var title = "";

        for (int i = 0; i < crumbs.Length; i++)
        {
            title += string.Format
                        (
                            "{0}{1}",
                            crumbs[i],
                            (i < crumbs.Length - 1) ? SeparatorTitle : string.Empty
                        );
        }

        title = title.Substring(0, title.Length <= MaxLenghtTitle ? title.Length :
MaxLenghtTitle).Trim();

        return title;
    }
```

نکته :
MaxLenghtTitle پیشنهادی برای عنوان سایت 60 می‌باشد.

2.متدی برای ایجاد متاتگ صفحات سایت

```
public enum CacheControlType
{
    [Description("public")]
    _public,
    [Description("private")]
    _private,
    [Description("no-cache")]
    _nocache,
    [Description("no-store")]
    _nostore
```

```
}



private const int MaxLenghtTitle = 60;
private const int MaxLenghtDescription = 170;
private const string FaviconPath = "~/cdn/ui/favicon.ico";
public static string GenerateMetaTag(string title, string description, bool allowIndexPage, bool
allowFollowLinks, string author = "", string lastmodified = "", string expires = "never", string
language = "fa", CacheControlType cacheControlType = CacheControlType._private)
{
        title = title.Substring(0, title.Length <= MaxLenghtTitle ? title.Length :
MaxLenghtTitle).Trim();
        description = description.Substring(0, description.Length <= MaxLenghtDescription ?
description.Length : MaxLenghtDescription).Trim();

        var meta = "";
        meta += string.Format("<title>{0}</title>\n", title);
        meta += string.Format("<link rel=\"shortcut icon\" href=\"{0}\"/>\n", FaviconPath);
        meta += string.Format("<meta http-equiv=\"content-language\" content=\"{0}\"/>\n", language);
        meta += string.Format("<meta http-equiv=\"content-type\" content=\"text/html; charset=utf-
8\"/>\n");
        meta += string.Format("<meta charset=\"utf-8\"/>\n");
        meta += string.Format("<meta name=\"description\" content=\"{0}\"/>\n", description);
        meta += string.Format("<meta http-equiv=\"Cache-control\" content=\"{0}\"/>\n",
EnumExtensions.EnumHelper<CacheControlType>.GetEnumDescription(cacheControlType.ToString()));
        meta += string.Format("<meta name=\"robots\" content=\"{0}, {1}\" />\n", allowIndexPage ?
"index" : "noindex", allowFollowLinks ? "follow" : "nofollow");
        meta += string.Format("<meta name=\"expires\" content=\"{0}\"/>\n", expires);

        if (!string.IsNullOrEmpty(lastmodified))
            meta += string.Format("<meta name=\"last-modified\" content=\"{0}\"/>\n", lastmodified);

        if (!string.IsNullOrEmpty(author))
            meta += string.Format("<meta name=\"author\" content=\"{0}\"/>\n", author);

        //-----------------------------------Google & Bing Doesn't Use Meta Keywords ...
        //meta += string.Format("<meta name=\"keywords\" content=\"{0}\"/>\n", keywords);

        return meta;
}
```

چند نکته :

MaxLenghtDescription پیشنهادی برای متاتگ توضیح سایت 170 می باشد.

آشنایی با متاتگ‌ها (Meta tags) و کاربرد آنها در صفحات وب (HTML)

برای کاربرد allowIndexPage و allowFollowLinks هم می‌توانید به لینک بالا و بررسی متاتگ robots بپردازید.

با توجه به اهمیت شبکه‌های اجتماعی متاتگ‌های شبکه‌های اجتماعی (+ و +) را هم نباید از قلم انداخت.

برای دریافت Description نوع سفارشی CacheControlType از پروژه متدهای الحاقی علیرضا اسم رام استفاده کردم.

3.متدی برای ایجاد Slug (اسلاگ آدرسی با مفهوم برای بکار بردن در URL ها است که دوست‌دار موتورهای جستجو می‌باشد)

```
private const int MaxLenghtSlug = 45;
public static string GenerateSlug(string title)
{
        var slug = RemoveAccent(title).ToLower();
        slug = Regex.Replace(slug, @"[^a-z0-9-\u0600-\u06FF]", "-");
        slug = Regex.Replace(slug, @"\s+", "-").Trim();
        slug = Regex.Replace(slug, @"-+", "-");
        slug = slug.Substring(0, slug.Length <= MaxLenghtSlug ? slug.Length : MaxLenghtSlug).Trim();

        return slug;
}

private static string RemoveAccent(string text)
{
        var bytes = Encoding.GetEncoding("UTF-8").GetBytes(text);
        return Encoding.UTF8.GetString(bytes);
}
```

نکته :

MaxLenghtSlug پیشنهادی برای عنوان سایت 45 می‌باشد.

نمونه ای از کاربرد توابع :

```
Head.InnerHtml = SEO.GenerateMetaTag
                            (
                                title: SEO.GeneratePageTitle(".NET Tips", "آرشیو مطالب", "ASP.NET MVC
#1"),
                                description: "چرا ASP.NET MVC با وجود فریم ورک پخته‌ای به نام ASP.NET web
forms، مطرح می‌شود این است: «برای چی؟». بنابراین تا به این سؤال ASP.NET MVC اولین سؤالی که حین سوئیچ به
پاسخ داده نشود، هر نوع بحث فنی در این مورد بی فایده است.",
                                allowIndexPage: true,
                                allowFollowLinks: true,
                                author: "وحید نصیری",
                                cacheControlType: SEO.CacheControlType._private
                            );
```

و خروجی در Page Source :

```
<title>.NET Tips - آرشیو مطالب - ASP.NET MVC #1</title>
<link rel="shortcut icon" href="../../cdn/images/ui/favicon.ico"/>
<meta http-equiv="content-language" content="fa"/>
<meta http-equiv="content-type" content="text/html; charset=utf-8"/>
<meta charset="utf-8"/>
<meta name="description" content="چرا ASP.NET MVC با وجود فریم ورک پخته‌ای به نام ASP.NET web forms؟
مطرح می‌شود این است: &#171;برای چی؟&#187;. بن ASP.NET MVC اولین سؤالی که حین سوئیچ به ..."/>
<meta http-equiv="Cache-control" content="private"/>
<meta name="robots" content="index, follow" />
<meta name="expires" content="never"/>
<meta name="author" content="وحید نصیری"/>
```

موفق باشید

نظرات خوانندگان

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۹/۰۸ ۱۵:۲۰

دوست من با این روش

```
Head.InnerHtml = SEO.GenerateMetaTag(....)
```

در صورتی که در قسمت head صفحه استایل یا متاتگهای دیگری تعریف شده باشد همگی حذف خواهند شد.

نویسنده: محسن نجف زاده
تاریخ: ۱۳۹۲/۰۹/۰۸ ۱۸:۴۵

بله صابر جان حتما همین طور است (برای مثال بیان شده است).
خودم به این صورت استفاده می‌کنم :

```
<head>
    <asp:Literal runat="server" ID="litHead"></asp:Literal>

    ....

</head>
```

و در قسمت *code - behind*

```
litHead.Text = SEO.GenerateMetaTag(...)
```

پ ن : چرا از کنترل Literal استفاده شده و از Label استفاده نشده ؟ به این دلیل که خروجی متن رندر شده Label توسط تگ یا <label> احاطه می‌شود. (+)

نویسنده: پوریا منفرد
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۲:۱۵

با سلام
امکان داره این خط کلاسش رو هم بزارید بنده دقیقا نتونستم با متد الحاقی که گفتین این قسمت رو پیاده سازی کنم

```
EnumExtensions.EnumHelper<CacheControlType>.GetEnumDescription(cacheControlType.ToString()));
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۲:۲۳

مراجعه کنید به پروژه DNT.Extensions و همچنین اینجا

نویسنده: پوریا منفرد
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۳:۲۸

بنده یه مشکلی دارم : از MasterPage استفاده می‌کنم در ASP.net Webform و فرمی رو بهش Connect کردم که:
1- در MPage یک سری متا تگ‌ها هست + JavaScript + CSS
2- وقتی در یک webForm متصل به Mpage کد:

```
Head.InnerHtml = SEO.GenerateMetaTag ....
```

رو قرار می‌دهم تمام تگ‌های Head حتی MasterPage هم پاک می‌شه
اینم می‌دونم این کلاس این کارو می‌کنه
راه حل چیه که تگ‌ها پاک نشه ؟ مخصوصا آدرس‌های CSS و JavaScript ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۳:۵۱

کمی بالاتر توضیح دادند ؛ استفاده از یک Literal برای مثال.

نویسنده: مهرداد پاک دل
تاریخ: ۱۳۹۲/۱۱/۰۲ ۱۶:۱۶

من این توابع را خواندم که در مورد seo نوشته بودید ولی متاسفانه نحوه استفاده از آن هارا نفهمیدم یعنی کجا تعریف میشن و از کجا پارامترهای خود را دریافت می‌کنند. لطف کنید اگر امکان داره یه پروژه کوچک همه این‌ها استفاده شده باشه اگر هست بگذارید .
نکته دیگه شما عنوان هر پست را داخل urlها نمایش میدهید منظور اینکه جزئی از url سایت است این کار را چگونه انجام می‌دهید

```
mysite.info/post/1
```

```
dotnettips.info/post/1200/seo
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۲ ۱۶:۳۸

- پروژه Iris هست. به section metatags اون دقت کن در فایل‌های View.

- مثلا متد Head.InnerHtml = SEO.GenerateMetaTag عنوان شده در این مطلب باید در Page_Load یک وب فرم فراخوانی شود. Id مطلب رو دارید. عنوان و متن و سایر مشخصات اون رو از دیتابیس دریافت کنید و بعد فقط یک جایگذاری است در متد تهیه شده.

- به اینکار Routing و Url rewrite می‌گن. بحثش در MVC و وب فرم‌ها کمی با هم فرق می‌کنه.

نویسنده: امین قادری
تاریخ: ۱۳۹۳/۰۵/۰۹ ۳:۳۵

لطفا مثالی را در مورد نحوه‌ی استفاده از " **متدی برای ایجاد Slug** " در عمل ارسال نمائید. با تشکر

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۸:۹

سلام منم کدها رو تویه فایل csبه نام سئو دخیره کردم اما اینجا وقتی ران میکنم ارور میده

```
public enum CacheControlType
{
    [Description("public")]
    _public,
```

```
    [Description("private")]
    _private,
    [Description("no-cache")]
    _nocache,
    [Description("no-store")]
    _nostore
}
```

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۹:۵

« ارور می‌ده » برای رفع مشکل کافی نیست. چه خطایی می‌ده دقیقا؟

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۳ ۸:۴۰

درسته حق با شماست. این ارور هست؛ یه نگاهی بندازین فکر کنم با فرم وورکم مشکل داره؟

```
Attribute 'Description' is not valid on this declaration type. It is only valid on 'method'
declarations.
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۳ ۹:۲۹

- ابتدا باید اسمبلی استاندارد System.ComponentModel.DataAnnotations به ارجاعات پروژه اضافه شود.
- سپس باید از فضای نام مرتبط استفاده کنید (و نه از فضاهای نام مشابه)

```
enum MyEnum {
    [System.ComponentModel.Description("Blah")]
    MyValue
}
```

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۳ ۱۵:۵۸

بازم تشکر بله مشکل همین بود و طبق دستور شما استفاده کردم و در headهم از یه لیترال استفاده کردم وخیلی خوب جواب داد.زنده باشین

عنوان: **تنظیمات امنیتی Glimpse**
نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۲/۰۸/۱۴ ۰:۴۵
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net, MVC, Glimpse, Profiler

در مورد glimpse پیشتر مطالبی در سایت منتشر شده است :

آشنایی و بررسی ابزار Glimpse

بعد از آپلود سایت ما می‌توانیم دسترسی به تنظیمات خاص glimpse را تنها به کاربران عضو محدود کنیم:

```
<location path="Glimpse.axd" >
    <system.web>
        <authorization>
            <allow users="Administrator" />
            <deny users="*" />
        </authorization>
    </system.web>
</location>
```

یا می‌توانیم آنرا غیرفعال کنیم :

```
<glimpse defaultRuntimePolicy="Off" xdt:Transform="SetAttributes">
</glimpse>
```

همچنین می‌توانیم با پیاده سازی اینترفیس IRuntimePolicy سیاست‌های مختلف نمایش تب‌های glimpse را تعیین کنیم :

```
using Glimpse.AspNet.Extensions;
using Glimpse.Core.Extensibility;

namespace Test
{
    public class GlimpseSecurityPolicy:IRuntimePolicy
    {
        public RuntimePolicy Execute(IRuntimePolicyContext policyContext)
        {
            // You can perform a check like the one below to control Glimpse's permissions within your
application.
// More information about RuntimePolicies can be found at http://getglimpse.com/Help/Custom-Runtime-
Policy
var httpContext = policyContext.GetHttpContext();
            if (!httpContext.User.IsInRole("Administrator "))
            {
                return RuntimePolicy.Off;
            }

            return RuntimePolicy.On;
        }

        public RuntimeEvent ExecuteOn
        {
            get { return RuntimeEvent.EndRequest; }
        }
    }
}
```

زمانیکه glimpse را از طریق Nuget نصب می‌کنید کلاس فوق به صورت اتوماتیک به پروژه اضافه می‌شود با این تفاوت که به صورت کامنت شده است تنها کاری شما باید انجام بدید کدهای فوق را از حالت کامنت خارج کنید و Role مربوطه را جایگزین کنید.

نکته : کلاس فوق نیاز به رجیستر شدن ندارد و تشخیص آن توسط Glimpse به صورت خودکار انجام می‌شود.

عنوان: **نحوه‌ی بستن یک بازه‌ی IP در IIS**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۱۱ ۸:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, Security

یک سری از ربات‌ها مدام سایت‌ها را برای یافتن یک سری از اسکریپت‌های خاص اسکن می‌کنند. IPهای آن‌ها نیز عموما متعلق است به چین و همسایگان آن. مشکلی که با این ربات‌ها وجود دارد این است که از یک IP خاص نشات نمی‌گیرند و به نظر صدها سرور آلوده را جهت مقاصد خود مورد استفاده قرار می‌دهند. به همین جهت نیاز است بتوان یک بازه‌ی IP را در IIS بست.

بستن یک بازه‌ی IP در IIS 6

در IIS6 باید به خواص وب سایت و برگه‌ی Directory security آن مراجعه کرد. سپس در این قسمت، در حین افزودن IP مد نظر، باید گزینه‌ی Group of computers را انتخاب نمود.

در اینجا برای مثال برای بستن IPهایی که با 194 شروع می‌شوند، باید 194.0.0.0 را وارد کرد و در این حالت subnet mask را نیز باید 255.0.0.0 انتخاب کرد. با این subnet mask خاص، اعلام می‌کنیم که فقط اولین قسمت IP وارد شده برای ما اهمیت دارد و سه قسمت بعدی خیر. به این ترتیب تمام IPهای شروع شده با 194 با هر سه جزء دیگر دلخواهی، بسته خواهند شد.

بستن یک بازه‌ی IP در IISهای 7 به بعد

در IISهای 7 به بعد نیاز است مراحل زیر را طی کرده و نقش « IP and Domain Restrictions » را نصب کنید.

```
Administrative Tools -> Server Manager -> expand Roles
-> Web Server (IIS) ->  Role Services -> Add Role Services -> select IP and Domain Restrictions
```

پس از آن با استفاده از تنظیمات ذیل در فایل web.config می‌توان یک IP و یا بازه‌ای از IPها را بست:

```
<system.webServer>
      <security>
         <ipSecurity>
            <add ipAddress="192.168.100.1" />
            <add ipAddress="169.254.0.0" subnetMask="255.255.0.0" />
         </ipSecurity>
      </security>
   </system.webServer>
```

البته علاوه بر نصب نقش یاد شده، باید Feature Delegation نیز جهت استفاده از آن فعال گردد:

```
IIS 7 -> root server -> Feature Delegation -> IP and Domain Restrictions
->  Change the delegation to Read/Write
```

نظرات خوانندگان

نویسنده: آرمان فرقانی
تاریخ: ۱۳۹۲/۰۹/۱۷ ۱۲:۴۶

ضمن تشکر بفرمایید چرا از طریق فایروال دسترسی IPهای یاد شده را مسدود نمی‌کنید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۱۷ ۱۲:۴۷

این هم یک روش است؛ البته برای کسی که ادمین است. اگر ادمین نباشد و ادمین قبلا قابلیت ذکر شده مخصوص IIS 7 را افزوده باشد، کاربران یک هاست اشتراکی هم می‌توانند راسا و بدون نیاز به ادمین، فقط با ویرایش کردن فایل web.config، اقدام کنند.

نویسنده: آرمان فرقانی
تاریخ: ۱۳۹۲/۰۹/۱۷ ۱۳:۲

بله. البته منابع سیستمی که اسکن‌های یاد شده مصرف می‌کنند، معمولاً مسدود نمودن آنها را در حیطه وظایف مدیر سرور قرار می‌دهد و البته در مورد IIS 6 دسترسی به وب سرور لازم است. و احتمالاً مسدود نمودن از طریق فایروال از نظر سربار و مصرف منابع ارجح است. از طرفی باید در نظر داشت که بسیار دیده شده که همان سرورهای چینی علاقه مند به اسکن سایت‌ها علاقه مند به حملات DDOS و ... هم هستند که الزاماً از مسیر IIS نمی‌گذرد.

در هر صورت ممنون از بیان مطلب فوق که به هر حال دانستن آن بهتر از ندانستن است.سوال کردم چون گفتم شاید دلیل خاصی دارد که این روش را بیان فرمودید.

عنوان: **انتقال دائمی آدرس ها (Permanent Ridirect) در ASP.NET Webform**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۹/۱۱ ۲۰:۱۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, ASP.Net 4.5, SEO, FriendlyUrls, routing

یکی از مسائلی که همیشه برای وب سایت، از نظر موتورهای جستجو بسیار مهم است، مقوله SEO Friendly بودن آدرس‌هاست. خوشبختانه در ASP.NET MVC این مقوله به خوبی پوشش داده شده است. برای مثال این قابلیت در ASP.NET Webforms توسط بسته نرم افزاری به نام ASP.NET FriendlyUrls توسط تیم توسعه دهنده ASP.NET ارائه شده است. در این زمینه قبلا آقای کاویانی در همین سایت در صفحه مروری بر قابلیت جدید ASP.NET FriendlyUrls به این مقوله پرداخته‌اند و با مثال، نحوه‌ی استفاده از این بسته‌ی نرم افزاری را آموزش داده‌اند.
یکی از مسائلی که در ASP.NET Webforms با آن مواجه ~~بودیم~~ انتقال دائمی آدرس‌های صفحات یا همان Permanent Redirect بود. در واقع زمانیکه شما در حالت عادی از بسته‌ی نرم افزاری مذکور استفاده می‌کنید، این افزونه باعث SEO Friendly شدن آدرس‌های صفحات شما می‌گردد. با در نظر گرفتن اینکه شما مطلب مروری بر قابلیت جدید ASP.NET FriendlyUrls را مطالعه کرده اید، به فرض مثال کد زیر:

```
public static class RouteConfig
    {
        public static void RegisterRoutes(RouteCollection routes)
        {
            routes.EnableFriendlyUrls();
            routes.MapPageRoute("Sitemap", "sitemap.xml", "~/sitemap.aspx");
        }
    }
```

و رجیستر کردن آن در RouteTable سایت به صورت زیر در فایل Global

```
protected void Application_Start(object sender, EventArgs e)
{
     RouteConfig.RegisterRoutes(RouteTable.Routes);
}
```

باعث می‌شود که با زدن آدرس http://www.dotnettips.info/sitemap.xml آدرس http://www.dotnettips.info/sitemap.aspx لود شود؛ اما در نوار آدرس، همان آدرس sitemap.xml باقی خواهد ماند.
در این روش در صورتیکه کاربر آدرس http://www.dotnettips.info/sitemap.aspx را در مرورگر خود تایپ کند باز هم سایت کار می‌کند (آدرس مورد نظر در دسترس است) و فاجعه زمانی شروع می‌شود که موتور جستجو، با محتوی تکراری در سایت در دو آدرس مختلف مواجه شود (پست آقای نصیری را در بهبود SEO در ASP.NET MVC بررسی کنید) .
یا زمانیکه شما صفحات قدیمی دارید و می‌خواهید آنها را به آدرس‌های جدید منتقل کنید. برای رفع این مشکل طوری‌که هر زمان آدرس قدیمی هم وارد سیستم مسیریابی سایت شود به صورت اتوماتیک به آدرس جدید به صورت دائم منتقل شود، می‌توانید کد بالا را به صورت زیر اصلاح نمایید:

```
public static void RegisterRoutes(RouteCollection routes)
 {
            routes.EnableFriendlyUrls(new FriendlyUrlSettings()
            {
                AutoRedirectMode = RedirectMode.Permanent,
                ResolverCachingMode = ResolverCachingMode.Dynamic
            });
            routes.MapPageRoute("Sitemap", "sitemap.xml", "~/sitemap.aspx");
}
```

با این تنظیمات، سیستم مسیریابی ملزم به انتقال دائمی آدرس، به آدرس جدید خواهد شد. یعنی شما حتی اگر آدرس http://www.dotnettips.info/sitemap.aspx را وارد نمایید مسیریاب سایت برای آدرس مورد نظر، وضعیت Moved 301 Permanently را مقدار دهی کرده و فقط آدرس http://www.dotnettips.info/sitemap.xml به عنوان آدرس اصلی در نظر گرفته خواهد شد.

نظرات خوانندگان

نویسنده: طاهر
تاریخ: ۱۳۹۲/۰۹/۱۲ ۱۴:۱۸

سلام
ASP.NET FriendlyUrls چه تفاوتی با Routing که در دات نت 4 به وب فرم اضافه شده دارد؟

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۰۹/۱۲ ۱۵:۳۹

از همون سیستم Routing دات نت استفاده می‌کنه با این تفاوت که کار باهاش راحتره،

نویسنده: سالار
تاریخ: ۱۳۹۳/۰۵/۲۹ ۲۲:۴۱

سلام؛ ممنون به خاطر مطلب مفیدی که قرار دادید. من این تنظیمات رو اعمال کردم ولی بازم با درخواست هر دو آدرس ، صفحه بارگذاری میشه. در واقع باید هر دو آدرس رو اگر تایپ کردیم مشکلی پیش نیاد درسته؟ با asp.net webforms 4.5 کار میکنم.ممنون میشم بگید مشکل از چیه

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۰۵/۳۱ ۱۴:۲۱

لطفا نمونه کد خودتون بذارید بررسی کنم

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۳۱ ۱۸:۱۱

در مثال پیوست شده زمانیکه sitemap معمولی فراخوانی میشه به sitemap.xml هدایت نمیشه (وضعیت Moved 301 Permanently در مرورگر قابل مشاهده نیست): FriendlyUrlsTests.zip

نویسنده: غلامرضا
تاریخ: ۱۳۹۳/۰۵/۳۱ ۲۰:۳۷

ممنون به خاطر پاسختون . پس دقیقا به این صورت هستش که هر دو آدرس رو اگر درخواست کنیم صفحه مورد نظر نمایش داده میشه؟ ولی مسیر یاب گوگل این رو مسیر تکراری به حساب نمیاره؟ خطا رو نمایش نمیده پس؟

عنوان: **جلوگیری از درج صفحات سایت در سایتی دیگر از طریق iframeها**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۴ ۸:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, JavaScript, MVC, Security

یکی از مواردی را که به کرات در ارجاع دهنده‌های سایت مشاهده می‌کنم، درج صفحات سایت به صورت iframe داخل یک سری سایت‌های تبلیغاتی بسیار سطحی است. سؤال: چگونه می‌توان جلوی این حرکت نامطلوب را گرفت؟
برای این‌کار نیاز است اسکریپت ذیل را به ابتدای اسکریپت‌های سایت خود اضافه کنید:

```
function defrm() {
    document.write = '';
    window.top.location = window.self.location;
    setTimeout(function () {
        document.body.innerHTML = '';
    }, 0);
    window.self.onload = function (evt) {
        document.body.innerHTML = '';
    };
}
if (window.top !== window.self) {
    try {
        if (window.top.location.host)
        { /* will throw */ }
        else {
            defrm(); /* chrome */
        }
    } catch (ex) {
        defrm(); /* everyone else */
    }
}
```

اگر شیء window.top با شیء window.self یکی نبود (یعنی window جاری، window رویی نبود)، بنابراین صفحه‌ی جاری، داخل iframe قرار گرفته‌است. همچنین برای اطمینان بیشتر می‌توان window.top.location.host را نیز فراخوانی کرد. مرورگرهای جدید امکان دسترسی به دومین والد را از طریق یک iframe نمی‌دهند و همینجا یک استثناء صادر خواهد شد.
در مرحله بعد، با فراخوانی window.top.location = window.self.location و سایر کدهایی که مشاهده می‌کنید، کل صفحه را به صورت خودکار به سایت خودمان هدایت خواهیم کرد.

این مورد را نیز می‌توان به مجموعه best practices اجباری تهیه یک سایت اضافه کرد.

نظرات خوانندگان

نویسنده: حامد سبزیان
تاریخ: ۱۳۹۲/۰۹/۲۴ ۹:۴

استفاده از X-FRAME-OPTIONS در هدر درخواست با مقدار SAMEORIGIN با DENY کفایت نمیکنه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۴ ۹:۲۰

خوبه ؛ فقط redirect بحث شده در مطلب را نداره

نویسنده: امید قربانی
تاریخ: ۱۳۹۲/۰۹/۲۷ ۹:۱۹

با سلام؛ اگر بخواهیم این موردی رو که اشاره کردید رو دور بزنیم چطور باید عمل کنیم منظورم اینه که اگر سایتی همچین قابلیتی رو اضافه کرده بود چطور از بین ببریمش ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۷ ۹:۲۶

حداقل با استفاده از جاوا اسکریپت، به دلایل امنیتی ، امکان دسترسی به محتویات یک iframe بارگذاری شده از سایت دیگری را ندارید؛ مگر اینکه سمت سرور، محتویات آن صفحه را دریافت و در ادامه این صفحه‌ی محلی شده را نشان دهید.

نویسنده: behrouz
تاریخ: ۱۳۹۲/۱۲/۲۸ ۲۲:۱۰

سلام
آیا سمت سرور راهی هست که بتوان تشخیص داد صفحاتی که درخواست داده می‌شوند از درون یک Iframe فرخوانی می‌شوند یا خیر(به طور خاص در ASP.NET MVC)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۸ ۲۲:۵۹

خیر. راه حل تشخیص سمت سرور، ندارد. هیچ هدر خاصی برای مشخص سازی درخواست اجرای یک صفحه از درون یک IFrame طراحی نشده‌است. حداکثر می‌توان Request.UrlReferrer را بررسی کرد (ولی اطمینانی به آن نیست. چون عوامل زیادی در تغییر آن دخیل هستند؛ از مرورگر تا فایروال و غیره)

عنوان: **افزودن هدرهای Content Security Policy به برنامه‌های ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۲۹ ۱۳:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Security

مرورگرهای جدید تحت زیر مجموعه‌ای به نام Content Security Policy، قابلیت‌های توکاری را اضافه کرده‌اند تا حملاتی مانند XSS را حتی در برنامه‌ی وبی که برای این نوع حملات تمهیداتی را درنظر نگرفته‌است، خنثی کنند. این قابلیت‌ها به صورت پیش فرض فعال نبوده و نیاز است برنامه نویس صراحتا درخواست فعال شدن آن‌ها را از طریق افزودن تعدادی هدر مشخص به Response، ارائه دهد. در ادامه این هدرها را بررسی خواهیم کرد.

غیرفعال کردن اجرای اسکریپت‌های inline

عمده‌ی حملات XSS زمانی قابلیت اجرا پیدا می‌کنند که مهاجم بتواند به طریقی (ورودی‌های اعتبارسنجی نشده)، اسکریپتی را به درون صفحه‌ی جاری تزریق کند. بنابراین اگر ما به مرورگر اعلام کنیم که دیگر اسکریپت‌های inline را پردازش نکن، سایت را تا حد زیادی در مقابل حملات XSS مقاوم کرده‌ایم. این قابلیت به صورت پیش فرض خاموش است؛ چون به طور قطع فعال سازی آن بسیاری از سایت‌هایی را که عادت کرده‌اند اسکریپت‌های خود را داخل صفحات وب مدفون کنند، از کار می‌اندازد. این نوع سایت‌ها باید به روز شده و اسکریپت‌ها را از طریق فایل‌های خارجی js، به سایت و صفحات خود الحاق کنند.
برای فعال سازی این قابلیت، فقط کافی است هدرهای زیر به Response اضافه شوند:

```
Content-Security-Policy: script-src 'self'
X-WebKit-CSP: script-src 'self'
X-Content-Security-Policy: script-src 'self'
```

سطر اول به زودی تبدیل به یک استاندارد W3 خواهد شد؛ اما فعلا فقط توسط کروم 25 به بعد پشتیبانی می‌شود. سطر دوم توسط مرورگرهایی که از موتور WebKit استفاده می‌کنند، پشتیبانی می‌شود و سطر سوم مخصوص فایرفاکس است و IE 10 به بعد.
بعد از فعال شدن این قابلیت، فقط اسکریپت‌هایی که از طریق دومین شما به صفحه الحاق شده‌اند، قابلیت اجرا را خواهند یافت و کلیه اسکریپت‌های مدفون شده داخل صفحات، دیگر اجرا نخواهد شد. در این حالت اگر از CDN برای الحاق اسکریپتی استفاده می‌کنید، مثلا مانند الحاق jQuery به صفحه، نیاز است مسیر آن‌را صراحتا در این هدر ذکر کنید:

```
Content-Security-Policy: script-src 'self' https://youcdn.com
X-WebKit-CSP: script-src 'self' https://yourcdn.com
X-Content-Security-Policy: script-src 'self' https://yourcdn.com
```

علاوه بر آن حتی می‌شود پردازش تمام منابع مورد استفاده را نیز مانند تصاویر، شیوه‌نامه‌ها، فایل‌های فلش و غیره، به دومین جاری محدود کرد:

```
Content-Security-Policy: default-src 'self' https://youcdn.com
X-WebKit-CSP: default-src 'self' https://yourcdn.com
X-Content-Security-Policy: default-src 'self' https://yourcdn.com
```

بدیهی است پس از آشنایی با این مورد، احتمالا در پروژه‌های جدید خود از آن استفاده کنید (چون inline scriptهای فعلی شما را کاملا از کار می‌اندازد).

نحوه‌ی اضافه کردن هدرهای Content Security Policy به برنامه‌های ASP.NET

روشی که با هر دو برنامه‌های وب فرم و MVC کار می‌کند، تهیه یک HTTP module است؛ به شرح ذیل:

```
using System;
using System.Web;

namespace AntiXssHeaders
{
```

```
    public class SecurityHeadersConstants
    {
        public static readonly string XXssProtectionHeader = "X-XSS-Protection";
        public static readonly string XFrameOptionsHeader = "X-Frame-Options";
        public static readonly string XWebKitCspHeader = "X-WebKit-CSP";
        public static readonly string XContentSecurityPolicyHeader = "X-Content-Security-Policy";
        public static readonly string ContentSecurityPolicyHeader = "Content-Security-Policy";
        public static readonly string XContentTypeOptionsHeader = "X-Content-Type-Options";
    }

    public class ContentSecurityPolicyModule : IHttpModule
    {
        public void Dispose()
        { }

        public void Init(HttpApplication app)
        {
            app.BeginRequest += AppBeginRequest;
        }

        void AppBeginRequest(object sender, EventArgs e)
        {
            var app = (HttpApplication)sender;
            var response = app.Context.Response;
            setHeaders(response);
        }

        private static void setHeaders(HttpResponse response)
        {
            response.Headers.Set(SecurityHeadersConstants.XFrameOptionsHeader, "SameOrigin");

            // For IE 8+
            response.Headers.Set(SecurityHeadersConstants.XXssProtectionHeader, "1; mode=block");
            response.Headers.Set(SecurityHeadersConstants.XContentTypeOptionsHeader, "nosniff");

            //todo: Add /Home/Report --> public JsonResult Report() { return Json(true); }

            const string cspValue = "default-src 'self';";
            // For Chrome 16+
            response.Headers.Set(SecurityHeadersConstants.XWebKitCspHeader, cspValue);

            // For Firefox 4+
            response.Headers.Set(SecurityHeadersConstants.XContentSecurityPolicyHeader, cspValue);
            response.Headers.Set(SecurityHeadersConstants.ContentSecurityPolicyHeader, cspValue);
        }
    }
}
```

و یا در برنامه‌های ASP.NET MVC می‌توان یک فیلتر جدید را تعریف کرد و سپس آن‌را به صورت عمومی معرفی نمود:

```
//// RegisterGlobalFilters -> filters.Add(new ContentSecurityPolicyFilterAttribute());
public class ContentSecurityPolicyFilterAttribute : ActionFilterAttribute
{
    public override void OnActionExecuting(ActionExecutingContext filterContext)
    {
        var response = filterContext.HttpContext.Response;
        response.AddHeader("Content-Security-Policy", "script-src 'self'");
        // the rest ...
        base.OnActionExecuting(filterContext);
    }
}
```

در ماژول تهیه شده چند مورد دیگر را نیز مشاهده می‌کنید:

الف) X-XSS-Protection مربوط است به IE 8 به بعد

ب) تنظیم هدر X-Frame-Options به SameOrigin سبب می‌شود تا صفحات سایت شما دیگر توسط Iframeها در سایت‌های دیگر قابل نمایش نباشد و فقط در سایت جاری بتوان صفحه‌ای را از همان دومین در صورت نیاز توسط Iframeها نمایش داد.

ج) تنظیم X-Content-Type-Options به nosniff سبب می‌شود تا IE سعی نکند با اجرای یک محتوا سعی در تشخیص mime-type آن کند و به این ترتیب امنیت دسترسی و مشاهده اشیاء قرار گرفته در صفحه (و یا تزریق شده توسط مهاجمین) به شدت بالا خواهد رفت.

برای مطالعه بیشتر

Security through HTTP response headers

پروژه‌ی کاملی مخصوص افزودن هدرهای یاد شده

https://nwebsec.codeplex.com/

یک نکته تکمیلی

توصیه شده‌است تا دیگر از روال رویدادگردان PreSendRequestHeaders برای ارسال هدرها استفاده نکنید؛ چون با پردازش‌های غیرهمزمان تداخل ایجاد می‌کند.

نظرات خوانندگان

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۲/۰۹/۳۰ ۹:۲۳

آیا اگر اجرای Inline غیر فعال شود و پردازش تمام منابع مورد استفاده به دامین جاری محدود شود مهاجم نمی‌تواند یک ارجاع به اسکریپت مخرب خود به صفحه اضافه کند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۳۰ ۹:۲۷

اجرا نمی‌شود؛ مگر اینکه به هدر ارسالی مانند CDN ذکر شده در متن، آدرس آن اضافه و صراحتا مجوز استفاده از دومینی دیگر توسط برنامه صادر شود.

نویسنده: ناصر طاهری
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۲:۲۲

یعنی کلیه اسکری‌ت‌ها حتی اسکری‌ت هایی مانند :

```
<script type="text/javascript">
        $(document).ready(function () {
            PopupForm.ShowForm({
                renderFormUrl : "/postreply/renderreplyform",
                .....
          });
        });
</script>
```

اجازه‌ی اجرا ندارند؟ و باید در یک فایل جدا به صفحه تزریق شوند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۳:۱۸

بله. این مثال را در کروم اجرا کنید:

AntiXssHeaders.zip

در صفحه اول آن

```
<script type="text/javascript">
        alert('test');
    </script>
```

درج شده

نویسنده: ناصر طاهری
تاریخ: ۱۳۹۲/۰۹/۳۰ ۱۳:۳۷

ممنون از مثالتون. یک پروژه فروشگاه دارم که داخل هر صفحه‌ی آن پر است از اسکریپت‌های به قول شما مدفون شده. حتی فکر پیدا کردن و منتقل کردن هر کدوم از اونها به داخل یک فایل خارجی وحشت آور است.

نویسنده: رسول آذری
تاریخ: ۱۳۹۲/۱۰/۰۴ ۱۳:۴۷

با سلام . تشکر

من با مرورگر ie9 تست کردم . ولی کد اجرا شد. مگه نگفتین واسه ie8 به بعد؟
و اینکه بعد از استفاده از این روش چطور میتونم از فایل‌های js و jquery خودم توی برنامه استفاده کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۰۴ ۱۴:۱۸

- رفتار IE در مورد X-XSS-Protection متفاوت است. یعنی باید واقعا تشخیص بدهد که اسکریپت در حال اجرا یک حمله محسوب می‌شود. یک مثال دریافت کوکی برای امتحان
- مانند قبل. در متن ذکر شده: « سایت‌ها باید به روز شده و اسکریپت‌ها را *از طریق فایل‌های خارجی js* ، به سایت و صفحات خود الحاق کنند »

نویسنده: رسول آذری
تاریخ: ۱۳۹۲/۱۰/۰۴ ۱۴:۳۵

ممنون از پاسخ تون.
یه سری از کدها را میشه توی فایل خارجی قرار داد.
ولی کدهای جاوایی که گریدویو تولید میکنه،چی کار میشه کرد؟
با استفاده از این روش رویدادهایی مثل SelectedIndexChanged در dropdownlist ،
RowCommand در گریدویو ، updatepanel و scriptmanager از کار می‌افته.آیا راهی برای این مشکل وجود داره؟
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۰۴ ۱۵:۲

- به ASP.NET MVC کوچ کنید تا کنترل کاملی بر روی عناصر اضافه شونده به صفحه داشته باشید.
- با استفاده از jQuery به هر عنصری در صفحه می‌توان رویدادهای خاصی را انتساب داد یا حذف کرد. انجام اینکار از طریق یک فایل js الحاق شده به صفحه نیز میسر است.

عنوان: **امکان اعتبارسنجی با تاخیر در ASP.NET 4.5**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۰۴ ۱:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Security

در نگارش‌های قبلی ASP.NET Web forms اگر نیاز به ارسال محتوای HTML ایی وجود داشت، می‌بایستی کل سیستم اعتبارسنجی حداقل یک صفحه را غیرفعال کرد. برای مثال:

```
<%@ Page Language="C#" ValidateRequest="false" %>
```

این نقیصه‌ی همه یا هیچ، در ASP.NET MVC وجود ندارد و می‌توان به ازای یک خاصیت خاص، اعتبارسنجی پیش فرض را با اعمال ویژگی AllowHtml موقتا غیرفعال کرد؛ اما مابقی فیلدها و خاصیت‌های فرم همچنان تحت نظر سیستم اعتبارسنجی‌های ورودی ASP.NET قرار خواهند داشت و به این ترتیب امکان ورود اطلاعات خطرناک، خصوصا از لحاظ مباحث XSS، حداقل در آن فیلدها وجود نخواهد داشت.
در ASP.NET 4.5 مفهوم جدیدی به نام Deferred validation معرفی شده‌است تا رفتار Web forms را نیز در برابر ورودی‌های ارسال شده همانند ASP.NET MVC کند. به این معنا که اگر جهت دسترسی به مقدار کوئری استرینگ lastName همانند قبل عمل کنید:

```
Request["lastName"]
```

و کاربر ورودی خطرناکی را وارد کرده باشد، همچنان استثنای A potentially dangerous request.form value was detected صادر می‌شود.
اما اگر در ASP.NET 4.5، کوئری استرینگ را به نحو ذیل دریافت کنید:

```
Request.Unvalidated.QueryString["lastName"]
```

به صورت خودکار سیستم اعتبارسنجی غیرفعال شده و امکان دسترسی به محتوای اصلی کوئری استرینگ lastName را خواهید داشت. به این ترتیب دیگر نیازی نخواهید داشت تا اعتبارسنجی کل صفحه را برای دسترسی به یک مقدار خاص غیرفعال نمائید.
برای دسترسی به مقادیر اعتبارسنجی نشده فیلدهای یک فرم ارسالی نیز می‌توان به صورت ذیل عمل کرد:

```
Request.Unvalidated.Form[txtData1.UniqueID]
```

کلاس Request.Unvalidated شامل خاصیت‌های Cookies، Files، Headers و امثال آن نیز می‌باشد.
در اینجا اگر دقت کرده باشید از UniqueID بجای Id استفاده شده‌است؛ از این جهت که مجموعه Form، حاوی Idهای واقعی دریافت شده از کاربر مانند ctl00$MainContent$MessageText می‌باشد.

روش دوم، استفاده از خاصیت جدید ValidateRequestMode یک کنترل سمت سرور است و در اینجا نیز می‌توان صرفا اعتبارسنجی یک کنترل را بجای کل صفحه، غیرفعال کرد:

```
<asp:TextBox ID="txtASPNet" ValidateRequestMode="Disabled" runat="server" />
```

تنظیم خاصیت ذکر شده برای کنترل‌های سمت سرور ضروری است. از این جهت که در حین تشکیل ViewState از محتوای این کنترل‌ها نیز استفاده می‌شود. اینجا است که اگر ValidateRequestMode غیرفعال نشده باشد، باز همان خطای ورودی خطرناک را دریافت خواهید کرد.

البته برای فعال سازی اعتبارسنجی با تاخیر نیاز است اصلاح ذیل را نیز به web.config برنامه اعمال کنید (مقدار پیش فرض آن 4 است):

```
<httpRuntime targetFramework="4.5" requestValidationMode="4.5" />
```

عنوان: معرفی ASP.NET Identity
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۴ ۹:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Entity framework, MVC, Security, ASP.NET Identity

سیستم ASP.NET Membership بهمراه ASP.NET 2.0 در سال 2005 معرفی شد، و از آن زمان تا بحال تغییرات زیادی در چگونگی مدیریت احزار هویت و اختیارات کاربران توسط اپلیکیشن‌های وب بوجود آمده است. ASP.NET Identity نگاهی تازه است به آنچه که سیستم Membership هنگام تولید اپلیکیشن‌های مدرن برای وب، موبایل و تبلت باید باشد.

پیش زمینه: سیستم عضویت در ASP.NET

ASP.NET Membership

ASP.NET Membership طراحی شده بود تا نیازهای سیستم عضویت وب سایت‌ها را تامین کند، نیازهایی که در سال 2005 رایج بود و شامل مواردی مانند مدل احراز هویت فرم، و یک پایگاه داده SQL Server برای ذخیره اطلاعات کاربران و پروفایل هایشان می‌شد. امروزه گزینه‌های بسیار بیشتری برای ذخیره داده‌های وب اپلیکیشن‌ها وجود دارد، و اکثر توسعه دهندگان می‌خواهند از اطلاعات شبکه‌های اجتماعی نیز برای احراز هویت و تعیین سطوح دسترسی کاربرانشان استفاده کنند. محدودیت‌های طراحی سیستم ASP.NET Membership گذر از این تحول را دشوار می‌کند:

الگوی پایگاه داده آن برای SQL Server طراحی شده است، و قادر به تغییرش هم نیستید. می‌توانید اطلاعات پروفایل را اضافه کنید، اما تمام داده‌ها در یک جدول دیگر ذخیره می‌شوند، که دسترسی به آنها نیز مشکل‌تر است، تنها راه دسترسی Profile Provider API خواهد بود.

سیستم تامین کننده (Provider System) امکان تغییر منبع داده‌ها را به شما می‌دهد، مثلا می‌توانید از بانک‌های اطلاعاتی MySQL یا Oracle استفاده کنید. اما تمام سیستم بر اساس پیش فرض هایی طراحی شده است که تنها برای بانک‌های اطلاعاتی relational درست هستند. می‌توانید تامین کننده (Provider) ای بنویسید که داده‌های سیستم عضویت را در منبعی به غیر از دیتابیس‌های relational ذخیره می‌کند؛ مثلا Windows Azure Storage Tables. اما در این صورت باید مقادیر زیادی کد بنویسید. مقادیر زیادی هم System.NotImplementedException باید بنویسید، برای متد هایی که به دیتابیس‌های NoSQL مربوط نیستند.

از آنجایی که سیستم ورود/خروج سایت بر اساس مدل Forms Authentication کار می‌کند، سیستم عضویت نمی‌تواند از OWIN استفاده کند. OWIN شامل کامپوننت هایی برای احراز هویت است که شامل سرویس‌های خارجی هم می‌شود (مانند Microsoft Accounts, Facebook, Google, Twitter). همچنین امکان ورود به سیستم توسط حساب‌های کاربری سازمانی (Organizational Accounts) نیز وجود دارد مانند Active Directory و Windows Azure Active Directory. این کتابخانه از JWT، OAuth 2.0 و CORS نیز پشتیبانی می‌کند.

ASP.NET Simple Membership

ASP.NET simple membership به عنوان یک سیستم عضویت، برای فریم ورک Web Pages توسعه داده شد. این سیستم با WebMatrix و Visual Studio 2010 SP1 انتشار یافت. هدف از توسعه این سیستم، آسان کردن پروسه افزودن سیستم عضویت به یک اپلیکیشن Web Pages بود. این سیستم پروسه کلی کار را آسان‌تر کرد، اما هنوز مشکلات ASP.NET Membership را نیز داشت. محدودیت هایی نیز وجود دارند:

ذخیره داده‌های سیستم عضویت در بانک‌های اطلاعاتی non-relational مشکل است.

نمی توانید از آن در کنار OWIN استفاده کنید.

با فراهم کننده‌های موجود ASP.NET Membership بخوبی کار نمی‌کند. توسعه پذیر هم نیست.

ASP.NET Universal Providers

ASP.NET Universal Providers برای ذخیره سازی اطلاعات سیستم عضویت در Windows Azure SQL Database توسعه پیدا کردند. با SQL Server Compact هم بخوبی کار می‌کنند. این تامین کننده‌ها بر اساس Entity Framework Code First ساخته شده بودند و بدین معنا بود که داده‌های سیستم عضویت را می‌توان در هر منبع داده ای که توسط EF پشتیبانی می‌شود ذخیره کرد. با انتشار این تامین کننده‌ها الگوی دیتابیس سیستم عضویت نیز بسیار سبک‌تر و بهتر شد. اما این سیستم بر پایه زیر ساخت ASP.NET

Membership نوشته شده است، بنابراین محدودیت‌های پیشین مانند محدودیت‌های SqlMembershipProvider هنوز وجود دارند. به بیان دیگر، این سیستم‌ها همچنان برای بانک‌های اطلاعاتی relational طراحی شده اند، پس سفارشی سازی اطلاعات کاربران و پروفایل‌ها هنوز مشکل است. در آخر آنکه این تامین کننده‌ها هنوز از مدل احراز هویت فرم استفاده می‌کنند.

ASP.NET Identity همانطور که داستان سیستم عضویت ASP.NET طی سالیان تغییر و رشد کرده است، تیم ASP.NET نیز آموخته‌های زیادی از بازخوردهای مشتریان شان بدست آورده اند. این پیش فرض که کاربران شما توسط یک نام کاربری و کلمه عبور که در اپلیکیشن خودتان هم ثبت شده است به سایت وارد خواهند شد، دیگر معتبر نیست. دنیای وب اجتماعی شده است. کاربران از طریق وب سایت‌ها و شبکه‌های اجتماعی متعددی با یکدیگر در تماس هستند، خیلی از اوقت بصورت زنده! شبکه هایی مانند Facebook و Twitter. همانطور که توسعه نرم افزارهای تحت وب رشد کرده است، الگوها و مدل‌های پیاده سازی نیز تغییر و رشد کرده اند. امکان Unit Testing روی کد اپلیکیشن‌ها، یکی از مهم‌ترین دلواپسی‌های توسعه دهندگان شده است. در سال 2008 تیم ASP.NET فریم ورک جدیدی را بر اساس الگوی Model-View-Controller (MVC) اضافه کردند. هدف آن کمک به توسعه دهندگان، برای تولید برنامه‌های ASP.NET با قابلیت Unit Testing بهتر بود. توسعه دهندگانی که می‌خواستند کد اپلیکیشن‌های خود را Unit Test کنند، همین امکان را برای سیستم عضویت نیز می‌خواستند.

با در نظر گرفتن تغییراتی که در توسعه اپلیکیشن‌های وب بوجود آمده ASP.NET Identity با اهداف زیر متولد شد:

یک سیستم هویت واحد (One ASP.NET Identity system)

سیستم ASP.NET Identity می‌تواند در تمام فریم ورک‌های مشتق از ASP.NET استفاده شود. مانند ASP.NET MVC, Web Forms, Web Pages, Web API و SignalR

از این سیستم می‌توانید در تولید اپلیکیشن‌های وب، موبایل، استور (Store) و یا اپلیکیشن‌های ترکیبی استفاده کنید.

سادگی تزریق داده‌های پروفایل درباره کاربران

روی الگوی دیتابیس برای اطلاعات کاربران و پروفایل‌ها کنترل کامل دارید. مثلا می‌توانید به سادگی یک فیلد، برای تاریخ تولد در نظر بگیرید که کاربران هنگام ثبت نام در سایت باید آن را وارد کنند.

کنترل ذخیره سازی/واکشی اطلاعات

بصورت پیش فرض ASP.NET Identity تمام اطلاعات کاربران را در یک دیتابیس ذخیره می‌کند. تمام مکانیزم‌های دسترسی به داده‌ها توسط EF Code First کار می‌کنند.

از آنجا که روی الگوی دیتابیس، کنترل کامل دارید، تغییر نام جداول و یا نوع داده فیلدهای کلیدی و غیره ساده است.

استفاده از مکانیزم‌های دیگر برای مدیریت داده‌های آن ساده است، مانند SharePoint, Windows Azure Storage Table و دیتابیس‌های NoSQL.

تست پذیری

ASP.NET Identity تست پذیری اپلیکیشن وب شما را بیشتر می‌کند. می‌توانید برای تمام قسمت هایی که از ASP.NET Identity استفاده می‌کنند تست بنویسید.

تامین کننده نقش (Role Provider)

تامین کننده ای وجود دارد که به شما امکان محدود کردن سطوح دسترسی بر اساس نقوش را می‌دهد. بسادگی می‌توانید نقش‌های جدید مانند "Admin" بسازید و بخش‌های مختلف اپلیکیشن خود را محدود کنید.

Claims Based

ASP.NET Identity از امکان احراز هویت بر اساس Claims نیز پشتیبانی می‌کند. در این مدل، هویت کاربر بر اساس دسته ای از اختیارات او شناسایی می‌شود. با استفاده از این روش توسعه دهندگان برای تعریف هویت کاربران، آزادی عمل بیشتری نسبت به مدل Roles دارند. مدل نقش‌ها تنها یک مقدار منطقی (bool) است؛ یا عضو یک نقش هستید یا خیر، در حالیکه با استفاده از روش Claims می‌توانید اطلاعات بسیار ریز و دقیقی از هویت کاربر در دست داشته باشید.

تامین کنندگان اجتماعی

به راحتی می‌توانید از تامین کنندگان دیگری مانند Microsoft, Facebook, Twitter, Google و غیره استفاده کنید و اطلاعات مربوط به کاربران را در اپلیکیشن خود ذخیره کنید.

Windows Azure Active Directory

برای اطلاعات بیشتر به این لینک مراجعه کنید.

یکپارچگی با OWIN

ASP.NET Identity بر اساس OWIN توسعه پیدا کرده است، بنابراین از هر میزبانی که از OWIN پشتیبانی می‌کند می‌توانید استفاده کنید. همچنین هیچ وابستگی ای به System.Web وجود ندارد. ASP.NET Identity یک فریم ورک کامل و مستقل برای OWIN است و می‌تواند در هر اپلیکیشنی که روی OWIN میزبانی شده استفاده شود.

ASP.NET Identity از OWIN برای ورود/خروج کاربران در سایت استفاده می‌کند. این بدین معنا است که بجای استفاده از Forms Authentication برای تولید یک کوکی، از OWIN CookieAuthentication استفاده می‌شود.

پکیج NuGet

ASP.NET Identity در قالب یک بسته NuGet توزیع می‌شود. این بسته در قالب پروژه‌های ASP.NET MVC, Web Forms و Web API که با Visual Studio 2013 منتشر شدند گنجانده شده است.

توزیع این فریم ورک در قالب یک بسته NuGet این امکان را به تیم ASP.NET می‌دهد تا امکانات جدیدی توسعه دهند، باگ‌ها را برطرف کنند و نتیجه را بصورت چابک به توسعه دهندگان عرضه کنند.

شروع کار با ASP.NET Identity

ASP.NET Identity در قالب پروژه‌های ASP.NET MVC, Web Forms, Web API و SPA که بهمراه Visual Studio 2013 منتشر شده اند استفاده می‌شود. در ادامه به اختصار خواهیم دید که چگونه ASP.NET Identity کار می‌کند.

یک پروژه جدید ASP.NET MVC با تنظیمات Individual User Accounts بسازید.

پروژه ایجاد شده شامل سه بسته می‌شود که مربوط به ASP.NET Identity هستند:

Microsoft.AspNet.Identity.EntityFramework این بسته شامل پیاده سازی ASP.NET Identity با Entity Framework می‌شود، که تمام داده‌های مربوطه را در یک دیتابیس SQL Server ذخیره می‌کند.
Microsoft.AspNet.Identity.Core این بسته محتوی تمام interfaceهای ASP.NET Identity است. با استفاده از این بسته می‌توانید پیاده سازی دیگری از ASP.NET Identity بسازید که منبع داده متفاوتی را هدف قرار می‌دهد. مثلا Windows Azure Storage Table و دیتابیس‌های NoSQL.

Microsoft.AspNet.Identity.OWIN این بسته امکان استفاده از احراز هویت OWIN را در اپلیکیشن‌های ASP.NET فراهم می‌کند. هنگام تولید کوکی‌ها از OWIN Cookie Authentication استفاده خواهد شد.

اپلیکیشن را اجرا کرده و روی لینک Register کلیک کنید تا یک حساب کاربری جدید ایجاد کنید.

Register - My ASP.NET Ap
localhost:1234/Account/Register
Application name
Register.
Create a new account.
User name
Password
Confirm password
Register
© 2013 - My ASP.NET Application

هنگامیکه بر روی دکمه‌ی Register کلیک شود، کنترلر Account، اکشن متد Register را فراخوانی می‌کند تا حساب کاربری جدیدی با استفاده از ASP.NET Identity API ساخته شود.

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser() { UserName = model.UserName };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            await SignInAsync(user, isPersistent: false);
            return RedirectToAction("Index", "Home");
        }
        else
        {
            AddErrors(result);
        }
    }

    // If we got this far, something failed, redisplay form
    return View(model);
}
```

اگر حساب کاربری با موفقیت ایجاد شود، کاربر توسط فراخوانی متد SignInAsync به سایت وارد می‌شود.

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser() { UserName = model.UserName };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            await SignInAsync(user, isPersistent: false);
            return RedirectToAction("Index", "Home");
        }
        else
        {
            AddErrors(result);
        }
    }

    // If we got this far, something failed, redisplay form
    return View(model);
}
```

```
private async Task SignInAsync(ApplicationUser user, bool isPersistent)
{
    AuthenticationManager.SignOut(DefaultAuthenticationTypes.ExternalCookie);

    var identity = await UserManager.CreateIdentityAsync(
       user, DefaultAuthenticationTypes.ApplicationCookie);

    AuthenticationManager.SignIn(
       new AuthenticationProperties() {
          IsPersistent = isPersistent
       }, identity);
}
```

از آنجا که ASP.NET Identity و OWIN Cookie Authentication هر دو Claims-based هستند، فریم ورک، انتظار آبجکتی از نوع ClaimsIdentity را خواهد داشت. این آبجکت تمامی اطلاعات لازم برای تشخیص هویت کاربر را در بر دارد. مثلا اینکه کاربر

مورد نظر به چه نقش هایی تعلق دارد؟ و اطلاعاتی از این قبیل. در این مرحله می‌توانید Claimهای بیشتری را به کاربر بیافزایید.

کلیک کردن روی لینک Log off در سایت، اکشن متد LogOff در کنترلر Account را اجرا می‌کند.

```
// POST: /Account/LogOff
[HttpPost]
[ValidateAntiForgeryToken]
public ActionResult LogOff()
{
    AuthenticationManager.SignOut();
    return RedirectToAction("Index", "Home");
}
```

همانطور که مشاهده می‌کنید برای ورود/خروج کاربران از AuthenticationManager استفاده می‌شود که متعلق به OWIN است. متد SignOut همتای متد FormsAuthentication.SignOut است.

کامپوننت‌های ASP.NET Identity

تصویر زیر اجزای تشکیل دهنده ASP.NET Identity را نمایش می‌دهد. بسته هایی که با رنگ سبز نشان داده شده اند سیستم کلی ASP.NET Identity را می‌سازند. مابقی بسته‌ها وابستگی هایی هستند که برای استفاده از ASP.NET Identity در اپلیکیشن‌های ASP.NET لازم اند.

دو پکیج دیگر نیز وجود دارند که به آنها اشاره نشد:

Microsoft.Security.Owin.Cookies این بسته امکان استفاده از مدل احراز هویت مبتنی بر کوکی (Cookie-based Authentication) را فراهم می‌کند. مدلی مانند سیستم ASP.NET Forms Authentication.

EntityFramework که نیازی به معرفی ندارد.

مهاجرت از Membership به ASP.NET Identity

تیم ASP.NET و مایکروسافت هنوز راهنمایی رسمی، برای این مقوله ارائه نکرده اند. گرچه پست‌های وبلاگ‌ها و منابع مختلفی وجود دارند که از جنبه‌های مختلفی به این مقوله پرداخته اند. امیدواریم تا در آینده نزدیک مایکروسافت راهنمایی‌های لازم را منتشر کند، ممکن است ابزار و افزونه هایی نیز توسعه پیدا کنند. اما در یک نگاه کلی می‌توان گفت مهاجرت بین این دو فریم ورک زیاد ساده نیست. تفاوت‌های فنی و ساختاری زیادی وجود دارند، مثلا الگوی دیتابیس‌ها برای ذخیره اطلاعات کاربران، مبتنی بودن بر فریم

ورک OWIN و غیره. اگر قصد اجرای پروژه جدیدی را دارید پیشنهاد می‌کنم از فریم ورک جدید مایکروسافت ASP.NET Identity استفاده کنید.

قدم‌های بعدی Create an ASP.NET MVC 5 App with Facebook and Google OAuth2 and OpenID Sign-on

در این مقاله خواهید دید چگونه اطلاعات پروفایل را اضافه کنید و چطور از ASP.NET Identity برای احراز هویت کاربران توسط Facebook و Google استفاده کنید. Deploy a Secure ASP.NET MVC app with Membership, OAuth, and SQL Database to a Windows Azure Web Site

Individual User Accounts

Organizational Accounts

Customizing profile information in ASP.NET Identity in VS 2013 templates

Get more information from Social providers used in the VS 2013 project templates

https://github.com/rustd/AspnetIdentitySample

پروژه نمونه ASP.NET Identity می‌تواند مفید باشد. در این پروژه نحوه کارکردن با کاربران و نقش‌ها و همچنین نیازهای مدیریتی رایج نمایش داده شده.

نظرات خوانندگان

نویسنده: ناظم
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۱:۴۶

سلام دوست عزیز
به خاطر این مطلب بسیار خوبتون سپاسگذارم.
آیا کتاب یا راهنمای جامعی برای کسی مثل من که اصلا تجربه کار با هیچ کدام از این سیستم‌های مدیریت کاربر رو نداره سراغ دارین؟ لینکهای بالا همشون موردی هستن

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۱۴ ۱۲:۳۰

مباحث ابتدایی Forms Authentication مربوط است به ASP.NET 1.x؛ دوره‌ای در این مورد . همچنین در MVC هم قابل استفاده است (^ و ^). مباحث membership هم مربوط است به ASP.NET 2.x. کتابی در این مورد .

نویسنده: مهران
تاریخ: ۱۳۹۲/۱۰/۱۴ ۲۱:۳۹

سلام
با تشکر از مطالب و ارائه خوبتون؛
خواستم بدونم چطور میشه، کاربر را بر اساس عملیات کنترلر (Actions) تعیین دسترسی کرد؟

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۴ ۲۱:۴۴

تا بحال با کتاب یا دوره جامعی درباره ASP.NET Identity مواجه نشدم، اگر منبع مناسبی پیدا کنم به اشتراک میذارم. در چند پست آتی بیشتر درباره این فریم ورک صحبت خواهم کرد و مثال هایی عملی نیز در نظر خواهم گرفت

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۴ ۲۳:۲۷

سوالتون رو دقیقا متوجه نشدم. از خاصیت Authorize میتونید استفاده کنید، که قابل اعمال بر روی تک تک اکشن متدها و یا کل کنترلر است. خاصیت AllowAnonymous برای دسترسی عمومی استفاده می‌شود. برای اطلاعات بیشتر درباره نحوه استفاده از ASP.NET Identity و ساختار کلی OWIN لطفا به لینک‌های ضمیمه شده مراجعه کنید.

```
[Authorize]
public controller account
{
    public ActionResult Index()  { }
    public ActionResult Manage() { }

    [AllowAnonymous]
    public ActionResult Info() { }
}

[Authorize(Roles="Admin")]
public controller admin
{
    public ActionResult Index() { }
}
```

نویسنده: مهران
تاریخ: ۱۳۹۲/۱۰/۱۵ ۰:۱

فکر میکنم با مطالعه بیشتر Claims Based که در مطلب اشاره کردید، به راه حل مورد نظر برسم.

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۵ ۲:۱۶

درسته، درباره Claims-based authorization هم یک یا دو پست می‌نویسم.

نویسنده: وحید
تاریخ: ۱۳۹۲/۱۰/۲۰ ۲:۱۸

با تشکر از مطلب مفیدتون سوالی داشتم
یک جا گفتید " مثلا الگوی دیتابیس‌ها برای ذخیره اطلاعات کاربران " منظورتون از دیتابیس Table هست؟
چرا از کد زیر task برای خروجی استفاده کردید

```
async Task<ActionResult> Register
```

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۱ ۶:۴۹

منظور دیتابیس سیستم عضویت است، همانطور که گفته شد این دیتابیس توسط EF ساخته می‌شود، بنابراین جداول، فیلدها و دیگر موارد را میتوانید سفارشی کنید.

همانطور که از امضای این متد مشخص است، عملیات بصورت Async پردازش می‌شوند. برای اطلاعات بیشتر به این لینک نمونه مراجعه کنید.

نویسنده: Programmer
تاریخ: ۱۳۹۲/۱۰/۲۳ ۱۷:۱۸

با عرض سلام و تشکر بابت پست‌های مفید که یقیناً خیلی براش زحمت می‌کشید.

اگر ممکنه کمی در مورد ساختار Identity و کلاس‌ها و اینترفیس‌ها و نحوه کار با اونها بصورتی که بتونیم خودمون هم Custom Implement اش رو انجام بدیم توضیح بدید.

اینکه اینترفیس هایی چون IUser و IUserStore و IUserPasswordStore و IUserSecurityStampStore و در طرف دیگه UserManager و UserStore و IdentityUser چی هستند و با چه هدفی تعریف شدند و قراره چیکار کنند و در آخر برای اینکه مطابق آموزش‌هایی که در سایت هست در مورد MVC و تعریف لایه‌های مختلف و سرویس‌های مورد نیاز، چطور باید از Identity استفاده کرد که هماهنگ با اون مطالب باشه؟

ممنون

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۳ ۲۰:۴

در پست‌های آتی پیاده سازی یک تامین کننده (Provider) برای MySQL رو بررسی می‌کنم. برای اطلاعات بیشتر به Implementing a Custom MySQL ASP.NET Identity Storage Provider مراجعه کنید.

هنوز مستندات کامل و رسمی برای این فریم ورک عرضه نشده اما مطالب مفید زیادی در اینترنت وجود دارند. چند لینک در زیر لیست شده:

ASP.NET Core Identity

The good, the bad and the ugly of ASP.NET Identity

درباره تطابق با آموزش‌های سایت: دقیقا متوجه نشدم منظورتون کدوم الگوها است، اما چند نکته تکمیلی:
بصورت پیش فرض وقتی ASP.NET Identity به پروژه اضافه میشه کلاسی بنام ApplicationDbContext ایجاد میشه که بعنوان DbContext پیش فرض برای دیتابیس عضویت استفاده خواهد شد. اگر موجودیت جدیدی برای پروفایل کاربران بسازید باید بعنوان یک DbSet<T> به این کلاس افزوده بشه. اگر نیازی به تفکیک دیتابیس سیستم عضویت و دیتابیس اپلیکیشن نیست، بهتره از یک DbContext استفاده کنید. لایه بندی مدل ها، سرویس‌ها و کد دسترسی به داده‌ها هم ساده است چرا که کل سیستم توسط EF Code First مدیریت میشه. بنابراین استفاده از الگوهای رایج مانند تزریق وابستگی‌ها و غیره مشابه دیگر سناریوهای EF Code First است.

نویسنده: کامران سادین
تاریخ: ۱۳۹۲/۱۰/۲۹ ۱۷:۱

با سلام.
بنده می‌خواستم بدونم آیا در .NET ورژن 4 هم میتوانیم استفاده کنیم؟
با ویژوال استادیو 2012

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۹ ۱۸:۴۰

با نصب پکیج‌های مربوط به ASP.NET Identity و غیرفعال کردن Forms Auth می‌تونید همچین کاری بکنید اما توصیه نمیشه.
سیستم Identity اکثر عملیاتش رو بصورت Async انجام میده که نیاز به NET 4.5. داره. دلایل دیگه ای هم وجود داره که اگر یک جستجوی ساده در اینترنت بکنید مطالب خوبی در این باره پیدا می‌کنید، مثلا لینک زیر:

http://stackoverflow.com/questions/19237285/using-asp-net-identity-in-mvc-4

نویسنده: سوران
تاریخ: ۱۳۹۲/۱۲/۰۱ ۱۲:۳۷

با تشکر از مطالب آموزنده شما ،
من در ارتباط با ارث بری از کلاس IdentityUser یک سوال داشتم. با توجه به نمونه کدهای تمپلیت vs2013 یک کلاس ApplicationUser از کلاس IdentityUser ارث بری می‌کنه و DbContext هم مربوط به این کلاس میشه، یعنی به صورت زیر

```
ApplicationDbContext : IdentityDbContext<ApplicationUser>
```

حال سوال من اینه که اگه چند کلاس داشته باشیم که بخوایم از IdentityUser ارث بری داشته باشند، چطور میشه اونها را در یک DbContext استفاده کرد؟
اگر مقاله یا مثالی در این مورد معرفی کنید ممنون میشم .
با تشکر

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۲/۰۳ ۵:۲۴

کلاس کاربر:

```
public class AppUser : IdentityUser
{
    public string Email { get; set; }
    public string ConfirmationToken { get; set; }
    public bool IsConfirmed { get; set; }

    public virtual UserProfile Profile { get; set; }
}
```

کلاس پروفایل کاربر:

```
public class UserProfile
{
    public int Id { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }

    public DateTime? Birthday { get; set; }

    public byte[] Avatar { get; set; }
}
```

کلاس کانتکست دیتابیس:

```
public class SampleDbContext : IdentityDbContext
{
    public SampleDbContext() : base("DefaultConnection") { }

    static SampleDbContext()
    {
        Database.SetInitializer(new DropCreateDatabaseIfModelChanges<SampleDbContext>());
    }

    public DbSet<UserProfile> UserProfiles { get; set; }
    public DbSet<Customer> Customers { get; set; }
    public DbSet<Product> Products { get; set; }
    ...
}
```

این کلاس‌ها می‌تونن تو لایه دیگری مثل Domain Models تعریف بشن.

نویسنده: رضا گرمارودی
تاریخ: ۱۳۹۲/۱۲/۰۳ ۹:۲۱

سلام؛ Identity در کار با SQLCE مشکل داره! هنگام چک کردن نام کاربری و کلمه عبور پیغامی مبنی بر استفاده از تابع Lowercase میدهد که گویا در SqlCe به شیوه دیگری باید صدا زده شود !

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۳ ۹:۳۶

این پروژه سورس باز هست. مشکلات آن‌را برای رفع در اینجا گزارش کنید. نحوه‌ی گزارش مشکل هم باید کمی فنی باشد. حداقل جزئیات exception و stack trace آن باید گزارش شوند.

نویسنده: احمد
تاریخ: ۱۳۹۳/۰۲/۲۳ ۱۰:۲۸

سلام
آیا در پروژه‌های windows application که از wcf استفاده میکنند هم میتوانیم از این سیستم استفاده کنیم.؟

نویسنده: داریوش حمیدی
تاریخ: ۱۳۹۳/۰۷/۱۸ ۲۰:۴

سلام ; در کلاس IdentityUser این خصوصیت دو به چی اشاره دارند ؟

1-SecurityStamp

این مقدار Random تغیر می‌کنید وقتی که اطلاعات کاربری تغیر پیدا می‌کند مثل تغیر رمز عبور ..

2-TwoFactorEnabled

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۳:۴۶

« اعمال تزریق وابستگی‌ها به مثال رسمی ASP.NET Identity »

عنوان: **متدهای احراز هویت در VS 2013**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۵ ۲:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Authentication, ASP.NET Identity, Security

ویژوال استودیو 2013 چندین گزینه برای احراز هویت در قالب‌های پیش فرض پروژه‌های ASP.NET Web Forms, MVC, Web API ارائه می‌کند:

No Authentication

Individual User Accounts

Organizational Accounts

Windows Authentication

No Authentication

اگر گزینه **No Authentication** را انتخاب کنید، پروژه ایجاد شده صفحاتی را برای ورود به سایت نخواهد ساخت. همچنین رابط کاربری ای برای نمایش کاربر فعلی، کلاس‌های موجودیت‌ها برای یک دیتابیس عضویت و رشته‌های اتصالی نیز وجود نخواهند داشت.

Individual User Accounts

اگر گزینه‌ی **Individual User Accounts** را انتخاب کنید، اپلیکیشن شما برای استفاده از ASP.NET Identity (که پیش از این با نام ASP.NET Membership شناخته می‌شد) پیکربندی می‌شود. ASP.NET Identity کاربران را قادر می‌سازد تا با ساختن حساب کاربری جدیدی در سایت و یا با استفاده از تامین کننده‌های ثالثی مانند Facebook, Google و غیره به سایت وارد شوند. این فریم ورک برای ذخیره‌ی داده‌های پروفایل کاربران، بصورت پیش فرض از یک دیتابیس SQL Server LocalDB استفاده می‌کند که می‌توانید بعدا آنرا بر روی SQL Server یا Windows Azure SQL Database نیز منتشر کنید.

این قابلیت‌ها در Visual Studio 2013 در نسخه قبلی نیز وجود داشتند، اما کد سیستم عضویت آن مجددا بازنویسی شده‌است. این بازنویسی دارای مزایای زیر است:

سیستم عضویت جدید بجای استفاده از ماژول ASP.NET Forms Authentication بر پایه OWIN نوشته شده است. این بدین معنا است که از یک مکانیزم احراز هویت واحد می‌توانید در اپلیکیشن‌های ASP.NET Web Forms, MVC, Web API و SignalR استفاده کنید. سیستم عضویت جدید توسط Entity Framework Code First مدیریت می‌شود و شامل تمامی کلاس‌هایی است که نماینده جداول و موجودیت‌ها هستند. این بدین معنا است که روی الگوی دیتابیس کنترل کامل دارید. سفارشی سازی و تغییر اطلاعات کاربران و پروفایل هایشان بسیار ساده‌تر است، تنها لازم است برای اعمال تغییرات از Code First Migrations استفاده کنید.

سیستم عضویت جدید بصورت خودکار در تمام قالب‌های پروژه پیش فرض، نصب و پیاده سازی می‌شود. این امکان برای تمام پروژه‌هایی که دات نت فریم ورک 4.5 را هدف قرار می‌دهند وجود دارد. ASP.NET Identity هنگام تولید وب سایت‌های اینترنتی که اکثر کاربرانشان خارجی (External) هستند گزینه خوبی است. اگر سازمان شما از Active Directory و یا Office 365 استفاده می‌کند و می‌خواهید پروژه‌تان قادر باشد تا احراز هویت کارمندان و شرکای تجاری تان را مدیریت کند، **Organizational Accounts** گزینه بهتری است.

برای اطلاعات بیشتر درباره‌ی Individual User Accounts به لینک‌های زیر مراجعه کنید:

asp.net/identity

Create an ASP.NET MVC 5 App with Facebook and Google OAuth2 and OpenID Sign-on

Web API - External Authentication Services

Adding External Logins to your ASP.NET application in Visual Studio 2013

Organizational Accounts

اگر گزینه **Organizational Accounts** را انتخاب کنید پروژه ایجاد شده برای استفاده از Windows Identity Foundation (WIF) پیکربندی خواهد شد. این فریم ورک برای احراز هویت کاربران از Windows Azure Active Directory (WAAD) استفاده می‌کند که شامل Office 365 نیز می‌شود.

Windows Authentication

اگر گزینه‌ی **Windows Authentication** را انتخاب کنید، پروژه ایجاد شده برای استفاده از Windows Authentication IIS Module پیکربندی خواهد شد. چنین اپلیکیشنی نام دامنه و نام کاربری را نمایش خواهد که یا از Active Directory می‌آید، یا از یک ماشین محلی (local machine). اما رابط کاربری ای برای ورود به سیستم وجود ندارد؛ چرا که اینگونه اپلیکیشن‌ها برای سایت‌های اینترانتی (Intranet) استفاده خواهند شد.

یک راه دیگر انتخاب گزینه On-Premises زیر شاخه Organizational Accounts است. این گزینه بجای استفاده از ماژول Windows Authentication از فریم ورک Windows Identity Foundation برای احراز هویت استفاده می‌کند. انجام چند مرحله دستی برای پیکربندی این گزینه لازم است، اما WIF امکاناتی را عرضه می‌کند که در ماژول احراز هویت ویندوز وجود ندارند. برای مثال، هنگام استفاده از WIF می‌توانید تنظیمات لازم را در Active Directory انجام دهید تا قادر به واکشی اطلاعات پوشه‌ها باشید (directory data querying).

گزینه‌های احراز هویت Organizational Accounts

دیالوگ **Configure Authentication** گزینه‌های متعددی برای احراز هویت توسط (Windows Azure Active Directory including Office 365) و Windows Server Active Directory در اختیارتان می‌گذارد:

Cloud - Single Organization

Cloud - Multi Organization

On-Premises

اگر می‌خواهید یکی از گزینه‌های WAAD را امتحان کنید اما حساب کاربری ای ندارید، روی این لینک کلیک کنید تا ثبت نام کنید.

نکته: اگر یکی از گزینه‌های WAAD را انتخاب کنید، باید اطلاعات هویتی (Credentials) یک مدیر کل را وارد کنید. برای نسخه نهایی Visual Studio 2013 برنامه هایی وجود دارد تا دیگر نیازی نباشد چنین مراحلی را تکمیل کنید. در این صورت ویژوال استودیو تنظیماتی را نمایش خواهد داد که یک مدیر می‌تواند بعدا از آنها استفاده کند تا اپلیکیشن را بصورت دستی در WAAD پیکربندی کند.

Cloud - Single Organization Authentication

از این گزینه برای احراز هویت کاربرانی استفاده کنید که در قالب یک OWIN Tenant تعریف می‌شوند. برای مثال سایتی با نام Company.com داریم که برای کارمندان این سازمان از طریق company.onmicrosoft.com قابل دسترسی خواهد بود. نمی‌توانید WAAD را طوری پیکربندی کنید که کاربران tenantهای دیگر نیز به اپلیکیشن شما دسترسی داشته باشند.

Domain

نام دامنه‌ای در WAAD که می‌خواهید اپلیکیشن را برای آن پیکربندی کنید، مثلا company.onmicrosoft.com. اگر از custom domain

ها استفاده می‌کنید مانند company.com بجای company.onmicrosoft.com می‌توانید این اطلاعات را اینجا وارد کنید.

سطح دسترسی

اگر اپلیکیشن نیاز به کوئری گرفتن یا بروز رسانی اطلاعات پوشه‌ها (directory information) را توسط Graph API دارد، از گزینه‌های **Single Sign-On, Read Directory Data** و یا **Single Sign-On, Read and Write Directory Data** استفاده کنید. در غیر اینصورت گزینه **Single Sign-On** را رها کنید. برای اطلاعات بیشتر به Application Access Levels و Using the Graph API to Query Windows Azure AD مراجعه کنید.

Application ID URI

بصورت پیش فرض، قالب پروژه یک شناسه application ID URI برای شما تولید می‌کند، که این کار با الحاق نام پروژه شما به نام دامنه WAAD صورت می‌گیرد. برای مثال، اگر نام پروژه Example باشد و نام دامنه contoso.onmicrosoft.com، شناسه خروجی https://contoso.onmicrosoft.com/Example می‌شود. اگر می‌خواهید بصورت دستی این فیلد را مقدار دهی کنید، گزینه **More Options** را انتخاب کنید. این شناسه باید با https:// شروع شود.

بصورت پیش فرض، اگر اپلیکیشنی که در WAAD تهیه و تدارک دیده شده است، شناسه‌ای یکسان با شناسه موجود در پروژه Visual Studio داشته باشد، پروژه شما به اپلیکیشن موجود در WAAD متصل خواهد شد. اگر می‌خواهید تدارکات جدیدی ببینید تیک گزینه **Overwrite the application entry if one with the same ID already exists** را حذف کنید.

اگر تیک این گزینه حذف شده باشد، و ویژوال استودیو اپلیکیشنی با شناسه‌ای یکسان را پیدا کند، عددی به آخر URI اضافه خواهد شد. مثلا فرض کنید نام پروژه Example است و اپلیکیشنی نیز با شناسه https://contoso.onmicrosoft.com/Example در WAAD وجود دارد. در این صورت اپلیکیشن جدیدی با شناسه ای مانند https://contoso.onmicrosoft.com/Example_ 20130619330903 ایجاد می‌شود.

تهیه و تدارک اپلیکیشن در WAAD

برای آنکه یک اپلیکیشن WAAD ایجاد کنید و یا پروژه را به یک اپلیکیشن موجود متصل کنید، ویژوال استودیو به اطلاعات ورود یک مدیر کل برای دامنه مورد نظر، نیاز دارد. هنگامی که در دیالوگ **Configure Authentication** روی **OK** کلیک می‌کنید، اطلاعات ورود یک مدیر کل از شما درخواست می‌شود و نهایتا هنگامیکه روی **Create Project** کلیک می‌کنید، ویژوال استودیو اپلیکیشن شما را در WAAD پیکربندی می‌کند.

برای اطلاعات بیشتر درباره نحوه استفاده از مدل احراز هویت **Cloud - Single Organization** به لینک‌های زیر مراجعه فرمایید:
Windows Azure Authentication

Adding Sign-On to Your Web Application Using Windows Azure AD

Developing ASP.NET Apps with Windows Azure Active Directory

مقالات مذکور برای ویژوال استودیو 2013 بروز رسانی نشده اند. برخی از مراحلی که در این مقالات بصورت دستی باید انجام شوند در Visual Studio 2013 مکانیزه شده است.

Cloud - Multi Organization Authentication

از این گزینه برای احراز هویت کاربرانی استفاده کنید که در WAADtenantهای متعددی تعریف شده‌اند. برای مثال، نام سایت contoso.com است و برای کارمندان دو سازمان از طریق آدرس‌های contoso.onmicrosoft.com و fabrikam.onmicrosoft.com قابل دسترسی خواهد بود. نحوه پیکربندی این مدل نیز مانند قسمت قبلی است.

برای اطلاعات بیشتر درباره احراز هویت **Cloud - Multi Organization** به لینک‌های زیر مراجعه کنید:

Easy Web App Integration with Windows Azure Active Directory, ASP.NET & Visual Studio

Developing Multi-Tenant Web Applications with Windows Azure AD

On-Premises Organizational Accounts

این گزینه را هنگامی انتخاب کنید که کاربران در Windows Server Active Directory (AD) تعریف شده اند و نمی‌خواهید از WAAD استفاده کنید. از این مدل برای ایجاد وب سایت‌های اینترنت و اینترانت می‌توانید استفاده کنید. برای یک وب سایت اینترنتی از Active Directory Federation Services (ADFS) استفاده کنید.

برای یک وب سایت اینترانتی، می‌توانید کلا این گزینه را رها کنید و از Windows Authentication استفاده کنید. در صورت استفاده از گزینه Windows Authentication لازم نیست تا آدرس سند متادیتا (metadata document URL) را فراهم کنید، همچنین توجه داشته باشید که Windows Authentication امکان کوئری گرفتن از پوشه‌ها و کنترل سطوح دسترسی در Active Directory را ندارد.

On-Premises Authority

آدرس سند متادیتا. این سند اطلاعاتی درباره مختصات Authority دارد که اپلیکیشن از آنها برای به پیش بردن روند احراز هویت و ورود به سایت استفاده می‌کند.

Application ID URI

یک شناسه منحصر به فرد که AD از آن برای شناسایی اپلیکیشن استفاده می‌کند. می‌توانید این فیلد را خالی رها کنید تا ویژوال استودیو بصورت خودکار اپلیکیشنی بدین منظور بسازد.

قدم‌های بعدی

در این مقاله با مدل‌های مختلف احراز هویت در اپلیکیشن‌های Visual Studio 2013 آشنا شدید و برخی تغییرات و امکانات جدید نیز بررسی شدند. برای اطلاعات تکمیلی به ASP.NET and Web Tools for Visual Studio 2013 Release Notes مراجعه کنید.

نظرات خوانندگان

نویسنده: هومن
تاریخ: ۱۳۹۳/۰۳/۱۹ ۱۹:۵۴

سلام
- بابت مطلب خوبتون ممنون.من یه سوال داشتم ، در پورتال‌های سازمانی و در حالتی که نیاز نداریم پالیسی رو از اکتیو بخونیم ، بهتره از حالت آخر یعنی Windows Authentication استفاده کنیم درسته؟
من از این حالت استفاده کردم، و یک مشکل و یک سوال برام پیش اومده...
- مشکلم اینه که در سیستمی که با یوزر اکتیو بالا اومده خوب کار میکنه و یوزر رو تشخیص میده، اما روی لپتاپم چون لوکال هست و طبیعیه اکتیو دایرکتوری موجود نیست نمیتونه یوزر رو تشخیص بده(پنجره لاگین مرورگر را باز میکند) و من نمیتونم برناممو گسترش بدم و برای تست نیاز به اجرای برنامه دارم ، این مشکلو چجوری برطرف کنم؟
- سوالم اینه که ما هیچ کجا اسم دامین یا آی پی سرور رو به برنامه نمیدیم ، پس سیستم احراز هویت چجوری دامین رو تشخیص میده و باهاش کار میکنه ؟

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۳/۲۰ ۹:۱۸

اگر از IE استفاده کنید، مشکلی نباید باشه. چون IE با سیستم اعتبارسنجی مبتنی بر ویندوز یکپارچه هست. اگر با IE صفحه لاگین مرورگر باز میشه، به تنظیمات امنیتی اون مراجعه کنید و سایت رو در قسمت trusted sites اضافه کنید:
http://support.microsoft.com/kb/258063

سمت سرور هم باید در تنظیمات IIS، گزینه‌ی اعتبارسنجی مبتنی بر ویندوز فعال باشه:

http://www.asp.net/mvc/tutorials/older-versions/security/authenticating-users-with-windows-authentication-cs

عنوان: **توسعه اپلیکیشن های ASP.NET با Windows Azure Active Directory**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۵ ۵:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Visual Studio 2013, ASP.NET Identity, Windows Azure Active Directory, Security

ابزار ASP.NET برای Windows Azure Active Directory فعال کردن احراز هویت در وب اپلیکیشن هایی که روی Windows Azure Web Sites میزبانی شده اند را ساده‌تر می‌کند. می‌توانید از Windows Azure Authentication برای احراز هویت کاربران Office 365 استفاده کنید، حساب‌های کاربری را از On-Premise Active Directory خود همگام سازی (Sync) کنید و یا از یک دامنه سفارشی Windows Azure Active Directory بهره ببرید. فعال سازی Windows Azure Authentication، اپلیکیشن شما را طوری پیکربندی می‌کند تا تمامی کاربران را با استفاده از یک Windows Azure Active Directory tenant احراز هویت کند.
این مقاله ساختن یک اپلیکیشن ASP.NET را که بر اساس organizational accounts پیکربندی شده و بر روی Windows Azure Active Directory میزبانی می‌شود، مرور می‌کند.
پیش نیاز ها

Visual Studio Express 2013 RC for Web یا Visual Studio 2013 RC

یک حساب کاربری در Windows Azure. می‌توانید یک حساب رایگان بسازید .

یک مدیر کلی به حساب کاربری Active Directory خود اضافه کنید

وارد Windows Azure Portal شوید.
یک حساب کاربری Windows Azure Active Directory (AD) انتخاب یا ایجاد کنید. اگر قبلا حساب کاربری ساخته اید از همان استفاده کنید در غیر اینصورت یک حساب جدید ایجاد کنید. مشترکین Windows Azure یک AD پیش فرض با نام **Default Directory** خواهند داشت.
در حساب کاربری AD خود یک کاربر جدید در نقش Global Administrator بسازید. اکانت AD خود را انتخاب کنید و **Add User** را کلیک کنید. برای اطلاعات کامل‌تر به Managing Windows Azure AD from the Windows Azure Portal 1- Sign Up with an Organizational Account مراجعه کنید.

یک نام کاربری انتخاب کرده و به مرحله بعد بروید.

نام کاربری را وارد کنید و نقش **Global Administrator** را به آن اختصاص دهید. مدیران کلی به یک آدرس ایمیل متناوب هم نیاز دارند. به مرحله بعد بروید.

بر روی **Create** کلیک کنید و کلمه‌ی عبور موقتی را کپی کنید. پس از اولین ورود باید کلمه عبور را تغییر دهید.

یک اپلیکیشن ASP.NET بسازید

در ویژوال استودیو یک پروژه جدید ASP.NET Web Forms یا MVC بسازید و روی **Change Authentication** کلیک کنید.

گزینه **Organizational Accounts** را انتخاب کنید. نام دامنه خود را وارد کنید و سپس گزینه **Single Sign On, Read directory data** را انتخاب کنید. به مرحله بعد بروید.

نکته: در قسمت **More Options** می توانید قلمرو اپلیکیشن (Application ID URI) را تنظیم کنید. تنظیمات پیش فرض برای اکثر

کاربران مناسب است اما در صورت لزوم می‌توانید آنها را ویرایش کنید، مثلا از طریق Windows Azure Portal دامنه‌های سفارشی خودتان را تنظیم کنید.

اگر گزینه **Overwrite** را انتخاب کنید اپلیکیشن جدیدی در Windows Azure برای شما ساخته خواهد شد. در غیر اینصورت فریم ورک سعی می‌کند اپلیکیشنی با شناسه یکسان پیدا کند (در پست متدهای احراز هویت در VS 2013 به تنظیمات این قسمت پرداخته شده).

اطلاعات مدیر کلی دامنه در Active Directory خود را وارد کنید (Credentials) و پروژه را با کلیک کردن بر روی **Create Project** بسازید.

با کلیدهای ترکیبی Ctrl + F5، اپلیکیشن را اجرا کنید. مرورگر شما باید یک اخطار SSL Certificate به شما بدهد. این بدین دلیل است که مدرک استفاده شده توسط IIS Express مورد اعتماد (trusted) نیست. این اخطار را بپذیرید و اجازه اجرا را به آن بدهید. پس از آنکه اپلیکیشن خود را روی Windows Azure منتشر کردید، این پیغام دیگر تولید نمی‌شود؛ چرا که Certificateهای استفاده شده trusted هستند.

با حساب کاربری سازمانی (organizational account) که ایجاد کرده‌اید، وارد شوید.

همانطور که مشاهده می‌کنید هم اکنون به سایت وارد شده اید.

توزیع اپلیکیشن روی Windows Azure

در Windows Azure Portal یک وب سایت را به‌همراه یک دیتابیس، ایجاد کنید. در پانل سمت چپ صفحه روی **Websites** کلیک کنید و بعد **New** را انتخاب کنید. سپس گزینه **Custom Create** را برگزینید.

اپلیکیشن را روی Windows Azure منتشر کنید. روی پروژه کلیک راست کرده و **Publish** را انتخاب کنید. در مرحله تنظیمات (Settings) مشاهده می‌کنید که احراز هویت حساب‌های سازمانی (organizational accounts) فعال است. همچنین سطح دسترسی به خواندن تنظیم شده است. در قسمت Database رشته اتصال دیتابیس را تنظیم کنید.

حال به وب سایت Windows Azure خود بروید و توسط حساب کاربری ایجاد شده وارد سایت شوید. در این مرحله دیگر نباید خطای امنیتی SSL را دریافت کنید.

خواندن اطلاعات پروفایل کاربران توسط Graph API

قالب پروژه ویژوال استودیو برای organizational accounts یک متد و نما بنام UserProfile به پروژه اضافه کرده است.

```
[Authorize]
```

```
public async Task<ActionResult> UserProfile()
{
    string tenantId = ClaimsPrincipal.Current.FindFirst(TenantSchema).Value;

    // Get a token for calling the Windows Azure Active Directory Graph
    AuthenticationContext authContext = new
AuthenticationContext(String.Format(CultureInfo.InvariantCulture, LoginUrl, tenantId));
    ClientCredential credential = new ClientCredential(AppPrincipalId, AppKey);
    AuthenticationResult assertionCredential = authContext.AcquireToken(GraphUrl, credential);
    string authHeader = assertionCredential.CreateAuthorizationHeader();
    string requestUrl = String.Format(
        CultureInfo.InvariantCulture,
        GraphUserUrl,
        HttpUtility.UrlEncode(tenantId),
        HttpUtility.UrlEncode(User.Identity.Name));

    HttpClient client = new HttpClient();
    HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Get, requestUrl);
    request.Headers.TryAddWithoutValidation("Authorization", authHeader);
    HttpResponseMessage response = await client.SendAsync(request);
    string responseString = await response.Content.ReadAsStringAsync();
    UserProfile profile = JsonConvert.DeserializeObject<UserProfile>(responseString);

    return View(profile);
}
```

کلیک کردن لینک **UserProfile** اطلاعات پروفایل کاربر جاری را نمایش می‌دهد.

اطلاعات بیشتر

Managing Windows Azure AD from the Windows Azure Portal 1– Sign Up with an Organizational Account

Adding Sign-On to Your Web Application Using Windows Azure AD

Using the Graph API to Query Windows Azure AD

نظرات خوانندگان

نویسنده: حسین
تاریخ: ۱۳۹۲/۱۰/۱۵ ۱۲:۲۰

خیلی ممنون از زحمات شما
چند وقت پیش قصد داشتم ثبت نام کنم، زمانی که میخواهید حساب کاربری ایجاد کنید نیاز به شماره موبایل داره تا اجازه بده مراحل ثبت نام کامل بشه ولی اصلا امکان انتخاب کشور ایران وجود ندارد و موفق نشدم

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۶ ۱۳:۱۴

از حساب‌های کاربری مایکروسافت می تونید استفاده کنید.

نویسنده: بهروز
تاریخ: ۱۳۹۲/۱۰/۱۹ ۱۲:۱

ممنون بابت مطلب خوبتون.

لطفا یکم شفاف‌تر بگین که چطور برا تو حساب کاربری ایجاد کنیم؟ من هرکاری میکنم این صفحه میاد!
https://manage.windowsazure.com/Error/NoSubscriptions

نویسنده: کامران سادین
تاریخ: ۱۳۹۲/۱۱/۱۲ ۴:۲۲

سلام. من شماره موبایل هم دادم اما اطلاعات مستر کارت و ویزا کارت میخواد!

عنوان: **جایگزینی اسکریپت‌های WebResource.axd با فایل‌های استاتیک در ASP.NET Web forms**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۰/۱۷ ۱۰:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net

مدتی قبل سؤال مهمی در سایت مطرح شد، به شرح زیر:
«من از کنترل‌های تلریک استفاده می‌کنم که یک سری اسکریپت را بصورت
http://localhost:1244/WebResource.axd?d=aklE6L8AEfPEgIS3T-oXc6mevPfbpi6VRp_ZTP2nBVrnt5ULOFYD3GNWRrDHwANC3VDQlL8dLAa5g35dzgHyuzAgAguIpYrf-_NXIJwNNu0YRSnH3-MgKMfnwKBKF_Lk2E5oeIcLL78uDlQ0se_GxQ2&t=635231470568640000
به فرم تزریق می‌کند و بعضی وقت‌ها داخلش xp و یا یک سری دستورات اس‌کیوال تولید می‌شوند. در این حالت این مسیرها توسط ISA Server در شبکه داخلی حمله تشخیص داده شده و بلاک خواهند شد و عملا برنامه از کار می‌افتد. آیا راهی برای خلاصی از دست آن‌ها هست؟»

پاسخ: بلی. از دات نت 3 و نیم به بعد، امکان جایگزینی کامل اسکریپت‌های خودکار مدفون شده در اسمبلی‌ها با فایل‌های استاتیک پیش بینی شده‌است که در ادامه نحوه‌ی استخراج و کار با آن‌ها را بررسی خواهیم کرد.

الف) یافتن اسکریپت‌های مدفون در اسمبلی‌ها

در ابتدا اسمبلی حاوی کنترل‌های وب فرم مدنظر خود را باید توسط برنامه‌های Reflector یا ILSpy و امثال آن‌ها گشوده و نام دقیق منبع و همچنین محتوای آن فایل اسکریپت را استخراج کنید. برای مثال:

در این تصویر، اسمبلی استاندارد System.Web.Extensions مورد بررسی قرار گرفته است. برای نمونه اگر بخواهید اسکریپت‌های متناظر با ScriptManager و UpdatePanel را با معادل‌های استاتیک آن‌ها جایگزین کنید، باید دو فایل MicrosoftAjaxWebForms.js و MicrosoftAjax.js را از این اسمبلی استخراج نمائید. (برنامه‌های یاد شده امکان ذخیره سازی منابع را نیز می‌دهند)

ب) وادار کردن ASP.NET به استفاده از نسخه‌ی استاتیک منابع

```
<asp:ScriptManager ID="Scriptmanager1" runat="server">
            <Scripts>
```

```
                <asp:ScriptReference Name="MicrosoftAjaxWebForms.js" Assembly="System.Web.Extensions"
                    Path="~/staticJS1.js" />
                <asp:ScriptReference Name="MicrosoftAjax.js" Assembly="System.Web.Extensions"
Path="~/staticJS2.js" />
            </Scripts>
    </asp:ScriptManager>
```

فرض کنید دو اسکریپت یاد شده را در فایل‌های staticJS1.js و staticJS2.js در ریشه‌ی سایت خود ذخیره کرده‌اید. اکنون یک ScriptManager را به صفحه اضافه کرده و مطابق کدهای فوق، اسکریپت‌های مدفون شده در اسمبلی System.Web.Extensions را به این فایل‌های استاتیک هدایت کنید. همانطور که عنوان شد نام این مداخل باید دقیقا با نام موجود در اسمبلی یکی باشد؛ در غیر اینصورت با خطای ذیل مواجه خواهید شد:

```
The assembly 'System.Web.Extensions' does not contain a Web resource that has the name 'xyz.js'.
Make sure that the resource name is spelled correctly.
Make sure that the application references the correct version of an ASP.NET AJAX Framework assembly.
```

اکنون اگر برنامه را اجرا کنید (با فرض قرار داشتن یک ScriptManager و UpdatePanel در صفحه)، اینبار دیگر در سورس صفحه، شاهد آدرس‌های طولانی WebResource.axd و ScriptResource.axd نخواهید بود. به صورت خودکار از دو فایل استاتیک تنظیم شده، استفاده می‌شود:

```
<script src="staticJS1.js" type="text/javascript"></script>
<script src="staticJS2.js" type="text/javascript"></script>
```

بدیهی است در صورت نیاز، باید تعاریف سایر اسکریپت‌های مدفون در اسمبلی یاد شده یا اسمبلی System.Web را نیز به صفحه از طریق ScriptManager اضافه کرد. در مورد کنترل‌های ثالث نیز وضع به همین صورت است و استاندارد آن تفاوتی نمی‌کند.

یک نکته‌ی تکمیلی

در مطلب « ASP.NET 4.5 ScriptManager Improvements in WebForms » مشاهده خواهید کرد که از ASP.NET 4.5 به بعد، طی دو بسته‌ی نیوگت که هر از چندگاهی به روز می‌شوند، کلیه اسکریپت‌های System.Web و System.Web.Extensions خارج از این اسمبلی‌ها نیز قابل دریافت بوده و با استفاده از سیستم bunding & minification می‌توان آن‌ها را فشرده و یکی کرد.

نظرات خوانندگان

نویسنده: سجـــــاد ف
تاریخ: ۱۳۹۲/۱۰/۱۷ ۱۰:۳۱

با تشکـــــــ

نیست این فایل‌ها را با هم یکی کنیم؟ (داخل یک اسمبلی گذاشتن به معنای یکی کردن فایل‌ها نیست) فشرده سازی خود فایل‌ها با حذف فواصل یا کوتاه کردن نام متغیرها چطور؟ اگر در این بین، سرور این‌ها را به صورت gzip ارائه دهد که خیلی خوب خواهد شد. اگر هدر کش کردن به مدت یکسال را هم در سمت کلاینت اضافه کنیم که عالی و به علاوه اگر فایلی در سمت سرور به روز شد، به صورت خودکار این کش دیگر قابل استفاده نباشد و به روز شود. اینجا است که سیستم minification & bundling دات نت متولد می‌شود. هم در MVC قابل استفاده است و هم در وب فرم‌ها. بنابراین طراحی سیستمی بهینه جهت ارائه اسکریپت‌ها و شیوه‌نامه‌ها، فراتر است از صرفا قرار دادن چند فایل در یک اسمبلی و ارائه خام آن‌ها.

عنوان: **افزودن ASP.NET Identity به یک پروژه Web Forms**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۸ ۱۳:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Entity framework, Security, ASP.Net MVC, ASP.NET Identity, Forms Authentication

با نصب و اجرای Visual Studio 2013 Express for Web یا Visual Studio 2013 شروع کنید.

یک پروژه جدید بسازید (از صفحه شروع یا منوی فایل)

گزینه Visual C# و سپس **ASP.NET Web Application** را انتخاب کنید. نام پروژه را به "WebFormsIdentity" تغییر داده و OK کنید.

در دیالوگ جدید ASP.NET گزینه **Empty** را انتخاب کنید.

دقت کنید که دکمه **Change Authentication** غیرفعال است و هیچ پشتیبانی ای برای احراز هویت در این قالب پروژه وجود ندارد.

افزودن پکیج‌های ASP.NET Identity به پروژه

روی نام پروژه کلیک راست کنید و گزینه **Manage NuGet Packages** را انتخاب کنید. در قسمت جستجوی دیالوگ باز شده عبارت "Identity.E" را وارد کرده و این پکیج را نصب کنید.

دقت کنید که نصب کردن این پکیج وابستگی‌ها را نیز بصورت خودکار نصب می‌کند: Entity Framework و ASP.NET Idenity Core.

افزودن فرم‌های وب لازم برای ثبت نام کاربران

یک فرم وب جدید بسازید.

در دیالوگ باز شده نام فرم را به **Register** تغییر داده و تایید کنید.

فایل ایجاد شده جدید را باز کرده و کد Markup آن را با قطعه کد زیر جایگزین کنید.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Register.aspx.cs"
Inherits="WebFormsIdentity.Register" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
```

```
</head>
<body style="
    <form id="form1" runat="server">
    <div>
        <h4 style="Register a new user</h4>
        <hr />
        <p>
            <asp:Literal runat="server" ID="StatusMessage" />
        </p>
        <div style="margin-bottom:10px">
            <asp:Label runat="server" AssociatedControlID="UserName">User name</asp:Label>
            <div>
                <asp:TextBox runat="server" ID="UserName" />
            </div>
        </div>
        <div style="margin-bottom:10px">
            <asp:Label runat="server" AssociatedControlID="Password">Password</asp:Label>
            <div>
                <asp:TextBox runat="server" ID="Password" TextMode="Password" />
            </div>
        </div>
        <div style="margin-bottom:10px">
            <asp:Label runat="server" AssociatedControlID="ConfirmPassword">Confirm
password</asp:Label>
            <div>
                <asp:TextBox runat="server" ID="ConfirmPassword" TextMode="Password" />
            </div>
        </div>
        <div>
            <div>
                <asp:Button runat="server" OnClick="CreateUser_Click" Text="Register" />
            </div>
        </div>
    </div>
    </form>
</body>
</html>
```

این تنها یک نسخه ساده شده *Register.aspx* است که از چند فیلد فرم و دکمه ای برای ارسال آنها به سرور استفاده می‌کند.

فایل کد این فرم را باز کرده و محتویات آن را با قطعه کد زیر جایگزین کنید.

```
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
using System;
using System.Linq;

namespace WebFormsIdentity
{
   public partial class Register : System.Web.UI.Page
   {
      protected void CreateUser_Click(object sender, EventArgs e)
      {
         // Default UserStore constructor uses the default connection string named: DefaultConnection
         var userStore = new UserStore<IdentityUser>();
         var manager = new UserManager<IdentityUser>(userStore);

         var user = new IdentityUser() { UserName = UserName.Text };
         IdentityResult result = manager.Create(user, Password.Text);

         if (result.Succeeded)
         {
            StatusMessage.Text = string.Format("User {0} was created successfully!", user.UserName);
         }
         else
         {
            StatusMessage.Text = result.Errors.FirstOrDefault();
         }
      }
   }
}
```

کد این فرم نیز نسخه ای ساده شده است. فایلی که بصورت خودکار توسط VS برای شما ایجاد می‌شود متفاوت است.

کلاس *IdentityUser* پیاده سازی پیش فرض EntityFramework از قرارداد IUser است. قرارداد IUser تعریفات حداقلی یک کاربر در ASP.NET Identity Core را در بر می‌گیرد.

کلاس *UserStore* پیاده سازی پیش فرض EF از یک فروشگاه کاربر (user store) است. این کلاس چند قرارداد اساسی ASP.NET Identity Core را پیاده سازی می‌کند: *IUserClaimStore*, *IUserLoginStore*, *IUserStore* و *IUserRoleStore*.

کلاس *UserManager* دسترسی به APIهای مربوط به کاربران را فراهم می‌کند. این کلاس تمامی تغییرات را بصورت خودکار در *UserStore* ذخیره می‌کند.

کلاس *IdentityResult* نتیجه یک عملیات هویتی را معرفی می‌کند (identity operations).

پوشه **App_Data** را به پروژه خود اضافه کنید.

فایل *Web.config* پروژه را باز کنید و رشته اتصال جدیدی برای دیتابیس اطلاعات کاربران اضافه کنید. این دیتابیس در زمان اجرا (runtime) بصورت خودکار توسط EF ساخته می‌شود. این رشته اتصال شبیه به رشته اتصالی است که هنگام ایجاد پروژه بصورت

خودکار برای شما تنظیم می‌شود.

```
<?xml version="1.0" encoding="utf-8"?>
<!--
  For more information on how to configure your ASP.NET application, please visit
  http://go.microsoft.com/fwlink/?LinkId=169433
  -->
<configuration>
  <configSections>
    <!-- For more information on Entity Framework configuration, visit
http://go.microsoft.com/fwlink/?LinkID=237468 -->
    <section name="entityFramework"
type="System.Data.Entity.Internal.ConfigFile.EntityFrameworkSection, EntityFramework, Version=6.0.0.0,
Culture=neutral, PublicKeyToken=b77a5c561934e089" requirePermission="false" />
  </configSections>
   <connectionStrings>
      <add name="DefaultConnection" connectionString="Data
Source=(LocalDb)\v11.0;AttachDbFilename=|DataDirectory|\WebFormsIdentity.mdf;Initial
Catalog=WebFormsIdentity;Integrated Security=True"
            providerName="System.Data.SqlClient" />
   </connectionStrings>
  <system.web>
    <compilation debug="true" targetFramework="4.5" />
    <httpRuntime targetFramework="4.5" />
  </system.web>
  <entityFramework>
    <defaultConnectionFactory type="System.Data.Entity.Infrastructure.LocalDbConnectionFactory,
EntityFramework">
      <parameters>
        <parameter value="v11.0" />
      </parameters>
    </defaultConnectionFactory>
    <providers>
      <provider invariantName="System.Data.SqlClient"
type="System.Data.Entity.SqlServer.SqlProviderServices, EntityFramework.SqlServer" />
    </providers>
  </entityFramework>
</configuration>
```

همانطور که مشاهده می‌کنید نام این رشته اتصال *DefaultConnection* است.

روی فایل *Register.aspx* کلیک راست کنید و گزینه **Set As Start Page** را انتخاب کنید. اپلیکیشن خود را با کلیدهای ترکیبی Ctrl + F5 اجرا کنید که تمام پروژه را کامپایل نیز خواهد کرد. یک نام کاربری و کلمه عبور وارد کنید و روی **Register** کلیک کنید.

ASP.NET Identity از اعتبارسنجی نیز پشتیبانی می‌کند، مثلا در این مرحله می‌توانید از اعتبارسنج هایی که توسط ASP.NET Identity Core عرضه می‌شوند برای کنترل رفتار فیلدهای نام کاربری و کلمه عبور استفاده کنید. اعتبارسنج پیش فرض کاربران (User) که UserValidator نام دارد خاصیتی با نام AllowOnlyAlphanumericUserNames دارد که مقدار پیش فرضش هم true است. اعتبارسنج پیش فرض کلمه عبور (MinimumLengthValidator) اطمینان حاصل می‌کند که کلمه عبور حداقل 6 کاراکتر باشد. این اعتبارسنج‌ها بصورت propertyها در کلاس UserManager تعریف شده اند و می‌توانید آنها را overwrite کنید و اعتبارسنجی سفارشی خود را پیاده کنید. از آنجا که الگوی دیتابیس سیستم عضویت توسط Entity Framework مدیریت می‌شود، روی الگوی دیتابیس کنترل کامل دارید، پس از Data Annotations نیز می‌توانید استفاده کنید.

تایید دیتابیس LocalDbIdentity که توسط EF ساخته می‌شود

از منوی **View** گزینه **Server Explorer** را انتخاب کنید.

MvcAuth - Microsoft Visual Studio
FILE
EDIT
VIEW
PROJECT
BUILD
DEBUG
TEAM
TOOLS
Solution Explorer Ctrl+Alt+L
Team Explorer Ctrl+\, Ctrl+M
Server Explorer Ctrl+Alt+S
Architecture Explorer Ctrl+\, Ctrl+R
SQL Server Object Explorer Ctrl+\, Ctrl+S
Bookmark Window Ctrl+K, Ctrl+W
Call Hierarchy Ctrl+Alt+K
Class View Ctrl+Shift+C
Code Definition Window Ctrl+\, D
Document Outline Ctrl+Alt+T
Object Browser Ctrl+Alt+J
Error List Ctrl+\, E
Output Ctrl+Alt+O
Start Page
Task List Ctrl+\, T
Toolbox Ctrl+Alt+X
Notifications Ctrl+W, N
Find Results
Other Windows
Toolbars
Full Screen Shift+Alt+Enter
All Windows Shift+Alt+M
Properties Window F4
Property Pages Shift+F4

گره **DefaultConnection (WebFormsIdentity)** و سپس **Tables** را باز کنید. روی جدول **AspNetUsers** کلیک راست کرده و **Show Table Data** را انتخاب کنید.

پیکربندی اپلیکیشن برای استفاده از احراز هویت OWIN

تا این مرحله ما تنها امکان ایجاد حساب‌های کاربری را فراهم کرده ایم. حال نیاز داریم امکان احراز هویت کاربران برای ورود آنها به سایت را فراهم کنیم. ASP.NET Identity برای احراز هویت مبتنی بر فرم (forms authentication) از OWIN Authentication استفاده می‌کند. OWIN Cookie Authentication مکانیزمی برای احراز هویت کاربران بر اساس cookieها و claimها است (claims-based). این مکانیزم می‌تواند توسط Entity Framework روی OWIN یا IIS استفاده شود.

با چنین مدلی، می‌توانیم از پکیج‌های احراز هویت خود در فریم ورک‌های مختلفی استفاده کنیم، مانند ASP.NET MVC و ASP.NET Web Forms. برای اطلاعات بیشتر درباره پروژه Katana و نحوه اجرای آن بصورت Host Agnostic به لینک Getting Started with the Katana Project مراجعه کنید.

نصب پکیج‌های احراز هویت روی پروژه

روی نام پروژه خود کلیک راست کرده و **Manage NuGet Packages** را انتخاب کنید. در قسمت جستجوی دیالوگ باز شده عبارت "*Identity.Owin*" را وارد کنید و این پکیج را نصب کنید.

به دنبال پکیجی با نام *Microsoft.Owin.Host.SystemWeb* بگردید و آن را نیز نصب کنید.

پکیج **Microsoft.Aspnet.Identity.Owin** حاوی یک سری کلاس Owin Extension است و امکان مدیریت و پیکربندی OWIN Authentication در پکیج‌های ASP.NET Identity Core را فراهم می‌کند.

پکیج **Microsoft.Owin.Host.SystemWeb** حاوی یک سرور OWIN است که اجرای اپلیکیشن‌های مبتنی بر OWIN را روی IIS و با استفاده از ASP.NET Request Pipeline ممکن می‌سازد. برای اطلاعات بیشتر به OWIN Middleware in the IIS integrated pipeline مراجعه کنید.

افزودن کلاس‌های پیکربندی Startup و Authentication

روی پروژه خود کلیک راست کرده و گزینه **Add** و سپس **Add New Item** را انتخاب کنید. در قسمت جستجوی دیالوگ باز شده عبارت "owin" را وارد کنید. نام کلاس را "*Startup*" تعیین کرده و تایید کنید.

فایل Startup.cs را باز کنید و قطعه کد زیر را با محتویات آن جایگزین کنید تا احراز هویت OWIN Cookie Authentication پیکربندی شود.

```
using Microsoft.AspNet.Identity;
using Microsoft.Owin;
using Microsoft.Owin.Security.Cookies;
using Owin;

[assembly: OwinStartup(typeof(WebFormsIdentity.Startup))]

namespace WebFormsIdentity
{
   public class Startup
   {
      public void Configuration(IAppBuilder app)
      {
         // For more information on how to configure your application, visit
http://go.microsoft.com/fwlink/?LinkID=316888
         app.UseCookieAuthentication(new CookieAuthenticationOptions
         {
            AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
            LoginPath = new PathString("/Login")
         });
      }
   }
}
```

این کلاس حاوی خاصیت OwinAttribute است که کلاس راه انداز OWIN را نشانه گذاری می‌کند. هر اپلیکیشن OWIN یک کلاس راه انداز (startup) دارد که توسط آن می‌توانید کامپوننت‌های application pipeline را مشخص کنید. برای اطلاعات بیشتر درباره این مدل، به OWIN Startup Class Detection مراجعه فرمایید.

افزودن فرم‌های وب برای ثبت نام و ورود کاربران

فایل *Register.cs* را باز کنید و قطعه کد زیر را وارد کنید. این قسمت پس از ثبت نام موفقیت آمیز کاربر را به سایت وارد می‌کند.

```
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
using Microsoft.Owin.Security;
using System;
using System.Linq;
using System.Web;

namespace WebFormsIdentity
{
   public partial class Register : System.Web.UI.Page
   {
      protected void CreateUser_Click(object sender, EventArgs e)
      {
         // Default UserStore constructor uses the default connection string named: DefaultConnection
         var userStore = new UserStore<IdentityUser>();
         var manager = new UserManager<IdentityUser>(userStore);
         var user = new IdentityUser() { UserName = UserName.Text };

         IdentityResult result = manager.Create(user, Password.Text);

         if (result.Succeeded)
         {
            var authenticationManager = HttpContext.Current.GetOwinContext().Authentication;
            var userIdentity = manager.CreateIdentity(user,
DefaultAuthenticationTypes.ApplicationCookie);
            authenticationManager.SignIn(new AuthenticationProperties() { }, userIdentity);
            Response.Redirect("~/Login.aspx");
         }
         else
         {
            StatusMessage.Text = result.Errors.FirstOrDefault();
         }
      }
   }
}
```

از آنجا که ASP.NET Identity و OWIN Cookie Authentication هر دو مبتنی بر Claims هستند، فریم ورک از برنامه نویس اپلیکیشن انتظار دارد تا برای کاربر یک آبجکت از نوع ClaimsIdentity تولید کند. این آبجکت تمام اطلاعات اختیارات کاربر را در بر می‌گیرد، مثلا اینکه کاربر به چه نقش هایی تعلق دارد. همچنین در این مرحله می‌توانید اختیارات (Claims) جدیدی به کاربر اضافه کنید.

شما با استفاده از AuthenticationManager که متعلق به OWIN است می‌توانید کاربر را به سایت وارد کنید. برای این کار شما متد SignIn را فراخوانی می‌کنید و آبجکتی از نوع ClaimsIdentity را به آن پاس می‌دهید. این کد کاربر را به سایت وارد می‌کند و یک کوکی برای او می‌سازد. این فراخوانی معادل همان FormAuthentication.SetAuthCookie است که توسط ماژول FormsAuthentication استفاده می‌شود.

روی پروژه خود کلیک راست کرده، فرم وب جدیدی با نام **Login** بسازید.

فایل *Login.aspx* را باز کنید و کد Markup آن را مانند قطعه کد زیر تغییر دهید.

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs"
Inherits="WebFormsIdentity.Login" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title></title>
</head>
<body style="font-family: Arial, Helvetica, sans-serif; font-size: small">
   <form id="form1" runat="server">
      <div>
```

```
         <h4 style="font-size: medium">Log In</h4>
         <hr />
         <asp:PlaceHolder runat="server" ID="LoginStatus" Visible="false">
            <p>
               <asp:Literal runat="server" ID="StatusText" />
            </p>
         </asp:PlaceHolder>
         <asp:PlaceHolder runat="server" ID="LoginForm" Visible="false">
            <div style="margin-bottom: 10px">
               <asp:Label runat="server" AssociatedControlID="UserName">User name</asp:Label>
               <div>
                  <asp:TextBox runat="server" ID="UserName" />
               </div>
            </div>
            <div style="margin-bottom: 10px">
               <asp:Label runat="server" AssociatedControlID="Password">Password</asp:Label>
               <div>
                  <asp:TextBox runat="server" ID="Password" TextMode="Password" />
               </div>
            </div>
            <div style="margin-bottom: 10px">
               <div>
                  <asp:Button runat="server" OnClick="SignIn" Text="Log in" />
               </div>
            </div>
         </asp:PlaceHolder>
         <asp:PlaceHolder runat="server" ID="LogoutButton" Visible="false">
            <div>
               <div>
                  <asp:Button runat="server" OnClick="SignOut" Text="Log out" />
               </div>
            </div>
         </asp:PlaceHolder>
      </div>
   </form>
</body>
</html>
```

محتوای فایل *Login.aspx.cs* را نیز مانند لیست زیر تغییر دهید.

```
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
using Microsoft.Owin.Security;
using System;
using System.Web;
using System.Web.UI.WebControls;

namespace WebFormsIdentity
{
   public partial class Login : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            if (User.Identity.IsAuthenticated)
            {
               StatusText.Text = string.Format("Hello {0}!", User.Identity.GetUserName());
               LoginStatus.Visible = true;
               LogoutButton.Visible = true;
            }
            else
            {
               LoginForm.Visible = true;
            }
         }
      }

      protected void SignIn(object sender, EventArgs e)
      {
         var userStore = new UserStore<IdentityUser>();
         var userManager = new UserManager<IdentityUser>(userStore);
         var user = userManager.Find(UserName.Text, Password.Text);

         if (user != null)
         {
            var authenticationManager = HttpContext.Current.GetOwinContext().Authentication;
            var userIdentity = userManager.CreateIdentity(user,
```

```
DefaultAuthenticationTypes.ApplicationCookie);

            authenticationManager.SignIn(new AuthenticationProperties() { IsPersistent = false },
userIdentity);
            Response.Redirect("~/Login.aspx");
         }
         else
         {
            StatusText.Text = "Invalid username or password.";
            LoginStatus.Visible = true;
         }
      }

      protected void SignOut(object sender, EventArgs e)
      {
         var authenticationManager = HttpContext.Current.GetOwinContext().Authentication;
         authenticationManager.SignOut();
         Response.Redirect("~/Login.aspx");
      }
   }
}
```

متد Page_Load حالا وضعیت کاربر جاری را بررسی می‌کند و بر اساس وضعیت Context.User.Identity.IsAuthenticated تصمیم گیری می‌کند.

نمایش نام کاربر جاری: فریم ورک ASP.NET Identity روی System.Security.Principal.Identity متدهایی نوشته است که به شما امکان دریافت نام و شناسه کاربر جاری را می‌دهد. این متدها در اسمبلی Microsoft.AspNet.Identity.Core وجود دارند. این متدها جایگزین HttpContext.User.Identity.Name هستند.

متد SignIn

این متد، متد CreateUser_Click را که پیشتر بصورت خودکار ایجاد شده جایگزین می‌کند و پس از ایجاد موفقیت آمیز حساب کاربری، کاربر جاری را به سایت وارد می‌کند. فریم ورک OWIN متدهایی روی System.Web.HttpContext افزوده است که به شما این امکان را می‌دهند که یک ارجاع از نوع IOwinContext بگیرید. این متدها در اسمبلی Microsoft.Owin.Host.SystemWeb وجود دارند. کلاس OwinContext خاصیتی از نوع IAuthenticationManager دارد که امکانات احراز هویت موجود برای درخواست جاری را معرفی می‌کند.

پروژه را با Ctrl + F5 اجرا کنید و کاربر جدیدی بسازید. پس از وارد کردن نام کاربری و کلمه عبور و کلیک کردن دکمه Register باید بصورت خودکار به سایت وارد شوید و نام خود را مشاهده کنید.

همانطور که مشاهده می‌کنید در این مرحله حساب کاربری جدید ایجاد شده و به سایت وارد شده اید. روی Log out کلیک کنید تا از سایت خارج شوید. پس از آن باید به صفحه ورود هدایت شوید.
حالا یک نام کاربری یا کلمه عبور نامعتبر وارد کنید و روی **Log in** کلیک کنید.

متد UserManager.Find مقدار null بر می‌گرداند، بنابراین پیام خطای "*Invalid username or password*" نمایش داده خواهد شد.

http://localhost:41969/Login.a
Log In
Invalid username or password.
User name
RaquelS
Password
Log in

نظرات خوانندگان

نویسنده: Programmer
تاریخ: ۱۳۹۲/۱۰/۱۹ ۱۴:۴۱

با عرض سلام و تشکر بابت ترجمه روانتون. خیلی وقت بود که منتظر همچین پستی بودم. اینکه بشه با EF عملیات احراز هویت رو با مکانیسمی قویتر از Memberdship سابق انجام داد. اگر ممکنه همین مثال رو در قالب پروژه MVC انجام بدید. ممنون

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۱۹ ۱۸:۹

سلام، متشکرم.
ASP.NET Identity بصورت پیش فرض در قالب پروژه‌های VS 2013 استفاده میشه. در پست‌های قبلی بیشتر درباره این فریم بحث شده که می‌تونید مراجعه کنید. برای اطلاعات بیشتر به ASP.NET Identity سر بزنید.

نویسنده: کامران
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۶:۰

سلام.
من اگر بخوام لیست کاربرایی که یک نقش خاص مثلا "Admin" رو دارن لیست کنم چطور میتونم دسترسی داشته باشم؟

نویسنده: غلامرضا
تاریخ: ۱۳۹۳/۰۵/۳۰ ۳:۴۹

سلام و میبخشید آیا به غیر از روش ایداعی این سایت روش دیگه ای برای استفاده از asp.net identity با EF db first وجود داره؟ اگه هست لطفا راهنمایی کنید.

نویسنده: غلامرضا
تاریخ: ۱۳۹۳/۰۵/۳۱ ۱۶:۳۲

جواب رو پیدا کردم اینم لینکش. فیلم

عنوان: **5 قابلیت برتر جدید در ASP.NET Web API 2**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۰ ۴:۳۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, ASP.NET Web API, ASP.NET Web API 2

ASP.NET Web API 2 بهمراه یک سری قابلیت جدید جالب منتشر شده است. در این پست 5 قابلیت برتر از این قابلیت‌های جدید را بررسی می‌کنیم.

1. Attribute Routing

در کنار سیستم routing فعلی، ASP.NET Web API 2 حالا از Attribute Routing هم پشتیبانی می‌کند. در مورد سیستم routing فعلی، می‌توانیم قالب‌های متعددی برای routing بنویسیم. هنگامی که یک درخواست به سرور میرسد، کنترلر مناسب انتخاب شده و اکشن متد مناسب فراخوانی می‌شود.
در لیست زیر قالب پیش فرض routing در Web API را مشاهده می‌کنید.

```
Config.Routes.MapHttpRoute(
     name: "DefaultApi",
     routeTemplate: "api/{Controller}/{id}",
     defaults: new { id = RouteParameter.Optional }
);
```

این رویکرد routing مزایای خود را دارد. از جمله اینکه تمام مسیرها در یک مکان واحد تعریف می‌شوند، اما تنها برای الگوهایی مشخص. مثلا پشتیبانی از nested routing روی یک کنترلر مشکل می‌شود.
در ASP.NET Web API 2 به سادگی می‌توانیم الگوی URI ذکرد شده را پشتیبانی کنیم. لیست زیر نمونه ای از یک الگوی URI با AttributeRouting را نشان می‌دهد.

```
URI Pattern --> books/1/authors

[Route("books/{bookId}/authors")]
public IEnumerable<Author> GetAuthorByBook(int bookId) { ..... }
```

2. CORS - Cross Origin Resource Sharing

بصورت نرمال، مرورگرها اجازه درخواست‌های cross-domain را نمی‌دهند، که بخاطر same-origin policy است. خوب، CORS (Cross Origin Resource Sharing) چیست؟
CORS یک مکانیزم است که به صفحات وب این را اجازه می‌دهد تا یک درخواست آژاکسی (Ajax Request) به دامنه ای دیگر ارسال کنند. دامنه ای به غیر از دامنه ای که صفحه وب را رندر کرده است. CORS با استانداردهای W3C سازگار است و حالا ASP.NET Web API در نسخه 2 خود از آن پشتیبانی می‌کند.

3. OWIN (Open Web Interface for .NET) self-hosting

ASP.NET Web API 2 بهمراه یک پکیج عرضه می‌شود، که *Microsoft.AspNet.WebApi.OwinSelfHost* نام دارد.
طبق گفته وب سایت http://owin.org :
OWIN یک اینترفیس استاندارد بین سرورهای دات نت و اپلیکیشن‌های وب تعریف می‌کند. هدف این اینترفیس جداسازی (decoupling) سرور و اپلیکیشن است. تشویق به توسعه ماژول‌های ساده برای توسعه اپلیکیشن‌های وب دات نت. و بعنوان یک استاندارد باز (open standard) اکوسیستم نرم افزارهای متن باز را تحریک کند تا ابزار توسعه اپلیکیشن‌های وب دات نت توسعه یابند.
بنابراین طبق گفته‌های بالا، OWIN گزینه ای ایده آل برای میزبانی اپلیکیشن‌های وب روی پروسس هایی به غیر از پروسس IIS است. پیاده سازی‌های دیگری از OWIN نیز وجود دارند، مانند Giacomo، Kayak,Firefly و غیره. اما **Katana** گزینه توصیه شده برای سرورهای مایکروسافت و فریم ورک‌های Web API است.

4. IHttpActionResult

در کنار دو روش موجود فعلی برای ساختن response اکشن متدها در کنترلر ها، ASP.NET Web API 2 حالا از مدل جدیدی هم

پشتیبانی می‌کند. *IHttpResponseMessage* یک اینترفیس است که بعنوان یک فاکتوری (factory) برای *HttpResponseMessage* کار می‌کند. این روش بسیار قدرتمند است بدلیل اینکه web api را گسترش می‌دهد. با استفاده از این رویکرد، می‌توانیم response هایی با هر نوع دلخواه بسازیم.

برای اطلاعات بیشتر به how to serve HTML with IHTTPActionResult مراجعه کنید.

5. Web API OData

پروتکل (Open Data Protocol) OData در واقع یک پروتکل وب برای کوئری گرفتن و بروز رسانی داده‌ها است. ASP.NET Web API 2 پشتیبانی از *$expand, $select* و *$value* را اضافه کرده است. با استفاده از این امکانات، می‌توانیم نحوه معرفی پاسخ سرور را کنترل کنیم، یعنی representation دریافتی از سرور را می‌توانید سفارشی کنید.

$expand: بصورت نرمال، هنگام کوئری گرفتن از یک کالکشن OData، پاسخ سرور موجودیت‌های مرتبط (related entities) را شامل نمی‌شود. با استفاده از $expand می‌توانیم موجودیت‌های مرتبط را بصورت inline در پاسخ سرور دریافت کنیم.

$select: از این متد برای انتخاب چند خاصیت بخصوص از پاسخ سرور استفاده می‌شود، بجای آنکه تمام خاصیت‌ها بارگذاری شوند.

$value: با این متد مقدار خام (raw) فیلدها را بدست می‌آورید، بجای دریافت آنها در فرمت OData.

چند مقاله خوب دیگر

Top 10 ASP.NET MVC Interview Questions

jQuery code snippets every web developer must have 7

WCF Vs ASMX Web Services

Top 10 HTML5 Interview Questions

JavaScript : Validating Letters and Numbers

عنوان: **مهاجرت از SQL Membership به ASP.NET Identity**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۱ ۱۸:۳۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Security, MemberShip, ASP.NET Identity

در این مقاله مهاجرت یک اپلیکیشن وب که توسط SQL Membership ساخته شده است را به سیستم جدید ASP.NET Identity بررسی می‌کنیم. برای این مقاله از یک قالب اپلیکیشن وب (Web Forms) که توسط Visual Studio 2010 ساخته شده است برای ساختن کاربران و نقش‌ها استفاده می‌کنیم. سپس با استفاده از یک SQL Script دیتابیس موجود را به دیتابیسی که ASP.NET Identity نیاز دارد تبدیل می‌کنیم. در قدم بعدی پکیج‌های مورد نیاز را به پروژه اضافه می‌کنیم و صفحات جدیدی برای مدیریت حساب‌های کاربری خواهیم ساخت. بعنوان یک تست، کاربران قدیمی که توسط SQL Membership ساخته شده بودند باید قادر باشند به سایت وارد شوند. همچنین کاربران جدید باید بتوانند بدون هیچ مشکلی در سیستم ثبت نام کنند. سورس کد کامل این مقاله را می‌توانید از این لینک دریافت کنید.

یک اپلیکیشن با SQL Membership بسازید

برای شروع به اپلیکیشنی نیاز داریم که از SQL Membership استفاده می‌کند و دارای داده هایی از کاربران و نقش‌ها است. برای این مقاله، بگذارید پروژه جدیدی توسط VS 2010 بسازیم.

حال با استفاده از ابزار ASP.NET Configuration دو کاربر جدید بسازید: **oldAdminUser** و **oldUser** .

Project
Build
Debug
Team
XML
Data
Tools
Architecture
Test
Add Class...
Add New Item...
Ctrl+Shift+A
Add Existing Item...
Shift+Alt+A
Add ASP.NET Folder
Build Deployment Package
Package/Publish Settings
Exclude From Project
Show All Files
Add Reference...
Add Service Reference...
Set as StartUp Project
Use IIS Express...
Add Deployable Dependencies...
Manage NuGet Packages...
Enable NuGet Package Restore
Refresh Project Toolbox Items
SQLMembership-Identity-OWIN Properties...
ASP.NET Configuration

نقش جدیدی با نام Admin بسازید و کاربر oldAdminUser را به آن اضافه کنید.

بخش جدیدی با نام Admin در سایت خود بسازید و فرمی بنام Default.aspx به آن اضافه کنید. همچنین فایل web.config این قسمت را طوری پیکربندی کنید تا تنها کاربرانی که در نقش Admin هستند به آن دسترسی داشته باشند. برای اطلاعات بیشتر به این لینک مراجعه کنید.

پنجره Server Explorer را باز کنید و جداول ساخته شده توسط SQL Membership را بررسی کنید. اطلاعات اصلی کاربران که برای ورود به سایت استفاده می‌شوند، در جداول **aspnet_Users** و **aspnet_Membership** ذخیره می‌شوند. داده‌های مربوط به نقش‌ها نیز در جدول **aspnet_Roles** ذخیره خواهند شد. رابطه بین کاربران و نقش‌ها نیز در جدول **aspnet_UsersInRoles** ذخیره می‌شود، یعنی اینکه هر کاربری به چه نقش هایی تعلق دارد.

برای مدیریت اساسی سیستم عضویت، مهاجرت جداول ذکر شده به سیستم جدید ASP.NET Identity کفایت می‌کند.

مهاجرت به Visual Studio 2013

برای شروع ابتدا Visual Studio Express 2013 for Web یا Visual Studio 2013 را نصب کنید.

حال پروژه ایجاد شده را در نسخه جدید ویژوال استودیو باز کنید. اگر نسخه ای از SQL Server Express را روی سیستم خود نصب نکرده باشید، هنگام باز کردن پروژه پیغامی به شما نشان داده می‌شود. دلیل آن وجود رشته اتصالی است که از SQL Server Express استفاده می‌کند. برای رفع این مساله می‌توانید SQL Express را نصب کنید، و یا رشته اتصال را طوری تغییر دهید که از LocalDB استفاده کند.

فایل web.config را باز کرده و رشته اتصال را مانند تصویر زیر ویرایش کنید.

```
<configuration>
  <connectionStrings>
    <add name="ApplicationServices"
         connectionString="data source=(LocalDb)\v11.0;Integrated Security=SSPI;AttachDBFilename=|DataDirectory|\aspnetdb.mdf
         providerName="System.Data.SqlClient" />
  </connectionStrings>
```

پنجره Server Explorer را باز کنید و مطمئن شوید که الگوی جداول و داده‌ها قابل رویت هستند.
سیستم ASP.NET Identity با نسخه 4.5 دات نت فریم ورک و بالاتر سازگار است. پس نسخه فریم ورک پروژه را به آخرین نسخه (4.5.1) تغییر دهید.

پروژه را Build کنید تا مطمئن شوید هیچ خطایی وجود ندارد.

نصب پکیج‌های NuGet

در پنجره Solution Explorer روی نام پروژه خود کلیک راست کرده، و گزینه Manage NuGet Packages را انتخاب کنید. در قسمت جستجوی دیالوگ باز شده، عبارت "Microsoft.AspNet.Identity.EntityFramework" را وارد کنید. این پکیج را در لیست نتایج انتخاب کرده و آن را نصب کنید. نصب این بسته، نیازمندهای موجود را بصورت خودکار دانلود و نصب می‌کند:

EntityFramework و **ASP.NET Identity Core** . حال پکیج‌های زیر را هم نصب کنید (اگر نمی‌خواهید OAuth را فعال کنید، 4 پکیج آخر را نادیده بگیرید).

Microsoft.AspNet.Identity.Owin

Microsoft.Owin.Host.SystemWeb

Microsoft.Owin.Security.Facebook

Microsoft.Owin.Security.Google

Microsoft.Owin.Security.MicrosoftAccount

Microsoft.Owin.Security.Twitter

مهاجرت دیتابیس فعلی به سیستم ASP.NET Identity

قدم بعدی مهاجرت دیتابیس فعلی به الگویی است، که سیستم ASP.NET Identity به آن نیاز دارد. بدین منظور ما یک اسکریپت SQL را اجرا می‌کنیم تا جداول جدیدی بسازد و اطلاعات کاربران را به آنها انتقال دهد. فایل این اسکریپت را می‌توانید از لینک https://github.com/suhasj/SQLMembership-Identity-OWIN دریافت کنید.

این اسکریپت مختص این مقاله است. اگر الگوی استفاده شده برای جداول سیستم عضویت شما ویرایش/سفارشی-سازی شده باید این اسکریپت را هم بر اساس این تغییرات بروز رسانی کنید.

پنجره Server Explorer را باز کنید. گره اتصال ApplicationServices را باز کنید تا جداول را مشاهده کنید. روی گره Tables کلیک راست کرده و گزینه New Query را انتخاب کنید.

در پنجره کوئری باز شده، تمام محتویات فایل *Migrations.sql* را کپی کنید. سپس اسکریپت را با کلیک کردن دکمه Execute اجرا کنید.

```sql
DROP TABLE AspNetUserRoles;

DROP TABLE AspNetUserClaims;

DROP TABLE AspNetUserLogins;

DROP TABLE AspNetRoles;

DROP TABLE AspNetUsers;

CREATE TABLE [dbo].[AspNetUsers] (
    [Id]                    NVARCHAR (128) NOT NULL,
    [UserName]              NVARCHAR (MAX) NULL,
    [PasswordHash]          NVARCHAR (MAX) NULL,
    [SecurityStamp]         NVARCHAR (MAX) NULL,
    [Discriminator]         NVARCHAR (128) NOT NULL,
    [ApplicationId]                        UNIQUEIDENTIFIER NOT NULL,
    [PasswordFormat]                       INT              DEFAULT ((0)) NULL,
    [PasswordSalt]                         NVARCHAR (128)   NULL,
    [LegacyPasswordHash]  NVARCHAR (MAX) NULL,
    [LoweredUserName]  NVARCHAR (256)   NOT NULL,
    [MobileAlias]      NVARCHAR (16)    DEFAULT (NULL) NULL,
    [IsAnonymous]      BIT              DEFAULT ((0)) NOT NULL,
    [LastActivityDate] DATETIME         NOT NULL,
    [MobilePIN]                            NVARCHAR (16)    NULL,
    [Email]                                NVARCHAR (256)   NULL,
    [LoweredEmail]                         NVARCHAR (256)   NULL,
    [PasswordQuestion]                     NVARCHAR (256)   NULL,
    [PasswordAnswer]                       NVARCHAR (128)   NULL,
    [IsApproved]                           BIT              NOT NULL,
    [IsLockedOut]                          BIT              NOT NULL,
    [CreateDate]                           DATETIME         NOT NULL,
    [LastLoginDate]                        DATETIME         NOT NULL,
    [LastPasswordChangedDate]              DATETIME         NOT NULL,
    [LastLockoutDate]                      DATETIME         NOT NULL,
    [FailedPasswordAttemptCount]           INT              NOT NULL,
    [FailedPasswordAttemptWindowStart]     DATETIME         NOT NULL,
    [FailedPasswordAnswerAttemptCount]     INT              NOT NULL,
```

ممکن است با اخطاری مواجه شوید مبنی بر آنکه امکان حذف (drop) بعضی از جداول وجود نداشت. دلیلش آن است که چهار عبارت اولیه در این اسکریپت، تمام جداول مربوط به Identity را در صورت وجود حذف می‌کنند. از آنجا که با اجرای اولیه این اسکریپت چنین جداولی وجود ندارند، می‌توانیم این خطاها را نادیده بگیریم. حال پنجره Server Explorer را تازه (refresh) کنید و خواهید دید که پنج جدول جدید ساخته شده اند.

لیست زیر نحوه Map کردن اطلاعات از جداول SQL Membership به سیستم Identity را نشان می‌دهد.

aspnet_Roles --> **AspNetRoles**

aspnet_Users, aspnet_Membership --> **AspNetUsers**

aspnet_UsersInRoles --> **AspNetUserRoles**

جداول AspNetUserClaims و AspNetUserLogins خالی هستند. فیلد تفکیک کننده (Discriminator) در جدول AspNetUsers باید مطابق نام کلاس مدل باشد، که در مرحله بعدی تعریف خواهد شد. همچنین ستون PasswordHash به فرم 'encrypted password salt|password format|password' می‌باشد. این شما را قادر می‌سازد تا از رمزنگاری برای ذخیره و بازیابی کلمه‌های عبور استفاده کنید. این مورد نیز در ادامه مقاله بررسی شده است.

ساختن مدل‌ها و صفحات عضویت

بصورت پیش فرض سیستم ASP.NET Identity برای دریافت و ذخیره اطلاعات در دیتابیس عضویت از Entity Framework استفاده می‌کند. برای آنکه بتوانیم با جداول موجود کار کنیم، می‌بایست ابتدا مدل هایی که الگوی دیتابیس را نمایندگی می‌کنند ایجاد کنیم. برای این کار مدل‌های ما یا باید اینترفیس‌های موجود در Identity.Core را پیاده سازی کنند، یا می‌توانند پیاده سازی‌های پیش فرض را توسعه دهند. پیاده سازی‌های پیش فرض در Microsoft.AspNet.Identity.EntityFramework وجود دارند.

در نمونه ما، جداول AspNetRoles, AspNetUserClaims, AspNetLogins و AspNetUserRole ستون هایی دارند که شباهت زیادی به پیاده سازی‌های پیش فرض سیستم Identity دارند. در نتیجه می‌توانیم از کلاس‌های موجود، برای Map کردن الگوی جدید استفاده کنیم. جدول AspNetUsers ستون‌های جدیدی نیز دارد. می‌توانیم کلاس جدیدی بسازیم که از **IdentityUser** ارث بری کند و آن را گسترش دهیم تا این فیلدهای جدید را پوشش دهد.

پوشه ای با نام Models بسازید (در صورتی که وجود ندارد) و کلاسی با نام User به آن اضافه کنید.

کلاس User باید کلاس IdentityUser را که در اسمبلی Microsoft.AspNet.Identity.EntityFramework وجود دارد گسترش دهد. خاصیت هایی را تعریف کنید که نماینده الگوی جدول AspNetUser هستند. خواص ID, Username, PasswordHash و SecurityStamp در کلاس IdentityUser تعریف شده اند، بنابراین این خواص را در لیست زیر نمی‌بینید.

```
public class User : IdentityUser
    {
        public User()
        {
            CreateDate = DateTime.Now;
            IsApproved = false;
            LastLoginDate = DateTime.Now;
            LastActivityDate = DateTime.Now;
            LastPasswordChangedDate = DateTime.Now;
            LastLockoutDate = DateTime.Parse("1/1/1754");
            FailedPasswordAnswerAttemptWindowStart = DateTime.Parse("1/1/1754");
            FailedPasswordAttemptWindowStart = DateTime.Parse("1/1/1754");
        }

        public System.Guid ApplicationId { get; set; }
        public string MobileAlias { get; set; }
        public bool IsAnonymous { get; set; }
        public System.DateTime LastActivityDate { get; set; }
        public string MobilePIN { get; set; }
        public string Email { get; set; }
        public string LoweredEmail { get; set; }
        public string LoweredUserName { get; set; }
        public string PasswordQuestion { get; set; }
        public string PasswordAnswer { get; set; }
        public bool IsApproved { get; set; }
        public bool IsLockedOut { get; set; }
        public System.DateTime CreateDate { get; set; }
        public System.DateTime LastLoginDate { get; set; }
        public System.DateTime LastPasswordChangedDate { get; set; }
        public System.DateTime LastLockoutDate { get; set; }
        public int FailedPasswordAttemptCount { get; set; }
        public System.DateTime FailedPasswordAttemptWindowStart { get; set; }
        public int FailedPasswordAnswerAttemptCount { get; set; }
        public System.DateTime FailedPasswordAnswerAttemptWindowStart { get; set; }
        public string Comment { get; set; }
    }
```

حال برای دسترسی به دیتابیس مورد نظر، نیاز به یک DbContext داریم. اسمبلی Microsoft.AspNet.Identity.EntityFramework کلاسی با نام **IdentityDbContext** دارد که پیاده سازی پیش فرض برای دسترسی به دیتابیس ASP.NET Identity است. نکته قابل توجه این است که IdentityDbContext آبجکتی از نوع **TUser** را می‌پذیرد. TUser می‌تواند هر کلاسی باشد که از IdentityUser ارث بری کرده و آن را گسترش می‌دهد.

در پوشه Models کلاس جدیدی با نام **ApplicationDbContext** بسازید که از IdentityDbContext ارث بری کرده و از کلاس

User استفاده می‌کند.

```
public class ApplicationDbContext : IdentityDbContext<User>
{

}
```

مدیریت کاربران در ASP.NET Identity توسط کلاسی با نام UserManager انجام می‌شود که در اسمبلی Microsoft.AspNet.Identity.EntityFramework قرار دارد. چیزی که ما در این مرحله نیاز داریم، کلاسی است که از UserManager ارث بری می‌کند و آن را طوری توسعه می‌دهد که از کلاس User استفاده کند.

در پوشه Models کلاس جدیدی با نام UserManager بسازید.

```
public class UserManager : UserManager<User>
{

}
```

کلمه عبور کاربران بصورت رمز نگاری شده در دیتابیس ذخیره می‌شوند. الگوریتم رمز نگاری SQL Membership با سیستم ASP.NET Identity تفاوت دارد. هنگامی که کاربران قدیمی به سایت وارد می‌شوند، کلمه عبورشان را توسط الگوریتم‌های قدیمی SQL Membership رمزگشایی می‌کنیم، اما کاربران جدید از الگوریتم‌های ASP.NET Identity استفاده خواهند کرد.

کلاس UserManager خاصیتی با نام **PasswordHasher** دارد. این خاصیت نمونه ای از یک کلاس را ذخیره می‌کند، که اینترفیس IPasswordHasher را پیاده سازی کرده است. این کلاس هنگام تراکنش‌های احراز هویت کاربران استفاده می‌شود تا کلمه‌های عبور را رمزنگاری/رمزگشایی شوند. در کلاس UserManager کلاس جدیدی بنام SQLPasswordHasher بسازید. کد کامل را در لیست زیر مشاهده می‌کنید.

```
public class SQLPasswordHasher : PasswordHasher
{
        public override string HashPassword(string password)
        {
            return base.HashPassword(password);
        }

        public override PasswordVerificationResult VerifyHashedPassword(string hashedPassword, string
providedPassword)
        {
            string[] passwordProperties = hashedPassword.Split('|');
            if (passwordProperties.Length != 3)
            {
                return base.VerifyHashedPassword(hashedPassword, providedPassword);
            }
            else
            {
                string passwordHash = passwordProperties[0];
                int passwordformat = 1;
                string salt = passwordProperties[2];
                if (String.Equals(EncryptPassword(providedPassword, passwordformat, salt),
passwordHash, StringComparison.CurrentCultureIgnoreCase))
                {
                    return PasswordVerificationResult.SuccessRehashNeeded;
                }
                else
                {
                    return PasswordVerificationResult.Failed;
                }
            }
        }

        //This is copied from the existing SQL providers and is provided only for back-compat.
        private string EncryptPassword(string pass, int passwordFormat, string salt)
        {
            if (passwordFormat == 0) // MembershipPasswordFormat.Clear
                return pass;

            byte[] bIn = Encoding.Unicode.GetBytes(pass);
```

```
            byte[] bSalt = Convert.FromBase64String(salt);
            byte[] bRet = null;

            if (passwordFormat == 1)
            { // MembershipPasswordFormat.Hashed
                HashAlgorithm hm = HashAlgorithm.Create("SHA1");
                if (hm is KeyedHashAlgorithm)
                {
                    KeyedHashAlgorithm kha = (KeyedHashAlgorithm)hm;
                    if (kha.Key.Length == bSalt.Length)
                    {
                        kha.Key = bSalt;
                    }
                    else if (kha.Key.Length < bSalt.Length)
                    {
                        byte[] bKey = new byte[kha.Key.Length];
                        Buffer.BlockCopy(bSalt, 0, bKey, 0, bKey.Length);
                        kha.Key = bKey;
                    }
                    else
                    {
                        byte[] bKey = new byte[kha.Key.Length];
                        for (int iter = 0; iter < bKey.Length; )
                        {
                            int len = Math.Min(bSalt.Length, bKey.Length - iter);
                            Buffer.BlockCopy(bSalt, 0, bKey, iter, len);
                            iter += len;
                        }
                        kha.Key = bKey;
                    }
                    bRet = kha.ComputeHash(bIn);
                }
                else
                {
                    byte[] bAll = new byte[bSalt.Length + bIn.Length];
                    Buffer.BlockCopy(bSalt, 0, bAll, 0, bSalt.Length);
                    Buffer.BlockCopy(bIn, 0, bAll, bSalt.Length, bIn.Length);
                    bRet = hm.ComputeHash(bAll);
                }
            }

            return Convert.ToBase64String(bRet);
    }
}
```

دقت کنید تا فضاهای نام System.Text و System.Security.Cryptography را وارد کرده باشید.

متد EncodePassword کلمه عبور را بر اساس پیاده سازی پیش فرض SQL Membership رمزنگاری می‌کند. این الگوریتم از System.Web گرفته می‌شود. اگر اپلیکیشن قدیمی شما از الگوریتم خاصی استفاده می‌کرده است، همینجا باید آن را منعکس کنید. دو متد دیگر نیز بنام‌های *HashPassword* و *VerifyHashedPassword* نیاز داریم. این متدها از EncodePassword برای رمزنگاری کلمه‌های عبور و تایید آنها در دیتابیس استفاده می‌کنند.

سیستم SQL Membership برای رمزنگاری (Hash) کلمه‌های عبور هنگام ثبت نام و تغییر آنها توسط کاربران، از PasswordHash, PasswordSalt و PasswordFormat استفاده می‌کرد. در روند مهاجرت، این سه فیلد در ستون PasswordHash جدول AspNetUsers ذخیره شده و با کاراکتر '|' جدا شده اند. هنگام ورود کاربری به سایت، اگر کله عبور شامل این فیلدها باشد از الگوریتم SQL Membership برای بررسی آن استفاده می‌کنیم. در غیر اینصورت از پیاده سازی پیش فرض ASP.NET Identity استفاده خواهد شد. با این روش، کاربران قدیمی لازم نیست کلمه‌های عبور خود را صرفا بدلیل مهاجرت اپلیکیشن ما تغییر دهند.

کلاس UserManager را مانند قطعه کد زیر بروز رسانی کنید.

```
public UserManager()
            : base(new UserStore<User>(new ApplicationDbContext()))
        {
            this.PasswordHasher = new SQLPasswordHasher();
 }
```

ایجاد صفحات جدید مدیریت کاربران

قدم بعدی ایجاد صفحاتی است که به کاربران اجازه ثبت نام و ورود را می‌دهند. صفحات قدیمی SQL Membership از کنترل هایی استفاده می‌کنند که با ASP.NET Identity سازگار نیستند. برای ساختن این صفحات جدید به این مقاله مراجعه کنید. از آنجا که در این مقاله پروژه جدید را ساخته ایم و پکیج‌های لازم را هم نصب کرده ایم، می‌توانید مستقیما به قسمت Adding Web Forms for registering users to your application بروید.

چند تغییر که باید اعمال شوند:

فایل‌های Register.aspx.cs و Login.aspx.cs از کلاس UserManager استفاده می‌کنند. این ارجاعات را با کلاس UserManager جدیدی که در پوشه Models ساختید جایگزین کنید.

همچنین ارجاعات استفاده از کلاس IdentityUser را به کلاس User که در پوشه Models ساختید تغییر دهید.

لازم است توسعه دهنده مقدار ApplicationId را برای کاربران جدید طوری تنظیم کند که با شناسه اپلیکیشن جاری تطابق داشته باشد. برای این کار می‌توانید پیش از ساختن حساب‌های کاربری جدید در فایل Register.aspx.cs ابتدا شناسه اپلیکیشن را بدست آورید و اطلاعات کاربر را بدرستی تنظیم کنید.

مثال: در فایل Register.aspx.cs متد جدیدی تعریف کنید که جدول aspnet_Applications را بررسی میکند و شناسه اپلیکیشن را بر اساس نام اپلیکیشن بدست می‌آورد.

```
private Guid GetApplicationID()
{
    using (SqlConnection connection = new
SqlConnection(ConfigurationManager.ConnectionStrings["ApplicationServices"].ConnectionString))
            {
                string queryString = "SELECT ApplicationId from aspnet_Applications WHERE
ApplicationName = '/'"; //Set application name as in database

                SqlCommand command = new SqlCommand(queryString, connection);
                command.Connection.Open();

                var reader = command.ExecuteReader();
                while (reader.Read())
                {
                    return reader.GetGuid(0);
                }

            return Guid.NewGuid();
        }
}
```

حال می‌توانید این مقدار را برای آبجکت کاربر تنظیم کنید.

```
var currentApplicationId = GetApplicationID();

User user = new User() { UserName = Username.Text,
ApplicationId=currentApplicationId, …};
```

در این مرحله می‌توانید با استفاده از اطلاعات پیشین وارد سایت شوید، یا کاربران جدیدی ثبت کنید. همچنین اطمینان حاصل کنید که کاربران پیشین در نقش‌های مورد نظر وجود دارند.

مهاجرت به ASP.NET Identity مزایا و قابلیت‌های جدیدی را به شما ارائه می‌کند. مثلا کاربران می‌توانند با استفاده از تامین کنندگان ثالثی مثل Facebook, Google, Microsoft, Twitter و غیره به سایت وارد شوند. اگر به سورس کد این مقاله مراجعه کنید خواهید دید که امکانات OAuth نیز فعال شده اند.

در این مقاله انتقال داده‌های پروفایل کاربران بررسی نشد. اما با استفاده از نکات ذکر شده می‌توانید پروفایل کاربران را هم بسادگی منتقل کنید. کافی است مدل‌های لازم را در پروژه خود تعریف کرده و با استفاده از اسکریپت‌های SQL داده‌ها را انتقال دهید.

نظرات خوانندگان

نویسنده: سمیرا قادری
تاریخ: ۱۳۹۲/۱۰/۲۳ ۱۳:۳۳

سلام ممنون از مطالب خوبتون .
زمانیکه در اپلیکیشن خودم asp.net Configuration و تب Security را انتخاب می‌نمایم با پیغام زیر مواجه می‌شوم Unable to connect to SQL Server database.
دستور زیر را هم اجرا کردم
C:\Windows\Microsoft.NET\Framework\v4.0.30319\aspnet_regsql.exe -S . -A all -d tt -U sa_tt -P asd123
در صورتیکه از login و password آن مطمئن هستم ولی مشکل حل نشد
در صورتیکه مشکل از permission هست چگونه باید آن را حل کنم
webconfig برنامه به شرح زیر است
Data Source=.;Initial Catalog=tt;User ID=sa_tt;Password=asd123
با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۳ ۱۴:۴۵

تنظیمات وب کانفیگ خودتون رو با نمونه MSDN برای تعریف membership مطابقت بدید. جزئیات قسمت‌های مختلف آن

نویسنده: حمید
تاریخ: ۱۳۹۲/۱۰/۲۳ ۲۱:۱۰

خیلی ممنون آموزش‌های مفیدتون!
فرض کنید ما یک سیستم Authentication مثل Membership داریم با 2 تا نقش. حالا می‌خواهیم نقش A به فرم F1 و بعضی از متدهاش دسترسی داشته و در فرم F2، نقش B به آن دسترسی داشته باشه. خوب تا اینجا فکر کنم پیاده سازیش راحت باشه. ولی مساله از جایی پیچیده میشه که ما بخواهیم در زمان اجرا این تنظیمات رو عوض کنیم مثلا: در فرم F1، نقش A دسترسیش به متدها و حتی به کل فرم تغییر کنه و یا اجازه دسترسی به فرم B بهش داده بشه و این شرایط رو برای چندین نقش و یا در سطح کاربر داشت توصیه چیه؟ از چه تکنولوژی و یا راهکاری میشه به این مقصود رسید.
خیلی ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۳ ۲۱:۳۷

پایه‌اش همین مسایل هست. فقط قسمتی که کار Authorization انجام میشه رو می‌توان سفارشی کرد. مثلا:

Dynamic Controller/Action Authorization in ASP.NET MVC
MVC Dynamic Authorization
ASP.NET MVC Custom Authorize Attribute with Roles Parser

نویسنده: کامران سادین
تاریخ: ۱۳۹۲/۱۱/۰۸ ۲:۱۳

ممنون از مطلب خوبتون.
من جداول خودم رو برای احراز هویت دارم آیا میشه همین جداول رو با ASP.NET Identity کار کنم؟
اگر نمیشه، این بانک ASP.NET Identity که توی SQL Server نمیسازه بانک Database.mdf رو میسازه، میشه توی SQL بسازیم از

همون ابتدای پروژه؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۸ ۹:۲۰

سیستم کارش EF Code first هست. این سیستم کدهاش گره خورده به بانک اطلاعاتی خاصی نیست. الان در این مثال رشته اتصالی به یک localdb اشاره می‌کنه. شما می‌تونید کلا این رشته و نحوه‌ی تعریف اون رو برای کار با SQL Server یا SQL CE یا هر بانک اطلاعاتی دیگری که پروایدر code first داره، تغییر بدید و استفاده کنید. (و اگر با ef code first آشنایی ندارید، کم کم در آینده نمی‌تونید با کتابخانه‌های کمکی و جانبی دات نت کار کنید)

نویسنده: کامران سادین
تاریخ: ۱۳۹۲/۱۱/۰۸ ۱۰:۲۷

مقاله ای نیز راجع به تغییر فیلدهاش هم بنویسید بد نیست دوستان علاقه دارند. با تشکر.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۸ ۱۰:۵۸

« سفارشی کردن ASP.NET Identity در MVC 5 » بیشتر در اینجا

نویسنده: مهدی
تاریخ: ۱۳۹۳/۰۱/۱۹ ۱۲:۳۴

دوست عزیز برای حل این مشکل شما باید پوشه سورس برنامه رو تغییر بدین. این مشکل به دلیل وجود خط فاصله تو آدرس (دایرکتوری) محل قرار گیری پروژه هست. اگه محل پروژه رو در جایی قرار دهید که در آدرس آن از فاصله استفاده نشده باشد به خوبی کار خواهد کرد.

عنوان: **انتشار پیش نمایش ASP.NET Identity 2.0.0-alpha1**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۳ ۱۶:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MemberShip, Visual Studio 2013, ASP.NET Identity

مایکروسافت در تاریخ 20 دسامبر 2013 پیش نمایش نسخه جدید ASP.NET Identity را معرفی کرد. تمرکز اصلی در این انتشار، رفع مشکلات نسخه 1.0 بود. امکانات جدیدی هم مانند Account Confirmation و Password Reset اضافه شده اند.

دانلود این انتشار

ASP.NET Identity را می‌توانید در قالب یک پکیج NuGet دریافت کنید. در پنجره Manage NuGet Packages می‌توانید پکیج‌های Preview را لیست کرده و گزینه مورد نظر را

نصب کنید. برای نصب پکیج‌های pre-release توسط Package Manager Console از فرامین زیر استفاده کنید.

```
Install-Package Microsoft.AspNet.Identity.EntityFramework -Version 2.0.0-alpha1 -Pre
Install-Package Microsoft.AspNet.Identity.Core -Version 2.0.0-alpha1 -Pre
Install-Package Microsoft.AspNet.Identity.OWIN -Version 2.0.0-alpha1 -Pre
```

دقت کنید که حتما از گزینه "Include Prerelease" استفاده می‌کنید. برای اطلاعات بیشتر درباره نصب پکیج‌های Pre-release لطفا به این لینک و یا این لینک مراجعه کنید.

در ادامه لیست امکانات جدید و مشکلات رفع شده را می‌خوانید.

Account Confirmation

سیستم ASP.NET Identity حالا از Account Confirmation پشتیبانی می‌کند. این یک سناریوی بسیار رایج است. در اکثر وب سایت‌های امروزی پس از ثبت نام، حتما باید ایمیل خود را تایید کنید. پیش از تایید ثبت نام قادر به انجام هیچ کاری در وب سایت نخواهید بود، یعنی نمی‌توانید Login کنید. این روش مفید است، چرا که از ایجاد حساب‌های کاربری نامعتبر (bogus) جلوگیری می‌کند. همچنین این روش برای برقراری ارتباط با کاربران هم بسیار کارآمد است. از آدرس‌های ایمیل کاربران می‌توانید در وب سایت‌های فروم، شبکه‌های اجتماعی، تجارت آنلاین و بانکداری برای اطلاع رسانی و دیگر موارد استفاده کنید.

نکته: برای ارسال ایمیل باید تنظیمات SMTP را پیکربندی کنید. مثلا می‌توانید از سرویس‌های ایمیل محبوبی مانند SendGrid استفاده کنید، که با Windows Azure براحتی یکپارچه می‌شود و از طرف توسعه دهنده اپلیکیشن هم نیاز به پیکربندی ندارد.

در مثال زیر نیاز دارید تا یک سرویس ایمیل برای ارسال ایمیل‌ها پیکربندی کنید. همچنین کاربران پیش از تایید ایمیل شان قادر به بازنشانی کلمه عبور نیستند.

Password Reset

این هم یک سناریوی رایج و استاندارد است. کاربران در صورتی که کلمه عبورشان را فراموش کنند، می‌توانند از این قابلیت برای بازنشانی آن استفاده کنند. کلمه عبور جدیدی بصورت خودکار تولید شده و برای آنها ارسال می‌شود. کاربران با استفاده از این رمز عبور جدید می‌توانند وارد سایت شوند و سپس آن را تغییر دهند.

Security Token Provider

هنگامی که کاربران کلمه عبورشان را تغییر می‌دهند، یا اطلاعات امنیتی خود را بروز رسانی می‌کنند (مثلا حذف کردن لاگین‌های خارجی مثل فیسبوک، گوگل و غیره) باید شناسه امنیتی (security token) کاربر را بازتولید کنیم و مقدار قبلی را Invalidate یا بی اعتبار سازیم. این کار بمنظور حصول اطمینان از بی اعتبار بودن تمام شناسه‌های قبلی است که توسط کلمه عبور پیشین تولید شده بودند. این قابلیت، یک لایه امنیتی بیشتر برای اپلیکیشن شما فراهم می‌کند. چرا که وقتی کاربری کلمه عبورش را تغییر بدهد از همه جا logged-out می‌شود. یعنی از تمام مرورگرهایی که برای استفاده از اپلیکیشن استفاده کرده خارج خواهد شد.

برای پیکربندی تنظیمات این قابلیت می‌توانید از فایل Startup.Auth.cs استفاده کنید. می‌توانید مشخص کنید که میان افزار OWIN cookie هر چند وقت یکبار باید شناسه امنیتی کاربران را بررسی کند. به لیست زیر دقت کنید.

```
// Enable the application to use a cookie to store information for the signed in user
// and to use a cookie to temporarily store information about a user logging in with a third party
login provider
// Configure the sign in cookie
app.UseCookieAuthentication(newCookieAuthenticationOptions {
    AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
    LoginPath = newPathString("/Account/Login"),
    Provider = newCookieAuthenticationProvider {
        OnValidateIdentity = SecurityStampValidator.OnValidateIdentity<ApplicationUserManager,
ApplicationUser>(
```

```
            validateInterval: TimeSpan.FromSeconds(5),
            regenerateIdentity: (manager, user) => user.GenerateUserIdentityAsync(manager))
    }
});
```

امکان سفارشی کردن کلیدهای اصلی Users و Roles

در نسخه 1.0 نوع فیلدهای کلید اصلی در جداول Users و Roles از نوع رشته (string) بود. این بدین معنا است که وقتی از Entity Framework و Sql Server برای ذخیره داده‌های ASP.NET Identity استفاده می‌کنیم داده‌های این فیلدها بعنوان nvarchar ذخیره می‌شوند. درباره این پیاده سازی پیش فرض در فروم هایی مانند سایت StackOverflow بسیار بحث شده است. و در آخر با در نظر گرفتن تمام بازخورد ها، تصمیم گرفته شد یک نقطه توسعه پذیری (extensibility) اضافه شود که توسط آن بتوان نوع فیلدهای اصلی را مشخص کرد. مثلا شاید بخواهید کلیدهای اصلی جداول Users و Roles از نوع int باشند. این نقطه توسعه پذیری مخصوصا هنگام مهاجرت داده‌های قبلی بسیار مفید است، مثلا ممکن است دیتابیس قبلی فیلدهای UserId را با فرمت GUID ذخیره کرده باشد.

اگر نوع فیلدهای کلید اصلی را تغییر دهید، باید کلاس‌های مورد نیاز برای Claims و Logins را هم اضافه کنید تا کلید اصلی معتبری دریافت کنند. قطعه کد زیر نمونه ای از نحوه استفاده این قابلیت برای تعریف کلیدهای int را نشان می‌دهد.

```
de Snippet
publicclassApplicationUser : IdentityUser<int, CustomUserLogin, CustomUserRole, CustomUserClaim>
{
}
publicclassCustomRole : IdentityRole<int, CustomUserRole>
{
    public CustomRole() { }
    public CustomRole(string name) { Name = name; }
}
publicclassCustomUserRole : IdentityUserRole<int> { }
publicclassCustomUserClaim : IdentityUserClaim<int> { }
publicclassCustomUserLogin : IdentityUserLogin<int> { }

publicclassApplicationDbContext : IdentityDbContext<ApplicationUser, CustomRole, int, CustomUserLogin,
CustomUserRole, CustomUserClaim>
{
}
```

پشتیبانی از IQueryable روی Users و Roles

کلاس‌های UserStore و RoleStore حالا از IQueryable پشتیبانی می‌کنند، بنابراین می‌توانید براحتی لیست کاربران و نقش‌ها را کوئری کنید.

بعنوان مثال قطعه کد زیر دریافت لیست کاربران را نشان می‌دهد. از همین روش برای دریافت لیست نقش‌ها از RoleManager می‌توانید استفاده کنید.

```
//
// GET: /Users/
public async Task<ActionResult> Index()
{
    return View(await UserManager.Users.ToListAsync());
}
```

پشتیبانی از عملیات Delete از طریق UserManager

در نسخه 1.0 اگر قصد حذف یک کاربر را داشتید، نمی‌توانستید این کار را از طریق UserManager انجام دهید. اما حالا می‌توانید مانند قطعه کد زیر عمل کنید.

```
var user = await UserManager.FindByIdAsync(id);

if (user == null)
{
    return HttpNotFound();
```

```
}
var result = await UserManager.DeleteAsync(user);
```

میان افزار UserManagerFactory

شما می‌توانید با استفاده از یک پیاده سازی Factory، وهله ای از UserManager را از OWIN context دریافت کنید. این الگو مشابه چیزی است که برای گرفتن AuthenticationManager در OWIN context استفاده می‌کنیم. این الگو همچنین روش توصیه شده برای گرفتن یک نمونه از UserManager به ازای هر درخواست در اپلیکیشن است.

قطعه کد زیر نحوه پیکربندی این میان افزار در فایل StartupAuth.cs را نشان می‌دهد.

```
// Configure the UserManager
app.UseUserManagerFactory(newUserManagerOptions<ApplicationUserManager>()
{
    DataProtectionProvider = app.GetDataProtectionProvider(),
    Provider = newUserManagerProvider<ApplicationUserManager>()
    {
        OnCreate = ApplicationUserManager.Create
    }
});
```

و برای گرفتن یک وهله از UserManager:

```
HttpContext.GetOwinContext().GetUserManager<ApplicationUserManager>();
```

میان افزار DbContextFactory

سیستم ASP.NET Identity از Entity Framework برای ذخیره داده هایش در Sql Server استفاده می‌کند. بدین منظور، ASP.NET Identity کلاس ApplicationDbContext را رفرنس می‌کند. میان افزار DbContextFactory به ازای هر درخواست در اپلیکیشن یک وهله از ApplicationDbContext را به شما تحویل می‌دهد.

می توانید پیکربندی لازم را در StartupAuth.cs انجام دهید.

```
app.UseDbContextFactory(ApplicationDbContext.Create);
```

Samples

امکانات جدید را می‌توانید در پروژه https://aspnet.codeplex.com پیدا کنید. لطفا به پوشه Identity در سورس کد مراجعه کنید. برای اطلاعاتی درباره نحوه اجرای پروژه هم فایل readme را بخوانید.

برای مستندات ASP.NET Identity 1.0 هم به http://www.asp.net/identity سر بزنید. هنوز مستنداتی برای نسخه 2.0 منتشر نشده، اما بزودی با انتشار نسخه نهایی مستندات و مثال‌های جدیدی به سایت اضافه خواهند شد.

Known Issues

در کنار قابلیت‌های جدیدی مانند Account Confirmation و Password Reset، دو خاصیت جدید به کلاس IdentityUser اضافه شده اند: 'Email' و 'IsConfirmed'. این تغییرات الگوی دیتابیسی که توسط ASP.NET Identity 1.0 ساخته شده است را تغییر می‌دهد. بروز رسانی پکیج‌ها از نسخه 1.0 به 2.0 باعث می‌شود که اپلیکیشن شما دیگر قادر به دسترسی به دیتابیس عضویت نباشد، چرا که مدل دیتابیس تغییر کرده. برای بروز رسانی الگوی دیتابیس می‌توانید از Code First Migrations استفاده کنید.

نکته: نسخه جدید به EntityFramework 6.1.0-alpha1 وابستگی دارد، که در همین تاریخ (20 دسامبر 2013) پیش نمایش شد. http://blogs.msdn.com/b/adonet/archive/2013/12/20/ef-6-1-alpha-1-available.aspx

EntityFramework 6.1.0-alpha1 بروز رسانی هایی دارد که سناریوی مهاجرت در ASP.NET Identity را تسهیل می‌کند، به همین دلیل از نسخه جدید EF استفاده شده. تیم ASP.NET هنوز باگ‌های زیادی را باید رفع کند و قابلیت‌های جدیدی را هم باید پیاده سازی کند. بنابراین پیش از نسخه نهایی RTM شاهد پیش نمایش‌های دیگری هم خواهیم بود که در ماه‌های آتی منتشر می‌شوند.

برای اطلاعات بیشتر درباره آینده ASP.NET Identity به لینک زیر سری بزنید.

https://aspnetidentity.codeplex.com/wikipage?title=Roadmap&version=1

نظرات خوانندگان

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۱۰/۳۰ ۲۰:۴۴

سلام

پیاده سازی Single Sign on در این سیستم کجا کار قرار داره؟ در واقع چطور میشه پیاده سازی ش کرد؟
در سیستم membership قبلی، اگر کلید اپلیکیشن رو در وب کانفیگ برنامه هاتون که دامین هاشون مشترک بود (در واقع ساب دامین ها)، یکسان وارد میکردی، برنامه‌ها بصورت SSO کار می‌کرد و احتیاجی به هیچ کاری نداشت. حالا در سیستم جدید همون روش جواب میده؟ برای برنامه‌های با دامین‌های متفاوت چطور؟

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۳۰ ۲۳:۷

CookieDomain رو باید تنظیم کنید.

عنوان: **تغییر فضای نام کلاس poco استفاده شده در wcf و از کار افتادن برنامه‌ی مشتری بدون دریافت پیام خطا**
نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۱۰/۲۴ ۱۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, WCF, namespace

چند وقت پیش در پروژه‌ای یک سرویس WCF داشتم که اطلاعاتی را در قالب یک کلاس poco برگشت می‌داد. اخیرا بعد از اصلاحاتی در پروژه متوجه شدم که سرویس کار نمی‌کند. هیچ خطایی هم وجود نداشت. شروع به دیباگ کردم و متوجه شدم که **سرویس برنامه اطلاعات را برگشت می‌دهد، اما برنامه‌ی مشتری تعداد اطلاعات دریافتی را صفر اعلام میکند و هیچ خطایی هم گزارش نمی‌شود.**

چون اطلاعات در قالب باینری در قسمتی از کلاس poco برگشت می‌شد، ابتدا حدسم حجم فایل بود.

اطلاعات کلاس poco:

```
public class OutgoingJob
    {
        public int Id;
        public string JobId;
        public string Subject;
        public string Reciver;
        public byte[][] Attachments;
    }
```

تنظیمات سایز ارسال و دریافت رو به حداکثر رسوندم. هیچ فایده ای نداشت. برنامه‌ی مشتری به راحتی به سرویس وصل می‌شد و با سایر متدهایی که خروجی‌های تایپ‌های اصلی مثل bool و string را برمی گرداند کار می‌کرد. فقط با متدی که لیست poco داشت، تعداد لیست اطلاعات دریافتی 0 اعلام می‌شد.

متد WCF برای برگشت اطلاعات و لاگ کردن وقایع:

```
public List<OutgoingJob> GetJobsList(int Count)
        {
            LogEvent("GetFaxsList Start...");

            List<OutgoingJob> OutgoingJobs = new List<OutgoingJob>();

           // business for fill list

            LogEvent("return job Count = " + OutgoingJobs.Count);
            return OutgoingJobs;
        }
```

```
[TestMethod]
        public void TestMethod1()
        {
            jobService ser = new jobService();
            var listjob = ser.GetJobsList(5);
            Assert.AreNotEqual(0, listjob.Count);
        }
```

لاگ‌های متد WCF، تعداد را 1 اعلام می‌کند، اما تست، نتیجه را صفر برمی‌گرداند.

بعد از کلی کلنجار با تنظیمات binding و serviceBehaviors متوجه شدم که اشکال کار یه نکته‌ی کوچک خیلی خیلی ساده است.

من هیچ تغییری در کلاس‌ها نداده بودم، اما برای مدیریت بهتر پروژه، **فضای نام کلاس‌ها را تغییر داده بودم و مسبب همه‌ی مشکلات و وقت کشی‌ها همین بود.**

راه حل هم که ساده ست. هنگامی که فضای نام کلاس‌های برگشتی را تغییر می‌دهید، حتما باید **update service reference** را در برنامه‌ی مشتری اجرا کنید تا اطلاعات سرویس بروز شود.

مشکل و راه حل خیلی ساده بود، ولی از من که خیلی وقت گرفت. امیدوارم وقت دوستان مثل من هدر نره
موفق باشید

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۰/۲۴ ۱۱:۴۱

استفاده از روش دیگر اتصال به سرور هم می‌تونه کمک کنه: مقایسه بین Proxy و ChannelFactory در WCF

نویسنده: افشار محبی
تاریخ: ۱۳۹۲/۱۰/۲۴ ۱۲:۵

بله کاملاً درسته. روش Channel Factory می‌تونه کمک خیلی خوبی به کاهش این گونه مشکلات بکند.

نویسنده: ابوالفضل رجب پور
تاریخ: ۱۳۹۲/۱۰/۲۴ ۱۲:۲۸

با تشکر از ارجاع خوبتون.
مطلب اینجا راجع به یک نکته و روش رفعش در روش اول بود. مزایا و معایب هر روش سرجای خود محفوظ

عنوان: فراخوانی متد Parent Page از User-Control
نویسنده: امیر هاشم زاده
تاریخ: ۱۳۹۲/۱۰/۲۶ ۲۳:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Delegate, User Control

در ASP.Net، ما user-control سفارشی را جهت استفاده مجدد و مستقل در صفحات ASPX ایجاد می‌کنیم. هر user-control دارای properties عمومی، متدها و یا delegateهای خاص خود است و زمانی که user-control در یک صفحه وب جاسازی (embedded) یا فرخوانی (load) می‌شود بوسیله صفحه وب قابل استفاده است.

بعد از درج user-control در صفحه وب و فراخوانی آن، ممکن است نیاز باشد مثلاً باکلیک بر روی دکمه‌ای از user-control متدی از صفحه اجرا شود. اما یک مشکل، زمانی که در حال ایجاد user-control هستید هیچ اطلاعی از صفحه ای که قرار است user-control در آن قرار بگیرد ندارید پس چگونه می‌توانیم به متدهای آن دسترسی داشته باشیم؟!

در کلاس Delegate، متدی بنام DynamicInvoke وجود دارد که برای فراخوانی (Invoke) متد اشاره شده در delegate استفاده می‌شود. ما از این متد برای صدا زدن یک متد صفحه وبی که user-control در آن قرار دارد استفاده می‌کنیم.

مثال:

```
public partial class CustomUserCtrl : System.Web.UI.UserControl
{
     private System.Delegate _delWithParam;
     private System.Delegate _delNoParam;

     // برای فراخوانی متدهایی از صفحه که دارای پارامتر هستند
     public Delegate PageMethodWithParamRef
     {
        set { _delWithParam = value; }
     }

     // برای فراخوانی متدهایی از صفحه که بدون پارامتر هستند
     public Delegate PageMethodWithNoParamRef
     {
        set { _delNoParam = value; }
     }

     protected void Page_Load(object sender, EventArgs e)
     {
     }

     protected void BtnMethodWithParam_Click(object sender, System.EventArgs e)
     {
        //Parameter to a method is being made ready
        object[] obj = new object[1];
        obj[0] = “Parameter Value” as object;
        _delWithParam.DynamicInvoke(obj);
     }

     protected void BtnMethowWithoutParam_Click(object sender, System.EventArgs e)
     {
        //Invoke a method with no parameter
        _delNoParam.DynamicInvoke();
     }
}
```

فرض کنید در user-control بالا، دو دکمه وجود دارد که متد BtnMethodWithParam_Click را به رویداد کلیک یک دکمه، و متد BtnMethowWithoutParam_Click به رویداد کلیک دکمه دیگر منتسب می‌کنیم، سپس دو عامل خصوصی (Private) را تعریف می‌کنیم و متد DynamicInvoke این عامل‌های خصوصی را در متدهای BtnMethodWithParam_Click و BtnMethowWithoutParam_Click فراخوانی می‌کنیم حال کافیست عامل‌هایی در صفحه تعریف کنیم که این عامل‌ها به متدهای مورد نظر صفحه اشاره کنند و این عامل‌های صفحه را در عامل‌های عمومی user-control قرار دهیم. در ادامه به پیاده سازی صفحه می‌پردازیم:

ابتدا دو عامل تعریف می‌کنیم:

```
public partial class _Default : System.Web.UI.Page
{
     delegate void DelMethodWithParam(string strParam);
     delegate void DelMethodWithoutParam();
```

در رویداد Page_Load، یک وهله از هر کدام از عامل‌های بالا که به متد (**توجه:** امضاء متدها با امضاء عامل‌ها یکسان است) مورد نظر ما در صفحه اشاره می‌کند ایجاد می‌کنیم:

```
protected void Page_Load(object sender, EventArgs e)
     {
        DelMethodWithParam delParam = new DelMethodWithParam(MethodWithParam);

        // عامل صفحه را به عامل عمومی تعریف شده در یوزر کنترل تخصیص می‌دهیم
        this.UserCtrl.PageMethodWithParamRef = delParam;
        DelMethodWithoutParam delNoParam = new DelMethodWithoutParam(MethodWithNoParam);

        // عامل صفحه را به عامل عمومی تعریف شده در یوزر کنترل تخصیص می‌دهیم
        this.UserCtrl.PageMethodWithNoParamRef = delNoParam;
     }
```

در زیر متدهایی خصوصی که در صفحه وجود دارند و قرار است با کلیک بر روی دکمه‌های user-control فراخوانی شوند را مشاهده می‌کنید:

```
// متد دارای پارامتری که قرار است در کنترل فراخوانی شود
     private void MethodWithParam(string strParam)
     {
        Response.Write("It has parameter: " + strParam);
     }

     // متد بدون پارامتری که قرار است در کنترل فراخوانی شود
     private void MethodWithNoParam()
     {
        Response.Write("It has no parameter.");
     }
```

و در نهایت کد پیاده سازی نهایی صفحه ما بشکل زیر خواهد شد:

```
public partial class _Default : System.Web.UI.Page
{
     delegate void DelMethodWithParam(string strParam);
     delegate void DelMethodWithoutParam();

     protected void Page_Load(object sender, EventArgs e)
     {
        DelMethodWithParam delParam = new DelMethodWithParam(MethodWithParam);

        // عامل صفحه را به عامل عمومی تعریف شده در یوزر کنترل تخصیص می‌دهیم
        this.UserCtrl.PageMethodWithParamRef = delParam;
        DelMethodWithoutParam delNoParam = new DelMethodWithoutParam(MethodWithNoParam);

        // عامل صفحه را به عامل عمومی تعریف شده در یوزر کنترل تخصیص می‌دهیم
        this.UserCtrl.PageMethodWithNoParamRef = delNoParam;
     }

     // متد دارای پارامتری که قرار است در کنترل فراخوانی شود
     private void MethodWithParam(string strParam)
     {
        Response.Write("It has parameter: " + strParam);
     }

     // متد بدون پارامتری که قرار است در کنترل فراخوانی شود
     private void MethodWithNoParam()
     {
        Response.Write("It has no parameter.");
     }
}
```

برداشتی آزاد از این مقاله .

عنوان: **شروع کار با ASP.NET Web API 2**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۰/۲۸ ۲۰:۲۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Web API, ASP.NET Web API 2

HTTP تنها برای به خدمت گرفتن صفحات وب نیست. این پروتکل همچنین پلتفرمی قدرتمند برای ساختن API هایی است که سرویس‌ها و داده را در دسترس قرار می‌دهند. این پروتکل ساده، انعطاف پذیر و در همه جا حاضر است. هر پلتفرمی که فکرش را بتوانید بکنید کتابخانه ای برای HTTP دارد، بنابراین سرویس‌های HTTP می‌توانند بازه بسیار گسترده ای از کلاینت‌ها را پوشش دهند، مانند مرورگرها، دستگاه‌های موبایل و اپلیکیشن‌های مرسوم دسکتاپ.

ASP.NET Web API فریم ورکی برای ساختن APIهای وب بر روی فریم ورک دات نت است. در این مقاله با استفاده از این فریم ورک، API وبی خواهیم ساخت که لیستی از محصولات را بر می‌گرداند. صفحه وب کلاینت، با استفاده از jQuery نتایج را نمایش خواهد داد.

یک پروژه Web API بسازید

در ویژوال استودیو 2013 پروژه جدیدی از نوع ASP.NET Web Application بسازید و نام آن را "ProductsApp" انتخاب کنید.

در دیالوگ New ASP.NET Project قالب Empty را انتخاب کنید و در قسمت "Add folders and core references for" گزینه Web API را انتخاب نمایید.

می توانید از قالب Web API هم استفاده کنید. این قالب با استفاده از ASP.NET MVC صفحات راهنمای API را خواهد ساخت. در این مقاله از قالب Empty استفاده میکنیم تا تمرکز اصلی، روی خود فریم ورک Web API باشد. بطور کلی برای استفاده از این فریم ورک لازم نیست با ASP.NET MVC آشنایی داشته باشید.

افزودن یک مدل

یک مدل (model) آبجکتی است که داده اپلیکیشن شما را معرفی می‌کند. ASP.NET Web API می‌تواند بصورت خودکار مدل شما را به JSON, XML و برخی فرمت‌های دیگر مرتب (serialize) کند، و سپس داده مرتب شده را در بدنه پیام HTTP Response بنویسد. تا وقتی که یک کلاینت بتواند فرمت مرتب سازی داده‌ها را بخواند، می‌تواند آبجکت شما را deserialize کند. اکثر کلاینت‌ها می‌توانند XML یا JSON را تفسیر کنند. بعلاوه کلاینت‌ها می‌توانند فرمت مورد نظرشان را با تنظیم Accept header در پیام HTTP Request مشخص کنند.

بگذارید تا با ساختن مدلی ساده که یک محصول (product) را معرفی میکند شروع کنیم.

کلاس جدیدی در پوشه Models ایجاد کنید.

نام کلاس را به "Product" تغییر دهید، و خواص زیر را به آن اضافه کنید.

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

افزودن یک کنترلر

در Web API کنترلرها آبجکت هایی هستند که درخواست‌های HTTP را مدیریت کرده و آنها را به اکشن متدها نگاشت می‌کنند. ما کنترلری خواهیم ساخت که می‌تواند لیستی از محصولات، یا محصولی بخصوص را بر اساس شناسه برگرداند. اگر از ASP.NET MVC استفاده کرده اید، با کنترلرها آشنا هستید. کنترلرهای Web API مشابه کنترلرهای MVC هستند، با این تفاوت که بجای ارث بری از کلاس Controller از کلاس ApiController مشتق می‌شوند.

کنترلر جدیدی در پوشه Controllers ایجاد کنید.

در دیالوگ Add Scaffold گزینه Web API Controller - Empty را انتخاب کرده و روی Add کلیک کنید.

در دیالوگ Add Controller نام کنترلر را به "ProductsController" تغییر دهید و روی Add کلیک کنید.

توجه کنید که ملزم به ساختن کنترلرهای خود در پوشه Controllers نیستید، و این روش صرفا قراردادی برای مرتب نگاه داشتن ساختار پروژه‌ها است. کنترلر ساخته شده را باز کنید و کد زیر را به آن اضافه نمایید.

```
using ProductsApp.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Net;
using System.Web.Http;

namespace ProductsApp.Controllers
{
    public class ProductsController : ApiController
    {
        Product[] products = new Product[]
        {
            new Product { Id = 1, Name = "Tomato Soup", Category = "Groceries", Price = 1 },
            new Product { Id = 2, Name = "Yo-yo", Category = "Toys", Price = 3.75M },
            new Product { Id = 3, Name = "Hammer", Category = "Hardware", Price = 16.99M }
```

```
        };

        public IEnumerable<Product> GetAllProducts()
        {
            return products;
        }

        public IHttpActionResult GetProduct(int id)
        {
            var product = products.FirstOrDefault((p) => p.Id == id);
            if (product == null)
            {
                return NotFound();
            }
            return Ok(product);
        }
}
```

برای اینکه مثال جاری را ساده نگاه داریم، محصولات مورد نظر در یک آرایه استاتیک ذخیره شده اند. مسلما در یک اپلیکیشن واقعی برای گرفتن لیست محصولات از دیتابیس یا منبع داده ای دیگر کوئری می‌گیرید.

کنترلر ما دو متد برای دریافت محصولات تعریف می‌کند:

متد GetAllProducts لیست تمام محصولات را در قالب یک IEnumerable<Product> بر می‌گرداند.
متد GetProductById سعی می‌کند محصولی را بر اساس شناسه تعیین شده پیدا کند.

همین! حالا یک Web API ساده دارید. هر یک از متدهای این کنترلر، به یک یا چند URI پاسخ می‌دهند:

Controller Method	URI
GetAllProducts	api/products/
GetProductById	api/products/ *id* /

برای اطلاعات بیشتر درباره نحوه نگاشت درخواست‌های HTTP به اکشن متدها توسط Web API به این لینک مراجعه کنید.

فراخوانی Web API با جاوا اسکریپت و jQuery

در این قسمت یک صفحه HTML خواهیم ساخت که با استفاده از AJAX متدهای Web API را فراخوانی می‌کند. برای ارسال درخواست‌های آژاکسی و بروز رسانی صفحه بمنظور نمایش نتایج دریافتی از jQuery استفاده میکنیم.

در پنجره Solution Explorer روی نام پروژه کلیک راست کرده و گزینه Add, New Item را انتخاب کنید.

در دیالوگ Add New Item قالب HTML Page را انتخاب کنید و نام فایل را به "index.html" تغییر دهید.

حال محتوای این فایل را با لیست زیر جایگزین کنید.

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
```

```
  <title>Product App</title>
</head>
<body>

  <div>
    <h2>All Products</h2>
    <ul id="products" />
  </div>
  <div>
    <h2>Search by ID</h2>
    <input type="text" id="prodId" size="5" />
    <input type="button" value="Search" onclick="find();" />
    <p id="product" />
  </div>

  <script src="http://ajax.aspnetcdn.com/ajax/jQuery/jquery-2.0.3.min.js"></script>
  <script>
    var uri = 'api/products';

    $(document).ready(function () {
      // Send an AJAX request
      $.getJSON(uri)
          .done(function (data) {
            // On success, 'data' contains a list of products.
            $.each(data, function (key, item) {
              // Add a list item for the product.
              $('<li>', { text: formatItem(item) }).appendTo($('#products'));
            });
          });
    });

    function formatItem(item) {
      return item.Name + ': $' + item.Price;
    }

    function find() {
      var id = $('#prodId').val();
      $.getJSON(uri + '/' + id)
          .done(function (data) {
            $('#product').text(formatItem(data));
          })
          .fail(function (jqXHR, textStatus, err) {
            $('#product').text('Error: ' + err);
          });
    }
  </script>
</body>
</html>
```

راه‌های مختلفی برای گرفتن jQuery وجود دارد، در این مثال از Microsoft Ajax CDN استفاده شده. می‌توانید این کتابخانه را از http://jquery.com دانلود کنید و بصورت محلی استفاده کنید. همچنین قالب پروژه‌های Web API این کتابخانه را به پروژه نیز اضافه می‌کنند.

گرفتن لیستی از محصولات

برای گرفتن لیستی از محصولات، یک درخواست HTTP GET به آدرس "/api/products" ارسال کنید.

تابع getJSON یک درخواست آژاکسی ارسال می‌کند. پاسخ دریافتی هم آرایه ای از آبجکت‌های JSON خواهد بود. تابع done در صورت موفقیت آمیز بودن درخواست، اجرا می‌شود. که در این صورت ما DOM را با اطلاعات محصولات بروز رسانی می‌کنیم.

```
$(document).ready(function () {
    // Send an AJAX request
    $.getJSON(apiUrl)
        .done(function (data) {
            // On success, 'data' contains a list of products.
            $.each(data, function (key, item) {
                // Add a list item for the product.
                $('<li>', { text: formatItem(item) }).appendTo($('#products'));
            });
        });
});
```

گرفتن محصولی مشخص

برای گرفتن یک محصول توسط شناسه (ID) آن کافی است یک درخواست HTTP GET به آدرس "/api/products/id" ارسال کنید.

```
function find() {
    var id = $('#prodId').val();
    $.getJSON(apiUrl + '/' + id)
        .done(function (data) {
            $('#product').text(formatItem(data));
        })
        .fail(function (jqXHR, textStatus, err) {
            $('#product').text('Error: ' + err);
        });
}
```

برای این کار هنوز از getJSON برای ارسال درخواست آژاکسی استفاده می‌کنیم، اما اینبار شناسه محصول را هم به آدرس درخواستی اضافه کرده ایم. پاسخ دریافتی از این درخواست، اطلاعات یک محصول با فرمت JSON است.

اجرای اپلیکیشن

اپلیکیشن را با F5 اجرا کنید. صفحه وب باز شده باید چیزی مشابه تصویر زیر باشد.

برای گرفتن محصولی مشخص، شناسه آن را وارد کنید و روی Search کلیک کنید.

اگر شناسه نامعتبری وارد کنید، سرور یک خطای HTTP بر می‌گرداند.

استفاده از F12 برای مشاهده درخواست‌ها و پاسخ ها

هنگام کار با سرویس‌های HTTP، مشاهده‌ی درخواست‌های ارسال شده و پاسخ‌های دریافتی بسیار مفید است. برای اینکار می‌توانید از ابزار توسعه دهندگان وب استفاده کنید، که اکثر مرورگرهای مدرن، پیاده سازی خودشان را دارند. در اینترنت اکسپلورر می‌توانید با F12 به این ابزار دسترسی پیدا کنید. به برگه Network بروید و روی Start Capturing کلیک کنید. حالا صفحه وب را مجددا بارگذاری (reload) کنید. در این مرحله اینترنت اکسپلورر ترافیک HTTP بین مرورگر و سرور را تسخیر می‌کند. می‌توانید تمام ترافیک HTTP روی صفحه جاری را مشاهده کنید.

F12

File Find Disable View Images Cache Tools Validate | Browser Mode: IE9

HTML CSS Console Script Profiler Network

Stop capturing | Go to detailed view

URL	Met...	Result	Type	Received	Taken
http://localhost:26481/	GET	200	text/html	1.66 KB	78 ms
/Content/css?v=ji3nO1pdg6VLv3CV...	GET	200	text/css	0.89 KB	47 ms
/Content/themes/base/css?v=UM62...	GET	200	text/css	24.73 KB	63 ms
/Scripts/js?v=4h5lPNUsLiFoa0vqrItj...	GET	200	text/javascript	325.88...	62 ms
/Scripts/jquery-1.6.2.min.js	GET	304	application/x-ja...	176 B	47 ms
/api/products/	GET	200	application/json	398 B	281 m

به دنبال آدرس نسبی "/api/products" بگردید و آن را انتخاب کنید. سپس روی Go to detailed view کلیک کنید تا جزئیات ترافیک را مشاهده کنید. در نمای جزئیات، می‌توانید headerها و بدنه درخواست‌ها و پاسخ‌ها را ببینید. مثلا اگر روی برگه Request headers کلیک کنید، خواهید دید که اپلیکیشن ما در Accept header داده‌ها را با فرمت "application/json" درخواست کرده است.

اگر روی برگه Response body کلیک کنید، می‌توانید ببینید چگونه لیست محصولات با فرمت JSON سریال شده است. همانطور که گفته شده مرورگرهای دیگر هم قابلیت‌های مشابهی دارند. یک ابزار مفید دیگر Fiddler است. با استفاده از این ابزار می‌توانید تمام ترافیک HTTP خود را مانیتور کرده، و همچنین درخواست‌های جدیدی بسازید که این امر کنترل کاملی روی HTTP headers به شما می‌دهد.

قدم‌های بعدی

برای یک مثال کامل از سرویس‌های HTTP که از عملیات POST,PUT و DELETE پشتیبانی می‌کند به این لینک مراجعه کنید.

برای اطلاعات بیشتر درباره طراحی واکنش گرا در کنار سرویس‌های HTTP به این لینک مراجعه کنید، که اپلیکیشن‌های تک صفحه ای (SPA) را بررسی می‌کند.

نظرات خوانندگان

نویسنده: پوریا منفرد
تاریخ: ۱۳۹۳/۰۳/۰۵ ۱:۲۶

من یه سوالی برام پیش اومده اینه که
میشه از Web API برای پروژه‌های بزرگ مبتنی بر روی HTTP استفاده کرد؟
منظورم از پروژه‌های بزرگ یعنی Request هایی که شاید اطلاعات برگشتی مثلا بیش از 1000 رکورد باشه
آیا شدنیه؟
یعنی منبع داده بتونه بوسیله Web API عملیات‌های Crud رو بر روی بستر اینترنت برای پروژه‌های این چنینی که امکان واکشی اطلاعات بیشمار و ورود اطلاعات همزمان بوسیله کاربرهای مختلف با دیوایس‌های مختلف وجود داره رو ارائه بده؟

نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۳/۰۳/۰۷ ۰:۷

بله. به طور کلی، هر پلتفرمی که دارای کتابخانه ای جهت کار با سرویس‌های Http است می‌تواند از سرویس‌های Asp.Net WebApi استفاده نماید.
اما در هنگام پیاده سازی پروژه‌های مقیاس بزرگ حتما به طراحی زیر ساخت توجه ویژه ای داشته باشید. اگر کتاب‌های
Designing Evolvable Web Api With Asp.Net یا
Pro Asp.Net Web Api : Http Web Service In Asp.Net را مطالعه نکردید بهتون پیشنهاد می‌کنم قبل از شروع به کار حتما نگاهی به آن‌ها بیندازید.
در همین رابطه:
»مقایسه بین امکانات Web Api و WCF

عنوان: **استفاده از Web API در ASP.NET Web Forms**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۲ ۲۳:۵۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, ASP.NET Web API, Web API, ASP.NET Web Forms

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

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

اضافه کردن یک کنترلر Web API که از کلاس **ApiController** مشتق می‌شود.
اضافه کردن مسیرهای جدید به متد **Application_Start** .

یک پروژه Web Forms بسازید

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

کنترلر و مدل اپلیکیشن را ایجاد کنید

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

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

```
}
```

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

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

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

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

    public class ProductsController : ApiController
    {

        Product[] products = new Product[]
        {
            new Product { Id = 1, Name = "Tomato Soup", Category = "Groceries", Price = 1 },
            new Product { Id = 2, Name = "Yo-yo", Category = "Toys", Price = 3.75M },
            new Product { Id = 3, Name = "Hammer", Category = "Hardware", Price = 16.99M }
        };

        public IEnumerable<Product> GetAllProducts()
        {
            return products;
        }

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

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

```
    }
}
```

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

اطلاعات مسیریابی را اضافه کنید

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

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

حال کد زیر را به متد Application_Start اضافه کنید.

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

برای اطلاعات بیشتر درباره مسیریابی در Web API به این لینک مراجعه کنید.

دریافت اطلاعات بصورت آژاکسی در کلاینت

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

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

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

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

حال در قسمت HeaderContent کتابخانه jQuery را ارجاع دهید.

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

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

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

زیر تگ jQuery اسکریپت زیر را اضافه کنید.

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

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

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

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

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

http://localh...
Home Page - My AS...
your logo here
Register
Log in
Home
About
Contact
Products
Name
Price
Tomato Soup
1
Yo-yo
3.75
Hammer
16.99
© 2012 - My ASP.NET Application

نظرات خوانندگان

نویسنده: ارشیا
تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۳:۲۷

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

نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۵ ۱۹:۵۰

می تونید از جاوا اسکریپت و Remote Validation استفاده کنید.

نویسنده: ارشیا
تاریخ: ۱۳۹۲/۱۱/۰۶ ۱۱:۲۲

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

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۰۶ ۱۲:۵

مفاهیم اعتبارسنجی در MVC با Web Api تقریبا یکی است.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۲/۱۵ ۱۶:۴۵

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

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۵ ۱۷:۲۵

- بله. گروه Web API و EF را در سایت پیگیری کنید.
- Web API یک بحث سمت سرور است. به آن به زبان ساده به چشم یک وب سرویس مدرن نگاه کنید. برای نمونه بجای وب‌متدهای استاتیک صفحات aspx یا فایل‌های ashx یا asmx و حتی سرویس‌های WCF از نوع REST و امثال آن، بهتر است از Web API استفاده کنید.
- برای نمونه پایه مباحثی مانند Forms Authentication در اینجا هم کاربرد دارد (البته این یک نمونه است).
- برای کار با Web API الزاما نیازی به ASP.NET ندارید (نه وب فرم‌ها و نه MVC)؛ به هیچکدام از نگارش‌های آن. سمت کاربر آن AngularJS و سمت سرور آن Web API باشد. کار می‌کند. (اهمیت این مساله در اینجا است که الان می‌شود یک فریم ورک جدید توسعه‌ی برنامه‌های وب را کاملا مستقل از وب فرم‌ها و MVC طراحی کرد)

عنوان: **ایجاد صفحات راهنما برای ASP.NET Web API**
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۲/۱۱/۰۳ ۰:۵۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, ASP.NET Web API, Web API

وقتی یک Web API می‌سازید بهتر است صفحات راهنمایی هم برای آن در نظر بگیرید، تا توسعه دهندگان بدانند چگونه باید سرویس شما را فراخوانی و استفاده کنند. گرچه می‌توانید مستندات را بصورت دستی ایجاد کنید، اما بهتر است تا جایی که ممکن است آنها را بصورت خودکار تولید نمایید.

بدین منظور فریم ورک ASP.NET Web API کتابخانه ای برای تولید خودکار صفحات راهنما در زمان اجرا (run-time) فراهم کرده است.

ایجاد صفحات راهنمای API

برای شروع ابتدا ابزار ASP.NET and Web Tools 2012.2 Update را نصب کنید. اگر از ویژوال استودیو 2013 استفاده می‌کنید این ابزار بصورت خودکار نصب شده است. این ابزار صفحات راهنما را به قالب پروژه‌های ASP.NET Web API اضافه می‌کند.

یک پروژه جدید از نوع ASP.NET MVC Application بسازید و قالب Web API را برای آن انتخاب کنید. این قالب پروژه کنترلری بنام ValuesController را بصورت خودکار برای شما ایجاد می‌کند. همچنین صفحات راهنمای API هم برای شما ساخته می‌شوند. تمام کد مربوط به صفحات راهنما در قسمت Areas قرار دارند.

اگر اپلیکیشن را اجرا کنید خواهید دید که صفحه اصلی لینکی به صفحه راهنمای API دارد. از صفحه اصلی، مسیر تقریبی Help/ خواهد بود.

این لینک شما را به یک صفحه خلاصه (summary) هدایت می‌کند.

نمای این صفحه در مسیر Areas/HelpPage/Views/Help/Index.cshtml قرار دارد. می‌توانید این نما را ویرایش کنید و مثلا قالب، عنوان، استایل‌ها و دیگر موارد را تغییر دهید.

بخش اصلی این صفحه متشکل از جدولی است که APIها را بر اساس کنترلر طبقه بندی می‌کند. مقادیر این جدول بصورت خودکار و توسط اینترفیس **IApiExplorer** تولید می‌شوند. در ادامه مقاله بیشتر درباره این اینترفیس صحبت خواهیم کرد. اگر کنترلر جدیدی به API خود اضافه کنید، این جدول بصورت خودکار در زمان اجرا بروز رسانی خواهد شد.

ستون "API" متد HTTP و آدرس نسبی را لیست می‌کند. ستون "Documentation" مستندات هر API را نمایش می‌دهد. مقادیر این ستون در ابتدا تنها placeholder-text است. در ادامه مقاله خواهید دید چگونه می‌توان از توضیحات XML برای تولید مستندات استفاده کرد.

هر API لینکی به یک صفحه جزئیات دارد، که در آن اطلاعات بیشتری درباره آن قابل مشاهده است. معمولا مثالی از بدنه‌های درخواست و پاسخ هم ارائه می‌شود.

GET api/Values

Documentation for 'Get'.

Response Information

Response body formats

application/json, text/json

Sample:

```
[
  "sample string 1",
  "sample string 2",
  "sample string 3"
]
```

application/xml, text/xml

Sample:

```
<ArrayOfstring xmlns:i="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://schemas.microsoft.com/2003/10/Serialization/Arrays">
  <string>sample string 1</string>
  <string>sample string 2</string>
  <string>sample string 3</string>
</ArrayOfstring>
```

افزودن صفحات راهنما به پروژه ای قدیمی

می توانید با استفاده از NuGet Package Manager صفحات راهنمای خود را به پروژه‌های قدیمی هم اضافه کنید. این گزینه مخصوصا هنگامی مفید است که با پروژه ای کار می‌کنید که قالب آن Web API نیست.

از منوی Tools گزینه‌های Library Package Manager, Package Manager Console را انتخاب کنید. در پنجره Package Manager Console فرمان زیر را وارد کنید.

```
Install-Package Microsoft.AspNet.WebApi.HelpPage
```

این پکیج اسمبلی‌های لازم برای صفحات راهنما را به پروژه اضافه می‌کند و نماهای MVC را در مسیر Areas/HelpPage می‌سازد.

اضافه کردن لینکی به صفحات راهنما باید بصورت دستی انجام شود. برای اضافه کردن این لینک به یک نمای Razor از کدی مانند لیست زیر استفاده کنید.

```
@Html.ActionLink("API", "Index", "Help", new { area = "" }, null)
```

همانطور که مشاهده می‌کنید مسیر نسبی صفحات راهنما "/Help" می‌باشد. همچنین اطمینان حاصل کنید که ناحیه‌ها (Areas) بدرستی رجیستر می‌شوند. فایل Global.asax را باز کنید و کد زیر را در صورتی که وجود ندارد اضافه کنید.

```
protected void Application_Start()
{
    // Add this code, if not present.
    AreaRegistration.RegisterAllAreas();

    // ...
}
```

افزودن مستندات API

بصورت پیش فرض صفحات راهنما از placeholder-text برای مستندات استفاده می‌کنند. می‌توانید برای ساختن مستندات از توضیحات XML استفاده کنید. برای فعال سازی این قابلیت فایل Areas/HelpPage/App_Start/HelpPageConfig.cs را باز کنید و خط زیر را از حالت کامنت درآورید:

```
config.SetDocumentationProvider(new XmlDocumentationProvider(
    HttpContext.Current.Server.MapPath("~/App_Data/XmlDocument.xml")));
```

حال روی نام پروژه کلیک راست کنید و **Properties** را انتخاب کنید. در پنجره باز شده قسمت **Build** را کلیک کنید.

زیر قسمت **Output** گزینه **XML documentation file** را تیک بزنید و در فیلد روبروی آن مقدار "App_Data/XmlDocument.xml" را وارد کنید.

حال کنترلر ValuesController را از مسیر /Controllers/ValuesController.cs باز کنید و یک سری توضیحات XML به متدهای آن اضافه کنید. بعنوان مثال:

```
/// <summary>
/// Gets some very important data from the server.
/// </summary>
public IEnumerable<string> Get()
{
    return new string[] { "value1", "value2" };
}

/// <summary>
/// Looks up some data by ID.
/// </summary>
/// <param name="id">The ID of the data.</param>
public string Get(int id)
{
    return "value";
}
```

اپلیکیشن را مجددا اجرا کنید و به صفحات راهنما بروید. حالا مستندات API شما باید تولید شده و نمایش داده شوند.

API	Description
GET api/Values	Gets some very important data from the server.
GET api/Values/{id}	Looks up some data by ID.

صفحات راهنما مستندات شما را در زمان اجرا از توضیحات XML استخراج می‌کنند. دقت کنید که هنگام توزیع اپلیکیشن، فایل XML را هم منتشر کنید.

توضیحات تکمیلی

صفحات راهنما توسط کلاس **ApiExplorer** تولید می‌شوند، که جزئی از فریم ورک ASP.NET Web API است. به ازای هر API این کلاس یک **ApiDescription** دارد که توضیحات لازم را در بر می‌گیرد. در اینجا منظور از "API" ترکیبی از متدهای HTTP و مسیرهای نسبی است. بعنوان مثال لیست زیر تعدادی API را نمایش می‌دهد:

GET /api/products

GET /api/products/{id}

POST /api/products

اگر اکشن‌های کنترلر از متدهای متعددی پشتیبانی کنند، ApiExplorer هر متد را بعنوان یک API مجزا در نظر خواهد گرفت. برای مخفی کردن یک API از ApiExplorer کافی است خاصیت **ApiExplorerSettings** را به اکشن مورد نظر اضافه کنید و مقدار خاصیت *IgnoreApi* آن را به true تنظیم نمایید.

```
[ApiExplorerSettings(IgnoreApi=true)]
public HttpResponseMessage Get(int id) {  }
```

همچنین می‌توانید این خاصیت را به کنترلرها اضافه کنید تا تمام کنترلر از ApiExplorer مخفی شود.

کلاس ApiExplorer متن مستندات را توسط اینترفیس **IDocumentationProvider** دریافت می‌کند. کد مربوطه در مسیر /Areas/HelpPage/XmlDocumentation.cs قرار دارد. همانطور که گفته شد مقادیر مورد نظر از توضیحات XML استخراج می‌شوند. نکته جالب آنکه می‌توانید با پیاده سازی این اینترفیس مستندات خود را از منبع دیگری استخراج کنید. برای اینکار باید متد الحاقی **SetDocumentationProvider** را هم فراخوانی کنید، که در **HelpPageConfigurationExtensions** تعریف شده است.

کلاس **ApiExplorer** بصورت خودکار اینترفیس IDocumentationProvider را فراخوانی می‌کند تا مستندات APIها را دریافت کند. سپس مقادیر دریافت شده را در خاصیت **Documentation** ذخیره می‌کند. این خاصیت روی آبجکت‌های **ApiDescription** و **ApiParameterDescription** تعریف شده است.

مطالعه بیشتر

Adding a simple Test Client to ASP.NET Web API Help Page

Making ASP.NET Web API Help Page work on self-hosted services

Design-time generation of help page (or client) for ASP.NET Web API

Advanced Help Page customizations

نظرات خوانندگان

نویسنده: سعید شیرزادیان
تاریخ: ۱۳۹۲/۱۱/۱۸ ۱۹:۵۵

سلام؛ می‌خواستم بدونم قابلیت فوق نیز بر روی پروژه‌های asp.net وب فرمز نیز فعال می‌گردد؟ با تشکر

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۸ ۲۰:۳۵

Enabling ASP.NET Web API Help Pages for ASP.NET Web Forms Applications

عنوان: **Url Routing در ASP.Net WebForms**
نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۰:۳۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, SEO, FriendlyUrls, UrlRewrite, routing

داشتن Urlهای تمیز و با معنا یکی از ویژگی‌های یک سایت خوب هست و بهترین روش برای بازنویسی URLها یا همان Url Routing، استفاده از امکانات توکار خود ASP.Net برای این کار است. در یک مثال کوچک برای WebFormها این کار را بررسی خواهیم کرد.

ابتدا باید فضای نام System.Web.Routing را در فایل Global.asax اضافه کنیم .

سپس توسط RouteTable.Routes.MapPageRoute در Application_Start میتوانیم الگوهای تبدیل Urlهای خود را بنویسیم.

```
protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.MapPageRoute("Product", "Product/{Name}", "~/Product.aspx");
        }
```

تابع RouteTable.Routes.MapPageRoute سه ورودی می‌گیرد. ورودی اول یک نام برای الگوی تبدیل ماست که در مثال بالا، "Product" نام گرفته است. ورودی دوم، الگوی تبدیل ماست که می‌گوید Url هایی به شکل Product/{Name} را به صفحه‌ی Product.aspx بفرست و در صفحه‌ی Product.aspx میتوانیم مقدار {Name} را توسط Page.RouteData.Values["Name"] بدست بیاوریم . ورودی سوم نیز صفحه‌ی مقصد می‌باشد.

با این الگو میتوانیم لینک هایی مثل "Product/لبتاب" ایجاد کنیم و در صفحه‌ای که میخواهیم اطلاعات محصولمان را نشان دهیم از Page.RouteData.Values["Name"] که همان مقدار لبتاب را بر میگرداند، استفاده کنیم.

یک مثال واقعی‌تر؛ مثالی که در URLهای مقالات همین سایت هم از آن استفاده شده البته از نوع MVC .

```
RouteTable.Routes.MapPageRoute("PostDetail", "Post/{pi}/{pt}", "~/PostShow.aspx");
```

این الگو یو آر الی مثل "Post/2/این-یک-تست-است" را به صفحه‌ی PostShow.aspx میفرستد و در صفحه‌ی مقصد میتوانیم توسط Page.RouteData.Values["pi"] که منظور از pi همون PostId یا کد مقاله ما است، خوانده و نمایش دهیم. دلیل ارسال قسمت {pt} یا همان "این-یک-تست-است" می‌باشد و از pt منظورمان PostTitile یا عنوان مقاله‌است که در SEO سایت تاثیر زیادی دارد.

نحوه استفاده از این تبدیل هم اینطور می‌تواند باشد.

ساخت لینک :

```
<a href='<%# string.Format("/Post/{0}/{1}",Eval("PostID"),Eval("PostTitle").ToString().Replace(" ","-
")) %>'>
```

وقتی روی لینک کلیک می‌شود، به طور مثال چنین آدرسی در قسمت آدرس مرورگر ظاهر می‌شود. دلیل استفاده از Replace هم برای زیباتر شدن Url هست. چون فضای خالی در Url زیبا نیست.

```
Post/12/چگونه-طراحان-وب-به-جهنم-می-روند!؟
```

و در صفحه‌ی PostShow.aspx به این نحو میتوانیم اطلاعات مقاله‌ی مورد نظر را بدست بیاریم:

```
int pi = int.Parse(Page.RouteData.Values["pi"].ToString());
Posts post = (from p in con.Posts where p.PostID == pi select p).FirstOrDefault();
```

یا حتی برای نشان دادن اطلاعات از این روش استفاده کنیم:

```
<asp:EntityDataSource ID="EntityDataSource1" runat="server"
        AutoGenerateWhereClause="True" ConnectionString="name=WebWorkEntities"
        DefaultContainerName="WebWorkEntities" EnableFlattening="False"
```

```
        EntitySetName="Posts" EntityTypeFilter="Posts" Where="" Select="">
        <WhereParameters>
            <asp:RouteParameter Name="PostID" RouteKey="pi" DbType="Int32"
                DefaultValue="0" />
        </WhereParameters>
    </asp:EntityDataSource>
```

منظورم استفاده از asp:RouteParameter در <WhereParameters> برای انتخاب مقاله است.

و برای زیباتر شدن کد نویسی، بهتر است الگوهای تبدیل را در یک تابع جدا بنویسیم:

```
protected void Application_Start(object sender, EventArgs e)
        {
            RoutingSite(RouteTable.Routes);
        }

        public static void RoutingSite(RouteCollection route)
        {
            route.MapPageRoute("PostDetail", "Post/{pi}/{pt}", "~/PostShow.aspx");
            route.MapPageRoute("RouteAbout", "About-Me", "~/About.aspx");
        }
```

مطالعه بیشتر ASP.NET 4.0 Web forms URL Routing

نظرات خوانندگان

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۰:۴۶

با تشکر از شما. تنها حذف فضای خالی برای تولید اصطلاحا slug کافی نیست. متد **GenerateSlug** مطلب « بهینه سازی برنامه‌های وب ASP.NET برای موتورهای جستجو (SEO) » بهتر است مدنظر باشد.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۰ ۱۶:۲۰

یک نمونه برنامه برای مثال Test-Routing.zip

نویسنده: مائده
تاریخ: ۱۳۹۲/۱۱/۱۴ ۱۹:۳۰

سوال من در مورد وب فرم هایی که توی فولدر هستند هنوز پابرجاست
شکل آدرس دهی این وب فرم‌ها چطوری میشه؟

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۴ ۱۹:۵۲

ممکنه بیشتر توضیح بدید! منظورتونو از فرم هایی که تو فولدر هستند متوجه نمیشم. مثلا فرم‌های قسمت مدیریتی رو میگید ، پوشه Admin ؟

نویسنده: مائده
تاریخ: ۱۳۹۲/۱۱/۱۴ ۲۰:۲۹

بله مثل همون پوشه ادمین
من کلا فرم هام رو توی یک پوشه جدا گذاشتم به اسم WebForms
حالا برای آدرس دهی اگه مثلا بنویسم Details/~ جواب نمیده اگه به صورت WebForms/Details/~ هم بنویسم بازم جواب نمیده!

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۴ ۲۱:۶

چرا جواب میده ، باید اینجوری بنویسید .

```
RouteTable.Routes.MapPageRoute("Folder", "ShowFolder/{Id}/{Name}", "~/Admin/Show.aspx");
```

در نمونه برنامه یه فولدر ساختم و کاری که خواستیدو انجام دادم. Test-Folder.zip

نویسنده: مائده
تاریخ: ۱۳۹۲/۱۱/۱۵ ۱۳:۸

ممنون
منم دقیقا همینطوری نوشتم ولی بازم ارور 404 میده
الآن امتحان کردم در این حالتی که شما نوشتید جواب میده اما اگه کل فرم‌ها توی پوشه باشه جواب نمیده.
شما هم اگه اون فرم Default رو بذارید توی پوشه ادمین می‌بینید که جواب نمیده!
و یه سوال: چی کار کردید که وقتی برنامه اجرا میشه اسم صفحه اول (Default) رو توی آدرس بار نشون نمیده؟

(پ.ن: همینطوری تغییرات کدها رو هم بنویسید متوجه میشم نیازی به نمونه برنامه نیست. با تشکر)

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۵ ۱۴:۳۷

زمانیکه فرم default رو در یک پوشه‌ی دیگر گذاشتید، برای اون یک route جدا تعریف کردید؟ الان آیا برنامه اطلاع داره زمانیکه به مسیر ریشه سایت وارد شد باید به کجا هدایت بشه؟

```
routes.MapPageRoute("defaultRoute", "{*value}", "~/folder/default.aspx");
```

نویسنده: رضا منصوری
تاریخ: ۱۳۹۲/۱۱/۱۵ ۱۷:۴۸

باید برای تعریف لینک‌ها اینگونه عمل کنید.

```
<a href='<%# string.Format("../ShowFolder/{0}/{1}", Eval("Id") ,Eval("Name").ToString().Replace(" ", "-
")) %>' ><%# Eval("Name") %></a>
```

.. باعث میشه یک فولدر به عقب برگرده و اینجوری درست کار میکنه.
در مورد اجرا شدن صفحه default هم دوستمون اشاره کردند.
بازم نمونه برنامه ضمیمه کردم . فرم default رو تو پوشه‌ی ادمین بردم. وقتی برنامه هم اجرا میشه صفحه default در داخل پوشه‌ی ادمینو نشون میده. Test-Folder-2.zip

نویسنده: مائده
تاریخ: ۱۳۹۲/۱۱/۱۶ ۲۲:۳۵

بازم ممنون. فقط یه سوال دیگه. فرض کنید من یه چیزی به آدرس اصلی صفحه اضافه کردم. حالا می‌خوام به این آدرس جدید دوباره یه چیز دیگه اضافه کنم. باید چی کار کنم؟ اینجا وقتی یک یا چند تا چک لیست انتخاب میشه یا مثلا چیزی از لیست باکس‌ها انتخاب میشه عبارت مربوطه اش به آدرس اضافه میشه. چطوری این کارو کرده؟ اینجوری که آدرس ثابت نیست و می‌تونه حالت‌های مختلفی داشته باشه! یا مثلا وقتی یک مورد انتخاب شده حذف میشه چطوری عبارت مربوطه اش از وسط آدرس حذف میشه؟!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۷ ۰:۴۱

این نوع routing خاص سمت کلاینت برنامه‌های تک صفحه‌ای وب را با استفاده از کتابخانه‌ای مانند path.js می‌شود انجام داد.
مثال‌های آن‌را به همراه سورس صفحات آن بررسی کنید.
نمونه آن‌را در مطلب « پیاده سازی دکمه «بیشتر» یا «اسکرول نامحدود» به کمک jQuery در ASP.NET MVC » (کامنت آخر آن) پیاده سازی شده می‌توانید مشاهده و دریافت کنید. از این قابلیت در سایت جاری در حال استفاده است. مثلا اگر به قسمت مطالب در بالای صفحه مراجعه کنید، چنین آدرسی قابل مشاهده است:
http://www.dotnettips.info/postsarchive **#/page/1/date/desc**
در پایین صفحه اگر دراپ داون‌های مرتب سازی را تغییر دهید، نام فیلد یا صعودی و نزولی بودن آدرس تغییر می‌کنند. شماره صفحه نیز با هر بار کلیک بر روی دکمه‌ی بیشتر یکی اضافه می‌شود. همچنین این آدرس‌ها را می‌شود ذخیره و عینا بازیابی کرد.

نویسنده: ابراهیم
تاریخ: ۱۳۹۲/۱۱/۱۷ ۱۲:۱۷

سلام؛ خیلی ممنون از مطلب مفیدتون.
من از دستورات route به شکل زیر استفاده میکنم

```
RouteTable.Routes.MapPageRoute("Post", "Post/{Id}/{title}", "/~showpost.aspx");
```

این دستوارت به خوبی کار میکنند و مشکلی هم نیست. اما مسئله ای که وجود داره اینه که وقتی کاربر بخواد url رو دستکاری کنه و مثلا عدد id روتغییر بده، title در url تغییر نمیکنه. من سایت‌های زیادی رو دیدم که این ایراد رو دارن. بسیاری از سایت‌های خبری. اما سایت جاری چنین مشکلی نداره. یعنی وقتی شما id رو در url تغییر میدید، عنوان پست در url هم تغییر میکنه. لطفا در مورد چگونگی پیاده سازی این روش توضیح بدید.
متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۷ ۱۲:۳۸

```
var id = Page.RouteData.Values["id"];
var title = Page.RouteData.Values["title"];

var blogPost = _postsService.FindPost(id);
if(blogPost == null)
{
  Response.RedirectPermanent("~/notfound");
}

if(blogPost.Title != title)
{
  Response.RedirectPermanent("~/post/" + id + "/" + blogPost.Title);
}
```

سیستم مسیریابی، اطلاعات id و title وارد شده توسط کاربر را به شما می‌دهد. زمانیکه می‌خواهید مطلب را نمایش دهید، این title دریافتی را با title واقعی مطلب مانند کدهای فوق مقایسه کنید. (البته بهتر است از GenerateSlug استفاده کنید و عناوین نهایی را بر این اساس مقایسه و اصلاح کنید)

نویسنده: ابراهیم
تاریخ: ۱۳۹۲/۱۱/۱۷ ۲۳:۳۶

سلام. سوال دیگه ای هم داشتم.
چطور جلوی دسترسی کاربران به آدرس اصلی فایل رو بگیریم؟
مثلا در مثال زیر :

```
route.MapPageRoute("PostDetail", "Post/{pi}/{pt}", "~/PostShow.aspx");
```

کاربر نتونه به فایل postshow.aspx به صورت مستقیم دسترسی داشته باشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۸ ۱:۲۷

در ASP.NET MVC برای اینکه کسی نتواند فایل‌های View را مستقیما فراخوانی کند، دسترسی به آن‌ها در فایل web.config کلا بسته شده :

```
<system.web>
    <httpHandlers>
      <remove verb="*" path="*.aspx" />
      <add path="*.aspx" verb="*" type="System.Web.HttpNotFoundHandler" />
    </httpHandlers>
  </system.web>

  <system.webServer>
    <validation validateIntegratedModeConfiguration="false" />
    <handlers>
      <add name="BlockViewHandler" path="*.aspx" verb="*" preCondition="integratedMode"
```

```
type="System.Web.HttpNotFoundHandler" />
    </handlers>
  </system.webServer>
```

نویسنده: ابراهیم
تاریخ: ۱۳۹۲/۱۱/۱۸ ۲:۵

جسارتا منظورم asp.net webform بود

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۸ ۲:۸

هدفم طرح یک ایده بود. امتحان کردید؟ (HttpNotFoundHandler مشترک است بین MVC و وب فرم‌ها)

نویسنده: ابراهیم
تاریخ: ۱۳۹۲/۱۱/۱۸ ۲:۱۳

نکته ای که الان یادم اومد اینه که در برنامه من، کاربران به صفحات اصلی سایت بصورت routing دسترسی دارند و در قسمت داشبورد به صورت ادرس مستقیم.
چطور میشه در webform این دو حالت رو در یک سایت از هم تفکیک کرد؟
متشکرم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۸ ۲:۱۷

فایل وب کانفیگی را که دسترسی به aspxها را not found اعلام می‌کند، داخل همان پوشه مدنظر قرار دهید. (در MVC هم دقیقا به همین نحو عمل شده‌است. یک فایل web.config دومی را در پوشه views قرار داده‌اند)

نویسنده: ابراهیم
تاریخ: ۱۳۹۲/۱۱/۱۹ ۲:۳

سلام
دستورات ذکر شده رو در فایل web.config قرار دادم. دسترسی به فایلهای aspx قطع شد و فقط میشد از طریق routing دسترسی داشت. اما مشکلی که بوجود اومد این بود که وقتی نام سایت رو در مرورگر وارد میکردم، خطای زیر رو دریافت کردم

```
The resource cannot be found.

Description: HTTP 404. The resource you are looking for (or one of its dependencies) could have been removed, had its name changed, or is temporarily unavailable.  Please review the following URL and make sure that it is spelled correctly.

Requested URL: /Default.aspx
```

به ذهنم رسید که در فایل global دستور زیر رو وارد کنم

```
routes.MapPageRoute("Default", "", "~/default.aspx", False)
```

وقتی این کارو کردم دیدم که مشکل دسترسی به فایل default.aspx حل شد اما دسترسی مستقیم به فایلهای aspx که در فولدرهای دیگر بودند دچار مشکل شد در وب جستجو کردم و به لینکهای روبرو رسیدم (+)
اما نتیجه ای نداد.

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۹ ۱۰:۱۱

دستورات not found handler رو در web.config ریشه اصلی نوشتید یا در یک وب کانفیگ جدید در پوشه‌ای که می‌خواهید؟ در ریشه اصلی روی تمام زیر پوشه‌ها به علت ارث بری تاثیر دارد.

نویسنده: ابراهیم
تاریخ: ۱۳۹۲/۱۱/۱۹ ۲۱:۴۳

چون اون فایلها در ریشه سایت هستند، بنابراین در web.config ریشه اصلی نوشتم
چه کاری میتونم بکنم؟

نویسنده: زیلایی
تاریخ: ۱۳۹۳/۰۲/۳۰ ۱۵:۹

سلام؛ تمام کارهایی که شما نوشتید رو نوشتم و انجام دادم. کار میکنه ولی دو تا مشکل داره:
1- استیل‌های صفحه رو بار گذاری نمیکنه. یعنی cssهایی که نوشتم و در پوشه ای به نام css ذخیره کردم رو نمیاره. البته من از مستر استفاده میکنم. استایل‌های مستر میاد اما استایل‌های شخصی اون صفحه نه.
2-مسیر سایتم دارای یک اسم به نام publicity است:

```
..../www.site.com/publicity
```

که وقتی آدرس‌ها رو درست میکنم این شکل میشه:
www.site.com/post/1325 و publicity حذف میشه از آدرس که باعث میشه برنامه ارور بده که همچین آدرسی نیست .

حالا باید چکار کنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۳۰ ۱۷:۲۶

در حالت کلی در وب فرم‌ها، برای مسیردهی سازگار با Routing باید از ResolveUrl استفاده کنید.
برای اسکریپت‌ها:

```
<script type='text/javascript' src='<%= ResolveUrl("~/Scripts/test.js") %>'></script>
```

برای شیوه‌نامه‌ها:

```
<link rel="stylesheet" href="<%= ResolveUrl("~/myStylysheet.css")%>"  type="text/css" />
```

متد ResolveUrl به صورت خودکار مسیر صحیح را تولید می‌کند.

اما ... این روش کار کردن صحیح نیست. چون به زودی به تعداد زیادی فایل اسکریپت و CSS لینک داده شده در صفحه می‌رسید. برای یکی کردن آن‌ها یا از ScriptManager استفاده کنید (برای اسکریپت‌ها) و یا از روش‌های bundling & minification که با فایل‌های CSS و JS سازگار است. مسیریابی‌ها را هم به صورت خودکار تصحیح می‌کند.

نویسنده: زیلایی
تاریخ: ۱۳۹۳/۰۲/۳۱ ۱۰:۴۰

تشکر.
اما من روش bunding & minification رو متوجه نشدم.مطلب آموزشیش رو از کجا باید بخونم؟
منظورم مطلب آموزشی فارسی است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۳۱ ۱۰:۵۰

روش‌های زیادی برای دسته بندی، فشرده و یکی سازی فایل‌های CSS و JS هست؛ منجمله:

- بهینه سازی فایلهای js و css در برنامه‌های ASP.NET با استفاده از Combres - قسمت اول
- فشرده سازی فایل‌های CSS و JavaScript بصورت خودکار توسط MS Ajax Minifier
- Adding Bundling and Minification to Web Forms
- Building a basic Web Forms application using Visual Studio 2013
- Bundling and Minification in ASP.Net 4.5

نویسنده: شایان
تاریخ: ۱۳۹۳/۰۴/۲۵ ۱۱:۳۰

با سلام و تشکر بابت مقاله خوبتون .
من یه صفحه به اسم add.aspx تو فولدر Admin سایتم که با asp.net web form ایجاد شده دارم و میخوام وقتی کاربر آدرس http://mysite/Admin را تو آدرس بار وارد کرد بطور اتوماتیک وارد اون صفحه(Add.aspx) بشه .چطوری میتونم اینکارو بکنم ؟ با تشکر

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۴/۲۵ ۱۲:۵

سلام به این صورت در فایل Global.asax.cs

```
// using System.Web.Routing; //Must Be Added

        public static void RoutingSite(RouteCollection route)
        {
            //تعریف  Route
            route.MapPageRoute("Admin", "Admin", "~/admin/add.aspx");
        }

        protected void Application_Start(object sender, EventArgs e)
        {
            RoutingSite(RouteTable.Routes);
        }
```

نویسنده: شایان
تاریخ: ۱۳۹۳/۰۴/۲۶ ۱۳:۱۵

ممنون بابت جوابتون .من اینکارو انجام میدم ولی وقتی کاربر آدرس http://mysite/Admin را تو آدرس بار وارد میکنه اونو به دایرکتوری فولدر Admin که شامل چندین صفحست میبره و باز کاربر باید خودش رو لینک Add.aspx کلیک کنه تا بتونه وارد اون صفحه بشه ..امکانش هست کاربر اتوماتیک وارد صفحه Add.aspx بشه ؟

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۴/۲۶ ۱۳:۴۳

خواهش میکنم ، این مشکل به خاطر اینه که وقتی شما آدرس Admin را در مرورگر تایپ میکنید قبل از اینکه برنامه به کدهای روتینگ برسه برنامه ، Admin رو پوشه میدونه و فکر میکنه شما میخواهید پوشه‌ی ادمین رو باز کنید و این کارو انجام میده و به کدهای روتینگ نمیرسه. برای حل این مشکل باید نام روتینگ را به نام دیگه ای تغییر بدید مثل این

```
public static void RoutingSite(System.Web.Routing.RouteCollection route)
    {
        //تعریف  Route
        route.MapPageRoute("Admin", "AdminAdd", "~/admin/add.aspx");
    }

    protected void Application_Start(object sender, EventArgs e)
    {
        RoutingSite(System.Web.Routing.RouteTable.Routes);
```

```
    }
```

با این روتینگ وقتی شما آدرس http://mysite/AdminAdd رو میزنید صفحه‌ی add.aspx در پوشه‌ی ادمین رو اجرا میکنه . یه نمونه برنامه براتون ساختم و همین کاریو که خواستید رو انجام دادم. WebSite1-5077f441ae044a3ca59b8d34c9c42237.zip

نویسنده: زیلایی
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۳:۵۸

سلام؛ وقتی این کد رو توی کد بیهایند مستر پیج برای گرفتن ID کاربر قرار میدم بهم ارور میده

```
Page.RouteData.Values("pi").ToString()
```

چون مستر پیج خاصیتی به اسم page نداره. برای همین page.routeData رو null برمیگردونه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۴:۱۱

از this.Request.RequestContext.RouteData استفاده کنید.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۴:۱۷

سلام . معمولا در code behind صفحات داخلی ID رو میگیرند ، نه در مستر پیج و اینکه احتمالان اشتباه میکنید. Page یک متد توکار هستش و همیشه وجود داره . اگر به null برمیخورید احتمالان به خاطر استفاده از .ToString هستش . اگه مقداری که وجود نداره رو ازش .ToString() بگیریم به ارور **"object reference not set to an instance of an object"** بر میخوریم . در این آموزش نمونه برنامه‌های زیادی وجود داره ، ازشون کمک بگیرید.

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۴:۲۸

نکته‌ی دیگه ای اینه که گرفتن مقدار از یک Route به این صورت هستش.

```
var id = Page.RouteData.Values["pi"].ToString();
```

شما از () استفاده کردید.

نویسنده: زیلایی
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۵:۲۴

آخه من با Vb مینویسم و وقتی این کد رو مینویسم ارور میده که نباید از[] استفاده بشه.
کد رو به این تغییر دادم ولی بازم ارور دارم :

```
Me.Request.RequestContext.RouteData.Values("pi")
```

ارور اینه:

```
Object reference not set to an instance of an object.
```

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۵:۵۲

یه نمونه کد با زبان بسیار شیرین VB گذاشتم . ارور هم عرض کردم این ارور به خاطر این هستش که مقدار نال رو میخواهید بخونید . موفق باشید. WebApplication2-b172669ecf424c98846a0291a7ada1c9.zip

نویسنده: حسین فرهودی
تاریخ: ۱۳۹۳/۰۵/۲۸ ۱۰:۲۳

سلام ممکنه منو راهنمایی کنید

```
RouteTable.Routes.MapPageRoute("Gallery", "{PageName In Database}", "~/Main.aspx");
```

من جدولی در دیتابیس دارم که نام صفحات رو ذخیره میکنم چگونه میتونم مقدار PageName in Databse
چک کنه اگه داخل اون جدول بود وارد صفحه main بشه

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۵/۲۸ ۱۰:۴۱

سلام.
این روتینگ رو تعریف میکنید.

```
RouteTable.Routes.MapPageRoute("Gallery", "Page/{PageName}", "~/Main.aspx");
```

در صفحه‌ی Main.aspx

```
string pi = Page.RouteData.Values["PageName"] + "" ;
Pages page = (from p in context.Pages where p.PageName == pi select p).FirstOrDefault();
if (page ==  null )
        Response.Redirect("/");
  else
    // نمایش اطلاعات
```

نویسنده: حسین فرهودی
تاریخ: ۱۳۹۳/۰۵/۲۸ ۱۵:۴۴

سلام؛ ممنونم از پاسخ خوبتون. آیا امکان دارد بتوان در Page/{PageName لغت Page را حذف کرد و بعد دومین که زده میشود چک کند؟

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۵/۲۸ ۱۸:۸

این کار شدنی هستش که چنین روتینگی داشته باشیم

```
RouteTable.Routes.MapPageRoute("Gallery", "{PageName}", "~/Main.aspx");
```

ولی تمام آدرس‌ها به این صفحه ارجاع داده میشه و شما هر آدرسی که وارد کنید به صفحه‌ی ذکر شده میره و این مشخصه که سایتی که دارید مینویسید فقط همین صفحه رو نداره.
البته این نوع روتینگ کاربرد خاصی هم داره و من تو چند تا سایت که میخواستیم توسعه بدیم استفاده کردم به این صورت که صفحاتی که توسط گوگل ایندکس شده بود و نمیخواستیم از دست بدیم با این روتینگ چک میکردیم و صفحاتی که ایندکس بود رو به آدرس جدیدش میفرستادیم.

نویسنده: محمد
تاریخ: ۱۳۹۳/۰۵/۲۹ ۰:۷

سلام .من دقیقا مانند کدهای شما رو نوشتم ولی یه مشکلی هست
زمانی که از لینکی مثل زیر استفاده می‌کنم

```
<a href="article/da/name">...</a>
```

چیزی که در نوار آدرس به وجود میاد تقریبا مثل زیر خواهد بود :

```
localhost://article/da/name
```

و زمانی که می‌خوام در این حالت به یه آدرس دیگه برم مثل زیر

```
<a href="about">...</a>
```

درست کار نمیکنه و مقدار About رو جایگزین آخرین مقدار آدرس قبلی میکنه . مثلا :

```
localhost://article/da/about
```

منظورم اینه که به روت بر نمیگرده . باید چکار کرد برای حل این مشکل

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۲۹ ۰:۲۶

کمی بالاتر توضیح دادن ... در حالت کلی در وب فرم‌ها، برای مسیردهی سازگار با Routing باید از **ResolveUrl** استفاده کنید.

نویسنده: آریا
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۵:۱۰

پارامترهای صفحات من از نوع رشته هستند نه عدد بطور مثال :

```
localhost:56254/type/Brand/سمپلر/متغیر/5 میکرولیتر با سرنگ اضافه
```

برای کلیک روی لینک تگ A در قیمت href با استفاده از strinf.Format() خط را جایگزین فاصله کردم اما اطلاعاتم را نمیخواند چون در دیتابیس فاصله جای خط وجود دارد. من برای خواندن اطلاعات از دیتالیست به همراه EntityDataSource استفاده کرده ام مانند :

```
<asp:EntityDataSource ID="EntityDataSource1" runat="server"
            ConnectionString="name=mainfardbEntities"
            DefaultContainerName="mainfardbEntities" EnableFlattening="False"
            EntitySetName="T_Product"  Where="it.[CoName]=@category and it.[CatName]=@sub and
it.[SubCatName]=@thi and it.[ProdName]=@det">
            <WhereParameters>
                <asp:QueryStringParameter DbType="String" Name="category" QueryStringField="category"
/>
                <asp:QueryStringParameter DbType="String" Name="sub" QueryStringField="sub" />
                <asp:QueryStringParameter DbType="String" Name="thi" QueryStringField="thi" />
                <asp:QueryStringParameter DbType="String" Name="det" QueryStringField="det" />
            </WhereParameters>
        </asp:EntityDataSource>
```

چگونه میتوانم برای خواندن اطلاعات پارامتر خود را بصورت اولیه یعنی با فاصله بخوانم؟
ممنون از همه عزیزان

نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۵:۲۱

بجای خط از Uri.EscapeDataString باید استفاده کنید.

نویسنده: آریا
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۵:۵۶

ممنون از پاسخ شما
متاسفانه به نتیجه ای نرسیدم
ممکنه خودتون بیشتر توضیح بدین؟
باید بگم من از روش url rewrite استفاده میکنم

```
<rewrite url="~/type/(.+)/(.+)/(.+)/(.+)" to="~/type/Product-
Details.aspx?category=$1&amp;sub=$2&amp;thi=$3&amp;det=$4" />
```

من چطور میتونم به پارامتر ارسالی قبل از اینکه وارد شدن به شرط EntityDataSource دست پیدا کنم و اون رو تغییر بدم
البته من احساس میکنم مطلب گذاشته شده شما رو نتونستم درک بکنم

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۸:۰

هنگام تولید URL، اجزای قسمت‌های مختلف آن باید encode شوند؛ در غیر اینصورت عموما قابل پردازش نخواهند بود.

```
var url = "<a href='http://site/page/" + Uri.EscapeDataString(item.Category) + "'>Item1</a>";
```

برای مثال فاصله تبدیل می‌شود به 20% و امثال آن. از متد Uri.EscapeDataString برای encode کردن اجزای مختلف URL باید استفاده کنید تا به نتیجه‌ی صحیحی برسید. زمانیکه این encoding انجام شد، طرف کدهای دات نتی که این اجزاء یا کوئری استرینگ‌ها را دریافت می‌کند، نیازی به هیچ تغییری یا اصلاحی نخواهد بود، چون به صورت خودکار قابلیت پردازش encoding آدرس دریافتی را دارد (در پشت صحنه از متد HttpUtility.UrlDecode استفاده می‌کند). برای نمونه مقادیر دریافتی Page.RouteData.Values را بررسی کنید.

نویسنده: هادی
تاریخ: ۱۳۹۳/۰۸/۲۴ ۱۰:۸

سلام
واسه من تصاویر سایت هم نشان داده نمیشه ، برای تصاویر هم باید همین کار رو بکنم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۴ ۱۰:۳۷

```
<img src='<%=ResolveUrl("~/images/brain.png")%>' />
<asp:ImageButton ID="ImageButton1" ImageUrl="~/images/brain.png"  runat="server" />
<asp:Image runat="server" ID="imgHelp"  ImageUrl="~/Images/myImage.jpg" />
```

نویسنده: هادی
تاریخ: ۱۳۹۳/۰۸/۲۴ ۱۱:۴

ممنون
حالا فکر کنید من مطالب کاربر که حاوی تصاویر نیز هست در دیتابیس ذخیره کرده ام و می‌خوام در صفحه نمایش بدم ، حالا چجور میتونم تصاویری که تو مطلب هست که در زمان اجرا در صفحه ریخته میشه نشون بدم؟
ببخشید یک سوال دیگه ، اصلا میشه گفت فقط فایلهایی که به پسوند aspx ختم میشند رو از دسترسی مستقیم خارج کن و بقیه

فایلها آزاد باشند؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۴ ۱۱:۵۲

- این نوع تصاویر را بجای آدرس نسبی، با آدرس مطلق ذخیره کنید.
- از HTML Agility pack برای ویرایش و اصلاح src در حین رندر نهایی استفاده کنید.
- امکان خارج کردن منابع از سیستم URL rewrite وجود دارد :

```
<rule name="RewriteUserFriendlyURL1" stopProcessing="true">
  <match url="^([^/]+)/?$" />
  <conditions>
     <add input="{URL}" negate="true" pattern="\.png$" />
  </conditions>
</rule>
```

نویسنده: هادی
تاریخ: ۱۳۹۳/۰۸/۲۶ ۱۳:۳

با سلام؛ من از این روش از تکنیک ای‌جکس استفاده می‌کنم، ولی نمیدونم الان که Urlها رو با تکنیک شما بازنویسی کردم چگونه توابع رو فراخوانی کنم؟ به وب سرویس دسترسی دارم ولی تابع داخلش اجرا نمیشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۲۶ ۱۳:۱۱

از اینجا شروع کنید «نحوه استفاده از افزونه Firebug برای دیباگ برنامه‌های ASP.NET مبتنی بر jQuery »
+ « ASP.NET FriendlyUrls و دریافت خطای Unauthorized 401 در حین عملیات Ajax »

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۰ ۱۱:۳۹

با سلام؛ وسایتم خوب کار میکنه فقط وقتی این قسمت که به جای فضای خالی _ بزار رو گزاشتم مشکل پیدا کردم چطور درستش کنم.

```
<asp:HyperLink ID="HyperLink1" runat="server" NavigateUrl='<%# Eval("NewsID", "ShowNews.aspx?id={0}")+
Eval("NewsTitle","&_title={0}").ToString().Replace(" ","_")%>'
```

این خطهای اضافی دنباله خبر ماله چیه که درست شده اخه من برای درج خبر کاراکتر اضافی وارد نکردم که؟کلا چرا بعد از خبر هام این قدر-------- میزاره

```
http://localhost:28625/ShowNews.aspx?id=25&_title=%D8%A2%D8%AE%D8%B1%DB%8C%D9%86_%D8%AA%D8%B5%D8%A7%D9%
88%DB%8C%D8%B1_%D8%A7%D8%B2_%22%D9%85%D8%B1%D8%AA%D8%B6%DB%8C_%D9%BE%D8%A7%D8%B4%D8%A7%DB%8C%DB%8C%22__
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
_____________________________________________________________________________________________________
```

_____

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۰ ۱۱:۴۹

- از روش‌های ابداعی برای encoding رشته‌ها در وب استفاده نکنید. کمی بالاتر توضیح دادم که باید از Uri.EscapeDataString استفاده کنید؛ از این جهت که حروفی که باید escape شوند، به یک فاصله‌ی ساده ختم نمی‌شوند. یک لیست طولانی دارد.
- اگر قصدتان تولید slug است، از متد GenerateSlug مطرح شده در مطلب « بهینه سازی برنامه‌های وب ASP.NET برای موتورهای جستجو (SEO) » ایده بگیرید. در انتهای آن یک متد Trim هم فراخوانی شده‌است (برای حذف فواصل خالی در ابتدا و انتهای رشته).

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۹:۹

سلام برای خوندن اطلاعات از کوئری استرینگ باید چطور دستور سلکت رو نوشت من وقتی از روش بالایی که گفتم استفاده میکردم میگفتم که مقدار id رو بخون از کوئری استرینگ الان نوشتم where newsid=@pi
بعدم تو sqlدیتا سورس گفتم از کوئری استرینگ مقدار piرو بخون ولی ارور میده الان بگم چی رو بخون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۹:۲۴

```
SqlDataSource1.SelectParameters.Add("@param1", value1);
```

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۹:۴۵

مقداری که از کوئری استرینگ میخونه چیه همون pi? آخه من قبلا id میگفتم بخون وجواب میداد اما الان گفتم pi رو بخون ولی جواب نداد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۹:۴۹

- یکبار متن و نظرات را کامل مطالعه کنید. اگر از روش جدید مسیریابی استفاده کنید، دیگر نباید کوئری استرینگی در URL شما مشاهده شود (یکی از اهداف اصلی سیستم مسیریابی). در اینجا یک *الگو* را باید در متد RouteTable.Routes.MapPageRoute ثبت کنید.
- **pi** بر اساس همان الگوی ثبت شده‌ی Post/{ **pi** }/{pt} است؛ در مثالی که در متن زده شده.

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۳ ۹:۸

سلام من تو globalاینجوری نوشتم

```
<%@ Application Language="C#" %>
<script runat="server">

    void Application_Start(object sender, EventArgs e)
    {
        // Code that runs on application startup
      System.Web.Routing.RouteTable.Routes.MapPageRoute("NewsDatils", "News/{pi}/{pt}",
"~/ShowNews.aspx");

    }

    void Application_End(object sender, EventArgs e)
    {
        //  Code that runs on application shutdown
```

```
    }
```

بعد تو صفحه لینک اینجوری نوشتم

```
<asp:HyperLink ID="HyperLink2" runat="server" NavigateUrl='<%#
string.Format("/News/{0}/{1}",Eval("NewsID"),Eval("NewsTitle").ToString().Replace(" ","-")) %>'
Text='<%# Eval("NewsTitle") %>'></asp:HyperLink>
```

و تو shownews الان مقدارpi برابر null

```
int p = int.Parse(Page.RouteData.Values["pi"].ToString());
```

اینم اروری که نشون میده

```
Cannot use a leading .. to exit above the top directory.
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۳ ۹:۳۸

به احتمال زیاد در صفحه یک چنین آدرسی را دارید:

```
<asp:Image ImageUrl="..\foo.jpg" />
```

در اینجا .. به یک سطح بالاتر از ریشه سایت اشاره می‌کند که معتبر نیست. بجای .. از ~ استفاده کنید یا از روش ResolveUrl که کمی بالاتر در نظرات توضیح داده شد .

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۳ ۱۵:۲۸

ممنون. تو صفحه دوتا ادرس فایل‌های js رو به این صورت ../ داده بودم و عوض کردم با ~ و مشکل حل شد.

نویسنده: موسوی
تاریخ: ۱۳۹۳/۰۹/۲۳ ۱۶:۳۲

سلام؛ چطور یه لینک بسازه به خانه برگرده. آخه وقتی مثل قبل امتحان کردم اینطور:

```
localhost:28625/News/26/~/default.aspx
```

شد؟

نویسنده: رضا منصوری
تاریخ: ۱۳۹۳/۰۹/۲۳ ۲۳:۲۴

سلام ، به این صورت

```
<a href=<%= ResolveUrl("~") %> >Home</a>
 <a href="/" >Home</a>
```

نویسنده: موسوی

تاریخ: ۱۳۹۳/۰۹/۲۶ ۱۲:۴۳

با سلام؛ همه چیز عالی کار کرده فقط یه مشکل دارم اونه که تو url به کاراکترهای / ، : ، و.... گیر میده و iis اخطار میگیره راهشم اینه که تو ادرس دهی بیام به جای این کاراکترها - یا + بزارم تو این لینک چطور بنویسم که مشکلم حل شه

```
NavigateUrl='<%# string.Format("/News/{0}/{1}",Eval("NewsID"),Eval("NewsTitle").ToString().Replace(" ","-")
```

به جای کد زیرچطور بنویسم که درست کار کنه. تشکر

```
Replace(" ","-")
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۶ ۱۳:۱۶

کمی بالاتر پاسخ گرفتید « از روش‌های ابداعی برای encoding رشته‌ها در وب استفاده نکنید ... »

نویسنده: امیر نوروزیان
تاریخ: ۱۳۹۳/۱۰/۱۷ ۱۲:۱۶

با سلام؛ جدولی دارم که لیست دانلودها رو نشون میده. وقتی روی یک لینک از جدول کلیک میکنی، میخام چنین آدرسی رو نشون بده:

آشنایی با قوانین کار/ ReDownload/1127/

حالا برای اینکه لیست دانلودها که یکسری مقاله هست، توی جستجوی گوگل بیشتر به چشم بخوره، اومدم آدرس بالا رو تبدیل به این آدرس کردم:

.docx اشنایی با قوانین کار/ ReDownload/1127/

ولی متاسفانه وقتی روی لینک کلیک میکنی http error 404 -not found

حالا سوال اصلی اینه، کلا روش من غلطه یا باید تنظیمات خاصی رو اعمال کنم؟

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۱۰/۱۷ ۱۳:۳۲

مرجع رسمی مسیریابی در وب فرم‌ها عنوان می‌کنه که URL routing allows you to configure an application to accept request URLs that **do not** map to **physical files** . یعنی چون آدرس شما الان **پسوند** پیدا کرده، دیگه وارد سیستم مسیریابی نمیشه و به صورت یک فایل فیزیکی پردازش میشه. یعنی در مسیر و پوشه‌ای شبیه به آدرسی که نوشتید به دنبال اون فایل می‌گرده (که نیست و به همین جهت خطای 404 رو دریافت می‌کنید). البته با تنظیم RouteTable.Routes. **RouteExistingFiles** = true امکان تغییر این پیش فرض هست. در این حالت درخواست تمام فایل‌های فیزیکی وارد سیستم مسیریابی میشن. البته در این مورد خاص باید یک IRouteHandler بنویسید تا این درخواست فایل رسیده رو پردازش کنه.

نویسنده: امیر نوروزیان
تاریخ: ۱۳۹۳/۱۰/۱۷ ۱۴:۱۵

فایل فیزیکی وجود نداره پسوندم برای جستجو تو گوگل اضافه کردم. تنظیمان مورد نظرم رو اضافه کردم در اربطه handler بیشترم لطفا توضیح بدین ممنون میشم

stackoverflow

از توضیحات بالام استفاده کردم ولی جوابی نگرفتم

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۱۰/۱۷ ۱۶:۰

گوگل اینقدر هوشمند هست که بتونه تشخیص بده الان درخواست یک فایل word رو داده ولی جوابش html هست یعنی یکجا کار

مشکل داره و نمره‌ی منفی بهت می‌ده. بهتره وقت بذاری و مباحث اصلی SEO رو در سایت بخونی.

نویسنده: امیر نوروزیان
تاریخ: ۱۳۹۳/۱۰/۱۷ ۱۶:۱۷

من دنبال آدرس جعلی نیستم. تو بعضی سایت‌ها مثل این

somesite.ir/thesis/pdf/1.pdf

لینک رو انطوری داده کلیک که میکنی دانلود میکنه. گوگل هم این لینک رو به عنوان یه فایل pdf میشناسه.

عنوان: **استفاده از Awesomium.NET در برنامه‌های وب**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۵ ۱۵:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Awesomium

برای تهیه تصاویر سایت‌های معرفی شده در قسمت اشتراک‌های سایت ، پیشتر از کنترل WebBrowser دات نت که در پشت صحنه از امکانات IE کمک می‌گیرد، استفاده می‌کردم. بسیار ناپایدار است؛ به روز رسانی مشکلی داشته و وابسته است به سیستم عامل جاری سیستم. برای مثال مرتبا برای تهیه تصاویر بند انگشتی (Thumbnails) سایت‌های تهیه شده با بوت استرپ کرش می‌کرد و این کرش چون از نوع unmaged code است، عملا پروسه IIS وب سایت را از کار می‌انداخت و در این حالت سایت تا ری‌استارت بعدی IIS در دسترس نبود. برای حل این مشکل و بهبود کیفیت تصاویر تهیه شده، از پروژه Awesomium که در حقیقت مرورگر کروم را جهت استفاده در انواع و اقسام زبان‌های برنامه نویسی محصور می‌کند، کمک گرفته شد؛ که شرح آن‌را در ادامه ملاحظه خواهید کرد.

دریافت و نصب Awesomium

پروژه Awesomium دارای یک SDK است که از اینجا قابل دریافت می‌باشد. بعد از نصب آن در مسیر Awesomium SDK\1.7.3.0\wrappers\Awesomium.NET\Assemblies\Packed می‌توانید محصور کننده‌ی دات نتی آن‌را مشاهده کنید. منظور از Packed در اینجا، استفاده از DLLهای فشرده شده‌ی native آن است که در مسیر Awesomium SDK\1.7.3.0\build\bin\packed کپی شده‌اند. بنابراین برای توزیع این نوع برنامه‌ها نیاز است اسمبلی دات نتی Awesomium.Core.dll به همراه دو فایل بومی icudt.dll و awesomium.dll ارائه شوند.

تهیه تصاویر سایت‌ها به کمک Awesomium.NET

پس از نصب Awesomium اگر به مسیر Documents\Awesomium SDK Samples\1.7.3.0\Awesomium.NET\Samples\Core\CSharp\BasicSample مراجعه کنید، مثالی را در مورد تهیه تصاویر سایت‌ها به کمک Awesomium.NET، مشاهده خواهید کرد. خلاصه‌ی آن چند سطر ذیل است:

```
try
            {
                using (WebSession mywebsession = WebCore.CreateWebSession(
new WebPreferences() { CustomCSS = "::-webkit-scrollbar { visibility: hidden; }" }))
                {
                    using (var view = WebCore.CreateWebView(1240, 1000, mywebsession))
                    {
                        view.Source = new Uri("https://site.com/");

                        bool finishedLoading = false;
                        view.LoadingFrameComplete += (s, e) =>
                        {
                            if (e.IsMainFrame)
                                finishedLoading = true;
                        };

                        while (!finishedLoading)
                        {
                            Thread.Sleep(100);
                            WebCore.Update();
                        }

                        using (var surface = (BitmapSurface)view.Surface)
                        {
                            surface.SaveToJPEG("result.jpg");
                        }
                    }
                }
            }
            finally
            {
                WebCore.Shutdown();
            }
```

کار با ایجاد یک WebSession شروع می‌شود. سپس با مقدار دهی CustomCSS، اسکرول بار صفحات را جهت تهیه تصاویری بهتر مخفی می‌کنیم. سپس یک WebView آغاز شده و منبع آن به Url مدنظر تنظیم می‌شود. در ادامه باید اندکی صبر کنیم تا بارگذاری سایت خاتمه یافته و نهایتا می‌توانیم سطح این View را به صورت یک تصویر ذخیره کنیم.

مشکل! این روش در برنامه‌های ASP.NET کار نمی‌کند!

مثال همراه آن یک مثال کنسول ویندوزی است و به خوبی کار می‌کند؛ اما در برنامه‌های وب پس از چند روز سعی و خطا مشخص شد که:

الف) WebCore.Shutdown فقط باید در پایان کار یک برنامه فراخوانی شود. یعنی اصلا نیازی نیست تا در برنامه‌های وب فراخوانی شود.

```
System.InvalidOperationException: You are attempting to re-initialize the WebCore.
The WebCore must only be initialized once per process and must be shut down only when the process
exits.
```

ب) Awesomium فقط در یک ترد کار می‌کند. به این معنا که اگر کدهای فوق را در یک صفحه‌ی وب فراخوانی کنید، بار اول کار خواهد کرد. بار دوم برنامه کرش می‌کند؛ با این پیغام خطا:

```
System.AccessViolationException: Attempted to read or write protected memory.
This is often an indication that other memory is corrupt. at
Awesomium.Core.NativeMethods.WebCore_CreateWebView_1(HandleRef jarg1, Int32 jarg2, Int32 jarg3,
HandleRef jarg4)
```

چون هر صفحه‌ی وب در یک ترد مجزا اجرا می‌شود، عملا استفاده‌ی مستقیم از Awesomium در آن ممکن نیست.
خطای فوق هم از آن نوع خطاهایی است که پروسه‌ی IIS را درجا خاموش می‌کند.

استفاده از Awesomium در یک ترد پس زمینه

راه حلی که نهایتا پاسخ داد و به خوبی و پایدار کار می‌کند، شامل ایجاد یک ترد مجزای Awesomium در زمان آغاز برنامه‌ی وب و زنده نگه داشتن آن تا زمان پایان کار برنامه است.

```
using System;
using System.Collections.Concurrent;
using System.IO;
using System.Threading;
using System.Web;
using Awesomium.Core;

namespace AwesomiumWebModule
{
    public class AwesomiumModule : IHttpModule
    {
        private static readonly Thread WorkerThread = new Thread(awesomiumWorker);
        private static readonly ConcurrentQueue<AwesomiumRequest> TaskQueue = new
ConcurrentQueue<AwesomiumRequest>();
        private static bool _isRunning = true;

        static AwesomiumModule()
        {
            WorkerThread.Start();
        }

        private static void awesomiumWorker()
        {
            while (_isRunning)
            {
                if (TaskQueue.Count != 0)
                {
                    AwesomiumRequest outRequest;
                    if (TaskQueue.TryDequeue(out outRequest))
                    {
                        var img = AwesomiumThumbnail.FetchWebPageThumbnail(outRequest);
```

```
                        File.WriteAllBytes(outRequest.SavePath, img);
                        Thread.Sleep(500);
                    }
                }
                Thread.Sleep(5);
            }
        }

        public void Dispose()
        {
            _isRunning = false;
            WebCore.Shutdown();
        }

        public void Init(HttpApplication context)
        {
            context.EndRequest += endRequest;
        }

        static void endRequest(object sender, EventArgs e)
        {
            var url = HttpContext.Current.Items[Constants.AwesomiumRequest] as AwesomiumRequest;
            if (url!=null)
            {
                TaskQueue.Enqueue(url);
            }
        }
    }
}
```

در اینجا اگر در برنامه‌های وب فرم، از طریق HttpContext.Current.Items.Add و یا در برنامه‌های MVC به کمک this.HttpContext.Items.Add یک آیتم جدید، با کلید Constants.AwesomiumRequest و از نوع کلاس AwesomiumRequest دریافت گردد، مقدار آن به یک ConcurrentQueue اضافه خواهد شد. این صف در یک ترد مجزا مدام در حال بررسی است. اگر مقداری به آن اضافه شده‌است، از صف خارج شده و پردازش خواهد شد.

نمونه‌ی استفاده از آن، در سمت یک برنامه‌ی وب نیز به صورت زیر است. ابتدا ماژول تهیه شده باید در برنامه ثبت شود:

```
<?xml version="1.0"?>
<configuration>
  <system.web>
    <compilation debug="true" targetFramework="4.0" />
    <httpModules>
      <add name="AwesomiumWebModule" type="AwesomiumWebModule.AwesomiumModule"/>
    </httpModules>
  </system.web>

  <system.webServer>
    <validation validateIntegratedModeConfiguration="false"/>
    <modules>
      <add name="AwesomiumWebModule" type="AwesomiumWebModule.AwesomiumModule"/>
    </modules>
  </system.webServer>
</configuration>
```

سپس باید تنها مدیریت HttpContext.Current.Items در سمت برنامه صورت گیرد:

```
protected void btnStart_Click(object sender, EventArgs e)
        {
            var host = new Uri(txtUrl.Text).Host;
            HttpContext.Current.Items.Add(Constants.AwesomiumRequest, new AwesomiumRequest
            {
                Url = txtUrl.Text,
                SavePath = Path.Combine(HttpRuntime.AppDomainAppPath, "App_Data\\Thumbnails\\" + host +
".jpg"),
                TempDir = Path.Combine(HttpRuntime.AppDomainAppPath, "App_Data\\Temp")
            });
            lblInfo.Text = "Please wait. Your request will be served shortly.";
        }
```

در اینجا کاربر درخواست خود را در صف قرار می‌دهد. پس از مدتی کار آن در WorkerThread ماژول تهیه شده انجام گردیده و

تصویر نهایی تهیه می‌شود.

Url، آدرس وب سایتی است که می‌خواهید تصویر آن تهیه شود. SavePath مسیر کامل فایل jpg نهایی است که قرار است دریافت و ذخیره گردد. TempDir محل ذخیره سازی فایل‌های موقتی Awesomium است. Awesomium یک سری کوکی، تصاویر و فایل‌های هر سایت را به این ترتیب کش کرده و در دفعات بعدی سریعتر عمل می‌کند.

پروژه‌ی کامل آن‌را از اینجا می‌توانید دریافت کنید:

AwesomiumWebApplication_V1.0.zip

نظرات خوانندگان

نویسنده: سعید شیرزادیان
تاریخ: ۱۳۹۲/۱۱/۱۶ ۱۹:۱۳

پروژه فوق در ویژوال استودیو 2012 باز نمی‌شود پیغام عدم سازگاری می‌دهد. آیا راهنمایی می‌توانید بکنید
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۱/۱۶ ۱۹:۲۳

فایل‌های csproj و sln آن‌را حذف کنید. بعد دو پروژه وب فرم و MVC خالی درست کنید و فایل‌های موجود را به آن‌ها اضافه کنید.
ماژول آن هم یک class library ساده است.

نویسنده: Raees
تاریخ: ۱۳۹۳/۰۱/۰۵ ۲۰:۳

```
Hi Vahid
```

```
You have done an excellent job. But I have one query after generating one image the second one doesnot
generate i mean do i need to refresh the page or do something special. The files in the  folder gets
hold by the worker process and hence the subsequent request are not processed can you help us in this
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۵ ۲۰:۵۷

```
Remove the AwesomiumModule->Dispose method's body. It's not necessary
```

عنوان: **ایجاد نقشه سایت (Site Map) داینامیک**
نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۰۲ ۰:۳۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, C#, sitemap

همان طور که می‌دانید نقشه‌ی سایت علاوه بر استفاده از MetaTagها و Url Routingها و ... یکی از نکات اصلی برای سایت شماست که در نتایج گوگل برای جستجو کنندگان نمایش داده شود .

در این مقاله من قصد دارم که به شما چگونگی ایجاد و کار با یک SiteMap داینامیک را آموزش دهم .
منظور از SiteMap داینامیک این است که به ازای هر مطلبی که شما در سایت تان ایجاد ، ویرایش یا حذف می‌کنید چنانچه دارای صفحه ای در سایت باشد SiteMap تغییر می‌کند .

ساختار یک SiteMap به صورت زیر است :

```
<?xml version="1.0" encoding="utf-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
  <url>
<loc>Url Page 1</loc>
<lastmod>2014-02-20</lastmod>
<changefreq>always</changefreq>
<priority>1</priority>
  </url>
  <url>
<loc>Url Page 2</loc>
<lastmod>2014-02-20</lastmod>
<changefreq>always</changefreq>
<priority>1</priority>
  </url>
</urlset>
```

به ازای هر مطلبی که به سایتتان اضافه می‌کنید چنانچه آن مطلب جهت نمایش دارای Url باشد ، باید یک تگ Url به SiteMap اضافه شود.
تگ loc موجود در تگ url آدرس صفحه را مشخص می‌کند .
تگ lastmod تاریخ اضافه کردن یا آخرین ویرایش را نمایش می‌دهد .
تگ changefreq که دوره‌ی بروز رسانی صفحه را مشخص می‌کند .
تگ priority الویت صفحه را مشخص می‌کند .

که من در کد نویسی تگ changefreq را always و تگ priority را 1 قرار دادم.

در فایل ضمیمه یک کلاس به اسم updateSiteMap.cs وجود دارد که تابع آن شامل دو پارامتر ورودی مانند زیر است :

```
public void UpdateSiteMap(string Addr,string NewOpr)
```

پارامتر Addr که آدرس صفحه ای است که شما می‌خواهید به SiteMap اضافه شود .
پارامتر NewOpr که می‌تواند شامل یکی از سه مقدار زیر باشد: add ، edit ، delete

اگر پارامتر NewOpr دارای مقدار add باشد یعنی مقدار موجود در پارامتر Addr را بهSiteMap اضافه کن . یعنی یک تگ url به SiteMap اضافه کن .
اگر پارامتر NewOpr دارای مقدار edit باشد یعنی مقدار موجود در تگ <lastmod> را ویرایش کن . یعنی تاریخ آخرین بروزرسانی تغییر می‌کند .
اگر پارامتر NewOpr دارای مقدار delete باشد یعنی تگ url ایی که محتوای تگ loc آن برابر است با مقدار موجود در پارامتر

Addr را از SiteMap حذف کن.

این بخش از کد موجود در فایل ضمیمه uupdateSiteMaop.cs قسمت edit و delete نقشه‌ی سایت را انجام می‌دهد .

```
if (NewOpr != "add")
{
XmlElement xmlElement = xmlDoc.DocumentElement;
if (xmlElement.ChildNodes != null)
{
  foreach (XmlElement myElement in xmlDoc.DocumentElement)
  {
if (myElement.ChildNodes[0].InnerText == Addr)
{
  if (NewOpr != "delete")
myElement.ChildNodes[1].InnerText = DateTime.Now.ToString("yyyy-MM-dd");
  else
myElement.ParentNode.RemoveChild(myElement);
break;
}
  }
}
}
```

و بخش else دستور بالا قسمت Add را انجام می‌دهد .یعنی کدهای زیر:

```
else
{
string ns="http://www.sitemaps.org/schemas/sitemap/0.9";
XmlNode url = xmlDoc.CreateNode(XmlNodeType.Element, "url",ns );
XmlNode loc = xmlDoc.CreateNode(XmlNodeType.Element, "loc", ns);
XmlNode lastmod = xmlDoc.CreateNode(XmlNodeType.Element, "lastmod", ns);
XmlNode changefreq = xmlDoc.CreateNode(XmlNodeType.Element, "changefreq", ns);
XmlNode priority = xmlDoc.CreateNode(XmlNodeType.Element, "priority", ns);
loc.InnerText = Addr;
lastmod.InnerText = DateTime.Now.ToString("yyyy-MM-dd");
changefreq.InnerText = "always";
priority.InnerText = "1";
url.AppendChild(loc);
url.AppendChild(lastmod);
url.AppendChild(changefreq);
url.AppendChild(priority);
xmlDoc.DocumentElement.AppendChild(url);
}
```

اگر اطلاعاتی را به جدول اضافه می‌کنید و می‌خواهید Url صفحه‌ی مربوط به آن اطلاعات برای شما در SiteMap اضافه شود بعد از ذخیره شدن اطلاعات در جدول بلافاصله کد زیر را اضافه می‌کنید :

```
//
Add Info In Table
//
updateSiteMap updateSiteMap = new updateSiteMap();
updateSiteMap.UpdateSiteMap("Url Page", "add");
```

برای قسمت ویرایش هم پس از آنکه اطلاعات را ویرایش کردید چنانچه برای آن اطلاعات صفحه ای را در SiteMap درج کرده اید کد زیر را می‌نویسید :

```
updateSiteMap updateSiteMap = new updateSiteMap();
updateSiteMap.UpdateSiteMap("Url Page", "edit");
```

برای قسمت حذف هم اگر شما اطلاعاتی را از جدول حذف می‌کنید چنانچه برای آن اطلاعات صفحه ای در SiteMap درج کرده اید کد زیر را می‌نویسید :

```
updateSiteMap updateSiteMap = new updateSiteMap();
updateSiteMap.UpdateSiteMap("Url Page", "delete");
```

موفق باشید .

Files.zip

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۰۲ ۰:۴۶

چند نکته تکمیلی

- نسخه‌ی MVC این بحث را در اینجا می‌توانید مطالعه کنید. این نسخه را می‌توان تبدیل به یک فایل ashx وب فرم‌ها نیز کرد. context آن قابل انتقال است و با کد انتهای بحث یکی است.

- در وب فرم‌ها الزاما نیازی نیست تا حتما یک فایل XML ثابت را روی سخت دیسک ذخیره کنید. می‌شود با استفاده از مباحث routing در وب فرم‌ها ، فایلی‌را که وجود خارجی ندارد، ایجاد کرد:

```
void Application_Start(object sender, EventArgs e)
{
    RouteTable.Routes.Add(new Route(
        "sitemap.xml", new PageRouteHandler("~/sitemap.ashx")));
}
```

در اینجا مسیر sitemap.ashx که یک Generic handler است، به مسیر sitemap.xml نگاشت می‌شود (کاربر اگر sitemap.xml را در مرورگر وارد کند، مدیریت آن توسط sitemap.ashx انجام خواهد شد).

- سپس برای تولید فایل XML به صورت پویا فقط از sitemap.ashx استفاده کنید به همراه مباحث caching اطلاعات :

```
public class Sitemap : IHttpHandler
    {
        public void ProcessRequest(HttpContext context) {
            context.Response.ContentType = "text/xml";
            //مابقی نکات بحث در اینجا
            context.Response.End();
        }

        public bool IsReusable {
            get {  return false; }
        }
    }
```

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۲/۱۲/۰۲ ۲۳:۲۹

متشکر از زحمتی که برای این مطلب کشیدید.

یک ایراد ویرایشی کوچک:

که من در کد نویسی تگ changefreq را 1 و تگ priority را always قراار دادم.

ظاهرا جابجا نوشتین

نویسنده: سلمان کاظمی
تاریخ: ۱۳۹۲/۱۲/۲۸ ۱:۴۴

من یه مشکلی با سایت مپ دارم و اون اینه که من یک فرم دارم که ممکنه چند فرم مختلف این فرم را نمایش میدهند. حالا این روش شما مشکل را حل می‌کنه؟مثلا اگه اسم فرم من A باشه می‌خوام وقتی از فرم B فرم A را نمایش میدهم سایت مپ بصورت B-->A نشون داده بشه و اگه از فرم C فرم A را نشون میدم شکل سایت مپ بصورت C-->A باشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۸ ۹:۶

sitemap معرفی شده در مطلب جاری، صرفا برای موتورهای جستجو نوشته شده است و کاربرد مدنظر شما را ندارد.

نویسنده: محسن نجف زاده

تاریخ: ۱۳۹۳/۰۵/۲۸ ۹:۴۶

من از این روش (تولید فایل XML به صورت پویا از طریق sitemap.ashx) استفاده کردم ولی متاسفانه خطای زیر رو میده *(از دات نت 4.5 استفاده می‌کنم)*

```
Type 'sitemap' does not inherit from 'System.Web.UI.Page'.
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۸ ۱۰:۱۳

نیاز به کمی اصلاح دارد. جنریک هندلرها برای معرفی به سیستم مسیریابی باید از طریق یک IRouteHandler معرفی شوند:

```
// نحوه‌ی معرفی جنریک هندلر به سیستم مسیریابی
public class RouteHandler : System.Web.Routing.IRouteHandler
{
    public IHttpHandler GetHttpHandler(System.Web.Routing.RequestContext requestContext)
    {
        return new Sitemap();
    }
}

// Global.asax file
void Application_Start(object sender, EventArgs e)
{
    RouteTable.Routes.Add(new Route("sitemap.xml", new RouteHandler()));
}
```

خلاصه آن به صورت یک متد الحاقی

عنوان: **Owin چیست ؟ قسمت اول**
نویسنده: یاسر مرادی
تاریخ: ۱۳۹۲/۱۲/۰۲ ۱۸:۴۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, OWIN, Katana, helios

مطمئنا اکثر شما برنامه نویسان با معماری IIS و ASP.NET کمابیش آشنایی دارید
Request از سمت کلاینت به IIS ارسال می‌شود، و **عموما** بسته به نوع درخواست کلاینت یا به یک Static File مپ می‌شود(مثلا به یک عکس)، و یا به یک ISAPI
ISAPI کدی است که عموما با ++C نوشته می‌شود، و برای درخواست آمده از سمت کلاینت کاری را انجام می‌دهد
یکی از این ISAPIها برای ASP.NET است، که درخواست کلاینت را به یک کد مبتنی بر .NET. مپ می‌کند (به همین علت به آن ASP.NET می‌گویند)
نکته ای که در خطوط فوق به وضوح دیده می‌شود، وابستگی شدید ASP.NET به IIS است
بدیهتا کدی که بر روی بستر ASP.NET نوشته می‌شود نیز وابستگی فوق العاده ای به IIS دارد، که یکی از بدترین نوع این وابستگی‌ها در ASP.NET Web Forms دیده می‌شود.
خب، این مسئله چه مشکلاتی را ایجاد می‌کند ؟
مشکل اول که شاید کمتر به چشم بیاید، بحث کندی اجرای بار اول برنامه‌های ASP.NET است.
اما مشکل دوم عدم توانایی در نوشتن کد برنامه، بدون وابستگی به **وب سرور** (در اینجا IIS) است، که این مشکل دوم روز به روز در حال جدی‌تر شدن است.
این مشکل دوم را برنامه نویسان جاوا سالهاست که با آن درگیرند، نکته این است که بین دو وب سرور در نحوه پردازش یک درخواست کلاینت تفاوت هایی وجود دارد، که بالطبع این تفاوت‌ها در نحوه‌ی اجرای کد بالاخره خودش را جاهایی نشان می‌دهد، این که بگوییم رفتار وب سرورها نباید متفاوت باشد کمی مسخره است، زیرا تفاوت آنها با یکدیگر باعث شده که سرعت یکسان و امکانات یکسانی نداشته باشند و هر کدام برای یک سناریوی خاص مناسب‌تر باشند
این مسئله برای ما نیز روز به روز دارای اهمیت بیشتری می‌شود، دیگر این که Web Server ما فقط IIS صرف باشد، سناریوی متداول در پروژه‌های Enterprise نیست
در چه جاهایی می‌توان یک برنامه را هاست کرد ؟
IIS به همراه ASP.NET
IIS بدون ASP.NET (می‌خواهیم برنامه بر روی IIS هاست شود، ولی کاری با ASP.NET نداریم) CLR AppDomains

و وب سرورهای لینوکسی در صورت اجرای برنامه بر روی Mono
و ...
هم اکنون به میزان زیادی مشکل شفاف شده است، مطابق با معماری فعلی داریم
Request >> IIS >> aspnet_isapi.dll >> System.Web.dll >> Your codes
مشکل دیگری که وجود دارد این است که اگر تیمی بخواهد فریم ورکی برای برنامه نویسان نهایی فراهم کند، باید آنرا بر روی اکثر گزینه‌های هاست موجود سازگار کنید، برای مثال مشاهده می‌کنید که ASP.NET Signal R را هم می‌توان بر روی IIS و ASP.NET هاست کرد و هم بر روی یک App Domain کاملا معمولی و علاوه بر این که تیم SignalR باید این هزینه مضاعف را پرداخت کند، خروجی برای ما نیز چندان خوشایند نیست، برای مثال اجرای همزمان ASP.NET SignalR و ASP.NET Web API اگر چه که بر روی هاستی به غیر از ASP.NET نیز امکان پذیر است، اما متاسفانه به عنوان دو بازیگر جدا از هم کار می‌کنند و عملا تعاملی با یکدیگر ندارند، مگر این که بر روی ASP.NET هاست شوند، و یا بسیاری از امکانات Routing موجود در WCF بر روی بستری غیر از ASP.NET کار نمی‌کند. بدیهی است که این بازار پر آشوب به نفع هیچ کس نیست. و اما راه حل چیست ؟ تعدادی از برنامه نویسان حرفه ای .NET. دور یکدیگر جمع شدند و طی بررسی هایشان به این نتیجه رسیدند که هاست‌های مختلف نقاط اشتراک بسیار زیادی دارند و تفاوت‌ها نباید باعث این میزان مشکل شود.
پس استانداری را طراحی کردند با نام OWIN یا Open Web Interface for .NET
این استاندارد به صورت کاملا ریز به طراحی هر چیزی را که شما به آن فکر کنید پرداخته است، Request, Cookie, Response, Web Sokcet و ...
اما همانطور که از نامش مشخص است این یک استاندارد است و پیاده سازی ندارد، و هر هاستی باید یک بار این استاندارد را بر

روی خود پیاده سازی کند
خبر خوش این است که تا این لحظه اکثر هاست‌های مهم این استاندارد را پیاده سازی کرده اند و یا در دست پیاده سازی دارند
پروژه Helios برای IIS
پروژه Katana برای IIS به در کنار و سازگار با ASP.NET برای پروژه هایی که تا این لحظه از امکانات سطح پایین ASP.NET استفاده زیادی کرده اند و فرصت تغییر ساختاری ندارند
پروژه هایی برای App Domains و ...
مرحله‌ی بعدی این است که فریم ورک‌ها خوشان را با Owin سازگار کنند
معروف‌ترین فریم ورک هایی که تا این لحظه اقدام به انجام این کار کرده اند، عبارتند از:
ASP.NET Web API
ASP.NET MVC
ASP.NET Identity
ASP.NET Signal R (در حال حاضر Signal R فقط بر روی Owin قابلیت استفاده دارد)
بدیهی است که زمانی که پروژه ASP.NET Web API بر روی استاندارد OWIN نوشته می‌شود، دیگر نیازی به تحمل هزینه مضاعف برای سازگاری خود با انواع هاست ها ندارد و این مسئله توسط Helios، Katana و ... انجام شده است، که بالطبع بزرگترین نفع آن برای ما جلوگیری از چند باره کاری توسط تیم Web API و ... است که بالطبع در زمان کمتر امکانات بیشتری را به ما ارائه می‌دهند.
البته واضح است فریم ورک هایی که با کلاینت و درخواست‌ها کاری ندارند، با این مقولات کاری ندارند، پس Entity Framework و ... از این داستان مستثنا هستند. و علاوه بر این فریم ورک هایی با طراحی اشتباه و قدیمی مانند ASP.NET Web Forms به صورت کلی قابلیت سازگار شدن با این استاندارد را ندارند، زیرا کاملا به ASP.NET وابسته هستند
و در نهایت در مرحله‌ی بعدی لازم است شما نیز از فریم ورک هایی استفاده کنید که مبتنی بر OWIN هستند، یعنی برای مثال پروژه بعدی تان را مبتنی بر ASP.NET MVC و ASP.NET Web API و ASP.NET Identity پیاده سازی کنید، در این صورت شما می‌توانید برنامه ای بنویسید که به Web Server هیچ گونه وابستگی ندارد.
به این صورت کد زدن چند مزیت بزرگ دیگر هم دارد که از کم اهمیت‌ترین آنها شروع می‌کنیم:
1- سرعت بسیار بالاتر برنامه در هاست‌های غیر ASP.NET ای، مانند زمانی که شما از IIS به صورت مستقیم و بدون وابستگی به System.Web.dll استفاده می‌کنید.
توجه کنید که حتی در این حالت هم می‌توانید از ASP.NET Web API و Signal R و Identity استفاده کنید و تا 25% سرعت بیشتری داشته باشید (بسته به سناریو) 2- قابلیت توسعه آسانتر و با قابلیت نگهداری بالاتر پروژه‌های Enterprise، برای مثال در یکی از پروژه‌ها من مجبور بودم از ASP.NET Web API به صورتی استفاده کنم که هم توسط کلاینت JavaScript ای استفاده شود، و هم توسط کدهای Controllerهای MVC (بدون استفاده مستقیم از کد سرویس با رفرنس زدن به سرویس‌ها البته) که خوشبختانه OWIN به خوبی از پس این کار بر آمد، و عملا یک سرویس Web API را هم بر روی IIS هاست کردم و هم داخل یک AppDomain
3- در چند سال آینده که اکثریت مطلق سایت‌ها از این روش استفاده کنند (شما چه بدانید و چه ندانید اگر در برنامه خودتان از Signal R نسخه 2 دارید استفاده می‌کنید، حتما از OWIN استفاده کرده اید)، مایکروسافت می‌تواند دست به تغییرات اساسی‌تری بزند، برای مثال معماری جدیدی از IIS ارائه دهد که مشکلات ساختاری فراوان فعلی IIS را که از حوصله توضیح این مقاله خارج است را نداشته باشد، و فقط یک پیاده سازی OWIN جدید بر روی آن ارائه دهد و برنامه‌های ما **بدون تغییر** بر روی آن نیز کار کنند، و یا این که بتواند تعدادی از فریم ورک‌های با طراحی قدیمی را راحت‌تر از دور خارج کند، مانند Web Forms
نکته پایانی، اگر هم اکنون پروژه ای دارید که در داخل آن از ASP.NET استفاده شده، و برای مثال تعدای فرم ASP.NET Web Forms نیز دارد، نگران نباشید، کماکان می‌توانید از Owin برای سایر قسمت‌ها مانند Web API استفاده کنید، البته در این حالت تاثیری در بهبود سرعت اجرای برنامه مشاهده نخواهید کرد، اما برای مهاجرت و اعمال تغییرات این آسانترین روش ممکن است در قسمت بعدی، مثالی را شروع می‌کنیم مبتنی بر ASP.NET Web API، ASP.NET Identity و Helios

نظرات خوانندگان

نویسنده: ناظم
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۰:۵۱

سلام
ممنون بابت مطلب مفیدتون.

بدون وابستگی به IIS یعنی هر web server ی که OWIN را پیاده سازی کند امکان اجرای برنامه هایی که مثلا با asp.net mvc نوشته شدن رو خواهند داشت؟
همین که مثلا با asp.net mvc برنامه نوشته شده به معنی این هست که برنامه بر اساس استاندارد OWIN هست؟ یا کارهایی برای این منظور باید انجام داد؟

نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۱:۴۸

بدون وابستگی به IIS یعنی شما امکان هاست کردن سرویس‌های Web APi رو به صورت Windows Service یا پروژه Console هم خواهید داشت.
به صورت پیش فرض یک پروژه MVC بدون وابستگی به Owin پیاده سازی می‌شود و برای این منظور می‌توانید یکی از موارد زیر را انجام دهید:
»امکان هاست سرویس‌ها روی IIS. در این صورت Owin فقط به صورت یک Middleware عمل خواهد نمود و در این حالت دیگر نیاز به نوشتن HttpModuleها نخواهید داشت. البته این روش به System.Web وابستگی دارد(**Microsoft.Owin.Host.SystemWeb**)
»استفاده از OwinHost.Exe که در واقع یک پیاده سازی دیگر برای Owin است و عملیات bootstrapping را بر عهده خواهد داشت. در نتیجه شما فقط موارد مربوط به middleWare در application انجام خواهید داد.
»استفاده از Owin Self Hosting برای هاست سرویس‌ها در قالب برنامه Console یا Windows Service
(**Microsoft.Owin.Host.HttpListener**)

نویسنده: یاسر مرادی
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۲:۱۳

بله، به همین معنی است
البته دقت کنید، پیاده سازی OWIN کار ساده ای نیست، و به سرعت نمی‌توان شاهد پیاده سازی آن بر روی هاست‌های مختلف بود، و این پروسه با سرعت فعلی از نظر من مدتی طول خواهد کشید.
برای مثال Katana که یک پیاده سازی قابل استفاده و خوب از آن به شمار می‌رود کار شرکت مایکروسافت است و سایر پیاده سازی Open Source سایر تیم‌ها که بالطبع امکان مانور شرکت مایکروسافت را ندارند، کمی طول می‌کشد تا واقعا آماده استفاده شود.
و همچنین پیاده سازی‌های فعلی در قسمت هایی مانند Web Socketها و سایر مسائل پیچیده دارای ضعف هایی هستند.
درست مانند استاندارد HTML 5 که بر روی مرورگرهای مختلف به میزان‌های مختلفی پیاده سازی شده است.
به بیان دیگر پیاده سازی OWIN صفر و صدی نیست، بلکه هر پیاده سازی ممکن است در داخل خود دارای ضعف‌ها و یا نواقصی باشد.
علاوه بر این اگر شما در کد نویسی ASP.NET MVC خود، بی جهت به امکانات پایه ASP.NET ایجاد وابستگی کنید، نیز در این عمل دچار مشکل خواهید شد، برای همین بدیهتا کاری را که می‌توانید با Action Filter انجام دهید را نباید با یک Http Module انجام دهید و ...
مهم‌ترین کار طراحی برنامه هایی که می‌نویسید به صورت سازگار با OWIN است که در پست‌های بعدی قرار است به همین قسم از مطالب بپردازیم
البته من آینده خوبی برای OWIN قائلم، و نفع آن در کوتاه مدت و بلند مدت کاملا آشکار و واضح است، کما این که در مطلب به آن اشاره شد.

برای مشاهده پیاده سازی‌های مختلف OWIN می‌توانید به سایت owin.org مراجعه کنید.
موفق و پایدار باشید

نویسنده: یاسر مرادی
تاریخ: ۱۳۹۲/۱۲/۰۳ ۱۹:۲۶

ممنون از پاسختون، البته این رو در نظر داشته باشید که استفاده از IIS به همراه Owin لزوما به پیاده سازی Katana یا همان Microsoft.Owin.Host.SystemWeb وابسته نیست، در این حالت شما هیچ گونه بهبود سرعتی رو مشاهده نخواهید کرد و حتی به علت اضافه شدن Owin Middleware بر روی ASP.NET حتی کندتر هم خواهید شد، این حالت فقط برای پروژه هایی توصیه می‌شود که با استفاده از مواردی مانند Moduleهای ASP.NET و یا Web form به ASP.NET وابسته اند. برای پروژه‌های جدید استفاده از Helios که نه به System.Web احتیاجی دارد و نه به Owin.Host.SystemWeb توصیه می‌شود، به همراه SignalR ، Web API و MVC، که به نظر من این ۳ آنقدر کامل و کافی هستند که لزومی به استفاده از ASP.NET System.Web.dll و پیاده سازی Owin مربوطه ای که نام بردید نباشد، تا بتوان بیشتر از مزایای Owin به خصوص کارامدی بیشتر برنامه‌ها بهره برد

نویسنده: مسعود پاکدل
تاریخ: ۱۳۹۲/۱۲/۰۳ ۲۱:۵۵

ممنونم.
در حال حاضر من استفاده از helios رو پیشنهاد نمی‌کنم چون اولین محدودیتی که در helios جلب توجه می‌کند Minimum system requirements مورد نظر است.
برای توسعه پروژه‌های helios :
« Windows 8 یا Windows Server 2012
«NET Framework 4.5.1
«Visual Studio 2012 یا Visual Studio 2013

و برای Web Server نیز :
«Windows Server 2012
«NET Framework 4.5.1
«Full trust مورد نیاز است.
البته به گفته تیم توسعه پروژه helios، احتمال رفع این محدودیت‌ها در آینده وجود دارد. در نتیجه به نظر من Microsoft.AspNet.WebApi.OwinSelfHost گزینه بهتری برای Owin Self Hosting است و از آن جا که در حالت Owin Self Hosting هیچ گونه وابستگی به IIS و البته System.Web نیز وجود ندارد در نتیجه مشکل performance نیز برطرف خواهد شد.

نویسنده: یاسر مرادی
تاریخ: ۱۳۹۲/۱۲/۰۴ ۰:۲

روش برنامه نویسی مایکروسافت بیش از دو سالی می‌شود که به این شکل شده است که هر امکان و قابلیت جدیدی بر روی آخرین نسخه NET Framework. ارائه می‌شود و البته سپس به نسخ قبلی نیز تعمیم می‌یابد، در همین جا است که می‌بینید اکثر امکانات Entity Framework 5 & 6 ابتدا بر روی NET Framework 4.5. ارائه شدند، و سپس بر روی 4
اگر ما بخواهیم به NET Framework. به عنوان یک پیش نیاز دردسر زا نگاه کنیم، در اولین قدم خودمان را به دردسر انداخته ایم، چون نه برای Helios، بلکه برای صدها امکان دیگر مانند Data Flowهای جدید و ... نیز باید صبر کنیم، که عملا هزینه به فایده آن نمی‌صرفد. پس همیشه با فراغ بال از آخرین نسخه NET Framework. استقبال کنیم
نکته‌ی دیگر را که باید مد نظر داشته باشیم، این است که مطابق با سیاست هایی که مایکروسافت جدیدا اتخاذ کرده است، دیگر نباید خیلی نگران نسخه‌های جدید NET Framework. باشیم، چون دیگر از آن نسخه دهی‌های پشت سر هم و با حجم تغییرات بالا خبری نیست، بلکه اکثر فریم ورک‌های مهم جدا از NET. ارائه و به روز رسانی می‌شوند.
علاوه بر این، ارتقا به آخرین نسخه سیستم عامل ویندوز نیز به هیچ وجه مانند قبل (IIS 6 به IIS 7) دردسر زا نیست، و خوشبختانه این ارتقا (و یا تغییر هاست) بدون دردسر است.

به نظر من این ارتقاء را انجام دهید، چون نه فقط Helios که خیلی چیزهای دیگری را دارید از دست می‌دهید، مانند سرعت بالاتر توسعه برنامه بر روی Visual Studio 2013 و Windows 8.1 برای توسعه برنامه‌های وبی، سرعت و کارآمدی بسیار بالاتر NET Framework 4.5.1. با IIS 8.5 برای مشتری‌های برنامه و ...

به نظر من آنقدر این ارتقاء ارزشمند است، که ارزش Helios این میان کمتر ارزشش به چشم می‌آید.

یکی از دلایلی که برنامه‌های سمت وب به سرعت بر برنامه‌های دسکتاپی قدیمی چیره شدند، همین است: امکان ارتقای سرورها در مدت زمان کم و به شکل مدیریت شده و با کمترین تاثیر روی مشتری‌های نهایی، بارها این تصمیم را که در ابتدایش کمی سخت به نظر می‌آید را گرفته ام و در نهایت از مشتری تا برنامه نویس همه را راضی دیده ام، چون هیچ کسی از امکانات جدید که بدون دردسر حاصل شود بدش نمی‌آید، و خوشبختانه کیفیت محصولات مایکروسافت واقعا بهبود یافته و دیگر آن زمانی که از NET 2. به 3.5 می‌رفتیم و گرفتار چندین مشکل می‌شدیم گذشته است.

از این نگذرید که بالاخره روزی باید این مهاجرت‌ها را انجام دهید، پس چه بهتر که از سود آن زودتر بهره مند شوید، البته بی مهابا عمل کردن توصیه نمی‌شود، بد نیست زمانی شروع به ارتقاء کنید که صفحه Release Notes و سوالات موجود در سایت Stack over flow در رابطه با اشکالات رخ داده در زمان ارتقاء و Breaking Changes را از بر باشید، به این صورت عمل کنید تماما برد کرده اید.

عنوان: **ایجاد سیستم وضعیت آب و هوا مانند گوگل (بخش اول)**
نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۱۰ ۱۱:۳۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, C#, ASP.Net MVC, MVC, xml

در این آموزش قصد دارم چگونگی ایجاد یک سیستم اعلام وضعیت آب و هوا را مشابه آنچه که در سایت گوگل می‌بینید برای شما توضیح دهم. باید توجه داشت من این آموزش را با ASP.NET MVC نوشتم ولی شما می‌توانید با اندک تغییراتی در کدها، آنرا در ASP.NET وب فرمز نیز استفاده کنید. برای گرفتن آب و هوای هر شهر از Rssهای اعلام وضعیت آب و هوای یاهو استفاده می‌کنم و توضیح خواهم داد که چگونه با Rss آن کار کنید.
Rss آب و هوای هر شهر در یاهو به صورت یک لینک یکتا می‌باشد؛ به شکل زیر :

```
http://weather.yahooapis.com/forecastrss?w=WOEID&u=c
```

حال می‌خواهم کوئری استرینگ‌های این لینک را برای شما توضیح دهم. هر شهری بر روی کره‌ی زمین یک WOEID یکتا و منحصر بفرد دارد که شما به پارامتر w عدد WOEID شهر موردنظر خود را می‌دهید. بعد از مقداردهی پارامتر w، وقتی این لینک را در آدرس بار مرورگر خود می‌زنید، RSS مربوط به آب و هوای آن شهر را به شما می‌دهد. مثلا WOEID تهران عدد 28350859 می‌باشد.
و این لینک http://weather.yahooapis.com/forecastrss?w=28350859&u=c اطلاعات آب و هوای تهران را در قالب یک RSS به شما نمایش خواهد داد.

خوب، حالا پارامتر دوم یعنی پارامتر u چکاری را انجام می‌دهد؟
* چنانچه مقدار پارامتر u برابر c باشد، یعنی شما دمای آب و هوای شهر مد نظر را بر اساس سانتیگراد می‌خواهید.
* اگر مقدار پارامتر u برابر f باشد، یعنی شما دمای آب و هوای آن شهر مورد نظر را بر اساس فارنهایت می‌خواهید.

برای گرفتن WOEID شهرها هم به این سایت بروید http://woeid.rosselliot.co.nz و اسم هر شهری که می‌خواهید بزنید تا WOEID را به شما نمایش دهد.

در این مثال من از یک DropDown استفاده کردم که کاربر با انتخاب هر شهر از DropDown، آب و هوای آن شهر را مشاهده می‌کند.
Action مربوط به صفحه‌ی Index به صورت زیر می‌باشد :

```
[HttpGet]
        public ActionResult Index()
        {
           ViewBag.ProvinceList = _RPosition.Positions;
            ShowWeatherProvince(8);
            return View();
        }
```

در اینجا من لیست شهرها را از جدول می‌خوانم که البته این جدول را چون بخش مهمی نبود و فقط شامل ID و نام شهرها بود در فایل ضمیمه قرار ندادم و نام شهرها و ID آنها را بر عهده‌ی خودتان گذاشتم.
حال تابعی را که آب و هوای مربوط به هر شهر را نمایش می‌دهد، به شرح زیر است:

```
public ActionResult ShowWeatherProvince(int dpProvince)
        {
            XDocument rssXml=null;
            CountryName CountryName = new CountryName();
            if (dpProvince != 0)
            {
                switch (dpProvince)
                {
                    case 1:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345768&u=c");
```

```
                            CountryName = new CountryName() { Country = "Iran", City = "Azarbayejan-e
Sharqhi" };
                            break;
                        }
                    case 2:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345767&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Azarbayejan-e
Qarbi" };
                            break;
                        }
                    case 3:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254335&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Ardebil" };
                            break;
                        }
                    case 4:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=28350859&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Alborz" };
                            break;
                        }
                    case 5:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345787&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Esfahan" };
                            break;
                        }
                    case 6:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345775&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Ilam" };
                            break;
                        }
                    case 7:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254463&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Bushehr" };
                            break;
                        }
                    case 8:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=28350859&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Tehran" };
                            break;
                        }
                    case 9:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345769&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Chahar Mahall
va Bakhtiari" };
                            break;
                        }
                    case 10:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=56189824&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Razavi
Khorasan" };
                            break;
                        }
                    case 11:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345789&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Shomali
Khorasan" };
                            break;
                        }
                    case 12:
                        {
                            rssXml =
```

```
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345789&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Jonubi
Khorasan" };
                            break;
                        }
                    case 13:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345778&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Khuzestan" };
                            break;
                        }
                    case 14:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2255311&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Zanjan" };
                            break;
                        }
                    case 15:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345784&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Semnan" };
                            break;
                        }
                    case 16:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345770&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Sistan va
Baluchestan" };
                            break;
                        }
                    case 17:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345772&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Fars" };
                            break;
                        }
                    case 18:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=20070200&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Qazvin" };
                            break;
                        }
                    case 19:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2255062&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Qom" };
                            break;
                        }
                    case 20:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345779&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kordestan" };
                            break;
                        }
                    case 21:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254796&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kerman" };
                            break;
                        }
                    case 22:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254797&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kermanshah" };
                            break;
                        }
                    case 23:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345771&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Kohgiluyeh va
```

```
Buyer Ahmad" };
                            break;
                        }
                    case 24:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=20070201&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Golestan" };
                            break;
                        }
                    case 25:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345773&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Gilan" };
                            break;
                        }
                    case 26:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345782&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Lorestan" };
                            break;
                        }
                    case 27:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345783&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Markazi" };
                            break;
                        }
                    case 28:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345780&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Mazandaran" };
                            break;
                        }
                    case 29:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2254664&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Hamedan" };
                            break;
                        }
                    case 30:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2345776&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Hormozgan" };
                            break;
                        }
                    case 31:
                        {
                            rssXml =
XDocument.Load("http://weather.yahooapis.com/forecastrss?w=2253355&u=c");
                            CountryName = new CountryName() { Country = "Iran", City = "Yazd" };
                            break;
                        }
                }
                ViewBag.Location = CountryName;
                XNamespace yWeatherNS = "http://xml.weather.yahoo.com/ns/rss/1.0";
                List<YahooWeatherRssItem> WeatherList = new List<YahooWeatherRssItem>();
                for (int i = 0; i < 4; i++)
                {
                    YahooWeatherRssItem YahooWeatherRssItem = new YahooWeatherRssItem()
                    {
                        Code = Convert.ToInt32(rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("code").Value),
                        Day = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("day").Value,
                        Low = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("low").Value,
                        High = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("high").Value,
                        Text = rssXml.Descendants("item").Elements(yWeatherNS +
"forecast").ElementAt(i).Attribute("text").Value,
                    };

                    WeatherList.Add(YahooWeatherRssItem);
                }
```

```
                ViewBag.FeedList = WeatherList;
            }

                return PartialView("_Weather");
        }
```

قسمت SwitchCase، مقدار و Value مربوط به هر آیتم DropDown را که شامل یک اسم شهر است، میگیرد و RSS مربوط به آن شهر را بر می‌گرداند.
حالا کد مربوط به خواندن فایل Rss را برایتان توضیح می‌دهم : حلقه‌ی for 0 تا 4 (که در کد بالا مشاهده می‌کنید)یعنی اطلاعات 4 روز آینده را برایم برگردان.
من تگ‌های High ، Low ، Day ، Code و text فایل RSS را در این حلقه For می‌خوانم که البته مقادیر این 4 روز را در لیستی اضافه می‌کنم که نوع این لیست هم از نوع YahooWeatherRssItem می‌باشد. من این کلاس را در فایل ضمیمه قرار دادم. اکنون هر کدام از این تگ‌ها را برایتان توضیح می‌دهم:

code : هر آب و هوا کدی دارد .مثلا آب و هوای نیمه ابری یک کد ، آب و هوای آفتابی کدی دیگر و ...
Low: حداقل دمای آن روز را به ما می‌دهد .
High: حداکثر دمای آن روز را به می‌دهد .
day: نام روز از هفته را بر می‌گرداند مثلا شنبه ، یکشنبه و ....
text: که توضیحاتی می‌دهد مثلا اگر هوا آفتابی باشد مقدار sunny را بر می‌گرداند و ...

خوب، تا اینجا ما Rss مربوط به هر شهر را خواندیم حالا در قسمت Design باید چکار کنیم .
کدهای html صفحه‌ی Index ما شامل کدهای زیر است :

```
@{

    ViewBag.Title = "Weather";
}

<link href="~/Content/User/Weather/Weather.css" rel="stylesheet" />
@section scripts{
    <script src="@Url.Content("~/Scripts/jquery-1.6.2.min.js")" type="text/javascript"></script>
        <script src="@Url.Content("~/Scripts/jquery.unobtrusive-ajax.min.js")"
type="text/javascript"></script>
       <script type="text/javascript">
           $("#dpProvince").change(function () {
               $(this).parents("form").submit();
           });
    </script>
}
<h2>Weather</h2>
<div id="Progress">
    <img src="~/Images/User/Other/ajax-loader.gif" />
</div>
<div id="BoxContent"> @Html.Partial("_Weather")</div>

  @using (Ajax.BeginForm(actionName: "ShowWeatherProvince", ajaxOptions: new AjaxOptions {
UpdateTargetId = "BoxContent", LoadingElementId = "Progress", InsertionMode = InsertionMode.Replace }))
                {
<div style="padding-top:15px;">
        <div style="float:left; width:133px; ">Select Your Province</div>
        <div style="float:left">   @Html.DropDownList("dpProvince", new
SelectList(ViewBag.ProvinceList, "Id", "Name"),"Select Your Province", new { @class =
"webUserDropDown", @style = "width:172px" })</div>
    </div>
  }
```

و کدهای _Weather که Partial است به صورت زیر است:

```
@{
    List<Weather.YahooWeatherRssItem> Feeds = ViewBag.FeedList;
}
<div>
```

```
    @{
        HtmlString StartTable = new HtmlString("<table class='WeatherTable' cellspacing='0'
cellpadding='0'><tbody><tr>");
        HtmlString EndTable = new HtmlString("</tr></tbody></table>");
        HtmlString StartTD = new HtmlString("<td>");
        HtmlString EndTD = new HtmlString("</td>");
    }
    <div style="width: 300px;">
        @{
            @StartTable
            foreach (var item in Feeds)
            {
            @StartTD
            <div>@item.Day</div>
            <div>
                @{
                string FileName = "";
                switch (item.Code)
                {
                    case 0: { FileName = "/Images/User/Weather/Tornado.png"; break; }
                    case 1: { FileName = "/Images/User/Weather/storm2.gif"; break; }
                    case 2: { FileName = "/Images/User/Weather/storm2.gif"; break; }
                    case 3: { FileName = "/Images/User/Weather/storm2.gif"; break; }
                    case 4: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 5: { FileName = "/Images/User/Weather/29.gif"; break; }
                    case 6: { FileName = "/Images/User/Weather/29.gif"; break; }
                    case 7: { FileName = "/Images/User/Weather/29.gif"; break; }
                    case 8: { FileName = "/Images/User/Weather/26.gif"; break; }
                    case 9: { FileName = "/Images/User/Weather/drizzle.png"; break; }
                    case 10: { FileName = "/Images/User/Weather/26.gif"; break; }
                    case 11: { FileName = "/Images/User/Weather/18.gif"; break; }
                    case 12: { FileName = "/Images/User/Weather/18.gif"; break; }
                    case 13: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 14: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 15: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 16: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 17: { FileName = "/Images/User/Weather/Hail.png"; break; }
                    case 18: { FileName = "/Images/User/Weather/25.gif"; break; }
                    case 19: { FileName = "/Images/User/Weather/dust.png"; break; }
                    case 20: { FileName = "/Images/User/Weather/fog_icon.png"; break; }
                    case 21: { FileName = "/Images/User/Weather/hazy_icon.png"; break; }
                    case 22: { FileName = "/Images/User/Weather/2017737395.png"; break; }
                    case 23: { FileName = "/Images/User/Weather/32.gif"; break; }
                    case 24: { FileName = "/Images/User/Weather/32.gif"; break; }
                    case 25: { FileName = "/Images/User/Weather/31.gif"; break; }
                    case 26: { FileName = "/Images/User/Weather/7.gif"; break; }
                    case 27: { FileName = "/Images/User/Weather/38.gif"; break; }
                    case 28: { FileName = "/Images/User/Weather/6.gif"; break; }
                    case 29: { FileName = "/Images/User/Weather/35.gif"; break; }
                    case 30: { FileName = "/Images/User/Weather/7.gif"; break; }
                    case 31: { FileName = "/Images/User/Weather/33.gif"; break; }
                    case 32: { FileName = "/Images/User/Weather/1.gif"; break; }
                    case 33: { FileName = "/Images/User/Weather/34.gif"; break; }
                    case 34: { FileName = "/Images/User/Weather/2.gif"; break; }
                    case 35: { FileName = "/Images/User/Weather/freezing_rain.png"; break; }
                    case 36: { FileName = "/Images/User/Weather/30.gif"; break; }
                    case 37: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 38: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 39: { FileName = "/Images/User/Weather/15.gif"; break; }
                    case 40: { FileName = "/Images/User/Weather/12.gif"; break; }
                    case 41: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 42: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 43: { FileName = "/Images/User/Weather/22.gif"; break; }
                    case 44: { FileName = "/Images/User/Weather/39.gif"; break; }
                    case 45: { FileName = "/Images/User/Weather/thundershowers.png"; break; }
                    case 46: { FileName = "/Images/User/Weather/19.gif"; break; }
                    case 47: { FileName = "/Images/User/Weather/thundershowers.png"; break; }
                    case 3200: { FileName = "/Images/User/Weather/1211810662.png"; break; }
                }
                }
                <img alt='@item.Text' title='@item.Text' src='@FileName'>
            </div>
            <div>
                <span>@item.High°</span>
                <span>@item.Low°</span>
            </div>
            @EndTD
            }
        }
        @EndTable
    </div>
```

```
</div>
```

من عکس‌های مربوط به وضعیت آب و هوا را در فایل ضمیمه قرار دادم.
چنانچه در مورد RSS وضعیت آب و هوای یاهو اطلاعات دقیق‌تری را می‌خواهید بدانید به این لینک بروید.
در آموزش بعدی قصد دارم برایتان این بخش را توضیح دهم که بر اساس IP بازدید کننده سایت شما، اطلاعات آب و هوایی شهر بازدید کننده را برایش در سایت نمایش دهد.

Files-06bf65bac63d4dd694b15fc24d4cb074.zip

موفق باشید

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۱ ۰:۵

ممنون از شما. چند نکته جزئی در مورد بهبود کیفیت کدها. در Viewهای MVC باید کدنویسی صرفا به نمایش اطلاعات View خلاصه شود. یعنی switch داخل آن بهتر است تبدیل به یک extension method شده و نهایتا استفاده شود (برای تمیز کردن View).
همچنین استفاده از switch 1 و 2 و 10000 اصطلاحا به magic code مشهور هستند. یعنی مشخص نیست معنای این اعداد چی هست. این‌ها را عموما بهتر است تبدیل به enum کرد و بعد استفاده نهایی.

نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۸:۳۹

ممنونم از شما که مشکل کد نویسی من را گفتید . extension methodها منظورتون همون helperها هست ؟ لینکی دارید که در مورد این extension methodها آموزشی داده باشد چون من می‌خوام کدهامو تا جایی که می‌شود بهینه بنویسم .

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۱۲/۱۲ ۹:۱۵

یک مثال در اینجا
قسمت آخر (**Ifهای شرطی را در Viewها را در متدهای کمکی کپسوله کنید)**

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۹:۵۱

قسمت switch ایی را که در View نوشتید، تبدیل کنید به یک متد کمکی در کلاسی خارج از View: (مهم نیست متد الحاقی باشد یا خیر؛ فقط داخل View نباشد)

```
public static string GetFileName(int code)
{
    switch (code)
    {
       case 0: return "/Images/User/Weather/Tornado.png";
  //...
}
```

بعد یک خاصیت محاسباتی به نام FileName به مدل مورد استفاده اضافه کنید:

```
public class YahooWeatherRssItem
    {
        public int Code { get; set; }
        //...
        public string FileName
        {
          get { return Util.GetFileName(Code); }
        }
    }
```

به این صورت View از کدهای محاسبات یافتن FileName خالی می‌شود.

نویسنده: علی
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۷

یه راه دیگه هم استفاده از Yahoo Query Language هست، می تونید با کوئری زیر اطلاعات آب و هوای شهر مورد نظر رو در قالب JSON دریافت کنید

```
select * from weather.forecast where woeid in (select woeid from geo.placefinder where text="CityName")
```

نویسنده: شقایق اشتری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۱۳

یه سوال دیگه که داشتم اگر بین کدهایمان ، کد html وجود داشت باز هم باید آنها را از view جدا کنیم و به صورت extension method بنویسیم؟

یا اینکه جداسازی کدها از View را زمانی انجام می‌دهیم که چندین خط کد پشت سر هم نوشته باشیم و بین آنها کد html یی نباشد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۱۲ ۱۰:۴۱

- نه الزاما. اگر چندین جا استفاده و تکرار می‌شود یا منطق طولانی دارد، روش « نوشتن HTML Helpers ویژه، به کمک امکانات Razor » می‌تواند مفید باشد.
- منطق قرار گرفته در View فقط باید کار «نهایی» نمایشی را انجام دهد. اگر در View مثلا مستقیما با دیتابیس کار می‌کنید، محاسبات مرتبط با فیلد خاصی را انجام می‌دهید و کلا هر منطقی که «نهایی» بودن نمایش اطلاعات را زیر سؤال ببرد، باید از View جدا شده و به کنترلر و زیرساخت آن منتقل شود.

نویسنده: محسن تقی پور
تاریخ: ۱۳۹۲/۱۲/۲۰ ۰:۸

میشه راجبه این روش کمی بیشتر توضیح بدین

نویسنده: Anaritius
تاریخ: ۱۳۹۲/۱۲/۲۱ ۰:۴۴

سلام
ممنون از مقاله خوبتون
امکانش هست سورس کد webForm ش رو هم بذارید واسه دانلود؟
ممنون

نویسنده: علی
تاریخ: ۱۳۹۲/۱۲/۲۱ ۱۰:۸

اینجا یک مثال از نحوه کار با YQL رو توضیح داده

نویسنده: ع پارسایی
تاریخ: ۱۳۹۳/۱۱/۱۸ ۱۶:۳۶

با سلام
امکان داره بگید در اکشن، قسمت ViewBag.ProvinceList = _RPosition.Positions چه کار میکنه و RPosition کارش چیه؟

نویسنده: ع پارسایی
تاریخ: ۱۳۹۳/۱۱/۲۱ ۱۹:۳۴

با سلام؛ در مطلب گفته شده برای گرفتن WOEID شهرها به فلان سایت بروید و هر کشوری رو بخواییم باید بریم و دستی واردش اما من دیتابیس یا هر چیز دیگری رو میخوام که قابلیت این رو داشته باشه که کاربر با وارد کردن کشور و شهر بتونه آب و هوا رو

ببینه. شما میتونید راهنمایی ام کنید که چه کار باید بکنم و یا منبعی برای دیتابیس مختصات جغرافیایی کشورها وجود داره؟ ممنون میشم کمکم کنید. با تشکر

عنوان: **انجام کارهای زمانبندی شده در برنامه‌های ASP.NET توسط DNT Scheduler**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۱:۴۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Schedule

اگر به دو مطلب استفاده از Quartz.Net (^ و ^) و خصوصا نظرات آن دقت کرده باشید به این نتیجه خواهید رسید که ... این کتابخانه‌ی در اصل جاوایی گنگ طراحی شده‌است. در سایت جاری برای انجام کارهای زمانبندی شده (مانند ارسال ایمیل‌های روزانه خلاصه مطالب، تهیه خروجی PDF و XML سایت، تبدیل پیش نویس‌ها به مطالب، بازسازی ایندکس‌های جستجو و امثال آن) از یک Thread timer استفاده می‌شود که حجم نهایی کتابخانه‌ی محصور کننده و مدیریت کننده‌ی وظایف آن جمعا 8 کیلوبایت است؛ متشکل از ... سه کلاس. در ادامه کدهای کامل و نحوه‌ی استفاده از آن را بررسی خواهیم کرد.

دریافت کتابخانه DNT Scheduler و مثال آن

DNTScheduler.7z

در این بسته، کدهای کتابخانه‌ی DNT Scheduler و یک مثال وب فرم را، ملاحظه خواهید کرد. از این جهت که برای ثبت وظایف این کتابخانه، از فایل global.asax.cs استفاده می‌شود، اهمیتی ندارد که پروژه‌ی شما وب فرم است یا MVC. با هر دو حالت کار می‌کند.

نحوه‌ی تعریف یک وظیفه‌ی جدید

کار با تعریف یک کلاس و پیاده سازی ScheduledTaskTemplate شروع می‌شود:

```
public class SendEmailsTask : ScheduledTaskTemplate
```

برای نمونه :

```
using System;

namespace DNTScheduler.TestWebApplication.WebTasks
{
    public class SendEmailsTask : ScheduledTaskTemplate
    {
        /// <summary>
        /// اگر چند جاب در یک زمان مشخص داشتید، این خاصیت ترتیب اجرای آن‌ها را مشخص خواهد کرد
        /// </summary>
        public override int Order
        {
            get { return 1; }
        }

        public override bool RunAt(DateTime utcNow)
        {
            if (this.IsShuttingDown || this.Pause)
                return false;

            var now = utcNow.AddHours(3.5);
            return now.Minute % 2 == 0 && now.Second == 1;
        }

        public override void Run()
        {
            if (this.IsShuttingDown || this.Pause)
                return;

            System.Diagnostics.Trace.WriteLine("Running Send Emails");
        }

        public override string Name
        {
            get { return "ارسال ایمیل"; }
        }
    }
}
```

- در اینجا Order، ترتیب اجرای وظیفه‌ی جاری را در مقایسه با سایر وظیفه‌هایی که قرار است در یک زمان مشخص اجرا شوند، مشخص می‌کند.
- متد RunAt ثانیه‌ای یکبار فراخوانی می‌شود (بنابراین بررسی now.Second را فراموش نکنید). زمان ارسالی به آن UTC است و اگر برای نمونه می‌خواهید بر اساس ساعت ایران کار کنید باید 3.5 ساعت به آن اضافه نمائید. این مساله برای سرورهایی که خارج از ایران قرار دارند مهم است. چون زمان محلی آن‌ها برای تصمیم گیری در مورد زمان اجرای کارها مفید نیست.
در متد RunAt فرصت خواهید داشت تا منطق زمان اجرای وظیفه‌ی جاری را مشخص کنید. برای نمونه در مثال فوق، این وظیفه هر دو دقیقه یکبار اجرا می‌شود. یا اگر خواستید اجرای آن فقط در سال 23 و 33 دقیقه هر روز باشد، تعریف آن به نحو ذیل خواهد بود:

```
public override bool RunAt(DateTime utcNow)
        {
            if (this.IsShuttingDown || this.Pause)
                return false;

            var now = utcNow.AddHours(3.5);
            return now.Hour == 23 && now.Minute == 33 && now.Second == 1;
        }
```

- خاصیت IsShuttingDown موجود در کلاس پایه ScheduledTaskTemplate، توسط کتابخانه‌ی DNT Scheduler مقدار دهی می‌شود. این کتابخانه قادر است زمان خاموش شدن پروسه‌ی فعلی IIS را تشخیص داده و خاصیت IsShuttingDown را true کند. بنابراین در حین اجرای وظیفه‌ای مشخص، به مقدار IsShuttingDown دقت داشته باشید. اگر true شد، یعنی فقط 30 ثانیه وقت دارید تا کار را تمام کنید.
خاصیت Pause هر وظیفه را برنامه می‌تواند تغییر دهد. به این ترتیب در مورد توقف یا ادامه‌ی یک وظیفه می‌توان تصمیم گیری کرد. خاصیت ScheduledTasksCoordinator.Current.ScheduledTasks، لیست وظایف تعریف شده را در اختیار شما قرار می‌دهد.
- در متد Run، منطق وظیفه‌ی تعریف شده را باید مشخص کرد. برای مثال ارسال ایمیل یا تهیه‌ی بک آپ.
- Name نیز نام وظیفه‌ی جاری است که می‌تواند در گزارشات مفید باشد.

همین مقدار برای تعریف یک وظیفه کافی است.

نحوه‌ی ثبت و راه اندازی وظایف تعریف شده

پس از اینکه چند وظیفه را تعریف کردیم، برای مدیریت بهتر آن‌ها می‌توان یک کلاس ثبت و معرفی کلی را مثلا به نام ScheduledTasksRegistry ایجاد کرد:

```
using System;
using System.Net;

namespace DNTScheduler.TestWebApplication.WebTasks
{
    public static class ScheduledTasksRegistry
    {
        public static void Init()
        {
            ScheduledTasksCoordinator.Current.AddScheduledTasks(
                new SendEmailsTask(),
                new DoBackupTask());

            ScheduledTasksCoordinator.Current.OnUnexpectedException = (exception, scheduledTask) =>
            {
                //todo: log the exception.
                System.Diagnostics.Trace.WriteLine(scheduledTask.Name + ":" + exception.Message);
            };

            ScheduledTasksCoordinator.Current.Start();
        }

        public static void End()
        {
```

```
            ScheduledTasksCoordinator.Current.Dispose();
        }

        public static void WakeUp(string pageUrl)
        {
            try
            {
                using (var client = new WebClient())
                {
                    client.Credentials = CredentialCache.DefaultNetworkCredentials;
                    client.Headers.Add("User-Agent", "ScheduledTasks 1.0");
                    client.DownloadData(pageUrl);
                }
            }
            catch (Exception ex)
            {
                //todo: log ex
                System.Diagnostics.Trace.WriteLine(ex.Message);
            }
        }
    }
}
```

- شیء ScheduledTasksCoordinator.Current، نمایانگر تنها وهله‌ی مدیریت وظایف برنامه است.
- توسط متد ScheduledTasksCoordinator.Current.AddScheduledTasks، تنها کافی است کلاس‌های وظایف مشتق شده از ScheduledTaskTemplate، معرفی شوند.
- به کمک متد ScheduledTasksCoordinator.Current.Start، کار Thread timer برنامه شروع می‌شود.
- اگر در حین اجرای متد Run، استثنایی رخ دهد، آن‌را توسط یک Action delegate به نام ScheduledTasksCoordinator.Current.OnUnexpectedException می‌توانید دریافت کنید. کتابخانه‌ی DNT Scheduler برای اجرای وظایف، از یک ترد با سطح تقدم Below normal استفاده می‌کند تا در حین اجرای وظایف، برنامه‌ی جاری با اخلال و کندی مواجه نشده و بتواند به درخواست‌های رسیده پاسخ دهد. در این بین اگر استثنایی رخ دهد، می‌تواند کل پروسه‌ی IIS را خاموش کند. به همین جهت این کتابخانه کار try/catch استثناهای متد Run را نیز انجام می‌دهد تا از این لحاظ مشکلی نباشد.
- متد ScheduledTasksCoordinator.Current.Dispose کار مدیر وظایف برنامه را خاتمه می‌دهد.
- از متد WakeUp تعریف شده می‌توان برای بیدار کردن مجدد برنامه استفاده کرد.

استفاده از کلاس ScheduledTasksRegistry تعریف شده

پس از اینکه کلاس ScheduledTasksRegistry را تعریف کردیم، نیاز است آن‌را به فایل استاندارد global.asax.cs برنامه به نحو ذیل معرفی کنیم:

```
using System;
using System.Configuration;
using DNTScheduler.TestWebApplication.WebTasks;

namespace DNTScheduler.TestWebApplication
{
    public class Global : System.Web.HttpApplication
    {
        protected void Application_Start(object sender, EventArgs e)
        {
            ScheduledTasksRegistry.Init();
        }

        protected void Application_End()
        {
            ScheduledTasksRegistry.End();
            //نکته مهم این روش نیاز به سرویس پینگ سایت برای زنده نگه داشتن آن است
            ScheduledTasksRegistry.WakeUp(ConfigurationManager.AppSettings["SiteRootUrl"]);
        }
    }
}
```

- متد ScheduledTasksRegistry.Init در حین آغاز برنامه فراخوانی می‌شود.
- متد ScheduledTasksRegistry.End در پایان کار برنامه جهت پاکسازی منابع باید فراخوانی گردد.

همچنین در اینجا با فراخوانی ScheduledTasksRegistry.WakeUp، می‌توانید برنامه را مجددا زنده کنید! IIS مجاز است یک سایت ASP.NET را پس از مثلا 20 دقیقه عدم فعالیت (فعالیت به معنای درخواست‌های رسیده به سایت است و نه کارهای پس زمینه)، از حافظه خارج کند (این عدد در application pool برنامه قابل تنظیم است). در اینجا در فایل web.config برنامه می‌توانید آدرس یکی از صفحات سایت را برای فراخوانی مجدد تعریف کنید:

```
<?xml version="1.0"?>
<configuration>
  <appSettings>
      <add key="SiteRootUrl" value="http://localhost:10189/Default.aspx" />
  </appSettings>
</configuration>
```

همینکه درخواست مجددی به این صفحه برسد، مجددا برنامه توسط IIS بارگذاری شده و اجرا می‌گردد. به این ترتیب وظایف تعریف شده، در طول یک روز بدون مشکل کار خواهند کرد.

گزارشگیری از وظایف تعریف شده

برای دسترسی به کلیه وظایف تعریف شده، از خاصیت ScheduledTasksCoordinator.Current.ScheduledTasks استفاده نمائید:

```
var jobsList = ScheduledTasksCoordinator.Current.ScheduledTasks.Select(x => new
{
   TaskName = x.Name,
   LastRunTime = x.LastRun,
   LastRunWasSuccessful = x.IsLastRunSuccessful,
   IsPaused = x.Pause,
}).ToList();
```

لیست حاصل را به سادگی می‌توان در یک Grid نمایش داد.

نظرات خوانندگان

نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۹:۲۲

آیا یکی از کاربردهای این مطلب می‌تونه مثلا ارسال یک ایمیل یک هفته قبل از اتمام زمان شارژ کاربری باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۶ ۱۹:۳۷

بله. می‌توانید یک وظیفه‌ی جدید تعریف کنید که هر شب ساعت مثلا 11 و 15 دقیقه اجرا شود (نحوه‌ی تعریف متد RunAt). سپس در متد Run آن یک کوئری از دیتابیس گرفته، لیست موارد مدنظر را واکشی کرده و به آن‌ها ایمیل بزنید.

نویسنده: پیمان مهربانی
تاریخ: ۱۳۹۲/۱۲/۲۷ ۱۱:۴۷

کتابخانه سبکی بود اما نگرانی هایی مانند پیش آمدن همزمانی در اجرای وظایف را دارم.پیش از این از کتابخانه quartz scheduler در یک پروژه بزرگ استفاده کرده بودیم و نتیجه کار بسیار راضی کننده بود ، حتی می‌توان از Scheduler خود ویندوز و یا Jonهای SQL Server هم بهره برد.

این روش چه مزایا و معایبی نسبت به روش‌های موجود دارد و آیا توصیه شده برای استفاده در پروژه‌های بزرگ هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۷ ۱۲:۲

- «پیش آمدن همزمانی در اجرای وظایف»
خاصیت Order را برای وظایفی که قرار است در یک زمان مشخص اجرا شوند، مقدار دهی کنید. 1و 2 و 3 و الی آخر.
- «حتی می‌توان از Scheduler خود ویندوز و یا Jonهای SQL Server هم بهره برد».
بله. به شرطی‌که سرور در اختیار شما باشد و دسترسی کافی برای انجام اینکار را داشته باشید. البته در این حالت خاص، مدیریت آن یکپارچه با یک برنامه‌ی وب نیست.
در سرورهای اشتراکی روش ارائه شده در این مطلب بدون نیاز به سطح دسترسی خاصی کار می‌کند. ضمنا برای ASP.NET نوشته شده است و این قابلیت را دارد که به شما اعلام کند مثلا تا 30 ثانیه دیگر برنامه از سرور unload می‌شود؛ توسط خاصیت IsShuttingDown. همچنین حق تقدم ترد آن طوری تنظیم شده که سبب اختلال در عملیات و عملکرد متداول سایت نشود.
- «آیا توصیه شده برای استفاده در پروژه‌های بزرگ هست؟»
یک به اشتراک گذاری بود از قسمتی از کدهای زیر ساخت سایت جاری که هم اکنون مورد استفاده است (مقدمه بحث).

نویسنده: سلمان کاظمی
تاریخ: ۱۳۹۲/۱۲/۲۸ ۱:۳۲

سوالی که واسه من پیش اومده اینه که من یک نرم افزار Web Form دارم . هدفم اینه که یک روز قبل از تاریخ تولد اعضا ایمیلی با عنوان تولدتان مبارک ارسال شه .حالا من باید این کدهای گفته شده را در برنامه خود بیاورم یا یک وب سرویس بنویسم که اینکارو انجام بده؟اگه بخوام که در برنامه انجام بشه خوب کجا باید نوشت ؟ یا به عبارتی دیگه باید حتما برنامه اجرا بشه و اگه برنامه اصلا Run نشه چی میشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۱۲/۲۸ ۸:۵۹

- وب سرویس فقط با یک درخواست رسیده کار می‌کند. کار کتابخانه‌ی فوق، اجرا در پس زمینه‌ی برنامه به صورت مداوم است.
- فقط دو حالت وجود دارد که برنامه اجرا نشود:

الف) protected void Application_End فراخوانی شود. متد WakeUp نوشته شده برای این منظور و راه اندازی مجدد برنامه توسط آن، کافی است.
ب) کل سرور ری استارت شود (نه فقط برنامه). در این حالت کافی است آدرس برنامه را به یکی از سرویس‌هایی که هر از چندگاهی برنامه را ping می‌کنند، معرفی کنید.

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۳/۰۱/۰۱ ۲۳:۲۳

خیلی خیلی ممنون ... کتابخانه‌ی ساده و مفیدی است.
سوال من اینه که میشه کاری کرد که آدرس روت تعریف شده در فایل کانفیگ جهت بیدار شدن IIS را در کد نوشت مثل کد زیر :

```
var urlToWakeup = Request.Url.Scheme + Uri.SchemeDelimiter + Request.Url.Host +
            (Request.Url.IsDefaultPort ? "" : ":" + Request.Url.Port);
```

که مجبور نباشی در هر سایتی که طراحی میکنیم آدرس رو در فایل کانفیگ ست کنیم !
پی نوشت : من این کار رو در متد Application_End نوشتم و خطای در دسترس نبودن Request رو دریافت کردم ! (هر چند که میدونستم این خطا رو میده !)

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۰۲ ۰:۵۱

می‌شود در Application_BeginRequest اطلاعات آدرس ریشه سایت را در یک متغیر استاتیک ذخیره کرد. مقدار آن در Application_End قابل استفاده است:

```
namespace TestApp
{
    public static class App
    {
        public static string SiteRootUrl;
    }

    public class TestApplication : HttpApplication
    {
        protected void Application_BeginRequest(object sender, EventArgs e)
        {
            if (string.IsNullOrWhiteSpace(App.SiteRootUrl))
            {
                App.SiteRootUrl = Request.Url.GetLeftPart(UriPartial.Authority) +
Request.ApplicationPath;
            }
        }

        protected void Application_End()
        {
            // use App.SiteRootUrl
        }
    }
}
```

نویسنده: امیرحسین جلوداری
تاریخ: ۱۳۹۳/۰۱/۰۲ ۱۳:۲

خیلی خیلی ممنون :) ... اگه میشه نسخه‌ی nuget کتابخانه را هم بذارید

نویسنده: مهدی پایروند
تاریخ: ۱۳۹۳/۰۱/۲۵ ۱۴:۳۱

1.برای زنده نگهداشتن سایت میشه از خود همین کتابخانه استفاده کرد؟
2.برای چه تعداد جاب بنظرتون منطقیه که استفاده بشه؟
ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۵ ۱۶:۴۳

- کمی بالاتر توضیح دادم « فقط دو حالت وجود دارد که برنامه اجرا نشود: ... »
- این مورد فقط بستگی به توان سرور شما دارد.

نویسنده: fss
تاریخ: ۱۳۹۳/۰۱/۲۸ ۹:۶

در حالتی که "کل سرور ری استارت شود (نه فقط برنامه) " ، بعد از بالا آمدن سرور و شروع به کار IIS، تابع Application_Start کار شروع برنامه را خود به خود انجام نمی‌دهد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۸ ۹:۱۴

نه تا زمانیکه اولین درخواستی به برنامه برسد.
کل سرور ری استارت شده. IIS برنامه را فقط زمانی مجددا بارگذاری می‌کند که درخواست نمایش یکی از قسمت‌های سایت به آن ارسال شود.
البته IISهای جدید قابلیت Auto-Start هم دارند؛ ولی باید در تنظیمات Application pool برنامه انتخاب شود:

همچنین Application Initialization Module نیز برای اجرا خودکار برنامه پس از ری‌استارت سرور طراحی شده .

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۷ ۹:۶

یک نکته‌ی تکمیلی

در دات نت *4.5.2* ، متدی به نام HostingEnvironment.QueueBackgroundWorkItem اضافه شده‌است تا درخواست اجرای کارهای پس زمینه در ASP.NET به سادگی و همچنین با اطمینان بیشتری قابل انجام باشد.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۳:۴۰

یعنی با استفاده از این متد جدیدی که در دات نت 4.5.2 اضافه شده ، دیگه نیازی به Quartz.net و یا DNT scheduler نیست ؟ و میتوان برای کارهایی که نیاز به زمانبندی دارند از این متد استفاده کرد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۳:۴۲

فقط قسمت مدیریت تردهای آن با یک سطر QueueBackgroundWorkItem جایگزین می‌شود. مابقی قسمت‌های آن (مانند اینکه یک WorkItem چه زمانی در Queue قرار گیرد) تفاوتی نمی‌کند و مانند قبل است.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۴:۱۰

منظور شما این خط هست ؟

```
ScheduledTasksCoordinator.Current.AddScheduledTasks(
                new SendEmailsTask(),
                new DoBackupTask());
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۷ ۱۴:۲۳

این‌ها باقی خواهند ماند. قسمت new Thread و مدیریت آن جایگزین می‌شود.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۹ ۹:۲۵

یک نکته

مثالی از نحوه‌ی استفاده از متد جدید HostingEnvironment.QueueBackgroundWorkItem
+ یک مثال رسمی

نویسنده: Aria
تاریخ: ۱۳۹۳/۰۳/۱۹ ۲۳:۱۰

با سلام
من روی یه پروژه مالی دارم با ASP WebForm کار میکنم و میخواستم در اول هر ماه یک سری دستورات ویرایش یا update را که به صورت رکورد در یک جدول ذخیره می‌کنم انجام بشن. سوالم اینه که برای این که مجموعه دستورات من انجام بشن باید برنامه جتما Run باشه و چه جوری میشه اونو به صورت پس زمینه در حال اجرا نگه داشت.
با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۲۰ ۰:۱۹

- به اندازه کافی در نظرات این بحث در مورد زنده نگه داشتن یک برنامه ASP.NET بحث شده. کمی وقت بگذارید و آن‌ها را مطالعه کنید.
+ اگر برنامه مالی است، احتمالا دسترسی کاملی به سرور و همچنین SQL Server (اگر با آن کار می‌کنید) دارید. در این حالت برای به روز رسانی زمانبندی شده‌ی چند رکورد شاید بهتر باشد از سرویس معروف و همیشه در حال اجرای SQL Server agent استفاده کنید. در اینجا نیز می‌شود یک job را که متشکل از دستورات T-SQL است، در فواصل زمانی مشخصی اجرا کرد.

نویسنده: میثم
تاریخ: ۱۳۹۳/۰۳/۲۵ ۱۳:۴۸

سلام.
منظورتون فقط همین یک خط توی متد Start کلاس ScheduledTasksCoordinator بود که تغییر میکنه؟
یعنی فقط متد Start به شکل زیر بازنویسی میشه دیگه؟

```
public void Start()
        {
            _timer.OnTimerCallback = () =>
            {
                var now = DateTime.UtcNow;
                var taskToRun = _tasks.Where(x => !x.IsRunning && x.RunAt(now)).OrderBy(x =>
x.Order).ToList();
                if (_isShuttingDown || !taskToRun.Any())
                    return;

                HostingEnvironment.QueueBackgroundWorkItem(x => taskAction(taskToRun));
            };
        }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۲۵ ۱۵:۱

- بله. قسمت‌های HostingEnvironment.RegisterObject و IRegisteredObject آن هم باید حذف شوند چون در QueueBackgroundWorkItem وجود دارند و یک CancellationToken را تنظیم می‌کند.
+ زمانیکه از DNTScheduler استفاده می‌کنید، عملا نیازی به QueueBackgroundWorkItem ندارید. چون نکته‌ی HostingEnvironment.RegisterObject و IRegisteredObject در آن لحاظ شده. این نکته که خاموش شدن IIS را گزارش می‌کند، چند سال قبل، توسط یکی از اعضای قبلی تیم ASP.NET منتشر شده بود . دقیقا از همین نکته در QueueBackgroundWorkItem استفاده شده.

به صورت خلاصه، DNTScheduler با دات نت 4 به بعد سازگار است و نکات QueueBackgroundWorkItem دات نت 4.5.2 را به صورت توکار پیاده سازی کرده‌است.

نویسنده: حسین
تاریخ: ۱۳۹۳/۰۴/۱۷ ۱۱:۲

سلام. اگر بخواهیم یک کار نسبتاً زمانبر که IO هم هست را توسط این کتابخانه در فواصل زمانی معین اجرا کنیم، میشه از async و await استفاده کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۱۷ ۱۱:۲۰

بله. از روش و کتابخانه معرفی شده در مطلب « استفاده از async و await در برنامه‌های کنسول و سرویس‌های ویندوز NT » استفاده کنید.

نویسنده: حمید حسین وند
تاریخ: ۱۳۹۳/۰۵/۰۹ ۱۰:۱۹

سلام
من برای این سری کارها از ویندوز سرویس استفاده می‌کنم. مثلا ویندوز سرویس من از ساعت 8 صبح شروع به کار می‌کنه و رویدادهایی مثل سالروز تولد رو با استفاده از پیامک به کاربران پیام تبریک ارسال می‌کنه.
مهمترین عاملی که باعث شد من از ویندوز سرویس استفاده کنم اجرای مداوم وهمیشگی بدون ارسال درخواست به وب سایت من بود. ولی فکر می‌کنم این کتابخانه شما هم مثل ویندوز سرویس عمل می‌کنه و خودش همیشه در حال اجراست.
حالا به نظرتون آیا از ویندوز سرویس استفاده کنم بهتره و یا اینکه از این کتابخانه استفاده کنم؟
ممنون

نویسنده: محسن خان
تاریخ: ۱۳۹۳/۰۵/۰۹ ۱۱:۳

فرض کن داری از یک هاست اشتراکی استفاده می‌کنی. دسترسی ادمین هم روی سرور نداری برای اجرا سرویس ویندوز یا فرض کن در یک سازمان بهت گفتن ما فقط اجازه می‌دیم فایل‌های سایتت رو روی سرور کپی کنی. دسترسی بیشتری بهت نمی‌دیم. اون وقت چکار می‌کنی؟

نویسنده: Mohammad
تاریخ: ۱۳۹۳/۰۵/۱۵ ۲:۵۶

سلام. ممنون.

دوستان برای من وقتی به صورت ساعت می‌دم درست کار نمیکنه.مثلا من می‌خوام در ساعت دو و بیست و چهار دقیقه بامداد یه کاری رو انجام بده اینجوری نوشتم:در صورتی اگه بگم دو دقیقه دو دقیقه درست انجام میشه.به نظرتون مشکل از کجاست؟ممنون

```
var now = utcNow.AddHours(3.5);
    return now.Hour == 2 && now.Minute == 24 && now.Second == 1;
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۱۵ ۸:۴۶

مشکل از تنظیم نبودن ساعت سرور است. مقدار DateTime.UtcNow را روی سرور بررسی کنید و با مقدار واقعی تطابق بدید. بعد اختلافش را باید در همینجا اعمال کنید. مثلا اگر پس از بررسی متوجه شدید ساعت سرور یک ساعت عقب هست، now.Hour == 2 می‌شود now.Hour == 1 و امثال این نوع محاسبات.

نویسنده: علی
تاریخ: ۱۳۹۳/۰۶/۱۵ ۱۸:۳۴

ممنون. مفید بود. اگه بخواهیم یه کاری مثلا هر سه روز یکبار انجام بشه باید چه جوری زمان رو تعیین کنیم؟ ممنون

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۵ ۱۹:۵۷

```
public override bool RunAt(DateTime utcNow)
        {
            if (this.IsShuttingDown || this.Pause)
                return false;

            var now = utcNow.AddHours(3.5);
            return (now.Day % 3 == 0) && (now.Hour == 0 && now.Minute == 1 && now.Second == 1);
        }
```

نویسنده: مهدی
تاریخ: ۱۳۹۳/۰۶/۱۷ ۲۲:۲۲

یه سوال:اختلاف زمانی ساعت گرینویچ با تهران همیشه 3:30 هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۷ ۲۳:۸

اگر سرور شما تنظیمات daylight saving time صحیحی داشته باشد، بله. اگر نه، خودتان این یک ساعت تفاوت را 6 ماه یکبار باید محاسبه و اعمال کنید.

نویسنده: آرام
تاریخ: ۱۳۹۳/۰۶/۱۸ ۹:۵۷

سلام.
چرا هیچ ارجاعی به متد Stop در کلاس JobsRunnerTimer وجود نداره؟ برای این کار دلیلی دارید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۸ ۱۰:۱۴

نیازی نیست. چون طول عمر کل این ماژول دقیقا معادل طول عمر برنامه‌ی وب است. خاتمه‌ی آن هم به صورت خودکار با از حافظه خارج کردن AppDomain برنامه توسط IIS انجام می‌شود. تا زمانیکه برنامه در حال اجرا است این ماژول هم به همین ترتیب. هر زمان که IIS تصمیم به خاتمه‌ی برنامه گرفت، نه این ماژول، هیچ ماژول دیگری هم فرصت مقاومت پیدا نمی‌کند و راسا به همراه AppDomain جاری خاتمه می‌یابد.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۸:۴۳

ممنون، عالیه
فقط یه سوال:
چطور می‌تونیم یک Task خاص رو خودمون فراخوانی کنیم، مثلاً بعداً از اینکه کاربر در سایت ثبت‌نام کرد به صورت هفتگی براش ایمیل ارسال بشه یا مواردی از این دست، در مطلب بالا این طوری که من متوجه شدم، توسط دستور ScheduledTasksRegistry.Init(); در Application_Start تمام وظایف به صورت همزمان شروع میشن، در این حالت باید برای هر وظیفه یک کلاس برای ریجستر کردن اون بنویسیم؟ درسته؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۷ ۱۸:۴۸

بله. هر وظیفه نیاز به طراحی مجزا دارد. ثبت و معرفی آن توسط متد ScheduledTasksCoordinator.Current.AddScheduledTasks در ScheduledTasksRegistry.Init انجام می‌شود.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۰۸/۰۷ ۲۳:۵۰

ممنون،
آیا انجام چنین کاری توسط این کتابخانه میسر است؟
به عنوان مثال: ابتدا یک وظیفه را توسط ScheduledTasksRegistry.Init اجرا می‌کنم، این وظیفه قرار است به مدت 6 روز، روزانه کاری را انجام دهد، بعد از پایان روز 6 یعنی در روز هفتم یک وظیفه دیگر شروع خواهد شد(تعداد روزها توسط کاربر انجام می‌گیرد). و در نهایت وظایف یکی پس از دیگری بعد زمان تعیین شده برای آنها متوقف خواهند شد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۸ ۰:۱

در متد RunAt بازه زمانی را مشخص کنید. در این مثال‌ها یک سری عدد ثابت است؛ شما آن‌ها را از دیتابیس بخوانید. فقط دقت داشته باشید که خروجی true/false این متد ثانیه‌ای یکبار بررسی می‌شود. بنابراین مباحث caching را جهت واکشی اطلاعات از بانک اطلاعاتی در متد RunAt اعمال کنید تا بی‌جهت رفت و برگشت اضافی صورت نگیرد.

نویسنده: حسن
تاریخ: ۱۳۹۳/۰۸/۰۸ ۱:۴۵

سلام
ایا میشه از این کتابخانه به صورت ثانیه ای استفاده کرد ؟ در هر ثانیه چک کنه و در صورت داشتن وظایف در این ساعت و دقیقه و ثانیه (که لیست اونها در دیتابیس هست) بهشون عمل کنه !
مثلا اینطوری اگر کار کنه مشکلی پیش نمیاد ؟

```
return now.Second  % 1 == 0 && now.Second == 1;
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۰۸ ۱:۵۴

برای این حالت در متد RunAt فقط بنویسید return true. چون این متد ثانیه‌ای یکبار فراخوانی می‌شود.

نویسنده: سعید شیرعلی
تاریخ: ۱۳۹۳/۰۹/۰۲ ۲۳:۳۳

سلام؛ فایلی که برای دانلود گذاشتین پر از اشکاله. وقتی بازش میکنم با asp زیر خیلی از کداش خط قرمز میکشه، مشکل از کجاست، آیا سورس مشکل داره؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۳ ۰:۲۸

دقیقا چه خطایی گرفتید؟

نویسنده: سعید شیرعلی
تاریخ: ۱۳۹۳/۰۹/۰۳ ۱۰:۱۱

چند بار بازو بستش کردم خیلی از خطاهاش رفتم نمیدونم دلیلش چیه اما هنوز یک خطا داره
خط اول Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="DNTScheduler.TestWebApplication.Default" %>
```

خطا

```
Error 1 Could not load type 'DNTScheduler.TestWebApplication.Default'.
C:\Users\KING\Desktop\DNTScheduler\DNTScheduler.TestWebApplication\Default.aspx 1
```

نویسنده: سعید شیرعلی
تاریخ: ۱۳۹۳/۰۹/۰۳ ۱۰:۵۵

مشکل را حل کردم. وارد قسمت پروپرتیس شدم و استارترشو عوض کردم درست شد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۳ ۱۰:۵۵

- این پروژه باید یکبار کامل کامپایل شود و همچنین پوشه‌ی bin آن موجود باشد.
- خطای Could not load type صرفا به معنای عدم کامپایل شدن برنامه و یا نبود فایل‌های dll مرتبط در پوشه‌ی bin است.
مراجعه کنید به منوی Build و گزینه‌ی Rebuild solution را انتخاب کنید.

نویسنده: سعید شیرعلی
تاریخ: ۱۳۹۳/۰۹/۰۳ ۱۲:۴۶

سلام؛ میخوام هر 15 ثانیه یک بار کدم اجرا بشه. این کارو کردم اما نشد:

```
return now.Second % 15 == 0 && now.Second == 1;
```

لطفا راهنمایی کنید.

```
<add key="SiteRootUrl" value="http://localhost:10189/Default.aspx" />
```

«همینکه درخواست مجددی به این صفحه برسد، مجددا برنامه توسط IIS بارگذاری شده و اجرا می‌گردد. به این ترتیب وظایف تعریف شده، در طول یک روز بدون مشکل کار خواهند کرد.»

- یعنی صفحه ایندکس سایت را بهش بدیم مشکلی پیش نمیاد.
- اگر برنامه در حال انجام کار باشه بعد یک کاربری وارد سایت بشه کد در حال اجرا از کار میفته یا فقط تایمرو زنده نگه میداره

با تشکر

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۳ ۱۲:۵۲

- در شرطی که نوشتید «now.Second == 1» اضافی است.
- خیر؛ مشکلی نخواهد بود.
- این کارها پس زمینه هستند. اصلا ارتباطی به UI برنامه ندارند و در ترد دیگری اجرا می‌شوند.

نویسنده: مهدی خیراندیش
تاریخ: ۱۳۹۳/۰۹/۰۹ ۱۶:۳

ممنون؛ اما من یه مشکلی دارم ساعت اجرا رو 1 بامداد تنظیم کردم ساعت سرور رو هم چک کردم تنظیمه اما فکر کنم اجرای ترد به اون موقع نمیرسه! اوایل ترد اجرا میشه اما همیشه موقعی که متد بیدارکننده! رو صدا میزنه خطاهای Server 503 Unavailable و Internal Server Error 500 پرت (throw میشه).

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۹ ۱۸:۱۳

- از ELMAH برای ثبت خودکار ریز جزئیات خطاهای مدیریت نشده و بررسی آتی آن‌ها استفاده کنید.
- همچنین در کتابخانه‌ی فوق به قسمت ScheduledTasksCoordinator.Current.OnUnexpectedException هم دقت داشته باشد؛ مطابق مثال ارائه شده.

عنوان: **صفحه بندی پویا در Entity Framework**
نویسنده: محمد رضا صفری
تاریخ: ۱۳۹۳/۰۱/۲۶ ۹:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Entity framework, Paging

در اکثر برنامه‌ها ما نیازمند این موضوع هستیم که بتوانیم اطلاعاتی را به کاربر نشان دهیم. در بعضی از موارد این اطلاعات بسیار زیاد هستند و نیاز است در این حالت از صفحه بندی اطلاعات یا Data Paging استفاده کنیم. در ASP.NET برای ارائه اطلاعات به کاربر معمولا از کنترلهای ListView ، Gridview و امثالهم استفاده می‌شود. مشکل اساسی این کنترل‌ها این است که آنها اطلاعات را به صورت کامل از سرور دریافت کرده، سپس اقدام به نمایش صفحه بندی شده آن می‌نمایند که این موضوع باعث استفاده بی مورد از حافظه سرور شده و هزینه زیادی برای برنامه ما خواهد داشت.
صفحه بندی در سطح پایگاه داده بهترین روش برای استفاده بهینه از منابع است. برای رسیدن به این مقصود ما نیاز به یک کوئری خواهیم داشت که فقط همان صفحه مورد نیاز را به کنترلر تحویل دهد.
با استفاده از متد توسعه یافته زیر می‌توان به این مقصود دست یافت:

```
/// <summary>
/// صفحه بندی کوئری
/// </summary>
/// <param name="query">کوئری مورد نظر شما</param>
/// <param name="pageNum">شماره صفحه</param>
/// <param name="pageSize">سایز صفحه</param>
/// <param name="orderByProperty">ترتیب خواص</param>
/// <param name="isAscendingOrder">اگر برابر با <c>true</c> است صعودی باشد</param>
/// <param name="rowsCount">تعداد کل ردیف ها</param>
/// <returns></returns>
private static IQueryable<T> PagedResult<T, TResult>(IQueryable<T> query, int pageNum, int pageSize,
                Expression<Func<T, TResult>> orderByProperty, bool isAscendingOrder, out int rowsCount)
{
    if (pageSize <= 0) pageSize = 20;

    //مجموع ردیف‌های به دست آمده
    rowsCount = query.Count();

// اگر شماره صفحه کوچکتر از 0 بود صفحه اول نشان داده شود
    if (rowsCount <= pageSize || pageNum <= 0) pageNum = 1;

// محاسبه ردیف هایی که نسبت به سایز صفحه باید از آنها گذشت
    int excludedRows = (pageNum - 1) * pageSize;

    query = isAscendingOrder ? query.OrderBy(orderByProperty) :
query.OrderByDescending(orderByProperty);

// ردشدن از ردیف‌های اضافی و  دریافت ردیف‌های مورد نظر برای صفحه مربوطه
    return query.Skip(excludedRows).Take(pageSize);
}
```

نحوه استفاده :

فرض کنید که کوئری مورد نظر قرار است تا یکسری از مطالب را از جدول Articles نمایش دهد. برای دریافت 20 ردیف اول جهت استفاده در صفحه اول، از کد زیر استفاده می‌کنیم :

```
var articles = (from article in Articles
                where article.Author == "Abc"
                select article);

int totalArticles;

var firstPageData =  PagedResult(articles, 1, 20, article => article.PublishedDate, false, out
totalArticles);
```

یا به صورت ساده‌تر و قابل اجرا به صورت کلی‌تر :

```
var context = new AtricleEntityModel();
var query = context.ArticlesPagedResult(articles, <pageNumber>, 20, article => article.PublishedDate,
false, out totalArticles);
```

نظرات خوانندگان

نویسنده: محمد رضا ایزدی
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۱:۰

یه سوالی خط آخر چطوری اجرایی شده
شما تو کانتکس اون متد رو اوردین ؟
+

```
Helper.PagedResult(t, 1, 10, o=>true, false, out i);
```

یه اور لود هم اینطوری میشه براش نوشت اینطوری نیاز نیست حتما order در نظر گرفته شود

نویسنده: میثم99
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۱:۲۳

باسلام
مطلبت بسیار مفیدی بود. فقط اگر بتوانی خود صفحه بندی را هم قرار دهی بسیار عالی میشود.
منظورم چند تا لینک که صفحه اول و آخر و صفحه جاری و تعداد دارد.
ممنون

نویسنده: محسن عباس آبادعربی
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۱:۴۷

ضمن تشکر از مطلب فوق
اگر شما در ASP.net استفاد میکنید میتوانی از کنترل ObjectContainerDataSource استفاده کنی که چند مزیت دارد
1 سرعت بالایی دارد
2 امکان sql cache dependency رو فعال میکنه یعنی فقط در هنگامی که شما اطلاعات رو در داخل گرید لود می‌کنید برای دفعه بعد اگر اطلاعات در دیتابیس تغییری نکرده باشد دیگر به سمت دیتابیس مراجعه نمی‌کند و اطلاعات از cache خوانده می‌شود
2 امکان paging سمت سرور رو به شما میدهد .
3 برای پروژهای با دیتای بزرگ تست شده و جواب داده

نویسنده: میثم99
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۲:۹

در mvc هم یک هلپر برای اینکار ساخته شده است که خیلی خوب کار می‌کند.
ولی منظور من یک ماژول ساده دست ساز بود که با توجه به سایت بتوان به هر شکل دلخواهی آنرا تغییر داد. در بعضی از پروژه‌ها واقعا یک همچین چیزی بدرد می‌خورد

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۲:۳۳

برای طراحی Pager سازگار با بوت استرپ این مطلب مفید است:
A simple Bootstrap Pager Html Helper

نویسنده: ایزدی
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۴:۱۸

ObjectContainerDataSource میشه یه توضیحی در موردش بدین یا یه مقاله معرفی کنید

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۱/۲۶ ۱۴:۲۷

ObjectContainerDataSource Control

نویسنده: محمد رضا صفری
تاریخ: ۱۳۹۳/۰۱/۲۶ ۲۰:۳۲

کسانی که Table بی دردسر و Ajax ی میخوان از این پلاگین استفاده کنند :

/http://www.jtable.org

با MVC هم کاملا سازگار هست و نمونه هم داره .

آموزش کامل : http://www.codeproject.com/Articles/277576/AJAX-based-CRUD-tables-using-ASP-NET-MVC-and-jTa

Datatable هم هست .

نویسنده: محسن عباس آبادعربی
تاریخ: ۱۳۹۳/۰۱/۲۷ ۱۲:۲۲

کنترل ObjectContainerDataSource Control مربوط به فزیم ورک WCSF می‌باشد می‌توانید از سایت مایکروسافت دانلود نمایید.

نویسنده: ایزدی
تاریخ: ۱۳۹۳/۰۶/۰۸ ۱۶:۵۶

سلام
من یک برنامه تولید لایه business نوشتم از کد شما هم استفاده کردم با اجازتون یه تغییر کوچیک دادم توش خواستم اینجا هم بذارم که اگر کسی خواست استفاده کنه

```
private static IQueryable<T> PagedResult<T, TResult>(IQueryable<T> query, int pageNum, int pageSize,
                                                     Expression<Func<T, TResult>>
orderByProperty,
                                                     bool isAscendingOrder, out int rowsCount,
                                                     Expression<Func<T, bool>> whereClause =
null)
        {
            if (pageSize <= 0) pageSize = 20;

            //مجموع ردیف‌های به دست آمده
            rowsCount = query.Count();

            // اگر شماره صفحه کوچکتر از 0 بود صفحه اول نشان داده شود
            if (rowsCount <= pageSize || pageNum <= 0) pageNum = 1;

            // محاسبه ردیف هایی که نسبت به سایز صفحه باید از آنها گذشت
            int excludedRows = (pageNum - 1) * pageSize;

            query = isAscendingOrder ? query.OrderBy(orderByProperty) :
query.OrderByDescending(orderByProperty);

            //جستجو را در صورت لزوم انجام می‌دهد

            query = whereClause == null ? query : query.Where(whereClause);


            // ردشدن از ردیف‌های اضافی و دریافت ردیف‌های مورد نظر برای صفحه مربوطه
            return query.Skip(excludedRows).Take(pageSize);
        }
```

و برای فراخوانی هم اینطور استفاده کردم

```
public static List<t_Prodcts> GetPaging(int currentPage, int pageSize, out int count,
                                                Expression<Func<t_Prodcts, bool>> search = null)
        {
            using (var db = new asusIranDBConnection())
            {
                return PagedResult(db.t_Prodcts, currentPage, pageSize, o => true, false, out count,
search).ToList();

            }
        }
```

عنوان: **پالایش درخواست ها در IIS**
نویسنده: امین مصباحی
تاریخ: ۱۳۹۳/۰۱/۳۰ ۱۹:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, Security

شاید شما هم قصد داشته باشید تا از برخی درخواست‌ها به وب سایت یا اپلیکیشن خود ممانعت عمل بیاورید. نظیر درخواست‌های SQL Injection یا برخی Query Stringهای خاص یا برخی درخواست‌های مزاحم.

یکی از مزاحمت هایی که گریبانگیر وب سایت هاست، Botهای متفاوتی است که برای کپی اطلاعات، درج کامنت به صورت خودکار و مواردی از این دست، به آنها مراجعه میکنند. شاید در نگاه اول بد نباشد که این Botها به سراغ وب سایت ما بیایند و باعث افزایش تعداد ویزیت سایتمان شوند؛ ولی ضررهای ناشی از کپی و سرقت مطالب سایت، آنهم با سرعت بالا، بیشتر از منافع ناشی از بالا رفتن رنک سایت است. به طور مثال همین سایت NET Tips. دارای تعداد زیادی مقالات مفید است که افراد متعددی در نگارش و تهیه آن‌ها زحمت کشیده اند، یا وب سایتی برای جلب اعتماد مشتریان جهت درج اطلاعاتشان و یا آگهی هایشان زحمت زیادی کشیده است، Botهای آماده‌ی زیادی وجود دارد که با چند دقیقه صرف وقت جهت تنظیم شدن آماده میشوند تا مطالب را طبق ساختار تعیین شده، مورد به مورد کپی کنند.

برای خلاصی از این موارد روش‌های متعددی وجود دارد که از جمله آنها می‌توان به تنظیمات فایل htaccess در وب سرورهایی نظیر Apache و یا web.config در IIS اشاره کرد. در این مقاله این امکان را با IIS مرور میکنیم و برای فعال سازی آن کافی است در:

IIS 7.5 و بالاتر، همراه با انتخاب Request Filtering در مراحل نصب IIS

IIS 7.0 پس از نصب بسته آپدیت Microsoft Knowledge Base Article 957508 .

IIS 6.0 با نصب URLScan 3.0

در بخش <system.webServer> و سپس <security>، تگ requestFiltering را استفاده کنیم، در این تگ دستورالعمل‌های ویژه‌ی پالایشگر درخواست‌ها را مینویسیم (filteringRules) هر دستورالعمل پالایش دارای خصیصه‌های (Attributes) زیر است:

denyUnescapedPercent

مقدار Boolean و انتخابی

اگر برابر با true تنظیم گردد، درخواست هایی که دارای کاراکتر "درصد" (%) هستند و به وسیله escape character ها پوشش داده نشده باشند، رد می‌شوند. (جهت جلوگیری از حملات XSS و...)

مقدار پیش فرض true است.

name

عنوان دستورالعمل.

مقدار پیش فرض نداشته و درج کردن آن اجباری است.

scanAllRaw

مقدار Boolean و انتخابی

اگر برابر با true تنظیم گردد، پالایشگر درخواست‌ها موظف است تا با بررسی متن header های درخواست، در صورت یافتن یکی از واژه هایی که در خصیصه denyStrings ذکر کرده اید، درخواست را رد کند.

مقدار پیش فرض false است.

scanQueryString

مقدار Boolean و انتخابی

اگر برابر با true تنظیم گردد، پالایشگر درخواست‌ها موظف است تا Query string را بررسی کند تا در صورتی که یکی از واژه‌های درج شده در خصیصه denyStrings را بیابد، درخواست را رد کند.

اگر خصیصه‌ی unescapeQueryString از تگ < requestFiltering > برابر با true باشد، query string دوبار بررسی می‌شود: یکبار متن query string برای یافتن عبارات ممنوعه و بار دیگر برای یافتن کاراکترهای بدون پوشش scaped .

مقدار پیشفرض false است.

scanUrl

مقدار Boolean و انتخابی

اگر برابر با true تنظیم گردد، پالایشگر درخواست‌ها URL را برای یافتن واژه‌های ممنوعه‌ی ذکر شده در خصیصه denyStrings بررسی می‌نماید.

مقدار پیش فرض false است.

چند مثال:

مثال 1: در این مثال عنوان User-Agent هایی را که در موارد متعدد برای وب سایت هایی که روی آنها کار میکردم مزاحمت ایجاد میکردند را پالایش میکنیم. (لیست این Botها آپدیت میشود)

```
<requestFiltering>
  <filteringRules>
    <filteringRule name="BlockSearchEngines" scanUrl="false" scanQueryString="false">
      <scanHeaders>
        <clear />
        <add requestHeader="User-Agent" />
      </scanHeaders>
      <appliesTo>
        <clear />
      </appliesTo>
      <denyStrings>
        <clear />
        <add string="Python UrlLib" />
        <add string="WGet" />
        <add string="Apache HttpClient" />
        <add string="Unknown Bot" />
        <add string="Yandex Spider" />
        <add string="libwww-perl" />
        <add string="Nutch" />
        <add string="DotBot" />
        <add string="CCBot" />
        <add string="Majestic 12 Bot" />
        <add string="Java" />
        <add string="Link Checker" />
        <add string="Baiduspider" />
        <add string="Exabot" />
        <add string="PHP" />
      </denyStrings>
    </filteringRule>
  </filteringRules>
</requestFiltering>
```

مثال 2: ممانعت از SQL Injection

```
<requestFiltering>
   <filteringRules>
      <filteringRule name="SQLInjection" scanUrl="false" scanQueryString="true">
         <appliesTo>
            <clear />
            <add fileExtension=".asp" />
            <add fileExtension=".aspx" />
            <add fileExtension=".php" />
         </appliesTo>
         <denyStrings>
            <clear />
            <add string="--" />
            <add string=";" />
```

```
            <add string="/*" />
            <add string="@" />
            <add string="char" />
            <add string="alter" />
            <add string="begin" />
            <add string="cast" />
            <add string="create" />
            <add string="cursor" />
            <add string="declare" />
            <add string="delete" />
            <add string="drop" />
            <add string="end" />
            <add string="exec" />
            <add string="fetch" />
            <add string="insert" />
            <add string="kill" />
            <add string="open" />
            <add string="select" />
            <add string="sys" />
            <add string="table" />
            <add string="update" />
         </denyStrings>
         <scanHeaders>
            <clear />
         </scanHeaders>
      </filteringRule>
   </filteringRules>
</requestFiltering>
```

مثال 3: ممانعت از درخواست انواع خاصی از فایل ها

```
<requestFiltering>
   <filteringRules>
      <filteringRule name="Block Image Leeching" scanUrl="false" scanQueryString="false"
scanAllRaw="false">
         <scanHeaders>
            <add requestHeader="User-agent" />
         </scanHeaders>
         <appliesTo>
            <add fileExtension=".zip" />
            <add fileExtension=".rar" />
            <add fileExtension=".exe" />
         </appliesTo>
         <denyStrings>
            <add string="leech-bot" />
         </denyStrings>
      </filteringRule>
   </filteringRules>
</requestFiltering>
```

اطلاعات بیشتر در وب سایت رسمی IIS

نظرات خوانندگان

نویسنده: امین مصباحی
تاریخ: ۱۳۹۳/۰۲/۰۱ ۶:۳

تکمیلی 1: AhrefsBot هم از جمله‌ی Botهای مزاحم است، لذا:

```
</ "add string="AhrefsBot>
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۰۱ ۹:۳۵

لیست User-Agent هایی است که من در این سایت بستم تا امروز (از لاگ‌های خطای برنامه استخراج شدند):

bots.txt

عنوان: ASP.NET MVC و Identity 2.0 : مفاهیم پایه
نویسنده: آرمین ضیاء
تاریخ: ۱۳۹۳/۰۲/۰۸ ۱:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Security, ASP.NET Identity

در تاریخ 20 مارچ 2014 تیم ASP.NET نسخه نهایی Identity 2.0 را منتشر کردند . نسخه جدید برخی از ویژگی‌های درخواست شده پیشین را عرضه می‌کند و در کل قابلیت‌های احراز هویت و تعیین سطح دسترسی ارزنده ای را پشتیبانی می‌کند. این فریم ورک در تمام اپلیکیشن‌های ASP.NET می‌تواند بکار گرفته شود.

فریم ورک Identity در سال 2013 معرفی شد، که دنباله سیستم ASP.NET Membership بود. سیستم قبلی گرچه طی سالیان استفاده می‌شد اما مشکلات زیادی هم بهمراه داشت. بعلاوه با توسعه دنیای وب و نرم افزار، قابلیت‌های مدرنی مورد نیاز بودند که باید پشتیبانی می‌شدند. فریم ورک Identity در ابتدا سیستم ساده و کارآمدی برای مدیریت کاربران بوجود آورد و مشکلات پیشین را تا حد زیادی برطرف نمود. بعنوان مثال فریم ورک جدید مبتنی بر EF Code-first است، که سفارشی کردن سیستم عضویت را بسیار آسان می‌کند و به شما کنترل کامل می‌دهد. یا مثلا احراز هویت مبتنی بر پروتوکل OAuth پشتیبانی می‌شود که به شما اجازه استفاده از فراهم کنندگان خارجی مانند گوگل، فیسبوک و غیره را می‌دهد.

نسخه جدید این فریم ورک ویژگی‌های زیر را معرفی می‌کند (بعلاوه مواردی دیگر):

مدل حساب‌های کاربری توسعه داده شده. مثلا آدرس ایمیل و اطلاعات تماس را هم در بر می‌گیرد
احراز هویت دو مرحله ای (Two-Factor Authentication) توسط اطلاع رسانی ایمیلی یا پیامکی. مشابه سیستمی که گوگل، مایکروسافت و دیگران استفاده می‌کنند
تایید حساب‌های کاربری توسط ایمیل (Account Confirmation)
مدیریت کاربران و نقش‌ها (Administration of Users & Roles)
قفل کردن حساب‌های کاربری در پاسخ به Invalid log-in attempts

تامین کننده شناسه امنیتی (Security Token Provider) برای بازتولید شناسه‌ها در پاسخ به تغییرات تنظیمات امنیتی (مثلا هنگام تغییر کلمه عبور)

بهبود پشتیبانی از Social log-ins
یکپارچه سازی ساده با Claims-based Authorization

Identity 2.0 تغییرات چشم گیری نسبت به نسخه قبلی به‌وجود آورده است. به نسبت ویژگی‌های جدید، پیچیدگی‌هایی نیز معرفی شده‌اند. اگر به تازگی (مانند خودم) با نسخه 1 این فریم ورک آشنا شده و کار کرده اید، آماده شوید! گرچه لازم نیست از صفر شروع کنید، اما چیزهای بسیاری برای آموختن وجود دارد.

در این مقاله نگاهی اجمالی به نسخه‌ی جدید این فریم ورک خواهیم داشت. کامپوننت‌های جدید و اصلی را خواهیم شناخت و خواهیم دید هر کدام چگونه در این فریم ورک کار می‌کنند. بررسی عمیق و جزئی این فریم ورک از حوصله این مقاله خارج است، بنابراین به این مقاله تنها بعنوان یک نقطه شروع برای آشنایی با این فریم ورک نگاه کنید.

اگر به دنبال اطلاعات بیشتر و بررسی‌های عمیق‌تر هستید، لینک هایی در انتهای این مقاله نگاشت شده اند. همچنین طی هفته‌های آینده چند مقاله تخصصی‌تر خواهم نوشت تا از دید پیاده سازی بیشتر با این فریم ورک آشنا شوید.

در این مقاله با مقدار قابل توجهی کد مواجه خواهید شد. لازم نیست تمام جزئیات آنها را بررسی کنید، تنها با ساختار کلی این فریم ورک آشنا شوید. کامپوننت‌ها را بشناسید و بدانید که هر کدام در کجا قرار گرفته اند، چطور کار می‌کنند و اجزای کلی سیستم چگونه پیکربندی می‌شوند. گرچه، اگر به برنامه نویسی دات نت (ASP.NET, C#) تسلط دارید و با نسخه قبلی Identity هم کار کرده اید، درک کدهای جدید کار ساده ای خواهد بود.

Identity 2.0 با نسخه قبلی سازگار نیست

اپلیکیشن هایی که با نسخه 1.0 این فریم ورک ساخته شده اند نمی‌توانند بسادگی به نسخه جدید مهاجرت کنند. قابلیت هایی جدیدی که پیاده سازی شده اند تغییرات چشمگیری در معماری این فریم ورک بوجود آورده اند، همچنین API مورد استفاده در اپلیکیشن‌ها نیز دستخوش تغییراتی شده است. مهاجرت از نسخه 1.0 به 2.0 نیاز به نوشتن کدهای جدید و اعمال تغییرات متعددی دارد که از حوصله این مقاله خارج است. فعلا همین قدر بدانید که این مهاجرت نمی‌تواند بسادگی در قالب Plug-in and play صورت پذیرد!

شروع به کار : پروژه مثال‌ها را از NuGet دریافت کنید

در حال حاظر (هنگام نوشتن این مقاله) قالب پروژه استانداردی برای اپلیکیشن‌های ASP.NET MVC که ا ز Identity 2.0 استفاده کنند وجود ندارد. برای اینکه بتوانید از نسخه جدید این فریم ورک استفاده کنید، باید پروژه مثال را توسط NuGet دریافت کنید. ابتدا پروژه جدیدی از نوع ASP.NET Web Application بسازید و قالب Empty را در دیالوگ تنظیمات انتخاب کنید.

کنسول Package Manager را باز کنید و با اجرای فرمان زیر پروژه مثال‌ها را دانلود کنید.

```
PM> Install-Package Microsoft.AspNet.Identity.Samples -Pre
```

پس از آنکه NuGet کار خود را به اتمام رساند، ساختار پروژه ای مشابه پروژه‌های استاندارد MVC مشاهده خواهید کرد. پروژه شما شامل قسمت‌های Models, Views, Controllers و کامپوننت‌های دیگری برای شروع به کار است. گرچه در نگاه اول ساختار پروژه بسیار شبیه به پروژه‌های استاندارد ASP.NET MVC به نظر می‌آید، اما با نگاهی دقیق‌تر خواهید دید که تغییرات جدیدی ایجاد شده‌اند و پیچیدگی هایی نیز معرفی شده اند.

پیکربندی Identity : دیگر به سادگی نسخه قبلی نیست

به نظر من یکی از مهم‌ترین نقاط قوت فریم ورک Identity یکی از مهم‌ترین نقاط ضعفش نیز بود. سادگی نسخه 1.0 این فریم ورک کار کردن با آن را بسیار آسان می‌کرد و به سادگی می‌توانستید ساختار کلی و روند کارکردن کامپوننت‌های آن را درک کنید. اما همین سادگی به معنای محدود بودن امکانات آن نیز بود. بعنوان مثال می‌توان به تایید حساب‌های کاربری یا پشتیبانی از احراز هویت‌های دو مرحله ای اشاره کرد.

برای شروع نگاهی اجمالی به پیکربندی این فریم ورک و اجرای اولیه اپلیکیشن خواهیم داشت. سپس تغییرات را با نسخه 1.0 مقایسه می‌کنیم.

در هر دو نسخه، فایلی بنام *Startup.cs* در مسیر ریشه پروژه خواهید یافت. در این فایل کلاس واحدی بنام Startup تعریف شده است که متد ()ConfigureAuth را فراخوانی می‌کند. چیزی که در این فایل مشاهده نمی‌کنیم، خود متد ConfigureAuth است. این بدین دلیل است که مابقی کد کلاس Startup در یک کلاس پاره ای (Partial) تعریف شده که در پوشه *App_Start* قرار دارد. نام فایل مورد نظر *Startup.Auth.cs* است که اگر آن را باز کنید تعاریف یک کلاس پاره ای بهمراه متد ()ConfigureAuth را خواهید یافت. در یک پروژه که از نسخه Identity 1.0 استفاده می‌کند، کد متد ()ConfigureAuth مطابق لیست زیر است.

```
public partial class Startup
{
    public void ConfigureAuth(IAppBuilder app)
    {
        // Enable the application to use a cookie to
        // store information for the signed in user
        app.UseCookieAuthentication(new CookieAuthenticationOptions
        {
            AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
            LoginPath = new PathString("/Account/Login")
        });

        // Use a cookie to temporarily store information about a
        // user logging in with a third party login provider
        app.UseExternalSignInCookie(DefaultAuthenticationTypes.ExternalCookie);

        // Uncomment the following lines to enable logging
        // in with third party login providers
        //app.UseMicrosoftAccountAuthentication(
        //    clientId: "",
        //    clientSecret: "");

        //app.UseTwitterAuthentication(
        //   consumerKey: "",
        //   consumerSecret: "");

        //app.UseFacebookAuthentication(
        //   appId: "",
        //   appSecret: "");

        //app.UseGoogleAuthentication();
    }
}
```

در قطعه کد بالا پیکربندی لازم برای کوکی‌ها را مشاهده می‌کنید. همچنین کدهایی بصورت توضیحات وجود دارد که به منظور استفاده از تامین کنندگان خارجی مانند گوگل، فیسبوک، توییتر و غیره استفاده می‌شوند. حال اگر به کد این متد در نسخه Identity 2.0 دقت کنید خواهید دید که کد بیشتری نوشته شده است.

```
public partial class Startup {
    public void ConfigureAuth(IAppBuilder app) {
        // Configure the db context, user manager and role
        // manager to use a single instance per request
```

```
        app.CreatePerOwinContext(ApplicationDbContext.Create);
        app.CreatePerOwinContext<ApplicationUserManager>(ApplicationUserManager.Create);
        app.CreatePerOwinContext<ApplicationRoleManager>(ApplicationRoleManager.Create);

        // Enable the application to use a cookie to store information for the
        // signed in user and to use a cookie to temporarily store information
        // about a user logging in with a third party login provider
        // Configure the sign in cookie
        app.UseCookieAuthentication(new CookieAuthenticationOptions {
            AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
            LoginPath = new PathString("/Account/Login"),
            Provider = new CookieAuthenticationProvider {
                // Enables the application to validate the security stamp when the user
                // logs in. This is a security feature which is used when you
                // change a password or add an external login to your account.
                OnValidateIdentity = SecurityStampValidator
                    .OnValidateIdentity<ApplicationUserManager, ApplicationUser>(
                    validateInterval: TimeSpan.FromMinutes(30),
                    regenerateIdentity: (manager, user)
                    => user.GenerateUserIdentityAsync(manager))
            }
        });

        app.UseExternalSignInCookie(DefaultAuthenticationTypes.ExternalCookie);

        // Enables the application to temporarily store user information when
        // they are verifying the second factor in the two-factor authentication process.
        app.UseTwoFactorSignInCookie(
            DefaultAuthenticationTypes.TwoFactorCookie,
            TimeSpan.FromMinutes(5));

        // Enables the application to remember the second login verification factor such
        // as phone or email. Once you check this option, your second step of
        // verification during the login process will be remembered on the device where
        // you logged in from. This is similar to the RememberMe option when you log in.
        app.UseTwoFactorRememberBrowserCookie(
            DefaultAuthenticationTypes.TwoFactorRememberBrowserCookie);

        // Uncomment the following lines to enable logging in
        // with third party login providers
        //app.UseMicrosoftAccountAuthentication(
        //    clientId: "",
        //    clientSecret: "");

        //app.UseTwitterAuthentication(
        //   consumerKey: "",
        //   consumerSecret: "");

        //app.UseFacebookAuthentication(
        //   appId: "",
        //   appSecret: "");

        //app.UseGoogleAuthentication();
    }
}
```

اول از همه به چند فراخوانی متد app.CreatePerOwinContext بر می‌خوریم. با این فراخوانی‌ها Callback هایی را رجیستر می‌کنیم که آبجکت‌های مورد نیاز را بر اساس نوع تعریف شده توسط type arguments وهله سازی می‌کنند. این وهله‌ها سپس توسط فراخوانی متد context.Get() قابل دسترسی خواهند بود. این به ما می‌گوید که حالا Owin بخشی از اپلیکیشن ما است و فریم ورک Identity 2.0 از آن برای ارائه قابلیت هایش استفاده می‌کند.

مورد بعدی ای که جلب توجه می‌کند فراخوانی‌های دیگری برای پیکربندی احراز هویت دو مرحله‌ای است. همچنین پیکربندی‌های جدیدی برای کوکی‌ها تعریف شده است که در نسخه قبلی وجود نداشتند.

تا اینجا پیکربندی‌های اساسی برای اپلیکیشن شما انجام شده است و می‌توانید از اپلیکیشن خود استفاده کنید. بکارگیری فراهم کنندگان خارجی در حال حاضر غیرفعال است و بررسی آنها نیز از حوصله این مقاله خارج است. این کلاس پیکربندی‌های اساسی Identity را انجام می‌دهد. کامپوننت‌های پیکربندی و کدهای کمکی دیگری نیز وجود دارند که در کلاس *IdentityConfig.cs* تعریف شده اند.

پیش از آنکه فایل *IdentityConfig.cs* را بررسی کنیم، بهتر است نگاهی به کلاس ApplicationUser بیاندازیم که در پوشه Models قرار گرفته است.

کلاس جدید ApplicationUser در Identity 2.0

اگر با نسخه 1.0 این فریم ورک اپلیکیشنی ساخته باشید، ممکن است متوجه شده باشید که کلاس پایه IdentityUser محدود و شاید ناکافی باشد. در نسخه قبلی، این فریم ورک پیاده سازی IdentityUser را تا حد امکان ساده نگاه داشته بود تا اطلاعات پروفایل کاربران را معرفی کند.

```
public class IdentityUser : IUser
{
    public IdentityUser();
    public IdentityUser(string userName);

    public virtual string Id { get; set; }
    public virtual string UserName { get; set; }

    public virtual ICollection<IdentityUserRole> Roles { get; }
    public virtual ICollection<IdentityUserClaim> Claims { get; }
    public virtual ICollection<IdentityUserLogin> Logins { get; }

    public virtual string PasswordHash { get; set; }
    public virtual string SecurityStamp { get; set; }
}
```

بین خواص تعریف شده در این کلاس، تنها Id, UserName و Roles برای ما حائز اهمیت هستند (از دید برنامه نویسی). مابقی خواص عمدتا توسط منطق امنیتی این فریم ورک استفاده می‌شوند و کمک شایانی در مدیریت اطلاعات کاربران به ما نمی‌کنند.

اگر از نسخه Identity 1.0 استفاده کرده باشید و مطالعاتی هم در این زمینه داشته باشید، می‌دانید که توسعه کلاس کاربران بسیار ساده است. مثلا برای افزودن فیلد آدرس ایمیل و اطلاعات دیگر کافی بود کلاس ApplicationUser را ویرایش کنیم و از آنجا که این فریم ورک مبتنی بر EF Code-first است بروز رسانی دیتابیس و مابقی اپلیکیشن کار چندان مشکلی نخواهد بود.

با ظهور نسخه Identity 2.0 نیاز به برخی از این سفارشی سازی‌ها از بین رفته است. گرچه هنوز هم می‌توانید بسادگی مانند گذشته کلاس ApplicationUser را توسعه و گسترش دهید، تیم ASP.NET تغییراتی بوجود آورده اند تا نیازهای رایج توسعه دهندگان را پاسخگو باشد.

اگر به کد کلاس‌های مربوطه دقت کنید خواهید دید که کلاس ApplicationUser همچنان از کلاس پایه IdentityUser ارث بری می‌کند، اما این کلاس پایه پیچیده‌تر شده است. کلاس ApplicationUser در پوشه *Models* و در فایلی بنام *IdentityModels.cs* تعریف شده است. همانطور که می‌بینید تعاریف خود این کلاس بسیار ساده است.

```
public class ApplicationUser : IdentityUser {
    public async Task<ClaimsIdentity> GenerateUserIdentityAsync(
        UserManager<ApplicationUser> manager) {
        // Note the authenticationType must match the one
        // defined in CookieAuthenticationOptions.AuthenticationType
        var userIdentity =
            await manager.CreateIdentityAsync(this,
                DefaultAuthenticationTypes.ApplicationCookie);

        // Add custom user claims here
        return userIdentity;
    }
}
```

حال اگر تعاریف کلاس IdentityUser را بازیابی کنید (با استفاده از قابلیت Go To Definition در VS) خواهید دید که این کلاس خود از کلاس پایه دیگری ارث بری می‌کند. اگر به این پیاده سازی دقت کنید کاملا واضح است که ساختار این کلاس به کلی نسبت به نسخه قبلی تغییر کرده است.

```
public class IdentityUser<TKey, TLogin, TRole, TClaim> : IUser<TKey>
    where TLogin : Microsoft.AspNet.Identity.EntityFramework.IdentityUserLogin<TKey>
```

```
    where TRole : Microsoft.AspNet.Identity.EntityFramework.IdentityUserRole<TKey>
    where TClaim : Microsoft.AspNet.Identity.EntityFramework.IdentityUserClaim<TKey>
{
    public IdentityUser();

    // Used to record failures for the purposes of lockout
    public virtual int AccessFailedCount { get; set; }
    // Navigation property for user claims
    public virtual ICollection<TClaim> Claims { get; }
    // Email
    public virtual string Email { get; set; }
    // True if the email is confirmed, default is false
    public virtual bool EmailConfirmed { get; set; }
    // User ID (Primary Key)
    public virtual TKey Id { get; set; }
    // Is lockout enabled for this user
    public virtual bool LockoutEnabled { get; set; }
    // DateTime in UTC when lockout ends, any
    // time in the past is considered not locked out.
    public virtual DateTime? LockoutEndDateUtc { get; set; }

    // Navigation property for user logins
    public virtual ICollection<TLogin> Logins { get; }
    // The salted/hashed form of the user password
    public virtual string PasswordHash { get; set; }
    // PhoneNumber for the user
    public virtual string PhoneNumber { get; set; }
    // True if the phone number is confirmed, default is false
    public virtual bool PhoneNumberConfirmed { get; set; }
    // Navigation property for user roles
    public virtual ICollection<TRole> Roles { get; }

    // A random value that should change whenever a users
    // credentials have changed (password changed, login removed)
    public virtual string SecurityStamp { get; set; }
    // Is two factor enabled for the user
    public virtual bool TwoFactorEnabled { get; set; }
    // User name
    public virtual string UserName { get; set; }
}
```

اول از همه آنکه برخی از خواص تعریف شده هنوز توسط منطق امنیتی فریم ورک استفاده می‌شوند و از دید برنامه نویسی مربوط به مدیریت اطلاعات کاربران نیستند. اما به هر حال فیلدهای Email و PhoneNumber نیاز به ویرایش تعریف پیش فرض موجودیت کاربران را از بین می‌برد.

اما از همه چیز مهم‌تر امضا (Signature)ی خود کلاس است. این آرگومانهای جنریک چه هستند؟ به امضای این کلاس دقت کنید.

```
public class IdentityUser<TKey, TLogin, TRole, TClaim> : IUser<TKey>
    where TLogin : Microsoft.AspNet.Identity.EntityFramework.IdentityUserLogin<TKey>
    where TRole : Microsoft.AspNet.Identity.EntityFramework.IdentityUserRole<TKey>
    where TClaim : Microsoft.AspNet.Identity.EntityFramework.IdentityUserClaim<TKey>
```

نسخه جدید IdentityUser انواع آرگومانهای جنریک را پیاده سازی می‌کند که انعطاف پذیری بسیار بیشتری به ما می‌دهند. بعنوان مثال بیاد بیاورید که نوع داده فیلد Id در نسخه 1.0 رشته (string) بود. اما در نسخه جدید استفاده از آرگومانهای جنریک (در اینجا TKey) به ما اجازه می‌دهد که نوع این فیلد را مشخص کنیم. همانطور که مشاهده می‌کنید خاصیت Id در این کلاس نوع داده TKey را باز می‌گرداند.

```
public virtual TKey Id { get; set; }
```

یک مورد حائز اهمیت دیگر خاصیت Roles است که بصورت زیر تعریف شده.

```
public virtual ICollection<TRole> Roles { get; }
```

همانطور که می‌بینید نوع TRole بصورت جنریک تعریف شده و توسط تعاریف کلاس IdentityUser مشخص می‌شود. اگر به

محدودیت‌های پیاده سازی این خاصیت دقت کنید می‌بینید که نوع این فیلد به IdentityUserRole<TKey> محدود (constraint) شده است، که خیلی هم نسبت به نسخه 1.0 تغییری نکرده. چیزی که تفاوت چشمگیری کرده و باعث breaking changes می‌شود تعریف خود IdentityUserRole است.

در نسخه Identity 1.0 کلاس IdentityUserRole بصورت زیر تعریف شده بود.

```
public class IdentityUserRole
{
      public IdentityUserRole();
      public virtual IdentityRole Role { get; set; }
      public virtual string RoleId { get; set; }
      public virtual IdentityUser User { get; set; }
      public virtual string UserId { get; set; }
}
```

این کلاس را با پیاده سازی نسخه Identity 2.0 مقایسه کنید.

```
public class IdentityUserRole<TKey>
{
    public IdentityUserRole();
    public virtual TKey RoleId { get; set; }
    public virtual TKey UserId { get; set; }
}
```

پیاده سازی پیشین ارجاعاتی به آبجکت هایی از نوع IdentityRole و IdentityUser داشت. پیاده سازی نسخه 2.0 تنها شناسه‌ها را ذخیره می‌کند. اگر در اپلیکیشنی که از نسخه 1.0 استفاده می‌کند سفارشی سازی هایی انجام داده باشید (مثلا تعریف کلاس Role را توسعه داده باشید، یا سیستمی مانند Group-based Roles را پیاده سازی کرده باشید) این تغییرات سیستم قبلی شما را خواهد شکست.

بررسی پیاده سازی جدید IdentityUser از حوصله این مقاله خارج است. فعلا همین قدر بدانید که گرچه تعاریف پایه کلاس کاربران پیچیده‌تر شده است، اما انعطاف پذیری بسیار خوبی بدست آمده که شایان اهمیت فراوانی است.

از آنجا که کلاس ApplicationUser از IdentityUser ارث بری می‌کند، تمام خواص و تعاریف این کلاس پایه در ApplicationUser قابل دسترسی هستند.

کامپوننت‌های پیکربندی Identity 2.0 و کدهای کمکی

گرچه متد ()ConfigAuth در کلاس Startup، محلی است که پیکربندی Identity در زمان اجرا صورت می‌پذیرد، اما در واقع کامپوننت‌های موجود در فایل *IdentityConfig.cs* هستند که اکثر قابلیت‌های Identity 2.0 را پیکربندی کرده و نحوه رفتار آنها در اپلیکیشن ما را کنترل می‌کنند.

اگر محتوای فایل *IdentityConfig.cs* را بررسی کنید خواهید دید که کلاس‌های متعددی در این فایل تعریف شده اند. می‌توان تک تک این کلاس‌ها را به فایل‌های مجزایی منتقل کرد، اما برای مثال جاری کدها را بهمین صورت رها کرده و نگاهی اجمالی به آنها خواهیم داشت. بهرحال در حال حاظر تمام این کلاس‌ها در فضای نام ApplicationName.Models قرار دارند.

Application User Manager و Application Role Manager

اولین چیزی که در این فایل به آنها بر می‌خوریم دو کلاس ApplicationUserManager و ApplicationRoleManager هستند. آماده باشید، مقدار زیادی کد با انواع داده جنریک در پیش روست!

```
public class ApplicationUserManager : UserManager<ApplicationUser>
{
    public ApplicationUserManager(IUserStore<ApplicationUser> store)
        : base(store)
    {
    }
```

```
    public static ApplicationUserManager Create(
        IdentityFactoryOptions<ApplicationUserManager> options,
        IOwinContext context)
    {
        var manager = new ApplicationUserManager(
            new UserStore<ApplicationUser>(
                context.Get<ApplicationDbContext>()));

        // Configure validation logic for usernames
        manager.UserValidator =
            new UserValidator<ApplicationUser>(manager)
        {
            AllowOnlyAlphanumericUserNames = false,
            RequireUniqueEmail = true
        };

        // Configure validation logic for passwords
        manager.PasswordValidator = new PasswordValidator
        {
            RequiredLength = 6,
            RequireNonLetterOrDigit = true,
            RequireDigit = true,
            RequireLowercase = true,
            RequireUppercase = true,
        };

        // Configure user lockout defaults
        manager.UserLockoutEnabledByDefault = true;
        manager.DefaultAccountLockoutTimeSpan = TimeSpan.FromMinutes(5);
        manager.MaxFailedAccessAttemptsBeforeLockout = 5;

        // Register two factor authentication providers. This application uses
        // Phone and Emails as a step of receiving a code for verifying
        // the user You can write your own provider and plug in here.
        manager.RegisterTwoFactorProvider("PhoneCode",
            new PhoneNumberTokenProvider<ApplicationUser>
        {
            MessageFormat = "Your security code is: {0}"
        });

        manager.RegisterTwoFactorProvider("EmailCode",
            new EmailTokenProvider<ApplicationUser>
        {
            Subject = "SecurityCode",
            BodyFormat = "Your security code is {0}"
        });

        manager.EmailService = new EmailService();
        manager.SmsService = new SmsService();

        var dataProtectionProvider = options.DataProtectionProvider;

        if (dataProtectionProvider != null)
        {
            manager.UserTokenProvider =
                new DataProtectorTokenProvider<ApplicationUser>(
                    dataProtectionProvider.Create("ASP.NET Identity"));
        }

        return manager;
    }

    public virtual async Task<IdentityResult> AddUserToRolesAsync(
        string userId, IList<string> roles)
    {
        var userRoleStore = (IUserRoleStore<ApplicationUser, string>)Store;
        var user = await FindByIdAsync(userId).ConfigureAwait(false);

        if (user == null)
        {
            throw new InvalidOperationException("Invalid user Id");
        }

        var userRoles = await userRoleStore
            .GetRolesAsync(user)
            .ConfigureAwait(false);

        // Add user to each role using UserRoleStore
        foreach (var role in roles.Where(role => !userRoles.Contains(role)))
        {
            await userRoleStore.AddToRoleAsync(user, role).ConfigureAwait(false);
```

```
        }

        // Call update once when all roles are added
        return await UpdateAsync(user).ConfigureAwait(false);
    }

    public virtual async Task<IdentityResult> RemoveUserFromRolesAsync(
        string userId, IList<string> roles)
    {
        var userRoleStore = (IUserRoleStore<ApplicationUser, string>) Store;
        var user = await FindByIdAsync(userId).ConfigureAwait(false);

        if (user == null)
        {
            throw new InvalidOperationException("Invalid user Id");
        }

        var userRoles = await userRoleStore
            .GetRolesAsync(user)
            .ConfigureAwait(false);

        // Remove user to each role using UserRoleStore
        foreach (var role in roles.Where(userRoles.Contains))
        {
            await userRoleStore
                .RemoveFromRoleAsync(user, role)
                .ConfigureAwait(false);
        }

        // Call update once when all roles are removed
        return await UpdateAsync(user).ConfigureAwait(false);
    }
}
```

قطعه کد بالا گرچه شاید به نظر طولانی بیاید، اما در کل، کلاس ApplicationUserManager توابع مهمی را برای مدیریت کاربران فراهم می‌کند: ایجاد کاربران جدید، انتساب کاربران به نقش‌ها و حذف کاربران از نقش ها. این کلاس بخودی خود از کلاس UserManager<ApplicationUser> ارث بری می‌کند بنابراین تمام قابلیت‌های UserManager در این کلاس هم در دسترس است. اگر به متد Create() دقت کنید می‌بینید که وهله ای از نوع ApplicationUserManager باز می‌گرداند. بیشتر تنظیمات پیکربندی و تنظیمات پیش فرض کاربران شما در این متد اتفاق می‌افتد.

مورد حائز اهمیت بعدی در متد Create() فراخوانی context.Get<ApplicationDBContext>() است. بیاد بیاورید که پیشتر نگاهی به متد ConfigAuth() داشتیم که چند فراخوانی CreatePerOwinContext داشت که توسط آنها Callback هایی را رجیستر می‌کردیم. فراخوانی متد context.Get<ApplicationDBContext>() این Callbackها را صدا می‌زند، که در اینجا فراخوانی متد استاتیک ApplicationDbContext.Create() خواهد بود. در ادامه بیشتر درباره این قسمت خواهید خواهند.

اگر دقت کنید می‌بینید که احراز هویت، تعیین سطوح دسترسی و تنظیمات مدیریتی و مقادیر پیش فرض آنها در متد Create() انجام می‌شوند و سپس وهله ای از نوع خود کلاس ApplicationUserManager بازگشت داده می‌شود. همچنین سرویس‌های احراز هویت دو مرحله ای نیز در همین مرحله پیکربندی می‌شوند. اکثر پیکربندی‌ها و تنظیمات نیازی به توضیح ندارند و قابل درک هستند. اما احراز هویت دو مرحله ای نیاز به بررسی عمیق‌تری دارد. در ادامه به این قسمت خواهیم پرداخت. اما پیش از آن نگاهی به کلاس ApplicationRoleManager بیاندازیم.

```
public class ApplicationRoleManager : RoleManager<IdentityRole>
{
    public ApplicationRoleManager(IRoleStore<IdentityRole,string> roleStore)
        : base(roleStore)
    {
    }

    public static ApplicationRoleManager Create(
        IdentityFactoryOptions<ApplicationRoleManager> options,
        IOwinContext context)
    {
        var manager = new ApplicationRoleManager(
            new RoleStore<IdentityRole>(
                context.Get<ApplicationDbContext>()));

        return manager;
```

```
    }
}
```

مانند کلاس ApplicationUserManager مشاهده می‌کنید که کلاس ApplicationRoleManager از RoleManager<IdentityRole> ارث بری می‌کند. بنابراین تمام قابلیت‌های کلاس پایه نیز در این کلاس در دسترس هستند. یکبار دیگر متدی بنام Create() را مشاهده می‌کنید که وهله ای از نوع خود کلاس بر می‌گرداند.

سرویس‌های ایمیل و پیامک برای احراز هویت دو مرحله ای و تایید حساب‌های کاربری

دو کلاس دیگری که در فایل *IdentityConfig.cs* وجود دارند کلاس‌های EmailService و SmsService هستند. بصورت پیش فرض این کلاس‌ها تنها یک wrapper هستند که می‌توانید با توسعه آنها سرویس‌های مورد نیاز برای احراز هویت دو مرحله ای و تایید حساب‌های کاربری را بسازید.

```
public class EmailService : IIdentityMessageService
{
    public Task SendAsync(IdentityMessage message)
    {
        // Plug in your email service here to send an email.
        return Task.FromResult(0);
    }
}
```

```
public class SmsService : IIdentityMessageService
{
    public Task SendAsync(IdentityMessage message)
    {
        // Plug in your sms service here to send a text message.
        return Task.FromResult(0);
    }
}
```

دقت کنید که هر دو این کلاس‌ها قرارداد واحدی را بنام IIdentityMessageService پیاده سازی می‌کنند. همچنین قطعه کد زیر را از متد ApplicationUserManager.Create() بیاد آورید.

```
// Register two factor authentication providers. This application uses
// Phone and Emails as a step of receiving a code for verifying
// the user You can write your own provider and plug in here.
manager.RegisterTwoFactorProvider("PhoneCode",
    new PhoneNumberTokenProvider<ApplicationUser>
{
    MessageFormat = "Your security code is: {0}"
});

manager.RegisterTwoFactorProvider("EmailCode",
    new EmailTokenProvider<ApplicationUser>
{
    Subject = "SecurityCode",
    BodyFormat = "Your security code is {0}"
});

manager.EmailService = new EmailService();
manager.SmsService = new SmsService();
```

همانطور که می‌بینید در متد Create() کلاس‌های EmailService و SmsService وهله سازی شده و توسط خواص مرتبط به وهله ApplicationUserManager ارجاع می‌شوند.

کلاس کمکی SignIn

هنگام توسعه پروژه مثال Identity، تیم توسعه دهندگان کلاسی کمکی برای ما ساخته‌اند که فرامین عمومی احراز هویت کاربران و ورود آنها به اپلیکیشن را توسط یک API ساده فراهم می‌سازد. برای آشنایی با نحوه استفاده از این متدها می‌توانیم به کنترلر AccountController در پوشه Controllers مراجعه کنیم. اما پیش از آن بگذارید نگاهی به خود کلاس SignInHelper داشته

باشیم.

```
public class SignInHelper
{
    public SignInHelper(
        ApplicationUserManager userManager,
        IAuthenticationManager authManager)
    {
        UserManager = userManager;
        AuthenticationManager = authManager;
    }

    public ApplicationUserManager UserManager { get; private set; }
    public IAuthenticationManager AuthenticationManager { get; private set; }

    public async Task SignInAsync(
        ApplicationUser user,
        bool isPersistent,
        bool rememberBrowser)
    {
        // Clear any partial cookies from external or two factor partial sign ins
        AuthenticationManager.SignOut(
            DefaultAuthenticationTypes.ExternalCookie,
            DefaultAuthenticationTypes.TwoFactorCookie);

        var userIdentity = await user.GenerateUserIdentityAsync(UserManager);

        if (rememberBrowser)
        {
            var rememberBrowserIdentity =
                AuthenticationManager.CreateTwoFactorRememberBrowserIdentity(user.Id);

            AuthenticationManager.SignIn(
                new AuthenticationProperties { IsPersistent = isPersistent },
                userIdentity,
                rememberBrowserIdentity);
        }
        else
        {
            AuthenticationManager.SignIn(
                new AuthenticationProperties { IsPersistent = isPersistent },
                userIdentity);
        }
    }

    public async Task<bool> SendTwoFactorCode(string provider)
    {
        var userId = await GetVerifiedUserIdAsync();

        if (userId == null)
        {
            return false;
        }

        var token = await UserManager.GenerateTwoFactorTokenAsync(userId, provider);

        // See IdentityConfig.cs to plug in Email/SMS services to actually send the code
        await UserManager.NotifyTwoFactorTokenAsync(userId, provider, token);

        return true;
    }

    public async Task<string> GetVerifiedUserIdAsync()
    {
        var result = await AuthenticationManager.AuthenticateAsync(
            DefaultAuthenticationTypes.TwoFactorCookie);

        if (result != null && result.Identity != null
            && !String.IsNullOrEmpty(result.Identity.GetUserId()))
        {
            return result.Identity.GetUserId();
        }

        return null;
    }

    public async Task<bool> HasBeenVerified()
    {
        return await GetVerifiedUserIdAsync() != null;
    }
```

```
    public async Task<SignInStatus> TwoFactorSignIn(
        string provider,
        string code,
        bool isPersistent,
        bool rememberBrowser)
    {
        var userId = await GetVerifiedUserIdAsync();

        if (userId == null)
        {
            return SignInStatus.Failure;
        }

        var user = await UserManager.FindByIdAsync(userId);

        if (user == null)
        {
            return SignInStatus.Failure;
        }

        if (await UserManager.IsLockedOutAsync(user.Id))
        {
            return SignInStatus.LockedOut;
        }

        if (await UserManager.VerifyTwoFactorTokenAsync(user.Id, provider, code))
        {
            // When token is verified correctly, clear the access failed
            // count used for lockout
            await UserManager.ResetAccessFailedCountAsync(user.Id);
            await SignInAsync(user, isPersistent, rememberBrowser);

            return SignInStatus.Success;
        }

        // If the token is incorrect, record the failure which
        // also may cause the user to be locked out
        await UserManager.AccessFailedAsync(user.Id);

        return SignInStatus.Failure;
    }

    public async Task<SignInStatus> ExternalSignIn(
        ExternalLoginInfo loginInfo,
        bool isPersistent)
    {
        var user = await UserManager.FindAsync(loginInfo.Login);

        if (user == null)
        {
            return SignInStatus.Failure;
        }

        if (await UserManager.IsLockedOutAsync(user.Id))
        {
            return SignInStatus.LockedOut;
        }

        return await SignInOrTwoFactor(user, isPersistent);
    }

    private async Task<SignInStatus> SignInOrTwoFactor(
        ApplicationUser user,
        bool isPersistent)
    {
        if (await UserManager.GetTwoFactorEnabledAsync(user.Id) &&
            !await AuthenticationManager.TwoFactorBrowserRememberedAsync(user.Id))
        {
            var identity = new ClaimsIdentity(DefaultAuthenticationTypes.TwoFactorCookie);
            identity.AddClaim(new Claim(ClaimTypes.NameIdentifier, user.Id));

            AuthenticationManager.SignIn(identity);
            return SignInStatus.RequiresTwoFactorAuthentication;
        }

        await SignInAsync(user, isPersistent, false);
        return SignInStatus.Success;
    }

    public async Task<SignInStatus> PasswordSignIn(
```

```
        string userName,
        string password,
        bool isPersistent,
        bool shouldLockout)
    {
        var user = await UserManager.FindByNameAsync(userName);

        if (user == null)
        {
            return SignInStatus.Failure;
        }

        if (await UserManager.IsLockedOutAsync(user.Id))
        {
            return SignInStatus.LockedOut;
        }

        if (await UserManager.CheckPasswordAsync(user, password))
        {
            return await SignInOrTwoFactor(user, isPersistent);
        }

        if (shouldLockout)
        {
            // If lockout is requested, increment access failed
            // count which might lock out the user
            await UserManager.AccessFailedAsync(user.Id);

            if (await UserManager.IsLockedOutAsync(user.Id))
            {
                return SignInStatus.LockedOut;
            }
        }

        return SignInStatus.Failure;
    }
}
```

کد این کلاس نسبتا طولانی است، و بررسی عمیق آنها از حوصله این مقاله خارج است. گرچه اگر به دقت یکبار این کلاس را مطالعه کنید می‌توانید براحتی از نحوه کارکرد آن آگاه شوید. همانطور که می‌بینید اکثر متدهای این کلاس مربوط به ورود کاربران و مسئولیت‌های تعیین سطوح دسترسی است.

این متدها ویژگی‌های جدیدی که در Identity 2.0 عرضه شده اند را در بر می‌گیرند. متد آشنایی بنام SignInAsync() را می‌بینیم، و متدهای دیگری که مربوط به احراز هویت دو مرحله ای و external log-ins می‌شوند. اگر به متدها دقت کنید خواهید دید که برای ورود کاربران به اپلیکیشن کارهای بیشتری نسبت به نسخه پیشین انجام می‌شود.

بعنوان مثال متد Login در کنترلر AccountController را باز کنید تا نحوه مدیریت احراز هویت در Identity 2.0 را ببینید.

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    if (!ModelState.IsValid)
    {
        return View(model);
    }

    // This doen't count login failures towards lockout only two factor authentication
    // To enable password failures to trigger lockout, change to shouldLockout: true
    var result = await SignInHelper.PasswordSignIn(
        model.Email,
        model.Password,
        model.RememberMe,
        shouldLockout: false);

    switch (result)
    {
        case SignInStatus.Success:
            return RedirectToLocal(returnUrl);
        case SignInStatus.LockedOut:
            return View("Lockout");
```

```
        case SignInStatus.RequiresTwoFactorAuthentication:
            return RedirectToAction("SendCode", new { ReturnUrl = returnUrl });
        case SignInStatus.Failure:
        default:
            ModelState.AddModelError("", "Invalid login attempt.");

            return View(model);
    }
}
```

مقایسه Sign-in با نسخه Identity 1.0

در نسخه 1.0 این فریم ورک، ورود کاربران به اپلیکیشن مانند لیست زیر انجام می‌شد. اگر متد Login در کنترلر AccountController را باز کنید چنین قطعه کدی را می‌بینید.

```
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Login(LoginViewModel model, string returnUrl)
{
    if (ModelState.IsValid)
    {
        var user = await UserManager.FindAsync(model.UserName, model.Password);

        if (user != null)
        {
            await SignInAsync(user, model.RememberMe);
            return RedirectToLocal(returnUrl);
        }
        else
        {
            ModelState.AddModelError("", "Invalid username or password.");
        }
    }

    // If we got this far, something failed, redisplay form
    return View(model);
}
```

در قطعه کد بالا متدی در کلاس UserManager را فراخوانی می‌کنیم که مشابه قطعه کدی است که در کلاس SignInHelper دیدیم. همچنین متد SignInAsync را فراخوانی می‌کنیم که مستقیما روی کنترلر AccountController تعریف شده است.

```
private async Task SignInAsync(ApplicationUser user, bool isPersistent)
{
    AuthenticationManager.SignOut(
        DefaultAuthenticationTypes.ExternalCookie);

    var identity = await UserManager.CreateIdentityAsync(
        user, DefaultAuthenticationTypes.ApplicationCookie);

    AuthenticationManager.SignIn(
        new AuthenticationProperties() { IsPersistent = isPersistent }, identity);
}
```

مسلما با عرضه قابلیت‌های جدید در Identity 2.0 و تغییرات معماری این فریم ورک، پیچیدگی هایی معرفی می‌شود که حتی در امور ساده ای مانند ورود کاربران نیز قابل مشاهده است.

ApplicationDbContext

اگر از نسخه پیشین Identity در اپلیکیشن‌های ASP.NET MVC استفاده کرده باشید با کلاس ApplicationDbContext آشنا هستید. این کلاس پیاده سازی پیش فرض EF فریم ورک است، که اپلیکیشن شما توسط آن داده‌های مربوط به Identity را ذخیره و بازیابی می‌کند.

در پروژه مثال ها، تیم Identity این کلاس را بطور متفاوتی نسبت به نسخه 1.0 پیکربندی کرده اند. اگر فایل *IdentityModels.cs* را باز کنید تعاریف کلاس ApplicationDbContext را مانند لیست زیر خواهید یافت.

```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser> {
    public ApplicationDbContext()
        : base("DefaultConnection", throwIfV1Schema: false) {
    }

    static ApplicationDbContext() {
        // Set the database intializer which is run once during application start
        // This seeds the database with admin user credentials and admin role
        Database.SetInitializer<ApplicationDbContext>(new ApplicationDbInitializer());
    }

    public static ApplicationDbContext Create() {
        return new ApplicationDbContext();
    }
}
```

قطعه کد بالا دو متد استاتیک تعریف می‌کند. یکی Create() و دیگری ApplicationDbContext() که سازنده دیتابیس (database initializer) را تنظیم می‌کند. این متد هنگام اجرای اپلیکیشن فراخوانی می‌شود و هر پیکربندی ای که در کلاس ApplicationDbInitializer تعریف شده باشد را اجرا می‌کند. اگر به فایل *IdentityConfig.cs* مراجعه کنیم می‌توانیم تعاریف این کلاس را مانند لیست زیر بیابیم.

```
public class ApplicationDbInitializer
    : DropCreateDatabaseIfModelChanges<ApplicationDbContext>
{
    protected override void Seed(ApplicationDbContext context)
    {
        InitializeIdentityForEF(context);
        base.Seed(context);
    }

    public static void InitializeIdentityForEF(ApplicationDbContext db)
    {
        var userManager = HttpContext
            .Current.GetOwinContext()
            .GetUserManager<ApplicationUserManager>();

        var roleManager = HttpContext.Current
            .GetOwinContext()
            .Get<ApplicationRoleManager>();

        const string name = "admin@admin.com";
        const string password = "Admin@123456";
        const string roleName = "Admin";

        //Create Role Admin if it does not exist
        var role = roleManager.FindByName(roleName);

        if (role == null)
        {
            role = new IdentityRole(roleName);
            var roleresult = roleManager.Create(role);
        }

        var user = userManager.FindByName(name);

        if (user == null)
        {
            user = new ApplicationUser { UserName = name, Email = name };

            var result = userManager.Create(user, password);
            result = userManager.SetLockoutEnabled(user.Id, false);
        }

        // Add user admin to Role Admin if not already added
        var rolesForUser = userManager.GetRoles(user.Id);

        if (!rolesForUser.Contains(role.Name))
        {
            var result = userManager.AddToRole(user.Id, role.Name);
        }
    }
}
```

پیکربندی جاری در صورتی که مدل موجودیت‌ها تغییر کنند دیتابیس را پاک کرده و مجددا ایجاد می‌کند. در غیر اینصورت از دیتابیس موجود استفاده خواهد شد. اگر بخواهیم با هر بار اجرای اپلیکیشن دیتابیس از نو ساخته شود، می‌توانیم کلاس مربوطه را به DropCreateDatabaseAlways<ApplicationDbContext> تغییر دهیم. بعنوان مثال هنگام توسعه اپلیکیشن و بمنظور تست می‌توانیم از این رویکرد استفاده کنیم تا هر بار با دیتابیسی (تقریبا) خالی شروع کنیم.

نکته حائز اهمیت دیگر متد InitializeIdentityForEF() است. این متد کاری مشابه متد Seed() انجام می‌دهد که هنگام استفاده از مهاجرت‌ها (Migrations) از آن استفاده می‌کنیم. در این متد می‌توانید رکوردهای اولیه ای را در دیتابیس ثبت کنید. همانطور که مشاهده می‌کنید در قطعه کد بالا نقشی مدیریتی بنام Admin ایجاد شده و کاربر جدیدی با اطلاعاتی پیش فرض ساخته می‌شود که در آخر به این نقش منتسب می‌گردد. با انجام این مراحل، پس از اجرای اولیه اپلیکیشن کاربری با سطح دسترسی مدیر در اختیار خواهیم داشت که برای تست اپلیکیشن بسیار مفید خواهد بود.

در این مقاله نگاهی اجمالی به Identity 2.0 در پروژه‌های ASP.NET MVC داشتیم. کامپوننت‌های مختلف فریم ورک و نحوه پیکربندی آنها را بررسی کردیم و با تغییرات و قابلیت‌های جدید به اختصار آشنا شدیم. در مقالات بعدی بررسی هایی عمیق‌تر خواهیم داشت و با نحوه استفاده و پیاده سازی قسمت‌های مختلف این فریم ورک آشنا خواهیم شد.

مطالعه بیشتر

MSDN: Announcing RTM of ASP.NET Identity 2.0.0

Identity 2.0 on Codeplex

نظرات خوانندگان

نویسنده: محمّد زارع
تاریخ: ۱۳۹۳/۰۶/۰۷ ۱۹:۷

سلام. من از ASP.NET Identity توی پروژم استفاده کردم و همه چیز درست کار میکنه (در محیط توسعه). سوالی که دارم مربوط به بررسی نقش‌های کاربر هستش. میخواستم بدونم استفاده از HttpContext.Current.User.IsInRole توی ASP.NET Identity هم درست هست یا نه؟ من الان توی محیط توسعه هیچ مشکلی در استفاده ازش نمیبینم ولی چندجا توی اینترنت دیدم که گفته بودن استفاده از این روش قدیمی هستش و بعد از Publish کردن به مشکل میخوره و بهتر هستش که از UserManager.IsInRole استفاده بشه.

ممنون میشم اگر راهنمایی بفرمایید. https://aspnetidentity.codeplex.com/workitem/2189

http://codeverge.com/asp.net.security/user.isinrole-not-working-in-production/73992

نویسنده: هومن
تاریخ: ۱۳۹۳/۰۶/۱۱ ۱۳:۲۴

سلام
نمونه پروژه ای وجود داره که با استفاده از identity 2.0 سیستم گروپ بیس رو پیاده سازی کرده باشه؟
نمونه identity 1.0 رو پیدا کردم ولی هرکاری کردم نتونستم تبدیلش کنم به نسخه دوم

نویسنده: ایلیا
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۳:۲۰

اگه در حالت EF Code First نباشیم و دیتابیسمون به صورت Database First باشه و دیتااکسسمون ADO Entity Data Model یا همون edmx باشه.چطوری می‌تونیم از Identity استفاده کنیم؟(البته از MVC 5 و EF 6 دارم استفاده می‌کنم
و دوم اینکه جداول لازم رو خودمون باید بسازیم یا مثل Membership خودش میسازه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۸/۱۹ ۱۴:۰

- Use ASP.NET Identity on existing DB-Model PART 1
- ASP.NET Identity 2.0 on existing DB-model PART 2

نویسنده: نرگس حقیقی
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۲:۳

سلام؛ آیا میشه در یه پروژه MVC که از EF6 برای ارتباط با Sql Server استفاده میکنه و بدلیل زیاد بودن تعداد جداول از class Library جداگانه‌ای به‌منظور Migrate کردن بانک استفاده میکنه، از Identity هم استفاده کرد؛ با وجود اینکه به Identity Code کاربران در جداول دیگر هم نیاز هست؟ اصلا این کار ممکنه؟

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۰۹/۲۲ ۱۸:۵۳

سرویس خوبیه اما کمی کار باهاش در عین سادگی پیچیده اس.
مثلا پیاده سازی اون و تفکیک تو لایه‌های سرویس و دامین و لایه نمایش واقعا ادم گیج میکنه
هر چند زیاد هم نمی‌توان الگوی واحد کار باهاش به کار برد - البته نظر شخصی منه

نویسنده: وحید نصیری

تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۳:۴۲

« اعمال تزریق وابستگی‌ها به مثال رسمی ASP.NET Identity »

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۳:۴۳

بله. مراجعه کنید به این مثال: « اعمال تزریق وابستگی‌ها به مثال رسمی ASP.NET Identity »

عنوان: **ارسال PingBack در ASP.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۰۸ ۷:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Regular expressions, Html Agility Pack, SEO

Pingback یکی از روش‌های اطلاع رسانی به سایت‌های دیگر در مورد لینک دادن به آن‌ها در سایت خود است. برای مثال من لینکی از یکی از مطالب شما را در متن جاری خودم قرار می‌دهم. سپس به وسیله‌ی ارسال یک ping، در مورد انجام اینکار به شما اطلاع رسانی می‌کنم. حاصل آن عموما قسمت معروف ping-backs سایت‌ها است. این مورد نیز یکی از روش‌های مؤثر SEO در گرفتن backlink است و تبلیغ محتوا.
کار کردن با پروتکل Ping-back آنچنان ساده نیست؛ از این جهت که تبادل ارتباطات آن با پروتکل XML-RPC انجام می‌شود. XML-RPC نیز توسط PHP کارها بیشتر مورد استفاده قرار می‌گیرد؛ بجای استفاده از پروتکل‌های استاندارد وب سرویس‌ها مانند Soap و امثال آن. پیاده سازی‌های ابتدایی Pingback نیز مرتبط است به Wordpress معروف که با PHP تهیه شده‌است. در ادامه نگاهی خواهیم داشت به جزئیات پیاده سازی ارسال ping-back توسط برنامه‌های ASP.NET.

یافتن آدرس وب سرویس سایت پذیرای Pingback

اولین قدم در پیاده سازی Pingback، یافتن آدرسی است که باید اطلاعات مورد نظر را به آن ارسال کرد. این آدرس عموما به دو طریق ارائه می‌شود:
الف) در هدری به نام x-pingback و یا pingback
ب) در قسمتی از کدهای HTML صفحه به شکل

```
<link rel="pingback" href="pingback server">
```

برای مثال اگر به وبلاگ‌های MSDN دقت کنید، هدر x-pingback را می‌توانید در خروجی وب سرور آن‌ها مشاهده کنید:

همانطور که ملاحظه می‌کنید، نیاز است Response header را آنالیز کنیم.

```
private Uri findPingbackServiceUri()
        {
            var request = (HttpWebRequest)WebRequest.Create(_targetUri);
            request.UserAgent = UserAgent;
            request.Timeout = Timeout;
            request.ReadWriteTimeout = Timeout;
            request.Method = WebRequestMethods.Http.Get;
            request.AutomaticDecompression = DecompressionMethods.GZip | DecompressionMethods.Deflate;
            using (var response = request.GetResponse() as HttpWebResponse)
            {
                if (response == null) return null;

                var url = extractPingbackServiceUriFormHeaders(response);
                if (url != null)
                    return url;

                if (!isResponseHtml(response))
                    return null;

                using (var reader = new StreamReader(response.GetResponseStream()))
                {
                    return extractPingbackServiceUriFormPage(reader.ReadToEnd());
                }
            }
        }

        private static Uri extractPingbackServiceUriFormHeaders(WebResponse response)
        {
            var pingUrl = response.Headers.AllKeys.FirstOrDefault(header =>
                                header.Equals("x-pingback", StringComparison.OrdinalIgnoreCase) ||
                                header.Equals("pingback", StringComparison.OrdinalIgnoreCase));
```

```
            return getValidAbsoluteUri(pingUrl);
        }

        private static Uri extractPingbackServiceUriFormPage(string content)
        {
            if (string.IsNullOrWhiteSpace(content)) return null;
            var regex = new Regex(@"(?s)<link\srel=""pingback""\shref=""(.+?)""",
RegexOptions.IgnoreCase);
            var match = regex.Match(content);
            return (!match.Success || match.Groups.Count < 2) ? null :
getValidAbsoluteUri(match.Groups[1].Value);
        }

        private static Uri getValidAbsoluteUri(string url)
        {
            Uri absoluteUri;
            return string.IsNullOrWhiteSpace(url) || !Uri.TryCreate(url, UriKind.Absolute, out
absoluteUri) ? null : absoluteUri;
        }

        private static bool isResponseHtml(WebResponse response)
        {
            var contentTypeKey = response.Headers.AllKeys.FirstOrDefault(header =>
                                        header.Equals("content-type",
StringComparison.OrdinalIgnoreCase));
            return !string.IsNullOrWhiteSpace(contentTypeKey) &&
                    response.Headers[contentTypeKey].StartsWith("text/html",
StringComparison.OrdinalIgnoreCase);
        }
```

نحوه‌ی استخراج آدرس سرویس Pingback یک سایت را در کدهای فوق ملاحظه می‌کنید.

targetUri، آدرسی است از یک سایت دیگر که در سایت ما درج شده‌است. زمانیکه این صفحه را درخواست می‌کنیم، response.Headers.AllKeys حاصل می‌تواند حاوی کلید x-pingback باشد یا خیر. اگر بلی، همینجا کار پایان می‌یابد. فقط باید مطمئن شد که این آدرس مطلق است و نه نسبی. به همین جهت در متد getValidAbsoluteUri، بررسی بر روی UriKind.Absolute انجام شده‌است.

اگر هدر فاقد کلید x-pingback باشد، قسمت ب را باید بررسی کرد. یعنی نیاز است محتوای Html صفحه را برای یافتن link rel=pingback بررسی کنیم. همچنین باید دقت داشت که پیش از اینکار نیاز است حتما بررسی isResponseHtml صورت گیرد. برای مثال در سایت شما لینکی به یک فایل 2 گیگابایتی SQL Server درج شده‌است. در این حالت نباید ابتدا 2 گیگابایت فایل دریافت شود و سپس بررسی کنیم که آیا محتوای آن حاوی link rel=pingback است یا خیر. اگر محتوای ارسالی از نوع text/html بود، آنگاه کار دریافت محتوای لینک انجام خواهد شد.

ارسال Ping به آدرس سرویس Pingback

اکنون که آدرس سرویس pingback یک سایت را یافته‌ایم، کافی است ping ایی را به آن ارسال کنیم:

```
public void Send()
        {
            var pingUrl = findPingbackServiceUri();
            if (pingUrl == null)
                throw new NotSupportedException(string.Format("{0} doesn't support pingback.",
_targetUri.Host));

            sendPing(pingUrl);
        }

        private void sendPing(Uri pingUrl)
        {
            var request = (HttpWebRequest)WebRequest.Create(pingUrl);
            request.UserAgent = UserAgent;
            request.Timeout = Timeout;
            request.ReadWriteTimeout = Timeout;
            request.Method = WebRequestMethods.Http.Post;
            request.ContentType = "text/xml";
            request.ProtocolVersion = HttpVersion.Version11;
            makeXmlRpcRequest(request);
            using (var response = (HttpWebResponse)request.GetResponse())
            {
                response.Close();
```

```
            }
        }

        private void makeXmlRpcRequest(WebRequest request)
        {
            var stream = request.GetRequestStream();
            using (var writer = new XmlTextWriter(stream, Encoding.ASCII))
            {
                writer.WriteStartDocument(true);
                writer.WriteStartElement("methodCall");
                writer.WriteElementString("methodName", "pingback.ping");
                writer.WriteStartElement("params");

                writer.WriteStartElement("param");
                writer.WriteStartElement("value");
                writer.WriteElementString("string", Uri.EscapeUriString(_sourceUri.ToString()));
                writer.WriteEndElement();
                writer.WriteEndElement();

                writer.WriteStartElement("param");
                writer.WriteStartElement("value");
                writer.WriteElementString("string", Uri.EscapeUriString(_targetUri.ToString()));
                writer.WriteEndElement();
                writer.WriteEndElement();

                writer.WriteEndElement();
                writer.WriteEndElement();
            }
        }
```

اینبار HttpWebRequest تشکیل شده از نوع post است و نه get. همچنین مقداری را که باید ارسال کنیم نیاز است مطابق پروتکل XML-RPC باشد. برای کار با XML-RPC در دات نت یا می‌توان از کتابخانه‌ی Cook Computing's XML-RPC.Net استفاده کرد و یا مطابق کدهای فوق، دستورات آن‌را توسط یک XmlTextWriter کنار هم قرار داد و نهایتا در درخواست Post ارسالی درج کرد.

در اینجا sourceUri آدرس صفحه‌ای در سایت ما است که targetUri ایی (آدرسی از سایت دیگر) در آن درج شده‌است. در یک pinback، صرفا این دو آدرس به سرویس دریافت کننده‌ی pingback ارسال می‌شوند.

سپس سایت دریافت کننده‌ی ping، ابتدا sourceUri را دریافت می‌کند تا عنوان آن‌را استخراج کند و همچنین بررسی می‌کند که آیا targetUri، در آن درج شده‌است یا خیر (آیا spam است یا خیر)؟

تا اینجا اگر این مراحل را کنار هم قرار دهیم به کلاس Pingback ذیل خواهیم رسید:

Pingback.cs

نحوه‌ی استفاده از کلاس Pingback تهیه شده

کار ارسال Pingback عموما به این نحو است: هر زمانیکه مطلبی یا یکی از نظرات آن، ثبت یا ویرایش می‌شوند، نیاز است Pingbackهای آن ارسال شوند. بنابراین تنها کاری که باید انجام شود، استخراج لینک‌های خارجی یک صفحه و سپس فراخوانی متد Send کلاس فوق است.

یافتن لینک‌های یک محتوا را نیز می‌توان مانند متد extractPingbackServiceUriFormPage فوق، توسط یک Regex انجام داد و یا حتی با استفاده از کتابخانه‌ی معروف HTML Agility Pack :

```
var doc = new HtmlWeb().Load(url);
var linkTags = doc.DocumentNode.Descendants("link");
var linkedPages = doc.DocumentNode.Descendants("a")
                                  .Select(a => a.GetAttributeValue("href", null))
                                  .Where(u => !String.IsNullOrEmpty(u));
```

عنوان: استفاده از Froala WYSIWYG Editor در ASP.NET
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۸ ۱۵:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC

چندی قبل، معرفی ادیتور سبک وزن و مناسبی را تحت عنوان RedActor ، در این سایت ملاحظه کردید. زمانیکه این‌کار انجام شد، این ادیتور هم رایگان بود و هم سورس آخرین نگارش آن به سادگی در دسترس. بعد از مدتی، هر دو ویژگی یاد شده‌ی RedActor حذف شدند. پس از آن ادیتور مدرن و بسیار مناسب دیگری به نام Froala منتشر شد که هرچند نگارش‌های تجاری هم دارد، اما سورس آخرین نگارش آن برای عموم قابل دریافت است. در ادامه مروری خواهیم داشت بر نحوه‌ی یکپارچه سازی آن با ASP.NET MVC و همچنین ASP.NET Web forms.

دریافت آخرین نگارش Froala WYSIWYG Editor

برای دریافت فایل‌های آخرین نگارش این ادیتور وب می‌توانید به سایت آن، قسمت دریافت فایل‌ها مراجعه نمائید.

http://editor.froala.com/download

و یا به این آدرس مراجعه کنید:

https://github.com/froala/wysiwyg-editor/releases

ساختار پروژه و نحوه‌ی کپی فایل‌های آن

در هر دو مثالی که فایل‌های آن‌را از انتهای بحث می‌توانید دریافت کنید، این ساختار رعایت شده است:

فایل‌های CSS و فونت‌های آن، در پوشه‌ی Content قرار گرفته‌اند.

فایل‌های اسکریپت و زبان آن (که دارای زبان فارسی هم هست) در پوشه‌ی Scripts کپی شده‌اند.

یک نکته

فایل font-awesome.css را نیاز است کمی اصلاح کنید. مسیر پوشه‌ی فونت‌های آن اکنون با fonts شروع می‌شود.

تنظیمات اولیه

تفاوتی نمی‌کند که از وب فرم‌ها استفاده می‌کنید یا MVC، نحوه‌ی تعریف و افزودن پیش نیازهای این ادیتور به نحو ذیل است:

```
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>

    <link href="Content/font-awesome.css" rel="stylesheet" />
    <link href="Content/froala_editor.css" rel="stylesheet" />

    <script src="Scripts/jquery-1.10.2.min.js"></script>
    <script src="Scripts/froala_editor.min.js"></script>
    <script src="Scripts/langs/fa.js"></script>
</head>
<body>
    <form id="form1" runat="server">
    </form>
</body>
</html>
```

دو فایل CSS دارد (آیکن‌های آن و همچنین شیوه نامه‌ی اصلی ادیتور) به همراه سه فایل JS (جی‌کوئری، ادیتور و فایل زبان فارسی آن) که باید در فایل master یا layout سایت اضافه شوند.

استفاده از Froala WYSIWYG Editor در ASP.NET MVC

در ادامه نحوه‌ی فعال سازی ادیتور وب Froala را در یک View برنامه‌های ASP.NET MVC ملاحظه می‌کنید:

```
@{
    ViewBag.Title = "Index";
}

<style type="text/css">
    /*تنظیم فونت پیش فرض ادیتور*/
    .froala-element {
    }
</style>

@using (Html.BeginForm(actionName: "Index", controllerName: "Home"))
{
    @Html.TextArea(name: "Editor1")
    <input type="submit" value="ارسال" />
}

@section Scripts
{
    <script type="text/javascript">
        $(function () {
            $('#Editor1').editable({
                buttons: ["bold", "italic", "underline", "strikeThrough", "fontFamily",
                    "fontSize", "color", "formatBlock", "align", "insertOrderedList",
                    "insertUnorderedList", "outdent", "indent", "selectAll", "createLink",
                    "insertImage", "insertVideo", "undo", "redo", "html", "save",
"inserthorizontalrule"],
                inlineMode: false,
                inverseSkin: true,
                preloaderSrc: '@Url.Content("~/Content/img/preloader.gif")',
                allowedImageTypes: ["jpeg", "jpg", "png"],
                height: 300,
                language: "fa",
                direction: "rtl",
                fontList: ["Tahoma, Geneva", "Arial, Helvetica", "Impact, Charcoal"],
                autosave: true,
                autosaveInterval: 2500,
                saveURL: '@Url.Action("FroalaAutoSave", "Home")',
                saveParams: { postId: "123" },
                spellcheck: true,
                plainPaste: true,
                imageButtons: ["removeImage", "replaceImage", "linkImage"],
                borderColor: '#00008b',
                imageUploadURL: '@Url.Action("FroalaUploadImage", "Home")',
                imageParams: { postId: "123" },
                enableScript: false
            });
        });
```

```
    </script>
}
```

اگر می‌خواهید فونت پیش فرض آن را تنظیم کنید، باید مطابق کدهای ابتدای فایل، ویژگی‌های froala-element را تغییر دهید.
سپس این ادیتور را بر روی المان TextArea قرار گرفته در صفحه، فعال می‌کنیم.
در قسمت مقادیر buttons، تمام حالات ممکن پیش بینی شده‌اند. هر کدام را که نیاز ندارید، حذف کنید.
نحوه‌ی تعریف زبان و راست به چپ بودن این ادیتور را با مقدار دهی پارامترهای language و direction ملاحظه می‌کنید.

پارامترهای saveURL، autosave و saveParams کار تنظیم ارسال خودکار محتوای ادیتور را جهت ذخیره‌ی آن در سرور به عهده دارند. بر اساس مقدار autosaveInterval می‌توان مشخص کرد که هر چند میلی ثانیه یکبار این‌کار باید انجام شود.

```
/// <summary>
        /// ذخیره سازی خودکار
        /// </summary>
        [HttpPost]
        [ValidateInput(false)]
        public ActionResult FroalaAutoSave(string body, int? postId) // نام پارامتر بادی را تغییر ندهید
        {
            //todo: save body ...
            return new EmptyResult();
        }
```

در قسمت سمت سرور هم می‌توان این مقادیر ارسالی را در اکشن متدی که ملاحظه می‌کنید، دریافت کرد.
چون قرار است تگ‌های HTML به سرور ارسال شوند، ویژگی ValidateInput به false تنظیم شده‌است.
saveParams آن، برای مقدار دهی پارامترهای اضافی است که نیاز می‌باشند تا به سرور ارسال شوند. مثلا شماره مطلب جاری نیز به سرور ارسال گردد.
در اینجا نام پارامتری که ارسال می‌گردد، دقیقا مساوی body است. بنابراین آن‌را تغییر ندهید.

پارامترهای imageUploadURL و imageParams برای فعال سازی ذخیره تصاویر آن در سرور کاربرد دارند.
اکشن متد مدیریت کننده‌ی آن به نحو ذیل می‌تواند تعریف شود:

```
// todo: مسایل امنیتی آپلود را فراموش نکنید
        /// <summary>
        /// ذخیره سازی تصاویر ارسالی
        /// </summary>
        [HttpPost]
        public ActionResult FroalaUploadImage(HttpPostedFileBase file, int? postId) // نام پارامتر فایل را
تغییر ندهید
        {
            var fileName = Path.GetFileName(file.FileName);
            var rootPath = Server.MapPath("~/images/");
            file.SaveAs(Path.Combine(rootPath, fileName));
            return Json(new { link = "images/" + fileName }, JsonRequestBehavior.AllowGet);
        }
```

در اینجا نام پارامتری که به سرور ارسال می‌گردد، دقیقا معادل file است. بنابراین آن‌را تغییر ندهید.
خروجی آن برای مشخص سازی محل ذخیره سازی تصویر در سرور باید یک خروجی JSON دارای خاصیت و پارامتر link به نحو فوق باشد (این مسیر، یک مسیر نسبی است؛ نسبت به ریشه سایت).
imageParams آن برای مقدار دهی پارامترهای اضافی است که نیاز می‌باشند تا به سرور ارسال شوند. مثلا شماره مطلب جاری نیز به سرور ارسال گردد.

استفاده از Froala WYSIWYG Editor در ASP.NET Web forms

تمام نکاتی که در قسمت تنظیمات ASP.NET MVC در مورد ویژگی‌های سمت کلاینت این ادیتور ذکر شد، در مورد وب فرم‌ها نیز صادق است. فقط قسمت مدیریت سمت سرور آن اندکی تفاوت دارد.

```
<%@ Page Title="" Language="C#" MasterPageFile="~/Site.Master"
```

```
    ValidateRequest="false"
    EnableEventValidation="false"
    AutoEventWireup="true" CodeBehind="Default.aspx.cs" Inherits="FroalaWebFormsTest.Default" %>

<%--اعتبارسنجی ورودی غیرفعال شده چون باید تگ ارسال شود--%>
<%--همچنین در وب کانفیگ هم تنظیم دیگری نیاز دارد--%>

<asp:Content ID="Content1" ContentPlaceHolderID="head" runat="server">
</asp:Content>
<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" runat="server">
    <%--حالت کلاینت آی دی بهتر است تنظیم شود در اینجا--%>
    <asp:TextBox ID="txtEditor" ClientIDMode="Static"
        runat="server" Height="199px" TextMode="MultiLine" Width="447px"></asp:TextBox>
    <br />
    <asp:Button ID="btnSave" runat="server" OnClick="btnSave_Click" Text="ارسال" />

    <style type="text/css">
        /*تنظیم فونت پیش فرض ادیتور*/
        .froala-element {
        }
    </style>

    <script type="text/javascript">
        $(function () {
            $('#txtEditor').editable({
                buttons: ["bold", "italic", "underline", "strikeThrough", "fontFamily",
                    "fontSize", "color", "formatBlock", "align", "insertOrderedList",
                    "insertUnorderedList", "outdent", "indent", "selectAll", "createLink",
                    "insertImage", "insertVideo", "undo", "redo", "html", "save",
"inserthorizontalrule"],
                inlineMode: false,
                inverseSkin: true,
                preloaderSrc: 'Content/img/preloader.gif',
                allowedImageTypes: ["jpeg", "jpg", "png"],
                height: 300,
                language: "fa",
                direction: "rtl",
                fontList: ["Tahoma, Geneva", "Arial, Helvetica", "Impact, Charcoal"],
                autosave: true,
                autosaveInterval: 2500,
                saveURL: 'FroalaHandler.ashx',
                saveParams: { postId: "123" },
                spellcheck: true,
                plainPaste: true,
                imageButtons: ["removeImage", "replaceImage", "linkImage"],
                borderColor: '#00008b',
                imageUploadURL: 'FroalaHandler.ashx',
                imageParams: { postId: "123" },
                enableScript: false
            });
        });
    </script>
</asp:Content>
```

همانطور که ملاحظه می‌کنید، ValidateRequest صفحه به false تنظیم شده و همچنین در وب کانفیگ httpRuntime requestValidationMode به نگارش 2 تنظیم گردیده‌است تا بتوان توسط این ادیتور تگ‌های ارسالی را به سرور ارسال کرد. به علاوه ClientIDMode=Static نیز تنظیم شده‌است، تا بتوان از ID تکست باکس قرار گرفته در صفحه، به سادگی در کدهای سمت کاربر جی‌کوئری استفاده کرد.

اگر دقت کرده باشید، save urlها اینبار به فایل FroalaHandler.ashx اشاره می‌کنند. محتوای این Genric handler را ذیل مشاهده می‌کنید:

```
using System.IO;
using System.Web;
using System.Web.Script.Serialization;

namespace FroalaWebFormsTest
{
    public class FroalaHandler : IHttpHandler
    {
        //todo: برای اینکارها بهتر است از وب ای پی آی استفاده شود
        //todo: یا دو هندلر مجزا یکی برای تصاویر و دیگری برای ذخیره سازی متن

        public void ProcessRequest(HttpContext context)
        {
```

```
            var body = context.Request.Form["body"];
            var postId = context.Request.Form["postId"];
            if (!string.IsNullOrWhiteSpace(body) && !string.IsNullOrWhiteSpace(postId))
            {
                //todo: save changes

                context.Response.ContentType = "text/plain";
                context.Response.Write("");
                context.Response.End();
            }

            var files = context.Request.Files;
            if (files.Keys.Count > 0)
            {
                foreach (string fileKey in files)
                {
                    var file = context.Request.Files[fileKey];
                    if (file == null || file.ContentLength == 0)
                        continue;

                    //todo: در اینجا مسایل امنیتی آپلود فراموش نشود
                    var fileName = Path.GetFileName(file.FileName);
                    var rootPath = context.Server.MapPath("~/images/");
                    file.SaveAs(Path.Combine(rootPath, fileName));


                    var json = new JavaScriptSerializer().Serialize(new { link = "images/" + fileName
});
                    // البته اینجا یک فایل بیشتر ارسال نمی‌شود
                    context.Response.ContentType = "text/plain";
                    context.Response.Write(json);
                    context.Response.End();
                }
            }

            context.Response.ContentType = "text/plain";
            context.Response.Write("");
            context.Response.End();
        }

        public bool IsReusable
        {
            get { return false; }
        }
    }
}
```

در اینجا نحوه‌ی مدیریت سمت سرور auto save و همچنین ارسال تصاویر ادیتور Froala ، ذکر شده‌اند. با استفاده از context.Request.Form می‌توان به عناصر ارسالی به سرور دسترسی پیدا کرد. همچنین توسط context.Request.Files، اگر فایلی ارسال شده بود، ذخیره شده و نهایتا خروجی JSON مدنظر بازگشت داده می‌شود.

یک نکته‌ی امنیتی مهم

```
<location path="upload">
  <system.webServer>
    <handlers accessPolicy="Read" />
  </system.webServer>
</location>
```

تنظیم فوق را در web.config سایت، جهت Read only کردن پوشه‌ی ارسال تصاویر، حتما مدنظر داشته باشید. در اینجا فرض شده‌است که پوشه‌ی uploads قرار است قابلیت اجرای فایل‌های پویا را نداشته باشد.

کدهای کامل این مطلب را در ادامه می‌توانید دریافت کنید

FroalaWebFormsTest.zip

نظرات خوانندگان

نویسنده: علی رضایی
تاریخ: ۱۳۹۳/۰۲/۱۸ ۱۸:۱۱

سلام
ممنون برای این مطلب
فکر کنم این ادیتور با کروم یک مشکل داره:
برای مثال کلمه «تست» رو در اون تایپ کنید، سپس بعد از «س» یک حرف جدید مثل «ن» وارد کنید: نتیجه باید بشود «تسنت» اما میشود «تسنت». البته من در کروم 34 تست کردم. با IE 10 و Firefox 28 مشکلی نداشت.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۱۸ ۱۸:۵۷

مشکلات این ادیتور را لطفا در issue tracker آن‌ها ارسال کنید.

نویسنده: محسن
تاریخ: ۱۳۹۳/۰۲/۲۰ ۸:۳۰

عالیه؛ فقط قسمت آپلود فایل را کم داره.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۰ ۸:۵۶

به زودی .

نویسنده: میثم مهربانی
تاریخ: ۱۳۹۳/۰۲/۲۱ ۸:۵۴

برای Upload تصویر به صورتی که در Sample خود ادیتور با Progress انجام می‌شود چه باید کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۱ ۹:۹

سمت کلاینت آن به صورت پیش فرض فعال است و نیاز به تنظیم اضافه‌تری ندارد. فقط باید مدیریت سمت سرور آن‌را مطابق کدهایی که در مطلب جاری توضیح داده شدند، انجام دهید.

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۳/۰۹ ۱۱:۱۵

با سلام . برای ایجاد بلاک کد که در redactor موجود است، در این ادیتور چه باید کرد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۳/۰۹ ۱۴:۳

امکان افزونه نویسی هم دارد. اگر همان کدهای افزونه‌ای را که برای redactor نوشتیم، با این ادیتور تطابق دهیم، به دکمه‌ی سفارشی زیر خواهیم رسید:

```
buttons: [
                // .... ,
                "insertHTML" //custom button
        ],


    customButtons: {
                    insertHTML: {
                        title: 'Insert Code',
                        icon: {
                            type: 'font',
                            value: 'fa fa-dollar' // Font Awesome icon class fa fa-*
                        },
                        callback: function (editor) {
                            editor.saveSelection();

                            var codeModal = $("<div>").addClass("froala-modal").appendTo("body");
                            var wrapper = $("<div>").addClass("f-modal-wrapper").appendTo(codeModal);
                            $("<h4>").append('<span data-text="true">Insert Code</span>')
                                .append($('<i class="fa fa-times" title="Cancel">')
                                .click(function () {
                                    codeModal.remove();
                                }))
                                .appendTo(wrapper);

                            var dialog = "<textarea id='code_area' style='height: 211px; width: 538px;'
/><br/><label>Language:</label><select
id='code_lang'><option>CSharp</option><option>VB</option><option>JScript</option><option>Sql</option><o
ption>XML</option><option>CSS</option><option>Java</option><option>Delphi</option></select> <input
type='button' name='insert' id='insert_btn' value='Insert' /><br/>";
                            $(dialog).appendTo(wrapper);

                            $("#code_area").text(editor.text());

                            if (!editor.selectionInEditor()) {
                                editor.$element.focus();
                            }

                            $('#insert_btn').click(function () {
                                var lang = $("#code_lang").val();
                                var code = $("#code_area").val();
                                code = code.replace(/\s+$/, ""); // rtrim
                                code = $('<span/>').text(code).html(); // encode

                                var htmlCode = "<pre language='" + lang + "' name='code'>" + code +
"</pre></div>";

                                var codeBlock = "<div align='left' dir='ltr'>" + htmlCode +

"</div><br/>";


                                editor.restoreSelection();
                                editor.insertHTML(codeBlock);
                                editor.saveUndoStep();

                                codeModal.remove();
                            });
                        }
                    }
                }
```

نویسنده: احسان میرسعیدی
تاریخ: ۱۳۹۳/۰۳/۱۱ ۱:۱۳

آقای نصیری با تشکر از معرفی این ویرایش گر و ارائه این تکه کد. من این کد رو در گیت هاب دیدم که به اشتراک گذاشتید. به نظرم مشکلی با SyntaxHighliter داشت که در اینجا مطرح کردم.
با تغییری که اعمال کردم در وبلاگ من کار کرد و فوق العاده بود.
با تشکر از به اشتراک گزاری دانش تون

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۴/۰۱ ۱۱:۲۷

قطعه کد برای fullscreen این ادیتور:

```
fullscreen: {
                        title: 'FullScreen',
                        icon: {
                            type: 'font',
                            value: 'fa fa-external-link'
                        },
                        callback: function (editor) {
                            if (editor.$box.hasClass("froala-editor-full-screen")) {
                                editor.$box.removeClass("froala-editor-full-screen");
                                editor.$box.appendTo(editor.editorParent);
                            }
                            else {
                                if (editor.editorParent === undefined)
                                    editor.editorParent = editor.$box.parent();
                                editor.$box.addClass("froala-editor-full-screen");
                                editor.$box.appendTo("body");
                            }
                            editor.focus();
                        }
                    },
```

و کلاس css مربوطه :

```
.froala-editor-full-screen {
    position: fixed !important;
    z-index: 10000;
    top: 0;
    bottom: 0;
    left: 0;
    right: 0;
    background-color: #fff;
}

    .froala-editor-full-screen div.f-placeholder {
        height: 100% !important;
    }
```

نویسنده: حمید رضا رحیمی
تاریخ: ۱۳۹۳/۰۴/۲۴ ۱۳:۴۳

سلام و ممنون از انتشار مطالب مفیدتون
یک سوال : بنده (بنا به سفارش مشتری) احتیاج به ادیتوری دارم که در ثبت لینک امکان flow و noflow رو داشته باشه
تا به حال خودم این امکان رو ندیدم. شما موردی رو مشاهده کردید و یا میشه همین ادیتور رو دستکاری کرد ؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۲۴ ۱۳:۴۶

این قابلیت را به صورت توکار دارد. اطلاعات بیشتر

نویسنده: حمید رضا رحیمی
تاریخ: ۱۳۹۳/۰۴/۲۴ ۱۴:۱۲

ممنون از پاسخ تان اما این برای کلیه لینک‌ها می‌باشد من می‌خوام موردی به یک لینک این امکان داده شده و به یکی دیگه داده نشده Enables or disables adding the rel="nofollow" attribute on all links.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۲۴ ۱۴:۱۷

در قسمت مدیریت پروژه آن درخواست دهید تا یک checkbox مخصوص اینکار به صفحه دیالوگ استاندارد افزودن لینک آن اضافه کنند.

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۵/۲۷ ۲:۳۰

با سلام؛ توسط خاصیت imageUploadURL مشخص میکنیم که تصاویر کجا ذخیره شوند. یک خاصیت دیگه هم imagesLoadURL در خصوص این خاصیت میشه توضیح بدین.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۷ ۹:۲

برای فعال سازی media manager آن کاربرد دارد. یک افزونه است که باید جداگانه اضافه شود. مقدار imagesLoadUrl به مسیری در سرور اشاره می‌کند که قرار است آرایه‌ای از مسیرهای تصاویر قابل نمایش در آن‌را به فرمت JSON بازگشت دهد.

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۵/۲۷ ۲۲:۵۵

برای حذف یک تصویر از قسمت منیجر از خاصیت imageDeleteURL و برای ارسال پارامترها از imageDeleteParams استفاده می‌کنیم. خاصیت imageDeleteParams را باید به چه صورت مقداردهی کرد
موقعی که از کد زیر استفاده میکنم ادیتور لود نمیشه

```
imageDeleteParams: { src: $img.attr('src') }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۸ ۰:۷

- نیازی به تنظیم imageDeleteParams نیست. از آن برای ارسال یک سری اطلاعات اضافی و دستی مانند شماره id کاربر و امثال آن استفاده می‌شود. src تصویری که قرار است حذف شود، به صورت خودکار به مسیر مشخص شده توسط imageDeleteURL در سمت سرور ارسال می‌شود.
- برای تشخیص دقیق نام پارامترهای ارسالی به سرور از برگه‌ی network فایرباگ استفاده کنید.
- روش دوم تشخیص پارامترها و مقادیر ارسالی به سرور استفاده از FormCollection است.

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۵/۲۸ ۱۵:۱۵

ممنون؛ پارامتر src که آدرس تصویر رو در خودش داره به سمت سرور ارسال میشه.

نویسنده: رضایی
تاریخ: ۱۳۹۳/۰۵/۲۸ ۲۲:۱۲

با سلام

موقعی که از ادیتور توی یک modal استفاده میکنم تکست باکس‌های ادیتور غیر قابل فوکوس میشن
z-index رو هم تغییر دادم اما تاثیری نداشت و موقع کلیک ، فوکوس به تکست باکس‌های دیگه که روی modal هستند منتقل میشه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۸ ۲۲:۵۷

اینجا گزارش شده . در حال بررسی است.

نویسنده: Mohammadreza AAA
تاریخ: ۱۳۹۳/۰۶/۰۲ ۱۲:۵

سلام.
من این ادیتورو گذاشتم. فوق العاده راحت و قشنگه. فقط با بخش ارسال عکسش مشکل دارم.
مسئله اینکه عکس من توی پوشه مورد نظرم داره ذخیره میشه ولی اشکالش اینکه توی متنم موقع نمایش هیچی نمیاید. همه خصوصیات مثلا ایتالیک و بولد به متون اعمال میشن ولی عکس نیست!

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۰۲ ۱۲:۱۱

بررسی کنید چه خطایی در فایرباگ گزارش شده

نویسنده: M.Shakeri
تاریخ: ۱۳۹۳/۰۶/۰۴ ۸:۵۷

سلام
اگه داخل ادیتور متنی نوشته نشه مقدار پیش فرضش

```
"<p><br></p>"
```

هست که موقع اعتبار سنجی معتبره.
تو تنظیماتش هم گزینه ای برای خالی کردن ادیتور وجود نداره.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۰۵ ۱۲:۵۰

« اعتبار سنجی سمت کاربر wysiwyg-editorها در ASP.NET MVC »

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۷ ۱۳:۲۴

امکان آپلود فایل هم به آن اضافه شده: ^ و ^

نویسنده: هادی دانش پژوه
تاریخ: ۱۳۹۳/۰۶/۱۸ ۹:۴۳

با سلام
مثل اینکه این ادیتور دیگه رایگان نیست :(
می خواستم بدونم اگه از همان سورسی که برای دانلود گذاشتیند استفاده کنم مشکلی بوجود نمیاد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۸ ۱۰:۱۶

این مسایل مشکل ما نیست. چون فلان ابزار رایگان هست یا نیست، دلیلی بر این نیست و نخواهد بود که ما در مورد آن مطلبی تهیه نکنیم یا کاری انجام ندهیم.

نویسنده: مهرداد
تاریخ: ۱۳۹۳/۰۷/۰۸ ۱۷:۱۹

سلام؛ چطوری می‌توان به این ویرایشگر قابلیت Code اضافه کرد آیا شما قبلا تجربه همچین کاری را دارید و یا حتی جایی دیده‌اید؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۰۸ ۱۷:۲۹

کمی بالاتر توضیح دادم.

نویسنده: اصغر
تاریخ: ۱۳۹۳/۰۷/۲۵ ۱۱:۱۹

سلام. من از این قطعه کدی که شما گذاشتید برای ایجاد بلاک کد استفاده میکنم و درست هم کار میکنه منتها یک مشکلی که دارم این هستش که وقتی فرم رو پست میکنم تمام کد رو فقط در یک خط نشون میده. درحالی که اگر برم توی دیتابیسم و داخل تگ pre کدم رو پیست کنم درست نشون میده. اگر نکته ای داره ممنون میشم بهم بگید. تو اینترنت هم خیلی گشتم ولی چیز خاصی پیدا نکردم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۲۵ ۱۱:۴۳

مشکل CSS دارید. سطر:

```
var htmlCode = "<pre language='" + lang + "' name='code'>" + code + "</pre></div>";
```

مطابق تنظیمات شخصی است. اینجا class لازم را بر اساس تنظیمات CSS نمایشگر کدهای خود اضافه کنید (مثلا class='brush:' csharp).

```
var htmlCode = "<pre class='brush: " + lang + "' language='" + lang + "' name='code'>" + code +
"</pre></div>";
```

نویسنده: اصغر
تاریخ: ۱۳۹۳/۰۷/۲۵ ۱۲:۱۱

بله من قبلاً این کار رو کردم تا استایل نمایش کد درست شد ولی الان مشکلی که هست فقط اینه که چندین خط کدی که پیست میکنم همش توی یک خط نمایش داده میشه و اسکرول افقی میخوره. مثل شکل زیر:

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۲۵ ۱۲:۱۸

مشکل این ادیتور یا افزونه‌ی نوشته شده نیست. اگر از یک فشرده کننده HTML استفاده می‌کنید، ممکن است خطوط جدید را حذف کرده باشد. برای مثال این مورد در «بلاگر» گوگل مرسوم است. راه حل آن:

```
SyntaxHighlighter.config.bloggerMode = true;
SyntaxHighlighter.all();
```

برای مطالعه بیشتر

نویسنده: رضا ریاضی
تاریخ: ۱۳۹۳/۱۰/۱۴ ۱۹:۲۳

سلام؛ در زمان استفاده از Ajax.BeginForm در صورتی که از این ادیتور هم استفاده کنیم عمل POST انجام نمی‌شود. به چه دلیل؟ در شرایطی که با Html.BeginForm چنین مشکلی وجود ندارد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۴ ۲۱:۵۱

در مثالی که پیوست شد، از فایل jquery.unobtrusive-ajax استفاده نشده‌است. ابتدا این فایل را اضافه کنید:

```
PM> Install-Package Microsoft.jQuery.Unobtrusive.Ajax
```

بعد مدخل تعریف آن هم باید به فایل _Layout.cshtml اضافه شود:

```
<script src="~/Scripts/jquery-1.10.2.min.js"></script>
<script src="~/Scripts/jquery.validate.min.js"></script>
<script src="~/Scripts/jquery.unobtrusive-ajax.min.js"></script>
<script src="~/Scripts/jquery.validate.unobtrusive.min.js"></script>
```

به این ترتیب Ajax.BeginForm کار خواهد کرد (آزمایش شد).

عنوان: استفاده از نگارش سوم Google Analytics API در سرویس‌های ویندوز یا برنامه‌های وب
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۴ ۷:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, google analytics

در زمان نگارش این مطلب، آخرین نگارش API مخصوص Google Analytics ، نگارش سوم آن است و ... کار کردن با آن دارای مراحل خاصی است که حتما باید رعایت شوند. در غیر اینصورت عملا در یک برنامه‌ی وب یا سرویس ویندوز قابل اجرا نخواهند بود. زیرا در حالت متداول کار با API مخصوص Google Analytics، ابتدا یک صفحه‌ی لاگین به Gmail باز می‌شود که باید به صورت اجباری، مراحل آن را انجام داد تا مشخصات تائید شده‌ی اکانت در حال استفاده‌ی از API، در پوشه‌ی AppData ویندوز برای استفاده‌های بعدی ذخیره شود. این مورد برای یک برنامه‌ی دسکتاپ معمولی مشکل ساز نیست؛ زیرا کاربر برنامه، به سادگی می‌تواند صفحه‌ی مرورگری را که باز شده‌است، دنبال کرده و به اکانت گوگل خود وارد شود. اما این مراحل را نمی‌توان در یک برنامه‌ی وب یا سرویس ویندوز پیگیری کرد، زیرا عموما امکان لاگین از راه دور به سرور و مدیریت صفحه‌ی لاگین به Gmail وجود ندارد یا بهتر است عنوان شود، بی‌معنا است. برای حل این مشکل، گوگل راه حل دیگری را تحت عنوان اکانت‌های سرویس، ارائه داده است که پس از ایجاد آن، یک یک فایل X509 Certificate برای اعتبارسنجی سرویس، در اختیار برنامه نویس قرار می‌گیرد تا بدون نیاز به لاگین دستی به Gmail، بتواند از API گوگل استفاده کند. در ادامه نحوه‌ی فعال سازی این قابلیت و استفاده از آن‌را بررسی خواهیم کرد.

ثبت برنامه‌ی خود در گوگل و انجام تنظیمات آن

اولین کاری که برای استفاده از نگارش سوم Google Analytics API باید صورت گیرد، ثبت برنامه‌ی خود در Google Developer Console است. برای این منظور ابتدا به آدرس ذیل وارد شوید:

https://console.developers.google.com

سپس بر روی دکمه‌ی Create project کلیک کنید. نام دلخواهی را وارد کرده و در ادامه بر روی دکمه‌ی Create کلیک نمائید تا پروفایل این پروژه ایجاد شود.

تنها نکته‌ی مهم این قسمت، بخاطر سپردن نام پروژه است. زیرا از آن جهت اتصال به API گوگل استفاده خواهد شد.
پس از ایجاد پروژه، به صفحه‌ی آن وارد شوید و از منوی سمت چپ صفحه، گزینه‌ی Credentials را انتخاب کنید. در ادامه در صفحه‌ی باز شده، بر روی دکمه‌ی Create new client id کلیک نمائید.

در صفحه‌ی باز شده، گزینه‌ی Service account را انتخاب کنید. اگر سایر گزینه‌ها را انتخاب نمائید، کاربری که قرار است از API استفاده کند، باید بتواند توسط مرورگر نصب شده‌ی بر روی کامپیوتر اجرا کننده‌ی برنامه، یکبار به گوگل لاگین نماید که این مورد مطلوب برنامه‌های وب و همچنین سرویس‌ها نیست.

در اینجا ابتدا یک فایل مجوز p12 را به صورت خودکار دریافت خواهید کرد و همچنین پس از ایجاد client id، نیاز است، ایمیل آن‌را جایی یادداشت نمائید:

از این ایمیل و همچنین فایل p12 ارائه شده، جهت لاگین به سرور استفاده خواهد شد.

همچنین نیاز است تا به برگه‌ی APIs پروژه‌ی ایجاد شده رجوع کرد و گزینه‌ی Analytics API آن‌را فعال نمود:

تا اینجا کار ثبت و فعال سازی برنامه‌ی خود در گوگل به پایان می‌رسد.

دادن دسترسی به Client ID ثبت شده در برنامه‌ی Google Analytics

پس از اینکه Client ID سرویس خود را ثبت کردید، نیاز است به اکانت Google Analytics خود وارد شوید. سپس در منوی آن، گزینه‌ی Admin را پیدا کرده و به آن قسمت، وارد شوید:

در ادامه به گزینه‌ی User management آن وارد شده و به ایمیل Client ID ایجاد شده در قسمت قبل، دسترسی خواندن و آنالیز را اعطاء کنید:

در صورت عدم رعایت این مساله، کلاینت API، قادر به دسترسی به Google Analytics نخواهد بود.

استفاده از نگارش سوم Google Analytics API در دات نت

قسمت مهم کار، تنظیمات فوق است که در صورت عدم رعایت آن‌ها، شاید نصف روزی را مشغول به دیباگ برنامه شوید. در ادامه نیاز است پیشنیازهای دسترسی به نگارش سوم Google Analytics API را نصب کنیم. برای این منظور، سه بسته‌ی نیوگت ذیل را توسط کنسول پاورشل نیوگت، به برنامه اضافه کنید:

```
PM> Install-Package Google.Apis
PM> Install-Package Google.Apis.auth
PM> Install-Package Google.Apis.Analytics.v3
```

پس از نصب، کلاس GoogleAnalyticsApiV3 زیر، جزئیات دسترسی به Google Analytics API را کپسوله می‌کند:

```
using System;
using System.Linq;
using System.Security.Cryptography.X509Certificates;
using Google.Apis.Analytics.v3;
using Google.Apis.Analytics.v3.Data;
using Google.Apis.Auth.OAuth2;
using Google.Apis.Services;

namespace GoogleAnalyticsAPIv3Tests
{
    public class AnalyticsQueryParameters
    {
        public DateTime Start { set; get; }
        public DateTime End { set; get; }
        public string Dimensions { set; get; }
        public string Filters { set; get; }
        public string Metrics { set; get; }
    }

    public class AnalyticsAuthentication
    {
        public Uri SiteUrl { set; get; }
        public string ApplicationName { set; get; }
        public string ServiceAccountEmail { set; get; }
        public string KeyFilePath { set; get; }
        public string KeyFilePassword { set; get; }

        public AnalyticsAuthentication()
        {
            KeyFilePassword = "notasecret";
        }
    }

    public class GoogleAnalyticsApiV3
    {
        public AnalyticsAuthentication Authentication { set; get; }
        public AnalyticsQueryParameters QueryParameters { set; get; }

        public GaData GetData()
        {
            var service = createAnalyticsService();
            var profile = getProfile(service);
            var query = service.Data.Ga.Get("ga:" + profile.Id,
                                QueryParameters.Start.ToString("yyyy-MM-dd"),
                                QueryParameters.End.ToString("yyyy-MM-dd"),
                                QueryParameters.Metrics);
            query.Dimensions = QueryParameters.Dimensions;
            query.Filters = QueryParameters.Filters;
            query.SamplingLevel = DataResource.GaResource.GetRequest.SamplingLevelEnum.HIGHERPRECISION;
            return query.Execute();
        }

        private AnalyticsService createAnalyticsService()
        {
            var certificate = new X509Certificate2(Authentication.KeyFilePath,
```

```
Authentication.KeyFilePassword, X509KeyStorageFlags.Exportable);
            var credential = new ServiceAccountCredential(
                new ServiceAccountCredential.Initializer(Authentication.ServiceAccountEmail)
                {
                    Scopes = new[] { AnalyticsService.Scope.AnalyticsReadonly }
                }.FromCertificate(certificate));

            return new AnalyticsService(new BaseClientService.Initializer
            {
                HttpClientInitializer = credential,
                ApplicationName = Authentication.ApplicationName
            });
        }

        private Profile getProfile(AnalyticsService service)
        {
            var accountListRequest = service.Management.Accounts.List();
            var accountList = accountListRequest.Execute();
            var site = Authentication.SiteUrl.Host.ToLowerInvariant();
            var account = accountList.Items.FirstOrDefault(x =>
x.Name.ToLowerInvariant().Contains(site));
            var webPropertyListRequest = service.Management.Webproperties.List(account.Id);
            var webPropertyList = webPropertyListRequest.Execute();
            var sitePropertyList = webPropertyList.Items.FirstOrDefault(a =>
a.Name.ToLowerInvariant().Contains(site));
            var profileListRequest = service.Management.Profiles.List(account.Id, sitePropertyList.Id);
            var profileList = profileListRequest.Execute();
            return profileList.Items.FirstOrDefault(a => a.Name.ToLowerInvariant().Contains(site));
        }
    }
}
```

در اینجا در ابتدا بر اساس فایل p12 ایی که از گوگل دریافت شد، یک X509Certificate2 ایجاد می‌شود. پسورد این فایل مساوی است با ثابت notasecret که در همان زمان تولید اکانت سرویس در گوگل، لحظه‌ای در صفحه نمایش داده خواهد شد. به کمک آن و همچنین ServiceAccountEmail ایمیلی که پیشتر به آن اشاره شد، می‌توان به AnalyticsService لاگین کرد. به این ترتیب به صورت خودکار می‌توان شماره پروفایل اکانت سایت خود را یافت و از آن در حین فراخوانی service.Data.Ga.Get استفاده کرد.

مثالی از نحوه استفاده از کلاس GoogleAnalyticsApiV3

در ادامه یک برنامه‌ی کنسول را ملاحظه می‌کنید که از کلاس GoogleAnalyticsApiV3 استفاده می‌کند:

```
using System;
using System.Collections.Generic;
using System.Linq;

namespace GoogleAnalyticsAPIv3Tests
{
    class Program
    {

        static void Main(string[] args)
        {
            var statistics = new GoogleAnalyticsApiV3
            {
                Authentication = new AnalyticsAuthentication
                {
                    ApplicationName = "My Project",
                    KeyFilePath = "811e1d9976cd516b55-privatekey.p12",
                    ServiceAccountEmail = "10152bng4j3mq@developer.gserviceaccount.com",
                    SiteUrl = new Uri("http://www.dotnettips.info/")
                },
                QueryParameters = new AnalyticsQueryParameters
                {
                    Start = DateTime.Now.AddDays(-7),
                    End = DateTime.Now,
                    Dimensions = "ga:date",
                    Filters = null,
                    Metrics = "ga:users,ga:sessions,ga:pageviews"
                }
            }.GetData();
```

```
            foreach (var result in statistics.TotalsForAllResults)
            {
                Console.WriteLine(result.Key + " -> total:" + result.Value);
            }
            Console.WriteLine();

            foreach (var row in statistics.ColumnHeaders)
            {
                Console.Write(row.Name + "\t");
            }
            Console.WriteLine();

            foreach (var row in statistics.Rows)
            {
                var rowItems = (List<string>)row;
                Console.WriteLine(rowItems.Aggregate((s1, s2) => s1 + "\t" + s2));
            }

            Console.ReadLine();
        }
    }
}
```

چند نکته

ApplicationName همان نام پروژه‌ای است که ابتدای کار، در گوگل ایجاد کردیم.

KeyFilePath مسیر فایل مجوز p12 ایی است که گوگل در حین ایجاد اکانت سرویس، در اختیار ما قرار می‌دهد.

ServiceAccountEmail آدرس ایمیل اکانت سرویس است که در قسمت ادمین Google Analytics به آن دسترسی دادیم.

SiteUrl آدرس سایت شما است که هم اکنون در Google Analytics دارای یک اکانت و پروفایل ثبت شده‌است.

توسط AnalyticsQueryParameters می‌توان نحوه‌ی کوئری گرفتن از Google Analytics را مشخص کرد. تاریخ شروع و پایان گزارش گیری در آن مشخص هستند. در مورد پارامترهایی مانند Dimensions و Metrics بهتر است به مرجع کامل آن در گوگل مراجعه نمائید:

Dimensions & Metrics Reference

برای نمونه در مثال فوق، تعداد کاربران، سشن‌های آن و همچنین تعداد بار مشاهده‌ی صفحات، گزارشگیری می‌شود.

برای مطالعه بیشتر

Using Google APIs in Windows Store Apps

How To Use Google Analytics From C# With OAuth

#Google Analytic’s API v3 with C

NET Library for Accessing and Querying Google Analytics V3 via Service Account.

#Google OAuth2 C

نظرات خوانندگان

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۰:۲۸

سلام و تشکر از مقاله خوبتان.
به نظر شما بهتر است که یک اکانت در آنالیتیکس تعریف کنیم و تمام وبسایت‌های خودمان را به این اکانت اضافه کنیم یا اینکه برای هر وبسایت یک اکانت مجزا بسازیم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۰:۳۶

- برای متد getProfile نوشته شده تفاوتی نمی‌کند. چون تمام اکانت‌های موجود را جهت یافتن آدرس سایت مدنظر جستجو می‌کند تا شماره پروفایل آن‌را برای کوئری گرفتن استخراج کند.
+ اکانت Client ID ساخته شده، برای تمام سرویس‌های متفاوت گوگل کاربرد دارد؛ و منحصر به Google Analytics نیست. فقط باید در قسمت APIs آن پروژه، سرویس موردنظر را فعال کرد.
- همچنین مزیت این روش نسبت به پروتکل‌های قدیمی گوگل که از نام کاربری و کلمه‌ی عبور Gmail استفاده می‌کردند، عدم نیاز به ذخیره سازی اطلاعات حساس اکانت گوگل خود در فایل کانفیگ برنامه است. بنابراین نیازی نیست تا اکانت‌های متفاوت گوگلی را برای این‌کار خاص ایجاد کنید.

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۲:۳۱

با سلام.
همه چیز بر روی کلاینت درست کار میکند ولی وقتی سایت را بر روی سرور آپلود میکنم خطای زیر روی می‌دهد:

```
System.Security.Cryptography.CryptographicException: An internal error occurred
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۳:۲۷

این سه مورد را بررسی کنید:
- آیا مسیر فایل p12 را درست تنظیم کرده‌اید؟ در یک برنامه‌ی وب باید به این نحو باشد:

```
Server.MapPath("~/folder/" + filename)
```

- آیا در application pool برنامه‌ی شما load user profile فعال است؟ اگر خیر، crypto susbsystem کار نخواهد کرد:

- تنظیم زیر را هم امتحان کنید:

بجای

```
var certificate = new X509Certificate2(Authentication.KeyFilePath, Authentication.KeyFilePassword,
 X509KeyStorageFlags.Exportable);
```

بنویسید:

```
var certificate = new X509Certificate2(Authentication.KeyFilePath, Authentication.KeyFilePassword,
   X509KeyStorageFlags.MachineKeySet |
   X509KeyStorageFlags.PersistKeySet |
   X509KeyStorageFlags.Exportable);
```

نویسنده: ایلیا اکبری فرد
تاریخ: ۱۳۹۳/۰۲/۲۴ ۱۴:۲

حالت سوم پاسخ شما برای مورد من درست کار کرد. تشکر فراوان.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۲/۲۹ ۱۵:۲۸

یک نکته‌ی تکمیلی

اگر کتابخانه‌ی Google.Apis.Analytics.v3 را بر روی یک سیستم دات نت 4 اجرا کنید، احتمالا خطای ذیل را دریافت خواهید کرد:

```
Could not load type 'System.Net.HttpStatusCode' from assembly System.Net
```

علت اینجا است که دات نت 4 نیاز به وصله‌ی KB2468871 دارد تا بتواند portable libraries را بارگذاری کند.

نویسنده: کامران
تاریخ: ۱۳۹۳/۰۵/۲۱ ۲:۵۲

از این ایراد میگیره میگه null هست، من اکانت ساختم پراپرتی هم درست کردم

```
Line 77: var webPropertyListRequest = service.Management.Webproperties.List(account.Id);

Object reference not set to an instance of an object.
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۰:۱۱

«من اکانت ساختم پراپرتی هم درست کردم»
کافی نیست. تمام مراحل باید انجام شوند.
این خطاها صرفا به معنای غیرمعتبر بودن یکی از مراحل است که سبب شده‌اند اطلاعات آن مرحله خاص قابل دریافت نباشد.

نویسنده: کامران
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۰:۵۳

نه اینکه فقط اکانت و پراپرتی درست کرده باشم، همه مراحل رو انجام دادم اما از خطی که خطا میده متوجه شدم که پرپرتی‌های من رو پیدا نمیکنه یعنی خالی برمیگردونه، قبلا هم کار کرده بودم همین مشکل پیش میومد.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۱:۹

در متد getProfile :
- اگر account نال هست یعنی SiteUrl مشخص شده در ابتدای برنامه، در اکانت آنالیتیکس شما موجود نیست. روی سطر accountList.Items یک break-point قرار دهید تا لیست واقعی اکانت‌های ثبت شده را مشاهده کنید.
- اگر به مرحله‌ی اجرای accountList نمی‌رسید، یعنی ServiceAccountEmail معتبری وارد نشده‌است.

نویسنده: کامران
تاریخ: ۱۳۹۳/۰۵/۲۱ ۱۲:۳۲

متوجه اشکال شدم.
من کدهای شما رو کپی کردم و نخوندم چه کردید، کدها رو که بررسی کردم متوجه شدم شما توی لیست اکانتها که واکشی میکنید اکانتی که میخوایم باید همنام یا شبیه آدرس سایت(SiteUrl) باشه و طبیعتا من وقتی توی گوگل آنالیز اکانت و پروفایلهام رو میسازم نام دلخواه میدم، شما بجای استفاده از آی دی از اسم استفاده کردید که از روی اسم آی دی رو در بیارید برای همین چون اسم اکانت با آدرس سایتم یکی نبود چیزی پیدا نمیکرد :

```
var account = accountList.Items.FirstOrDefault(x => x.Name.ToLowerInvariant().Contains(site));
```

برای پراپرتی هم به همین صورت.

عنوان: **بهینه سازی فایلهای js و css در برنامه‌های ASP.NET با استفاده از Combres - قسمت اول**
نویسنده: سامان هاشمی
تاریخ: ۱۳۹۳/۰۲/۲۸ ۱۸:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Combres, Minification

یکی از مواردی که در توسعه وب نقش مهمی دارد، بهینه سازی فایلهای js و css است که با فشرده سازی و کش کردن آنها می‌توان سرعت بارگذاری را تا حد چشمگیری افزایش داد. برای درک بهتر، به مثال زیر توجه کنید.
یک پروژه ساده را ایجاد می‌کنیم و فایل‌های CSS و js را مانند شکل زیر، به آن اضافه می‌کنیم:

طبق تصویر فایل‌ها را به صفحه‌ای که ساختیم اضافه می‌کنیم:

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>combress</title>
    <link href="Styles/Site.css" rel="stylesheet" />
    <script src="Scripts/jquery-1.10.2.js"></script>
    <script src="Scripts/jquery-1.10.2.min.js"></script>
</head>
<body>
    <form id="form1" runat="server">
        <div>
        </div>
    </form>
</body>
</html>
```

پروژه را اجرا کرده و توسط افزونه‌ی firebug درخواست‌هایی را که از سرور شده‌است، بررسی می‌کنیم. مشاهده خواهید کرد که به ازای هر فایل، یک درخواست به سرور ارسال شده و هیچکدام از فایل‌ها توسط وب سرور فشرده سازی نشده‌اند و اطلاعاتی در مورد کش، به هدر آنها اضافه نشده است.

GET Site.css	200 OK
GET jquery-1.10.2.js	200 OK
GET jquery-1.10.2.min.js	200 OK

برای رفع این موارد، روشهای گوناگونی وجود دارد که امروز قصد داریم این کار را توسط کتابخانه Combress انجام دهیم ! **نصب کتابخانه Combres**

شما می‌توانید با استفاده از nuget این کتابخانه را به پروژه خود اضافه کنید. **ایجاد فایل تنظیمات**

پس از نصب کتابخانه، فایلی با نام combres.xml در فولدر app_data ساخته می‌شود که تمامی فعالیت‌های کتابخانه براساس آن انجام می‌شود و ساختار آن بصورت زیر است:

```
<?xml version="1.0" encoding="utf-8" ?>
<combres xmlns='urn:combres'>
  <filters>
    <filter type="Combres.Filters.FixUrlsInCssFilter, Combres" />
  </filters>
  <resourceSets url="~/combres.axd"
                defaultDuration="30"
                defaultVersion="auto"
                defaultDebugEnabled="false"
                defaultIgnorePipelineWhenDebug="true"
                localChangeMonitorInterval="30"
                remoteChangeMonitorInterval="60">
    <resourceSet name="siteCss" type="css">
      <resource path="~/content/Site.css" />
      <resource path="~/content/anotherCss.css" />
      <resource path="~/scripts/yetAnotherCss.css" />
    </resourceSet>
    <resourceSet name="siteJs" type="js">
      <resource path="~/scripts/jquery-1.4.4.js" />
```

```
      <resource path="~/scripts/anotherJs.js" />
      <resource path="~/scripts/yetAnotherJs.js" />
    </resourceSet>
  </resourceSets>
</combres>
```

ResourceSet : با استفاده از هر ResourceSet می‌توانید آن مجموعه فایل را در یک درخواست از سرور دریافت کنید.

پارامتر **url** : برای تولید لینک فایل‌ها از آن استفاده میکند.

پارامتر **defaultDuration** : این عدد به تعداد روزهای پیشفرض برای کش کردن فایلها اشاره میکند.

پارامتر **defaultVersion** :در صورتی که مقدار آن auto باشد به ازای هر تغییر، آدرس جدیدی برای فایل موردنظر ایجاد میشود.

پارامتر **defaultDebugEnabled** :با استفاده از آن میتوانید debug mode را مشخص کنید. در صورتیکه مقدار آن auto باشد، این مقدار مستقیما از وب‌کانفیگ خوانده میشود.

مقادیر پیش فرض برای تمامی ResourceSetها استفاده می‌شود و در صورت نیاز میتوان این مقادیر را برای هر ResourceSet بازنویسی کرد. فیلترها برای اعمال تغییراتی در فایل js و CSS استفاده می‌شوند که باید به این شکل معرفی شوند. در قسمت بعد با فیلترها بیشتر آشنا میشویم.

فایل cobmres.xml را به منظور استفاده در پروژه به صورت زیر تغییر می‌دهیم.

```
<?xml version="1.0" encoding="utf-8" ?>

<combres xmlns='urn:combres'>
  <filters>
    <filter type="Combres.Filters.FixUrlsInCssFilter, Combres" />
  </filters>
  <resourceSets url="~/combres.axd"
                defaultDuration="30"
                defaultVersion="auto"
                defaultDebugEnabled="false"
                defaultIgnorePipelineWhenDebug="true"
                localChangeMonitorInterval="30"
                remoteChangeMonitorInterval="60">
    <resourceSet name="siteCss" type="css">
      <resource path="~/Styles/Site.css" />
    </resourceSet>
    <resourceSet name="siteJs" type="js">
      <resource path="~/Scripts/jquery-1.10.2.js" />
      <resource path="~/Scripts/jquery-1.10.2.min.js" />
    </resourceSet>
  </resourceSets>
</combres>
```

اگر از نیوگت برای نصب کتابخانه استفاده کرده باشید تغییرات مورد نیاز در فایل وب کانفیگ به صورت خودکار اعمال می‌شود؛ در غیر اینصورت باید قسمتهای زیر را به آن اضافه کنید.

```
<configuration>
  <configSections>
    <section name="combres" type="Combres.ConfigSectionSetting, Combres, Version=2.2, Culture=neutral,
PublicKeyToken=1ca6b37997dd7536" />
  </configSections>
  <system.web>
    <pages>
      <namespaces>
        <add namespace="Combres" />
      </namespaces>
    </pages>
  </system.web>
  <combres definitionUrl="~/App_Data/combres.xml" />
  <appSettings>
    <add key="CombresSectionName" value="combres" />
  </appSettings>
  <runtime>
    <assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
      <dependentAssembly>
        <assemblyIdentity name="AjaxMin" publicKeyToken="21ef50ce11b5d80f" culture="neutral" />
        <bindingRedirect oldVersion="0.0.0.0-4.84.4790.14405" newVersion="4.84.4790.14405" />
      </dependentAssembly>
    </assemblyBinding>
  </runtime>
</configuration>
```

حال باید Route مربوط به Combres را به RouteTable اضافه کنیم. در صورتیکه از نیوگت استفاده کرده باشید، کلاسی به فولدر app_start اضافه شده است که با استفاده از WebActivator اینکار را در Application_Start انجام میدهد؛ در غیر اینصورت باید به صورت زیر این کار را انجام دهیم.

```
protected void Application_Start(object sender, EventArgs e)
        {
            RouteTable.Routes.AddCombresRoute("Combres Route");
        }
```

بعد از انجام مراحل قبل، نوبت به آن رسیده است که از لینکهای combres در صفحات خود استفاده کنیم. شیوه استفاده از آن در وب فرم به این صورت است:

```
<%@ Import Namespace="Combres" %>
<head runat="server">
    <%= WebExtensions.CombresLink("siteCss") %>
    <%= WebExtensions.CombresLink("siteJs") %>
</head>
```

و در MVC به صورت زیر می‌باشد:

```
<%= Url.CombresLink("siteCss") %>
<%= Url.CombresLink("siteJs") %>
```

در هر دو مورد نام ResourceSet برای تولید لینک به متد CombresLink ارسال میشود. پس از اجرای مجدد برنامه و با استفاده از firebug خواهیم دید که به ازای هر **Set** Resource ، یک درخواست به سرور ارسال شده است و حجم فایلها به صورت چشمگیری کاهش یافته است و اطلاعات مربوط به کش هم به آن اضافه شده است.

در ادامه می‌توانید فایل site.css قبلی و فعلی را مقایسه کنید!

GET /combres.axd/siteCss/13518664	200 OK	localhost:55439	220 B

Headers | Response | Cache

```
body{padding-top:50px;padding-bottom:20px}.body-content{padding-left:15px;padding-right:
,textarea{max-width:280px}.field-validation-error{color:#b94a48}.field-validation-valid{
}input.input-validation-error{border:1px solid #b94a48}input[type="checkbox"].input-vali
{border:0 none}.validation-summary-errors{color:#b94a48}.validation-summary-valid{displa
```

GET /combres.axd/siteJs/119039392	200 OK	localhost:55439	67.5 KB
GET site.css	200 OK	localhost:55439	463 B

Headers | Response | Cache

```
body {
    padding-top: 50px;
    padding-bottom: 20px;
}

/* Set padding to keep content from hitting the edges */
.body-content {
    padding-left: 15px;
    padding-right: 15px;
}

/* Set width on the form input elements since they're 100% wide by default */
input,
select,
textarea {
    max-width: 280px;
}
```

در قسمت بعد با سازوکار combres و روش استفاده از فیلترها، بیشتر آشنا میشویم.

CombresProject-67bd3b7299f24c5484edc35537fa990b.zip

نظرات خوانندگان

نویسنده: آرایه
تاریخ: ۱۳۹۳/۰۲/۲۹ ۹:۴۰

ضمن تشکر به خاطر توضیحات خوب، مزیت استفاده از Combres نسبت به امکان bundle و minification خود ASP.NET MVC چیست؟

نویسنده: سامان هاشمی
تاریخ: ۱۳۹۳/۰۲/۳۰ ۱۳:۴۴

شاید عمده مزیت‌های Combres این چهار مورد زیر باشد:
1- از کتابخانه‌های مختلفی برای بهینه سازی پشتیبانی میکند مانند Microsoft Ajax Minifier و YUI Compressor for .NET و Google Closure Compiler.
2- تعریف جداگانه اطلاعات کش به ازای هر فایل
3- فیلترها
4- لاگ کردن پردازشهای داخلی(به صورت پیشفرض توسط Log4Net انجام میگیرد)

نویسنده: rezaei
تاریخ: ۱۳۹۳/۰۲/۳۰ ۱۷:۳۶

با سلام؛ من تمامی مراحلی که فرمودید را انجام دادم. اما موقع اجرا خطای زیر رو می‌گیره

```
System.ArgumentException: Illegal characters in path
```

```
@Url.CombresLink("jquery")
```

نویسنده: سامان هاشمی
تاریخ: ۱۳۹۳/۰۲/۳۱ ۷:۵۷

احتمالا تو نوع آدرس دهی یکی از ResourceSetهای فایل xml یک مشکلی وجود داره! (بهتر بود نمونه کد یا پروژه اینجا قرار میدادید تا بهتر بتونم کمکتون کنم)

نویسنده: afsharsalar
تاریخ: ۱۳۹۳/۰۳/۱۳ ۱۱:۵۶

برای MVC از Combres.Mvc استفاده کنید و سپس از دستور

```
@Html.Raw(WebExtensions.CombresLink("siteCss"))
```

نویسنده: rezaei
تاریخ: ۱۳۹۳/۰۶/۱۱ ۲۳:۱۳

با سلام
موقع استفاده یکی از فایل‌های جاوا اسکریپت توی cache قرار میگیره و تغییرات روش اعمال نمیشه
اما وقتی به صورت عادی بدون combres فایل جاوا اسکریپت رو استفاده میکنم این مشکل وجود نداره
ولی در مورد فایل‌های Css همچین مشکلی وجود نداره

ممنون میشم راهنمایی کنید

عنوان: **انجام کارهای پس زمینه در ASP.NET 4.5.2**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۴/۰۲ ۱۰:۲۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Schedule, MVC

دات نت 4.5.2 قابلیت توکاری را به نام در صف قرار دادن یک کار پس زمینه، اضافه کرده‌است که در ادامه خلاصه‌ای از آن‌را مرور خواهیم کرد.

روش متداول ایجاد کارهای پس زمینه

ساده‌ترین روش انجام کارهای پس زمینه در برنامه‌های دات نتی، استفاده از متدهایی هستند که یک ترد جدید را ایجاد می‌کنند مانند Task.Run, Task.Factory.StartNew, Delegate.BeginInvoke, ThreadPool.QueueUserWorkItem و امثال آن. اما ... این روش‌ها در برنامه‌های ASP.NET ایده‌ی خوبی نیستند!
از این جهت که موتور ASP.NET در این حالات اصلا نمی‌داند که شما کار پس زمینه‌ای را ایجاد کرده‌اید. به همین جهت اگر پروسه‌ی برنامه پس از مدتی recycle شود ، تمام کارهای پس زمینه‌ی موجود نیز از بین خواهند رفت.

معرفی HostingEnvironment.QueueBackgroundWorkItem

متد HostingEnvironment.QueueBackgroundWorkItem به دات نت 4.5.2 اضافه شده‌است تا بتوان توسط آن یک کار پس زمینه را توسط موتور ASP.NET شروع کرد و نه مانند قبل، بدون اطلاع آن. البته باید دقت داشت که این کارهای پس زمینه مستقل از هر نوع درخواستی اجرا می‌شوند.
در این حالت چون موتور ASP.NET از وجود کار پس زمینه‌ی آغاز شده مطلع است، در صورت فرا رسیدن زمان recycle شدن برنامه، کل AppDomain را به یکباره نابود نخواهد کرد. البته این مورد فقط به این معنا است که در صورت فرا رسیدن زمان recycle شدن پروسه، با تنظیم یک CancellationToken، اطلاع رسانی خواهد کرد. در این حالت حداکثر 30 ثانیه فرصت خواهید داشت تا کارهای پس زمینه را بدون مشکل خاتمه دهید. اگر کار پس زمینه در این مدت به پایان نرسد، همانند قبل، کل AppDomain نابود خواهد شد.

این متد دو overload دارد و در هر دو حالت، تنظیم خودکار پارامتر CancellationToken توسط ASP.NET، بیانگر آغاز زمان خاتمه‌ی کل برنامه است:

```
public static void QueueBackgroundWorkItem(Action<CancellationToken> workItem);
public static void QueueBackgroundWorkItem(Func<CancellationToken, Task> workItem);
```

در متد اول، یک متد معمولی از نوع void قابل پردازش است. در متد دوم، می‌توان متدهای async Task دار را که قرار است کارهای async را پردازش کنند، معرفی نمود.
علت استفاده از Action و Func در اینجا، امکان تعریف خلاصه و inline یک متد و ارسال پارامتری به آن از طرف برنامه است، بجای تعریف یک اینترفیس جدید، نیاز به پیاده سازی آن اینترفیس و بعد برای مثال ارسال یک مقدار از طرف برنامه به متد Stop آن (بجای تعریف یک اینترفیس تک متدی، از Action و یا Func نیز می‌توان استفاده کرد).

نمونه‌ای از نحوه‌ی فراخوانی این دو overload را در ذیل مشاهده می‌کنید:

```
HostingEnvironment.QueueBackgroundWorkItem(cancellationToken =>
{
        //todo: ...
});

HostingEnvironment.QueueBackgroundWorkItem(async cancellationToken =>
{
        //todo: ...
        await Task.Delay(20000, cancellationToken);
});
```

پشت صحنه‌ی HostingEnvironment.QueueBackgroundWorkItem

روش استاندارد ثبت و معرفی یک کار پس زمینه در ASP.NET، توسط پیاده سازی اینترفیسی به نام IRegisteredObject انجام می‌شود. سپس توسط متد HostingEnvironment.RegisterObject می‌توان این کلاس را به موتور ASP.NET معرفی کرد. در این حالت زمانیکه AppDomain قرار است خاتمه یابد، متد Stop اینترفیس IRegisteredObject کار اطلاع رسانی را انجام می‌دهد.
توسط QueueBackgroundWorkItem دقیقا از همین روش به همراه فراخوانی ThreadPool.QueueUserWorkItemجهت اجرای متد معرفی شده‌ی به آن استفاده می‌شود.
از مکانیزم IRegisteredObject در DNT Scheduler نیز استفاده شده‌است.

پیشنیازها

ابتدا نیاز است به خواص پروژه مراجعه کرده و Target framework را بر روی 5.4.2 قرار داد. اگر به روز رسانی دوم VS 2013 را نصب کرده باشید، این نگارش هم اکنون بر روی سیستم شما فعال است. اگر خیر، امکان دریافت و نصب آن، به صورت جداگانه نیز وجود دارد:

NET Framework 4.5.2.

Developer pack

محدودیت‌های QueueBackgroundWorkItem

- از آن در خارج از یک برنامه‌ی وب ASP.NET نمی‌توان استفاده کرد.
- توسط آن، خاتمه‌ی یک AppDomain تنها به مدت 30 ثانیه به تاخیر می‌افتد؛ تا فرصت داشته باشید کارهای در حال اجرا را با حداقل خسارت به پایان برسانید.
- یک work item، اطلاعاتی را از فراخوان خود دریافت نمی‌کند. به این معنا که مستقل از زمینه‌ی یک درخواست اجرا می‌شود.
- استفاده‌ی از آن الزاما به این معنا نیست که کار درخواستی شما حتما اجرا خواهد شد. زمانیکه که کار خاتمه‌ی AppDomain آغاز می‌شود، فراخوانی‌های QueueBackgroundWorkItem دیگر پردازش نخواهند شد.
- اگر برنامه به مقدار CancellationToken تنظیم شده توسط ASP.NET دقت نکند، جهت پایان یافتن کار در حال اجرا، صبر نخواهد شد.

نظرات خوانندگان

نویسنده: سعید شیرعلی
تاریخ: ۱۳۹۳/۰۹/۰۸ ۱۵:۱۲

با سلام؛ من می‌خوام یک کار در پس زمینه انجام بدم به صورتی که یک تابع را با چند آرگمان صدا بزنم و این تابع در یک تسک جدید اجرا بشود، به طوری که اگر من این تابع را هر چند بار که صدا بزنم همه آنها در یک تسک جدید و مجزا از هم اجرا بشود.
من چطوری میتونم این کار را انجام بدهم؟ با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۰۸ ۱۵:۱۸

- مطلب فوق در همین مورد است. `BackgroundWorkItem` **`Queue`** به معنای در صف قرار دادن این کارها برای پردازش مجزا از هم است.
- از آن استفاده کردید؟ به چه مشکلی برخوردید؟

نویسنده: محمد سعید
تاریخ: ۱۳۹۳/۱۱/۰۸ ۱۶:۱۹

باسلام
من میخوام یک کار در پس زمینه انجام بدم که طول زمان انجامش بیشتر از خیلی بیشتر از سی ثانیه است. از اونجا که برای انجامش فقط سی ثانیه زمان دارم(طبق مطالب متن)، آیا هنوز میتونم از این روش استفاده کنم؟ اگه آره، چطوری؟
پیشاپیش از توجهتون تشکر میکنم.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۰۸ ۱۷:۴۷

«برای انجامش فقط سی ثانیه زمان دارم»
خیر؛ اینطور نیست. این مورد فقط مرتبط است به زمانیکه قرار است سرور یا برنامه ری‌استارت شوند (خاتمه‌ی `AppDomain`). در غیر اینصورت محدودیت زمانی ندارید.

عنوان: **بررسی مقدمات کتابخانه‌ی JSON.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۳ ۱۹:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, JSON, ASP.NET Web API, json.net

چرا JSON.NET؟

JSON.NET یک کتابخانه‌ی سورس باز کار با اشیاء JSON در دات نت است. تاریخچه‌ی آن به 8 سال قبل بر می‌گردد و توسط یک برنامه نویس نیوزیلندی به نام James Newton King تهیه شده‌است. اولین نگارش آن در سال 2006 ارائه شد؛ مقارن با زمانی که اولین استاندارد JSON نیز ارائه گردید.

این کتابخانه از آن زمان تا کنون، 6 میلیون بار دانلود شده‌است و به علت کیفیت بالای آن، این روزها پایه اصلی بسیاری از کتابخانه‌ها و فریم ورک‌های دات نتی می‌باشد؛ مانند RavenDB تا ASP.NET Web API و SignalR مایکروسافت و همچنین گوگل نیز از آن جهت تدارک کلاینت‌های کار با API خود استفاده می‌کنند.

هرچند دات نت برای نمونه در نگارش سوم آن جهت مصارف WCF کلاسی را به نام DataContractJsonSerializer ارائه کرد، اما کار کردن با آن محدود است به فرمت خاص WCF به همراه عدم انعطاف پذیری و سادگی کار با آن. به علاوه باید درنظر داشت که JSON.NET از دات نت 2 به بعد تا مونو، Win8 و ویندوز فون را نیز پشتیبانی می‌کند.

برای نصب آن نیز کافی است دستور ذیل را در کنسول پاورشل نیوگت اجرا کنید:

```
PM> install-package Newtonsoft.Json
```

معماری JSON.NET

کتابخانه‌ی JSON.NET از سه قسمت عمده تشکیل شده‌است:

الف) JsonSerializer

ب) LINQ to JSON

ج) JSON Schema

الف) JsonSerializer

کار JsonSerializer تبدیل اشیاء دات نتی به JSON و برعکس است. مزیت مهم آن امکانات قابل توجه تنظیم عملکرد و خروجی آن می‌باشد که این تنظیمات را به شکل ویژگی‌های خواص نیز می‌توان اعمال نمود. به علاوه امکان سفارشی سازی هر کدام نیز توسط کلاسی به نام JsonConverter، پیش بینی شده‌است.

یک مثال:

```
var roles = new List<string>
{
   "Admin",
   "User"
};
string json = JsonConvert.SerializeObject(roles, Formatting.Indented);
```

در اینجا نحوه‌ی استفاده از JSON.NET را جهت تبدیل یک شیء دات نتی، به معادل JSON آن مشاهده می‌کنید. اعمال تنظیم Formatting.Indented سبب خواهد شد تا خروجی آن دارای Indentation باشد. برای نمونه اگر در برنامه‌ی خود قصد دارید فرمت JSON تو در تویی را به نحو زیبا و خوانایی نمایش دهید یا چاپ کنید، همین تنظیم ساده کافی خواهد بود.

و یا در مثال ذیل استفاده از یک anonymous object را مشاهده می‌کنید:

```
var jsonString = JsonConvert.SerializeObject(new
{
   Id =1,
   Name = "Test"
}, Formatting.Indented);
```

به صورت پیش فرض تنها خواص عمومی کلاس‌ها توسط JSON.NET تبدیل خواهند شد.

تنظیمات پیشرفته‌تر JSON.NET

مزیت مهم JSON.NET بر سایر کتابخانه‌های موجود مشابه، قابلیت‌های سفارشی سازی قابل توجه آن است. در مثال ذیل نحوه‌ی معرفی JsonSerializerSettings را مشاهده می‌نمائید:

```
var jsonData = JsonConvert.SerializeObject(new
{
   Id = 1,
   Name = "Test",
   DateTime = DateTime.Now
}, new JsonSerializerSettings
{
   Formatting = Formatting.Indented,
   Converters =
   {
      new JavaScriptDateTimeConverter()
   }
});
```

در اینجا با استفاده از تنظیم JavaScriptDateTimeConverter، می‌توان خروجی DateTime استانداردی را به مصرف کنندگان جاوا اسکریپتی سمت کاربر ارائه داد؛ با خروجی ذیل:

```
{
  "Id": 1,
  "Name": "Test",
  "DateTime": new Date(1409821985245)
}
```

نوشتن خروجی JSON در یک استریم

خروجی متد JsonConvert.SerializeObject یک رشته‌است که در صورت نیاز به سادگی توسط متد File.WriteAllText در یک فایل قابل ذخیره می‌باشد. اما برای رسیدن به حداکثر کارآیی و سرعت می‌توان از استریم‌ها نیز استفاده کرد:

```
using (var stream = File.CreateText(@"c:\output.json"))
{
    var jsonSerializer = new JsonSerializer
   {
      Formatting = Formatting.Indented
   };
   jsonSerializer.Serialize(stream, new
   {
     Id = 1,
     Name = "Test",
     DateTime = DateTime.Now
   });
}
```

کلاس JsonSerializer و متد Serialize آن یک استریم را نیز جهت نوشتن خروجی می‌پذیرند. برای مثال response.Output برنامه‌های وب نیز یک استریم است و در اینجا نوشتن مستقیم در استریم بسیار سریعتر است از تبدیل شیء به رشته و سپس ارائه خروجی آن؛ زیرا سربار تهیه رشته JSON از آن حذف می‌گردد و نهایتا GC کار کمتری را باید انجام دهد.

تبدیل JSON رشته‌ای به اشیاء دات نت

اگر رشته‌ی jsonData ایی را که پیشتر تولید کردیم، بخواهیم تبدیل به نمونه‌ای از شیء User ذیل کنیم:

```
public class User
{
```

```
   public int Id { set; get; }
   public string Name { set; get; }
   public DateTime DateTime { set; get; }
}
```

خواهیم داشت:

```
var user = JsonConvert.DeserializeObject<User>(jsonData);
```

در اینجا از متد DeserializeObject به همراه مشخص سازی صریح نوع شیء نهایی استفاده شده‌است.
البته در اینجا با توجه به استفاده از JavaScriptDateTimeConverter برای تولید jsonData، نیاز است چنین تنظیمی را نیز در حالت DeserializeObject مشخص کنیم:

```
var user = JsonConvert.DeserializeObject<User>(jsonData, new JsonSerializerSettings
{
   Converters = {  new JavaScriptDateTimeConverter() }
});
```

مقدار دهی یک نمونه یا وهله‌ی از پیش موجود

متد JsonConvert.DeserializeObject یک شیء جدید را ایجاد می‌کند. اگر قصد دارید صرفا تعدادی از خواص یک وهله‌ی موجود، توسط JSON.NET مقدار دهی شوند از متد PopulateObject استفاده کنید:

```
JsonConvert.PopulateObject(jsonData, user);
```

کاهش حجم JSON تولیدی

زمانیکه از متد JsonConvert.SerializeObject استفاده می‌کنیم، تمام خواص عمومی تبدیل به معادل JSON آن‌ها خواهند شد؛ حتی خواصی که مقدار ندارند. این خواص در خروجی JSON، با مقدار null مشخص می‌شوند. برای حذف این خواص از خروجی JSON نهایی تنها کافی است در تنظیمات JsonSerializerSettings، مقدار NullValueHandling = NullValueHandling.Ignore مشخص گردد.

```
var jsonData = JsonConvert.SerializeObject(object, new JsonSerializerSettings
{
   NullValueHandling = NullValueHandling.Ignore,
   Formatting = Formatting.Indented
});
```

مورد دیگری که سبب کاهش حجم خروجی نهایی خواهد شد، تنظیم DefaultValueHandling = DefaultValueHandling.Ignore است. در این حالت کلیه خواصی که دارای مقدار پیش فرض خودشان هستند، در خروجی JSON ظاهر نخواهند شد. مثلا مقدار پیش فرض خاصیت int مساوی صفر است. در این حالت کلیه خواص از نوع int که دارای مقدار صفر می‌باشند، در خروجی قرار نمی‌گیرند.
به علاوه حذف Formatting = Formatting.Indented نیز توصیه می‌گردد. در این حالت فشرده‌ترین خروجی ممکن حاصل خواهد شد.

مدیریت ارث بری توسط JSON.NET

در مثال ذیل کلاس کارمند و کلاس مدیر را که خود نیز در اصل یک کارمند می‌باشد، ملاحظه می‌کنید:

```
public class Employee
{
    public string Name { set; get; }
}

public class Manager : Employee
{
    public IList<Employee> Reports { set; get; }
}
```

در اینجا هر مدیر لیست کارمندانی را که به او گزارش می‌دهند نیز به همراه دارد. در ادامه نمونه‌ای از مقدار دهی این اشیاء ذکر شده‌اند:

```
var employee = new Employee { Name = "User1" };
var manager1 = new Manager { Name = "User2" };
var manager2 = new Manager { Name = "User3" };
manager1.Reports = new[] { employee, manager2 };
manager2.Reports = new[] { employee };
```

با فراخوانی

```
var list = JsonConvert.SerializeObject(manager1, Formatting.Indented);
```

یک چنین خروجی JSON ایی حاصل می‌شود:

```
{
  "Reports": [
    {
      "Name": "User1"
    },
    {
      "Reports": [
        {
          "Name": "User1"
        }
      ],
      "Name": "User3"
    }
  ],
  "Name": "User2"
}
```

این خروجی JSON جهت تبدیل به نمونه‌ی معادل دات نتی خود، برای مثال جهت رسیدن به manager1 در کدهای فوق، چندین مشکل را به همراه دارد:

- در اینجا مشخص نیست که این اشیاء، کارمند هستند یا مدیر. برای مثال مشخص نیست User2 چه نوعی دارد و باید به کدام شیء نگاشت شود.
- مشکل دوم در مورد کاربر User1 است که در دو قسمت تکرار شده‌است. این شیء JSON اگر به نمونه‌ی معادل دات نتی خود نگاشت شود، به دو وهله از User1 خواهیم رسید و نه یک وهله‌ی اصلی که سبب تولید این خروجی JSON شده‌است.

برای حل این دو مشکل، تغییرات ذیل را می‌توان به JSON.NET اعمال کرد:

```
var list = JsonConvert.SerializeObject(manager1, new JsonSerializerSettings
{
   Formatting = Formatting.Indented,
   TypeNameHandling = TypeNameHandling.Objects,
   PreserveReferencesHandling = PreserveReferencesHandling.Objects
});
```

با این خروجی:

```
{
  "$id": "1",
  "$type": "JsonNetTests.Manager, JsonNetTests",
  "Reports": [
    {
      "$id": "2",
      "$type": "JsonNetTests.Employee, JsonNetTests",
      "Name": "User1"
    },
    {
      "$id": "3",
      "$type": "JsonNetTests.Manager, JsonNetTests",
      "Reports": [
        {
          "$ref": "2"
        }
      ],
      "Name": "User3"
    }
  ],
  "Name": "User2"
}
```

- با تنظیم TypeNameHandling = TypeNameHandling.Objects سبب خواهیم شد تا خاصیت اضافه‌ای به نام $type به خروجی JSON اضافه شود. این نوع، در حین فراخوانی متد JsonConvert.DeserializeObject جهت تشخیص صحیح نگاشت اشیاء بکار گرفته خواهد شد و اینبار مشخص است که کدام شیء، کارمند است و کدامیک مدیر.
- با تنظیم PreserveReferencesHandling = PreserveReferencesHandling.Objects شماره Id خودکاری نیز به خروجی JSON اضافه می‌گردد. اینبار اگر به گزارش دهنده‌ها با دقت نگاه کنیم، مقدار $ref=2 را خواهیم دید. این مورد سبب می‌شود تا در حین نگاشت نهایی، دو وهله متفاوت از شیء با Id=2 تولید نشود.

باید دقت داشت که در حین استفاده از JsonConvert.DeserializeObject نیز باید JsonSerializerSettings یاد شده، تنظیم شوند.

ویژگی‌های قابل تنظیم در JSON.NET

علاوه بر JsonSerializerSettings که از آن صحبت شد، در JSON.NET امکان تنظیم یک سری از ویژگی‌ها به ازای خواص مختلف نیز وجود دارند.
- برای نمونه ویژگی JsonIgnore معروفترین آن‌ها است:

```
public class User
{
   public int Id { set; get; }

   [JsonIgnore]
   public string Name { set; get; }

   public DateTime DateTime { set; get; }
}
```

JsonIgnore سبب می‌شود تا خاصیتی در خروجی نهایی JSON تولیدی حضور نداشته باشد و از آن صرفنظر شود.

- با استفاده از ویژگی JsonProperty اغلب مواردی را که پیشتر بحث کردیم مانند TypeNameHandling، NullValueHandling و غیره، می‌توان تنظیم نمود. همچنین گاهی از اوقات کتابخانه‌های جاوا اسکریپتی سمت کاربر، از اسامی خاصی که از روش‌های نامگذاری دات نتی پیروی نمی‌کنند، در طراحی خود استفاده می‌کنند. در اینجا می‌توان نام خاصیت نهایی را که قرار است رندر شود نیز صریحا مشخص کرد. برای مثال:

```
[JsonProperty(PropertyName = "m_name", NullValueHandling = NullValueHandling.Ignore)]
public string Name { set; get; }
```

همچنین در اینجا امکان تنظیم Order نیز وجود دارد. برای مثال مشخص کنیم که خاصیت X در ابتدا قرار گیرد و پس از آن خاصیت Y رندر شود.

- استفاده از ویژگی JsonObject به همراه مقدار OptIn آن به این معنا است که از کلیه خواصی که دارای ویژگی JsonProperty نیستند، صرفنظر شود. حالت پیش فرض آن OptOut است؛ یعنی تمام خواص عمومی در خروجی JSON حضور خواهند داشت منهای مواردی که با JsonIgnore مزین شوند.

```
[JsonObject(MemberSerialization.OptIn)]
public class User
{
    public int Id { set; get; }

    [JsonProperty]
    public string Name { set; get; }

    public DateTime DateTime { set; get; }
}
```

- با استفاده از ویژگی JsonConverter می‌توان نحوه‌ی رندر شدن مقدار خاصیت را سفارشی سازی کرد. برای مثال:

```
[JsonConverter(typeof(JavaScriptDateTimeConverter))]
public DateTime DateTime { set; get; }
```

تهیه یک JsonConverter سفارشی

با استفاده از JsonConverterها می‌توان کنترل کاملی را بر روی اعمال serialization و deserialization مقادیر خواص اعمال کرد. مثال زیر را در نظر بگیرید:

```
public class HtmlColor
{
   public int Red { set; get; }
   public int Green { set; get; }
   public int Blue { set; get; }
}

var colorJson = JsonConvert.SerializeObject(new HtmlColor
{
  Red = 255,
  Green = 0,
  Blue = 0
}, Formatting.Indented);
```

در اینجا علاقمندیم، در حین عملیات serialization، بجای اینکه مقادیر اجزای رنگ تهیه شده به صورت int نمایش داده شوند، کل رنگ با فرمت hex رندر شوند. برای اینکار نیاز است یک JsonConverter سفارشی را تدارک دید:

```
public class HtmlColorConverter : JsonConverter
    {

        public override bool CanConvert(Type objectType)
        {
            return objectType == typeof(HtmlColor);
        }

        public override object ReadJson(JsonReader reader, Type objectType,
                                        object existingValue, JsonSerializer serializer)
        {
            throw new NotSupportedException();
        }

        public override void WriteJson(JsonWriter writer, object value, JsonSerializer serializer)
        {
            var color = value as HtmlColor;
            if (color == null)
```

```
                return;

            writer.WriteValue("#" + color.Red.ToString("X2")
                + color.Green.ToString("X2") + color.Blue.ToString("X2"));
        }
    }
```

کار با ارث بری از کلاس پایه JsonConverter شروع می‌شود. سپس باید تعدادی از متدهای این کلاس پایه را بازنویسی کرد. در متد CanConvert اعلام می‌کنیم که تنها اشیایی از نوع کلاس HtmlColor را قرار است پردازش کنیم. سپس در متد WriteJson منطق سفارشی خود را می‌توان پیاده سازی کرد.
از آنجائیکه این تبدیلگر صرفا قرار است برای حالت serialization استفاده شود، قسمت ReadJson آن پیاده سازی نشده‌است.

در آخر برای استفاده از آن خواهیم داشت:

```
var colorJson = JsonConvert.SerializeObject(new HtmlColor
{
  Red = 255,
  Green = 0,
  Blue = 0
},  new JsonSerializerSettings
    {
      Formatting = Formatting.Indented,
      Converters = { new HtmlColorConverter() }
    });
```

نظرات خوانندگان

نویسنده: افتابی
تاریخ: ۱۳۹۳/۰۶/۱۳ ۲۱:۵۴

سلام؛ من مطالب مربوطه رو خوندم فقط اینکه توی یه صفحه rozar در mvc من به چه نحو میتونم از آن استفاده کنم ، حتی توی سایت خودش هم رفتم و sample ها رو دیدم فقط میخوام در یک پروژه به چه نحو ازش استفاده کنم و کجا کارش ببرم؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۳ ۲۱:۵۹

در یک اکشن متد، بجای return Json پیش فرض و توکار، می‌شود نوشت:

```
return Content(JsonConvert.SerializeObject(obj));
```

البته این ساده‌ترین روش استفاده از آن است؛ برای مقاصد Ajax ایی.
و یا برای ذکر Content type می‌توان به صورت زیر عمل کرد:

```
return new ContentResult
{
  Content = JsonConvert.SerializeObject(obj),
  ContentType = "application/json"
};
```

نویسنده: رحمت اله رضایی
تاریخ: ۱۳۹۳/۰۶/۱۴ ۱۴:۳

"ASP.NET Web API و SignalR از این کتابخانه استفاده می‌کنند". دلیلی دارد هنوز ASP.NET MVC از این کتابخانه استفاده نکرده است؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۴ ۱۴:۱۹

- تا ASP.NET MVC 5 از JavaScriptSerializer در JsonResult استفاده می‌شود.
- در نگارش بعدی ASP.NET MVC که با Web API یکی شده (یعنی در یک کنترلر هم می‌توانید ActionResult داشته باشید و هم خروجی‌های متداول Web API را با هم) اینبار تامین کننده‌ی JsonResult از طریق تزریق وابستگی‌ها تامین می‌شود و می‌تواند هر کتابخانه‌ای که صلاح می‌دانید باشد. البته یک مقدار پیش فرض هم دارد که دقیقا از JSON.NET استفاده می‌کند.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۷/۱۸ ۱۲:۳۵

یک نکته‌ی تکمیلی

استفاده از استریم‌ها برای کار با فایل‌ها در JSON.NET

```
public static T DeserializeFromFile<T>(string filePath, JsonSerializerSettings settings = null)
        {
            if (!File.Exists(filePath))
                return default(T);

            using (var fileStream = File.OpenRead(filePath))
            {
                using (var streamReader = new StreamReader(fileStream))
                {
                    using (var reader = new JsonTextReader(streamReader))
                    {
                        var serializer = settings == null ? JsonSerializer.Create() :
JsonSerializer.Create(settings);
```

```
                        return serializer.Deserialize<T>(reader);
                    }
                }
            }
        }

        public static void SerializeToFile(string filePath, object data, JsonSerializerSettings
settings = null)
        {
            using (var fileStream = new FileStream(filePath, FileMode.Create))
            {
                using (var streamReader = new StreamWriter(fileStream))
                {
                    using (var reader = new JsonTextWriter(streamReader))
                    {
                        var serializer = settings == null ? JsonSerializer.Create() :
JsonSerializer.Create(settings);
                        serializer.Serialize(reader, data);
                    }
                }
            }
        }
```

عنوان: **تنظیمات و نکات کاربردی کتابخانه‌ی JSON.NET**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۴ ۱۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, JSON, ASP.NET Web API, json.net, Entity framework

پس از بررسی مقدماتی امکانات کتابخانه‌ی JSON.NET، در ادامه به تعدادی از تنظیمات کاربردی آن با ذکر مثال‌هایی خواهیم پرداخت.

گرفتن خروجی CamelCase از JSON.NET

یک سری از کتابخانه‌های جاوا اسکریپتی سمت کلاینت، به نام‌های خواص CamelCase نیاز دارند و حالت پیش فرض اصول نامگذاری خواص در دات نت عکس آن است. برای مثال بجای UserName به userName نیاز دارند تا بتوانند صحیح کار کنند.
روش اول حل این مشکل، استفاده از ویژگی JsonProperty بر روی تک تک خواص و مشخص کردن نام‌های مورد نیاز کتابخانه‌ی جاوا اسکریپتی به صورت صریح است.
روش دوم، استفاده از تنظیمات ContractResolver می‌باشد که با تنظیم آن به CamelCasePropertyNamesContractResolver به صورت خودکار به تمامی خواص به صورت یکسانی اعمال می‌گردد:

```
var json = JsonConvert.SerializeObject(obj, new JsonSerializerSettings
{
   ContractResolver = new CamelCasePropertyNamesContractResolver()
});
```

درج نام‌های المان‌های یک Enum در خروجی JSON

اگر یکی از عناصر در حال تبدیل به JSON، از نوع enum باشد، به صورت پیش فرض مقدار عددی آن در JSON نهایی درج می‌گردد:

```
using Newtonsoft.Json;

namespace JsonNetTests
{
    public enum Color
    {
        Red,
        Green,
        Blue,
        White
    }

    public class Item
    {
        public string Name { set; get; }
        public Color Color { set; get; }
    }

    public class EnumTests
    {
        public string GetJson()
        {
            var item = new Item
            {
                Name = "Item 1",
                Color = Color.Blue
            };

            return JsonConvert.SerializeObject(item, Formatting.Indented);
        }
    }
}
```

با این خروجی:

```
{
  "Name": "Item 1",
  "Color": 2
}
```

اگر علاقمند هستید که بجای عدد 2، دقیقا مقدار Blue در خروجی JSON درج گردد، می‌توان به یکی از دو روش ذیل عمل کرد:
الف) مزین کردن خاصیت از نوع enum به ویژگی JsonConverter از نوع StringEnumConverter:

```
[JsonConverter(typeof(StringEnumConverter))]
  public Color Color { set; get; }
```

ب) و یا اگر می‌خواهید این تنظیم به تمام خواص از نوع enum به صورت یکسانی اعمال شود، می‌توان نوشت:

```
return JsonConvert.SerializeObject(item, new JsonSerializerSettings
{
   Formatting = Formatting.Indented,
   Converters = { new StringEnumConverter() }
});
```

تهیه خروجی JSON از مدل‌های مرتبط، بدون Stack overflow

دو کلاس گروه‌های محصولات و محصولات ذیل را درنظر بگیرید:

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

        public virtual ICollection<Product> Products { get; set; }

        public Category()
        {
            Products = new List<Product>();
        }
    }

    public class Product
    {
        public int Id { get; set; }
        public string Name { get; set; }

        public virtual Category Category { get; set; }
    }
```

این نوع طراحی در Entity framework بسیار مرسوم است. در اینجا طرف‌های دیگر یک رابطه، توسط خاصیتی virtual معرفی می‌شوند که به آن‌ها خواص راهبری یا navigation properties هم می‌گویند.
با توجه به این دو کلاس، سعی کنید مثال ذیل را اجرا کرده و از آن، خروجی JSON تهیه کنید:

```
using System.Collections.Generic;
using Newtonsoft.Json;
using Newtonsoft.Json.Converters;

namespace JsonNetTests
{
    public class SelfReferencingLoops
    {
        public string GetJson()
        {
            var category = new Category
            {
                Id = 1,
                Name = "Category 1"
            };
            var product = new Product
```

```
            {
                Id = 1,
                Name = "Product 1"
            };

            category.Products.Add(product);
            product.Category = category;

            return JsonConvert.SerializeObject(category, new JsonSerializerSettings
            {
                Formatting = Formatting.Indented,
                Converters = { new StringEnumConverter() }
            });
        }
    }
}
```

برنامه با این استثناء متوقف می‌شود:

```
An unhandled exception of type 'Newtonsoft.Json.JsonSerializationException' occurred in
Newtonsoft.Json.dll
Additional information: Self referencing loop detected for property 'Category' with type
'JsonNetTests.Category'. Path 'Products[0]'.
```

اصل خطای معروف فوق «Self referencing loop detected» است. در اینجا کلاس‌هایی که به یکدیگر ارجاع می‌دهند، در حین عملیات Serialization سبب بروز یک حلقه‌ی بازگشتی بی‌نهایت شده و در آخر، برنامه با خطای stack overflow خاتمه می‌یابد.

راه حل اول:

به تنظیمات JSON.NET، مقدار ReferenceLoopHandling = ReferenceLoopHandling.Ignore را اضافه کنید تا از حلقه‌ی بازگشتی بی‌پایان جلوگیری شود:

```
return JsonConvert.SerializeObject(category, new JsonSerializerSettings
{
   Formatting = Formatting.Indented,
   ReferenceLoopHandling = ReferenceLoopHandling.Ignore,
   Converters = { new StringEnumConverter() }
});
```

راه حل دوم:

به تنظیمات JSON.NET، مقدار PreserveReferencesHandling = PreserveReferencesHandling.Objects را اضافه کنید تا مدیریت ارجاعات اشیاء توسط خود JSON.NET انجام شود:

```
return JsonConvert.SerializeObject(category, new JsonSerializerSettings
{
   Formatting = Formatting.Indented,
   PreserveReferencesHandling = PreserveReferencesHandling.Objects,
   Converters = { new StringEnumConverter() }
});
```

خروجی حالت دوم به این شکل است:

```
{
  "$id": "1",
  "Id": 1,
  "Name": "Category 1",
  "Products": [
    {
      "$id": "2",
      "Id": 1,
      "Name": "Product 1",
      "Category": {
        "$ref": "1"
      }
    }
```

```
  ]
}
```

همانطور که ملاحظه می‌کنید، دو خاصیت $id و $ref توسط JSON.NET به خروجی JSON اضافه شده‌است تا توسط آن بتواند ارجاعات و نمونه‌های اشیاء را تشخیص دهد.

نظرات خوانندگان

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۲۳ ۱۵:۱۷

گرفتن خروجی مرتب شده بر اساس نام خواص (جهت مقاصد نمایشی):

تعریف DefaultContractResolver :

```
public class OrderedContractResolver : DefaultContractResolver
    {
        protected override IList<JsonProperty> CreateProperties(
            System.Type type, MemberSerialization memberSerialization)
        {
            return base.CreateProperties(type, memberSerialization).OrderBy(p =>
p.PropertyName).ToList();
        }
    }
```

و بعد معرفی آن به نحو ذیل:

```
return JsonConvert.SerializeObject(data, new JsonSerializerSettings
            {
                ContractResolver = new OrderedContractResolver()
            });
```

عنوان: LINQ to JSON به کمک JSON.NET
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۶/۱۹ ۲۳:۵۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, JSON, ASP.NET Web API, json.net

عموما از امکانات LINQ to JSON کتابخانه‌ی JSON.NET زمانی استفاده می‌شود که ورودی JSON تو در توی حجیمی را دریافت کرده‌اید اما قصد ندارید به ازای تمام موجودیت‌های آن یک کلاس معادل را جهت نگاشت به آن‌ها تهیه کنید و صرفا یک یا چند مقدار تو در توی آن جهت عملیات استخراج نهایی مدنظر است. به علاوه در اینجا LINQ to JSON واژه‌ی کلیدی dynamic را نیز پشتیبانی می‌کند.

همانطور که در تصویر مشخص است، خروجی‌های JSON عموما ترکیبی هستند از مقادیر، آرایه‌ها و اشیاء. هر کدام از این‌ها در LINQ to JSON به اشیاء JArray، JValue و JObject نگاشت می‌شوند. البته در حالت JObject هر عضو به یک JProperty و JValue تجزیه خواهد شد.

برای مثال آرایه [1,2] تشکیل شده‌است از یک JArray به همراه دو JValue که مقادیر آن‌را تشکیل می‌دهند. اگر مستقیما بخواهیم یک JArray را تشکیل دهیم می‌توان از شیء JArray استفاده کرد:

```
var array = new JArray(1, 2, 3);
var arrayToJson = array.ToString();
```

و اگر یک JSON رشته‌ای دریافتی را داریم می‌توان از متد Parse مربوط به JArray کمک گرفت:

```
var json = "[1,2,3]";
var jArray= JArray.Parse(json);
var val = (int)jArray[0];
```

خروجی JArray یک لیست از JTokenها است و با آن می‌توان مانند لیست‌های معمولی کار کرد.

در حالت کار با اشیاء، شیء JObject امکان تهیه اشیاء JSON ایی را دارا است که می‌تواند مجموعه‌ای از JPropertyها باشد:

```
var jObject = new JObject(
new JProperty("prop1", "value1"),
new JProperty("prop2", "value2")
);
var jObjectToJson = jObject.ToString();
```

با JObject به صورت dynamic نیز می‌توان کار کرد:

```
dynamic jObj = new JObject();
jObj.Prop1 = "value1";
jObj.Prop2 = "value2";
jObj.Roles = new[] {"Admin", "User"};
```

این روش بسیار شبیه است به حالتی که با اشیاء جاوا اسکریپتی در سمت کلاینت می‌توان کار کرد.
و حالت عکس آن توسط متد JObject.Parse قابل انجام است:

```
var json = "{ 'prop1': 'value1', 'prop2': 'value2'}";
var jObj = JObject.Parse(json);
var val1 = (string)jObj["prop1"];
```

اکنون که با اجزای تشکیل دهنده‌ی LINQ to JSON آشنا شدیم، مثال ذیل را درنظر بگیرید:

```
var array = @"[
{
  'prop1': 'value1',
  'prop2': 'value2'
},
{
  'prop1': 'test1',
  'prop2': 'test2'
}
]";
var objects = JArray.Parse(array);
var obj1 = objects.FirstOrDefault(token => (string) token["prop1"] == "value1");
```

خروجی JArray یا JObject از نوع IEnumerable است و بر روی آن‌ها می‌توان کلیه متدهای LINQ را فراخوانی کرد. برای مثال در اینجا اولین شیءایی که مقدار خاصیت prop1 آن مساوی value1 است، یافت می‌شود و یا می‌توان اشیاء را بر اساس مقدار خاصیتی مرتب کرده و سپس آن‌ها را بازگشت داد:

```
var values = objects.OrderBy(token => (string) token["prop1"])
.Select(token => new {Value = (string) token["prop2"]})
.ToList();
```

امکان انجام sub queries نیز در اینجا پیش بینی شده‌است:

```
var array = @"[
{
  'prop1': 'value1',
  'prop2': [1,2]
},
{
  'prop1': 'test1',
  'prop2': [1,2,3]
}
]";
var objects = JArray.Parse(array);
var objectContaining3 = objects.Where(token => token["prop2"].Any(v => (int)v == 3)).ToList();
```

در این مثال، خواص prop2 از نوع آرایه‌ای از اعداد صحیح هستند. با کوئری نوشته شده، اشیایی که خاصیت prop2 آن‌ها دارای عضو 3 است، یافت می‌شوند.

عنوان: **حذف همزمان چندین رکورد GridView با استفاده از CheckBox در ASP.NET**
نویسنده: عثمان رحیمی
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۳:۵
آدرس: www.dotnettips.info
گروه‌ها: ADO.NET, ASP.Net, GridView, CheckBox

همانطور که می‌دانید GridView جزء جداناپذیر از اکثر پروژه‌های برنامه نویسان ASP.NET Web forms می‌باشد. اکثرا روشی که در میان برنامه نویسان بیشتر استفاده می‌شود، قرار دادن یک دکمه/لینک در هر ردیف از GridView برای حذف رکورد مورد نظر می‌باشد. در این مقاله قصد دارم روشی را ارائه کنم تا کاربر قادر باشد هر تعداد رکورد را که مدنظر دارد، انتخاب کرده و با فشردن دکمه "حذف" رکوردهای انتخاب شده را حذف کند.

برای درک بهتر، ابتدا جدولی به اسم "Emploee" را در SQL Server با مشخصات زیر ساخته :

```
CREATE TABLE [dbo].[Employee] (
    [EmpId]     INT          NOT NULL,
    [FirstName] VARCHAR (20) NOT NULL,
    [LastName]  VARCHAR (20) NOT NULL,
    [City]      VARCHAR (20) NOT NULL,
    PRIMARY KEY CLUSTERED ([EmpId] ASC)
);
```

1- یک GridView به صفحه افزوده و خاصیت AutoGenerateColumns آن را برابر False قرار دهید . 2- فیلدهایی را که قصد نمایش آنها در GridView را دارید به صورت زیر به GridView بیفزایید :

```
<asp:BoundField DataField="FirstName" HeaderText="First Name" />
```

3- برای قرار دادن کنترل‌های Asp.net که در اینجا منظور CheckBox می‌باشد می‌بایست از TemplateField و قرار دادن تگ ItemTemplate درون آن، به صورت زیر استفاده نمایید :

```
<asp:TemplateField>
                    <ItemTemplate>
                        <asp:CheckBox ID="chkDel" runat="server" />
                    </ItemTemplate>
                </asp:TemplateField>
```

و بعد از تگ GridView دکمه‌ای را برای حذف موارد انتخابی در فرم قرار دهید :

```
<asp:Button ID="btnDeleteRecord" runat="server" OnClick="btnDeleteRecord_Click" Text="Delete"  />
```

برای نمایش یک پیغام به کاربر به منظور Confirm کردن دستور حذف در سمت کلاینت، قطعه کد Javascript زیر را قرار می‌دهیم:

```
function DeleteConfirm()
        {
            var Ans = confirm("Do you want to Delete Selected Employee Record?");
            if (Ans)
            {
                return true;
            }
            else
            {
                return false;
            }
        }
```

و در رویداد Page_Load کدهای زیر را جهت نمایش مقادیر در GridView و افزودن تابع فوق به دکمه، قبل از حذف رکوردها می‌افزاییم :

```
protected void Page_Load(object sender, EventArgs e)
```

```
{
    if(!IsPostBack)
    {
        //Displaying the Data
        showData();
        //Adding an Attribute to Server Control(i.e. btnDeleteRecord)
        btnDeleteRecord.Attributes.Add("onclick", "javascript:return DeleteConfirm()");
    }
}
```

```
//Method for Displaying Data
protected void showData()
{
    DataTable dt = new DataTable();
    SqlConnection con = new SqlConnection(cs);
    SqlDataAdapter adapt = new SqlDataAdapter("select * from Employee",con);
    con.Open();
    adapt.Fill(dt);
    con.Close();
    GridView1.DataSource = dt;
    GridView1.DataBind();
}
```

ابتدا تابع DeleteRecode را به صورت زیر پیاده سازی میکنیم :

که یک پارمتر را از ورودی دریافت میکند که ID رکورد انتخاب شده می‌باشد و با استفاده از ID، رکورد مورد نظر را حذف میکنیم :

```
protected void DeleteRecord(int empid)
{
    SqlConnection con = new SqlConnection(cs);
    SqlCommand com = new SqlCommand("delete from Employee where EmpId=@ID",con);
    com.Parameters.AddWithValue("@ID",empid);
    con.Open();
    com.ExecuteNonQuery();
    con.Close();
}
```

و اما بخش مهم مربوط به رویداد دکمه می‌باشد. در هنگام کلیک بر روی دکمه باید تمامی رکوردهای GridView را چک و تمامی رکوردهایی را که CheckBox آنها تیک خورده است گرفته و ID رکورد مورد نظر را به تابع DeleteRecode فرستاد و در پایان برای اعمال تغییرات، متد ShowDate را فراخوانی و GridView را مجددا Bind می‌کنیم.

```
protected void btnDeleteRecord_Click(object sender, EventArgs e)
{
    foreach (GridViewRow grow in GridView1.Rows)
    {
        //Searching CheckBox("chkDel") in an individual row of Grid
        CheckBox chkdel = (CheckBox)grow.FindControl("chkDel");
        //If CheckBox is checked than delete the record with particular empid
        if(chkdel.Checked)
        {
            int empid = Convert.ToInt32(grow.Cells[1].Text);
            DeleteRecord(empid);
        }
    }
    //Displaying the Data in GridView
    showData();
}
```

نظرات خوانندگان

نویسنده: محمد حسین فخراوری
تاریخ: ۱۳۹۳/۰۷/۰۳ ۱۸:۱۹

شرط بر اساس

```
where id in (........)
```

در (chkdel.Checked)
شما ID بگیرید

نویسنده: عثمان رحیمی
تاریخ: ۱۳۹۳/۰۷/۰۳ ۲۱:۴۱

بله حق با شماست می‌توان کدهای فوق را نسبت به چیزی که گذاشته ام بهینه‌تر نوشت . ممنون از شما

عنوان: **آپلود همزمان چندین فایل در Asp.Net Web Forms**
نویسنده: عثمان رحیمی
تاریخ: ۱۳۹۳/۰۷/۰۴ ۱۷:۵۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, FileUpload

تا قبل از آمدن html5 امکان آپلود چندین فایل در Asp.net web forms امکان پذیر نبود و کاربران می‌بایستی فایل‌های مورد نظر خود را یکی یکی انتخاب و آپلود می‌کردند که تا حد زیادی سخت و حوصله زیادی هم می‌خواست. اما با معرفی html5 یک attribute به تگ مربوط به آپلود فایل به اسم AllowMultiple افزودن شد که مقادیر قابل قبول این attribute مقادیر بولی true,false می‌باشند. اگر این attribute به تگ‌های مربوط به آپلود فایل اضافه نشود، به صورت پیش فرض قادر خواهید بود که هر بار فقط یک فایل را برای آپلود انتخاب کنید. با مقدار دهی این Attribute با مقدار true این اجازه به شما داده می‌شود که در هر بار بتوانید چندین فایل را به صورت همزمان آپلود نمایید.
کد زیر نمایشی از چگونگی بکاربردن این attribute در تگ input می‌باشد:

```
<input type="file" multiple="multiple" name="FileUpload1" id="FileUpload1" />
```

این Attribute از نسخه ASP.NET 4.5 Framework به Asp.net Web forms اضافه شده و برنامه نویسان می‌توانند در صورت نیاز از این امکان استفاده کنند. کد زیر، نحوه افزودن این Attribute به تگ FileUpload در ASP.Net Web Forms می‌باشد:

```
<asp:FileUpload runat="server" ID="FileUploadMultiple" AllowMultiple="true" />
```

و اما به چه روشی می‌توانیم فایل یا فایل‌های انتخاب شده را در این حالت آپلود کرد؟
در صورتیکه از این روش در پروژه‌هایتان استفاده کنید فقط کافیست با یک حلقه تمامی کنترل‌های مورد نظر را پیمایش و هر کدام از فایل‌ها را آپلود و ذخیره نمایید.
برای درک بهتر مطلب، پروژه جدیدی در Asp.net web Forms ایجاد کرده و کنترل‌های زیر را به آن اضافه کنید:

```
<asp:FileUpload runat="server" ID="FileUploadMultiple" AllowMultiple="true" />
        <asp:Button runat="server" ID="btnUlpad" Text="Upload" OnClick="btnUlpad_Click" />
       <asp:Label runat="server" ID="lblMessage"></asp:Label>
```

و در رویداد مربوط به دکمه آپلود، قطعه کد زیر را قرار دهید :

```
int Count = 0;
            foreach (var item in FileUploadMultiple.PostedFiles)
            {
                string Extension = Path.GetExtension(item.FileName);
                string FileName = new Random().Next(1, 50).ToString()+Extension;
                item.SaveAs(Server.MapPath("~")+"//File//"+FileName);
                Count++;
            }
            if (Count == FileUploadMultiple.PostedFiles.Count)
                lblMessage.Text = string.Format("فایل‌های انتخابی با موفقیت آپلود شدند");
            else
                lblMessage.Text = string.Format("{0} فایل با موفقیت آپلود شد {1} از", Count,
FileUploadMultiple.PostedFiles.Count);
```

کدهای پروژه ; Upload-multiple-.rar

عنوان: **آشنایی با چالش های امنیتی در توسعه برنامه‌های تحت وب، بخش اول**
نویسنده: سید مهران موسوی
تاریخ: ۱۳۹۳/۰۷/۱۱ ۸:۵۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Security, xss, Cross-site scripting

در پروژه‌های بزرگ نرم افزاری، از قدیم بحث تامین امنیت پروژه، یکی از چالش‌های مهم بوده است. از دیدگاه شخصی بنده، یک مدیر نرم افزار یا حتی یک توسعه دهنده‌ی برنامه‌های تحت وب، لازم است علاوه بر صرف وقت مطالعاتی و آشنایی و تسلط بر مباحث طراحی معماری سیستم‌های تحت وب، که از اهمیت بالا و مقیاس بزرگی برخوردارند آشنایی لازم را با چالش‌های امنیتی در پیاده سازی اینگونه سیستم‌ها داشته باشد. امنیت در یک سیستم بزرگ و ارائه دهنده خدمات، باعث می‌شود تا کاربر علاوه بر یک تجربه کاربری (user experience) خوب از سیستم که حاصل پیاده سازی صحیح سیستم می‌باشد، اعتماد ویژه‌ای به سیستم مذکور داشته باشد. گاها کاربران به علت بی اعتمادی به شرایط امنیتی حاکم بر یک سیستم، از تجربه کاربری خوب یک سیستم چشم پوشی می‌کنند. اهمیت این مسئله تا جاییست که غول‌های تکنولوژی دنیا همچون Google درگیر این چالش می‌باشند و همیشه سعی بر تامین امنیت کاربران علاوه بر ایجاد تجربه کاربری خوب دارند. پس عدم توجه به این موضوع میتواند خسارات وارده جبران ناپذیری را به یک سیستم از جهت‌های مختلف وارد کند.

در این سری از مقالات، بنده سعی دارم تا حد توان در رابطه با چالش‌های امنیتی موجود در زمینه توسعه برنامه‌های تحت وب، مطالبی را منتشر کنم. از این رو امیدوارم تا این سری از مقالات برای دوستان مفید واقع گردد.

در این سری از مقالات چالش‌های امنیتی زیر مورد بحث و بررسی واقع خواهند گردید

XSS , LDAPi ,RFI ,LFI ,SQLi ,RFD ,LFD ,SOF ,BSQLI ,DNN ,BOF ,CRLF ,CSRF ,SSI ,PCI ,SCD ,AFD ,RCE

در بخش اول از این سری مقالات ، به بررسی آسیب پذیری **Cross-site scripting** میپردازیم .

واژه XSS مخفف Cross-site scripting ، نوعی از آسیب پذیریست که در برنامه‌های تحت وب نمود پیدا میکند. به طور کلی و خلاصه، این آسیب پذیری به فرد نفوذ کننده اجازه تزریق اسکریپت‌هایی را به صفحات وب، می‌دهد که در سمت کاربر اجرا می‌شوند (Client Side scripts) . در نهایت این اسکریپت‌ها توسط سایر افرادی که از صفحات مورد هدف قرار گرفته بازدید می‌کنند اجرا خواهد شد.

هدف از این نوع حمله :

بدست آوردن اطلاعات کوکی‌ها و سشن‌های کاربران (مرتبط با آدرسی که صفحه آلوده شده در آن قرار دارد) است. سپس فرد نفوذ کننده متناسب با اطلاعات بدست آمده می‌تواند به اکانت شخصی کاربران مورد هدف قرار گرفته، نفوذ کرده و از اطلاعات شخصی آن‌ها سوء استفاده کند .

به صورت کلی دو طبقه بندی برای انواع حملات Cross-site scripting وجود دارند.

حملات XSS ذخیره سازی شده (Stored XSS Attacks) :

در این نوع ، کدهای مخرب تزریق شده، در سرور سایت قربانی ذخیره میشوند. محل ذخیره سازی می‌تواند دیتابیس سایت یا هر جای دیگری که داده‌ها توسط سایت یا برنامه تحت وب بازیابی می‌شوند و نمایش داده می‌شوند باشد. اما اینکه چگونه کدهای مخرب در منابع یاد شده ذخیره میشوند؟

فرض کنید در سایت جاری آسیب پذیری مذکور وجود دارد. راه‌های ارسال داده‌ها به این سایت چیست؟ نویسندگان میتوانند مطلب ارسال کنند و کاربران میتوانند نظر دهند. حال اگر در یکی از این دو بخش بررسی‌های لازم جهت مقابله با این آسیب پذیری

وجود نداشته باشد و نوشته‌های کاربران که می‌تواند شامل کدهای مخرب باشد مستقیما در دیتابیس ذخیره شده و بدون هیچ اعتبار سنجی نمایش داده شود چه اتفاقی رخ خواهد داد؟ مسلما با بازدید صفحه آلوده شده، کدهای مخرب بر روی مرورگر شما اجرا و کوکی‌های سایت جاری که متعلق به شما هستند برای هکر ارسال میشود و ...

حملات XSS منعکس شده (Reflected XSS Attacks) :

در این نوع از حمله، هیچ نوع کد مخربی در منابع ذخیره سازی وبسایت یا اپلیکیشن تحت وب توسط فرد مهاجم ذخیره نمی‌شود ! بلکه از ضعف امنیتی بخش‌هایی همچون بخش جستجو وب سایت، بخش‌های نمایش پیغام خطا و ... استفاده میشود ... اما به چه صورت؟

در بسیاری از سایت‌ها، انجمن‌ها و سیستم‌های سازمانی تحت وب، مشاهده می‌شود که مثلا در بخش جستجو، یک فیلد برای وارد کردن عبارت جستجو وجود دارد. پس از وارد کردن عبارت جستجو و submit فرم، علاوه بر نمایش نتایج جستجو، عبارت جستجو شده نیز به نمایش گذاشته میشود و بعضا در بسیاری از سیستم‌ها این عبارت قبل از نمایش اعتبار سنجی نمی‌شود که آیا شامل کدهای مخرب می‌باشد یا خیر. همین امر سبب میشود تا اگر عبارت جستجو شامل کدهای مخرب باشد، آن‌ها به همراه نتیجه‌ی جستجو اجرا شوند.

اما این موضوع چگونه مورد سوء استفاده قرار خواهد گرفت؟ مگر نه اینکه این عبارت ذخیره نمیشود پس با توضیحات فوق، کد فقط بر روی سیستم مهاجم که کد جستجو را ایجاد می‌کند اجرا می‌شود، درست است؟ بله درست است ولی به نقطه ضعف زیر توجه کنید ؟

```
www.test.com/search?q=PHNjcmlwdD5hbGVydChkb2N1bWVudC5jb29raWUpOzwvc2NyaXB0Pg==
```

این آدرس حاصل submit شدن فرم جستجو وب‌سایت test (نام وب‌سایت واقعی نیست و برای مثال است) و ارجاع به صفحه نتایج جستجو میباشد. در واقع این لینک برای جستجوی یک کلمه یا عبارت توسط این وبسایت تولید شده و از هر کجا به این لینک مراجعه کنید عبارت مورد نظر مورد جستجو واقع خواهد شد. در واقع عبارت جستجو به صورت Base64 به عنوان یک query String به وبسایت ارسال می‌شود؛ علاوه بر نمایش نتایج، عبارت جستجو شده نیز به کاربر نشان داده شده و اگر آسیب پذیری مورد بحث وجود داشته باشد و عبارت شامل کدهای مخرب باشد، کدهای مخرب بر روی مرورگر فردی که این لینک را باز کرده اجرا خواهد شد!

در این صورت کافیست فرد مهاجم لینک مخرب را به هر شکلی به فرد مورد هدف بدهد (مثلا ایمیل و ...). حال در صورتیکه فرد لینک را باز کند (با توجه به اینکه لینک مربوط به یک سایت معروف است و عدم آگاهی کاربر از آسیب پذیری موجود در لینک، باعث باز کردن لینک توسط کاربر می‌شود)، کدها بر روی مرورگرش اجرا شده و کوکی‌های سایت مذکور برای مهاجم ارسال خواهد شد ... به این نوع حمله XSS ، نوع انعکاسی می‌گویند که کاملا از توضیحات فوق الذکر، دلیل این نامگذاری مشخص می‌باشد.

اهمیت مقابله با این حمله :

برای نمونه این نوع باگ حتی تا سال گذشته در سرویس ایمیل یاهو وجود داشت. به شکلی که یکی از افراد انجمن hackforums به صورت Private این باگ را به عنوان Yahoo 0-Day XSS Exploit در محیط زیر زمینی و بازار سیاه هکرها به مبلغ چند صد هزار دلار به فروش می‌رساند. کاربران مورد هدف کافی بود تا فقط یک ایمیل دریافتی از هکر را باز کنند تا کوکی‌های سایت یاهو برای هکر ارسال شده و دسترسی ایمیل‌های فرد قربانی برای هکر فراهم شود ... (در حال حاظر این باگ در یاهو وجو ندارد).

چگونگی جلوگیری از این آسیب پذیری

در این سری از مقالات کدهای پیرامون سرفصل‌ها و مثال‌ها با ASP.net تحت فریم ورک MVC و به زبان C# خواهند بود. هر چند کلیات مقابله با آسیب پذیری هایی از این دست در تمامی زبان‌ها و تکنولوژی‌های تحت وب یکسان میباشند.

خوشبختانه کتابخانه‌ای قدرتمند برای مقابله با حمله مورد بحث وجود دارد با نام AntiXSS که میتوانید آخرین نسخه آن را با فرمان زیر از طریق nugget به پروژه خود اضافه کنید. البته ذکر این نکته **حائز اهمیت** است که Asp.net و فریم ورک MVC به صورت توکار تا حدودی از بروز این حملات جلوگیری می‌کند. برای مثال به این صورت که در View ها شما تا زمانی که از MvcHtmlString استفاده نکنید تمامی محتوای مورد نظر برای نمایش به صورت Encode شده رندر می‌شوند. این داستان برای Url ها هم که به صورت پیش فرض encode میشوند صدق می‌کند. ولی گاها وقتی شما برای ورود اطلاعات مثلا از یک ادیتور WYSWYG استفاده می‌کنید و نیاز دارید داده‌ها را بدون encoding رندر کنید. آنگاه به ناچار مجاب بر اعمال یک سری سیاست‌های خاص‌تر بر روی داده مورد نظر برای رندر می‌شوید و نمی‌توانید از encoding توکار فوق الذکر استفاده کنید. آنگاه این کتابخانه در اعمال سیاست‌های جلوگیری از بروز این آسیب پذیری می‌تواند برای شما مفید واقع شود.

```
PM> Install-Package AntiXSS
```

این کتابخانه مجموعه‌ای از توابع کد کردن عبارات است که از مواردی همچون Html, XML, Url, Form, LDAP, CSS, JScript and VBScript پشتیبانی می‌کند. استفاده از آن بسیار ساده می‌باشد. کافیست ارجاعات لازم را به پروژه خود افزوده و به شکل زیر از توابع ارائه شده توسط این کتابخانه استفاده کنید:

```
…
var reviewContent = model.UserReview;
reviewContent = Microsoft.Security.Application.Encoder.HtmlEncode(review);
…
```

امیدوارم در اولین بخش از این سری مقالات، به صورت خلاصه مطالب مهمی که باعث ایجاد فهم کلی در رابطه با حملات Xss وجود دارد، برای دوستان روشن شده و پیش زمینه فکری برای مقابله با این دست از حملات برایتان به وجود آمده باشد.

نظرات خوانندگان

نویسنده: مجید و مسعود منظوری
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱۲:۱۳

سلام
جایی دیدیم که نوعی از حملات هست که هکر تو یکی از پوشه‌های سایت (مثلا پوشه Images) یک اسکریپت یا یک صفحه asp قرار میده و کاربر رو به سایت خودش هدایت میکنه، یا اینکه یک عکس نشون میده
ظاهرا این حملات بیشتر برای استفاده از سایت هدف برای تبلیغات به کار میره
راه مقابله با این حملات چی میتونه باشه؟
این حملات به نام حملات CLRF شناخته میشن گویا

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱۲:۵۴

دسترسی اجرایی را از این پوشه‌ها با تنظیم ذیل در وب کانفیگ سایت، بگیرید:

```
<location path="upload">
  <system.webServer>
    <handlers accessPolicy="Read" />
  </system.webServer>
</location>
```

اطلاعات بیشتر: ^ و ^

نویسنده: سید مهران موسوی
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱۳:۲۳

در واقع آپلود فایل‌های مخرب نوعی حفره امنیتی در توسعه اپلیکیشن هست که موجب سوء استفاده میتونه واقع بشه . این حفره امنیتی به Unrestricted File Upload معروفه که با نکته ای که آقای نصیری ذکر کردن قابل حل هست در asp net (لینک‌های ارجاعی رو مطالعه بفرمایید)

در رابطه با ارجاع کاربران به سایت هدف نوع حمله همون CLRF هست که شامل xss هم میشه .
مقابله با این نوع حملات ساده و استوار بر دو اصل اساسی هست :
1 : همیشه بر این قانون که به ورودی داده‌های کاربر اعتماد نکنید استوار باشید
2 : تمامی ورودی‌های کاربران را که قرار است مورد مشاهده در عموم و خصوص کاربران در مرورگر باشد تا حد ممکن پاکسازی کنید
مطالعه بیشتر ^

عنوان: **آشنایی با WebDav و نحوه استفاده از آن**
نویسنده: رضا بازرگان
تاریخ: ۱۳۹۳/۰۹/۱۳ ۱۸:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, MS Office, webdav, cloud

WebDAV استانداردی است بر روی پروتکل HTTP که Requestها و Responseهای مدیریت یک فایل را بر روی سرویس دهنده وب، تشریح می‌کند.

برای درک چرایی وجود این استاندارد بهتر است ذهن خود را معطوف به نحوه‌ی عملکرد سیستم فایل در OS کنیم که شامل APIهای خاص برای دسترسی نرم افزارهای گوناگون به فایل‌های روی یک سیستم است.
حال فکر کنید یک سرور Cloud راه اندازی نموده‌اید که قرار است مدیریت فایل‌ها و پرونده‌های Office را بر عهده داشته باشد و چون امکان ویرایش اسناد Office بر روی وب را ندارید، نیاز است تا اجازه دهید نرم افزارهای Office مستقیما فایل‌ها را از روی سرور شما باز کنند و بعد از تغییرات، به جای ذخیره در سیستم local، محتوا را به فایل روی سرور ارسال کنند.
در مفهوم web عملا این کار غیر استاندارد و نادرست است. همه درخواست‌ها و جواب‌ها باید بر روی پروتکل Http باشند. خوب حال تصور کنید نرم افزارهای Office قابلیت آن را داشته باشند که به جای تحویل محتوا به سیستم عامل برای ذخیره‌ی آن بر روی سیستم local، محتوا را به یک آدرس ارسال نمایند و پشت آن آدرس، متدی باشد که بتواند به درخواست رسیده، به درستی پاسخ دهد.
این یعنی باید سمت سرور متدی با قابلیت ارسال پاسخ‌های درست و در سمت کلاینت نرم افزاری با قابلیت ارسال درخواست‌های مناسب وجود داشته باشد.
WebDAV استاندارد تشریح محتوای درخواست‌ها و پاسخ‌های مربوط به مدیریت فایل‌ها است.
خوشبختانه نرم افزارهای Office و بسیاری از نرم افزارهای دیگر، استاندارد WebDAV را پشتیبانی می‌کنند و فقط لازم است برای سرورتان متدی با قابلیت پشتیبانی از درخواست‌های WebDAV پیاده سازی نمایید و البته متاسفانه کتابخانه‌های سورس باز چندانی برای WebDAV در سرور دات نت وجود ندارد. من ماژولی را برای کار با WebDAV نوشتم و سورسش را در Git قرار دادم. برای این مثال هم از همین کتابخانه استفاده می‌کنم.
ابتدا یک پروژه‌ی وب MVC ایجاد نمایید و پکیج xDav را از nugget نصب کنید.

```
PM> Install-Package xDav
```

اگر به web.config نگاهی بیاندازیم می‌بینیم یک module به نام xDav به وب سرور اضافه شده که بررسی درخواست‌های WebDAV را به عهده دارد.

```
<system.webServer>
        <modules>
            <add name="XDav" type="XDav.XDavModule, XDav" />
        </modules>
  </system.webServer>
```

همچنین یک Section جدید هم به config برای پیکربندی xDav اضافه شده است.

```
<XDavConfig Name="xdav">
    <FileLocation URL="xdav" PathType="Local"></FileLocation>
  </XDavConfig>
```

خاصیت Name برای xDav نشانگر درخواست‌هایی است که باید توسط این ماژول اجرا شوند. در اینجا یعنی درخواست‌هایی که آدرس آن‌ها شامل "/xdav/" باشد، توسط این ماژول Handle می‌شوند. عبارت بعد از مقدار Name در URL هم طبیعتا نام فایل مورد نظر شماست.

FileLocation آدرس پوشه ای است که فایل‌ها در آن ذخیره و یا بازخوانی می‌شوند. اگر FileType با مقدار Local تنظیم شود،

یعنی باید یک پوشه به نام خاصیت URL که در اینجا xdav است در پوشه‌ی اصلی وب شما وجود داشته باشد و اگر با Server مقدار دهی شود URL باید یک آدرس فیزیکی بر روی سرور داشته باشد . مثل "c:\webdav"URL=

ما در این مثال مقادیر را به صورت پیش‌فرض نگه می‌داریم. یعنی باید در پوشه‌ی وب، یک Folder با نام xdav ایجاد کنیم.

در ادامه چند فایل word را برای تست در این پوشه کپی می‌کنم.

می خواهیم در صفحه Index، لیستی از فایل‌های درون این پوشه را نمایش دهیم طوری که در صورت کلیک بر روی هر کدام از آن‌ها، آدرس WebDav فایل مورد نظر را به Word ارسال کنیم.

بعد از نصب Office ، در registry چند نوع Url تعریف می‌شود که معرف اپلیکیشنی است که آدرس به آن فرستاده شود. این دقیقا همان چیزیست که ما به آن نیاز داریم. کافیست آدرس WebDav فایل را بعد از عبارت " |ms-word:ofe|u " در یک لینک قرار دهیم تا آدرس به نرم افزار Word ارسال شود. یعنی آدرس URL باید این شکلی باشد:

```
ms-word:ofe|u|http://Webaddress/xdav/filename
```

Webaddress آدرس وبسایت و filename نام فایل مورد نظرمان است. عبارت /xdav/ هم که نشان می‌دهد درخواست هایی که این آدرس را دارند باید توسط ماژول xDav پردازش شوند.

کلاسی با نام DavFile در پوشه‌ی Model ایجاد می‌کنم:

```
public class DavFile
    {
        public string Name { get; set; }
```

```
        public string Href(string webAddress)
        {
            return string.Format("ms-word:ofe|u|http://{0}/xdav/{1}", webAddress, Name);
        }
    }
```

اکشن متد Index را در Home Controller، مانند زیر تغییر دهید:

```
var dir = new DirectoryInfo(XDav.Config.ConfigManager.DavPath);
            var model = dir.GetFiles().ToList()
                                        .Select(f =>
                                            new DavFile() {
                                              Name = f.Name
                                          });
            return View(model);
```

یک لیست از فایل هایی که در پوشه‌ی webDav قرار دارند تهیه می‌کنیم و به View ارسال می‌کنیم. View را هم مثل زیر بازنویسی می‌کنیم.

```
@model IEnumerable<WebDavServer.Models.DavFile>
<h1>
    File List
</h1>

<ul>
    @foreach (var item in Model)
    {
        <li> <a href="@Html.Raw(item.Href(ViewContext.HttpContext.Request.Url.Authority))">
@Html.Raw(item.Name) </a></li>
    }
</ul>
```

قرار است به ازای هر فایل، لینکی نمایش داده شود که با کلیک بر روی آن، آدرس فایل به word ارسال می‌شود. بعد از ثبت تغییرات، word محتوا را به همان آدرس ارسال می‌کند و ماژول xDav محتوا را در فایل فیزیکی سرور ذخیره خواهد کرد.

برنامه را اجرا کنید و بر روی فایل‌ها کلیک نمایید. اگر نرم افزار Office روی کامپیوترتان باز باشد با کلیک بر روی هر کدام از فایل‌ها، فایل word باز شده و می‌توانید محتوا را تغییر داده و ذخیره نمایید.

نرم افزار کلاینت (word) درخواست هایی با verbهای مشخص که در استاندارد WebDav ذکر شده به آدرس مورد نظر می‌فرستد. سرور WebDav درخواست را بر اساس Verb آن Request پردازش کرده و Response استاندارد را ایجاد میکند.

نرم افزار word پس از دریافت یک URL، به جای فرمت فیزیکی فایل، درخواست هایی را با تایپ‌های Option, Head, lock, get, post و unlock ارسال می‌کند. محتوای درخواست و پاسخ هر کدام از تایپ‌ها در استاندارد webDav تعریف شده و ماژول xDav آن را پیاده سازی نموده است.

دریافت پروژه مثال

نظرات خوانندگان

نویسنده: نمو
تاریخ: ۱۳۹۳/۰۹/۱۵ ۱۱:۴۹

سلام؛ ممنون.
آیا از مرورگر خاصی باید استفاده شود؟ پروژه نمونه هم کار نمی‌کند و وقتی روی لینک‌های کلیک می‌کنم هیچ اتفاقی نمی‌افتد.

نویسنده: رضا بازرگان
تاریخ: ۱۳۹۳/۰۹/۱۵ ۱۳:۴۶

با سلام.
مطمئن شوید که Office رو سیستم شما نصب است. من Packageها رو از پروژه نمونه حذف کردم . لطفا مجددا آن را نصب کنید.
فرقی در استفاده از مرورگر هم نیست. می‌توانید از پروژه ای که روی Git گذاشتم هم استفاده کنید که کاملتر است.

عنوان: **کمی درباره httpmodule**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۰۹/۲۶ ۲۰:۱۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IHttpModule, IhttpHandler

قبل از اینکه به httpmodulehaها بپردازیم، اجازه بدید کمی در در مورد httphandler اطلاعات کسب کنیم. httphandler ویژگی است که از asp.net به بعد ایجاد شد و در asp کلاسیک خبری از آن نیست.
یک httphandler کامپوننتی است که از طریق اینترفیس System.Web.IHttpHandler پیاده سازی میشود و به پردازش درخواست‌های رسیده از httprequest رسیدگی می‌کند.
فرض کنید کاربری درخواست صفحه default.aspx را کرده است و سرور هم پاسخ آن را می‌دهد. در واقع پردازش اینکه چه پاسخی باید به کاربر یا کلاینت ارسال شود بر عهده این کامپوننت می‌باشد. برای وب سرویس هم موضوع به همین صورت است؛ هر نوع درخواست HTTP از این طریق انجام می‌شود.
حال به سراغ httpmodule می‌رویم. httpmodulehaها اسمبلی یا ماژول‌هایی هستن که بر سر راه هر درخواست کاربر از سرور قرار گرفته و قبل از اینکه درخواست شما به httphandler برسد، اول از فیلتر این‌ها رد می‌شود. در واقع موقعی که شما درخواست صفحه default.aspx را می‌کنید، درخواست شما به موتور asp.net ارسال می‌شود و از میان فیلترهایی رد می‌شود تا به دست httphandler برای پردازش خروجی برسد. برای همین اگر گاهی به جای گفتن asp.net engine عبارت asp.net pipeline هم میگویند همین هست؛ چون درخواست شما از بین بخش‌های زیادی می‌گذرد تا به httphandler برسد که httpmodule یکی از آن بخش هاست. با هر درخواستی که سرور ارسال می‌شود، httpmodulehaها صدا زده می‌شوند و به برنامه نویس امکان بررسی اطلاعات درخواستی و پردازش درخواست‌ها را در ورودی و خروجی، می‌دهد و شما میتوانید هر عملی را که نیاز دارید انجام دهید.
تعدادی از این ماژول‌های آماده، همان stateها و Authentication می‌باشند.
تصویر زیر نحوه‌ی ارسال و بازگشت یک درخواست را به سمت httphandler نشان می‌دهد

برنامه نویس هم میتواند با استفاده از اینترفیس‌های IHttpModule و IHttpHandler در درخواست‌ها دخالت نماید.
برای شروع یک کلاس ایجاد کنید که اینترفیس IHttpModule را پیاده سازی می‌کند. شما دو متد را باید در این کلاس بنویسید؛ یکی Init و دیگر Dispose. همانطور که مطلع هستید، اولی در ابتدای ایجاد شیء و دیگر موقع از دست رفتن شی صدا زده می‌شود.
متد Init یک آرگومان از نوع httpapplication دارد که مانند رسم نامگذاری متغیرها، بیشتر به اسم context یا app نام گذاری می‌شوند:

```
public void Init(HttpApplication app)
{
   app.BeginRequest += new EventHandler(OnBeginRequest);
}

public void Dispose(){ }
```

همانطور که می‌بینید این شیء یک رویداد دارد که ما این رویداد را به تابعی به نام OnBeginRequest متصل کردیم. سایر رویدادهای موجود در httpapplication به شرح زیر می‌باشند:

BeginRequest	این رویداد اولین رویدادی است که اجرا می‌شود، هر نوع عملی که میخواهید در ابتدای ارسال درخواست انجام دهید، باید در این قسمت قرار بگیرد؛ مثلا قرار دادن یک بنر بالای صفحه
AuthenticateRequest	خود دانت از یک سیستم امنیتی توکار بهره مند است و اگر می‌خواهید در مورد آن خصوصی سازی انجام بدهید، این رویداد می‌تواند کمکتان کند
AuthorizeRequest	بعد از رویداد بالا، این رویداد برای شناسایی انجام می‌شود. مثلا دسترسی ها؛ دسترسی به قسمت هایی خاصی از منابع به او داده شود و قسمت هایی بعضی از منابع از او گرفته شود.
ResolveRequestCache	این رویداد برای کش کردن اطلاعات استفاده می‌شود. خود دانت تمامی این رویدادها را به صورت تو کار فراهم آورده است؛ ولی اگر باز خصوصی سازی خاصی مد نظر شماست می‌توانید در این قسمت‌ها، تغییراتی را اعمال کنید. مثلا ایجاد file caching به جای memory cache و ...
AcquireRequestState	این قسمت برای مدیریت state می‌باشد مثلا مدیریت session ها
PreRequestHandlerExecute	این رویداد قبل از httphandler اجرا می‌شود.
PostRequestHandlerExecute	این رویداد بعد از httphandler اجرا می‌شود.
ReleaseRequestState	این رویداد برای این صدا زده می‌شود که به شما بگوید عملیات درخواست پایان یافته است و باید stateهای ایجاد شده را release یا رها کنید.
UpdateRequestCache	برای خصوصی سازی output cache بکار می‌رود.
EndRequest	عملیات درخواست پایان یافته است. در صورتیکه قصد نوشتن دیباگری در طی تمامی عملیات دارید، میتواند به شما کمک کند.
PreSendRequestHeaders	این رویداد قبل از ارسال طلاعات هدر هست. اگر قصد اضافه کردن اطلاعاتی به هدر دارید، این رویداد را به کار ببرید.
PreSendRequestContent	این رویداد موقعی صدا زده می‌شود که متد response.flush فراخوانی شود.، اگر می‌خواهید به محتوا چیزی اضافه کنید، از اینجا کمک بگیرید.

BeginRequest	این رویداد اولین رویدادی است که اجرا می‌شود، هر نوع عملی که میخواهید در ابتدای ارسال درخواست انجام دهید، باید در این قسمت قرار بگیرد؛ مثلا قرار دادن یک بنر بالای صفحه
Error	این رویداد موقعی رخ می‌دهد که یک استثنای مدیریت نشده رخ بدهد. برای نوشتن سیستم خطایابی خصوصی از این قسمت عمل کنید.
Disposed	این رویداد موقعی صدا زده میشود که درخواست، بنا به هر دلیلی پایان یافته است. برای عملیات پاکسازی و .. می‌شود از آن استفاده کرد. مثلا یک جور rollback برای کارهای انجام گرفته.

کد زیر را در نظر بگیرید:

کد زیر یک رویداد را تعریف کرده و سپس خود httpapplication را به عنوان sender استفاده می‌کند.

در اینجا قصد داریم یکی از صفحات را در خروجی تغییر دهیم. آدرس تایپ شده همان باشد ولی صفحه‌ی درخواست شده، صفحه‌ی دیگری است. این کار موقعی بیشتر کاربردی است که آدرس یک صفحه تغییر کرده و کاربر آدرس قبلی را وارد می‌کند. حالا یا از طریق بوک مارک یا از طریق یک لینک، در یک جای دیگر و شما میخواهید او را به صفحه‌ای جدید انتقال دهید، ولی در نوار آدرس، همان آدرس قبلی باقی بماند. همچنین کار دیگری که قرار است انجام بگیرد محاسبه مدت زمان رسیدگی به درخواست را محاسبه کند ، برای همین در رویداد BeginRequest زمان شروع درخواست را ذخیره و در رویداد EndRequest با به دست آوردن اختلاف زمان فعلی با زمان شروع به مدت زمان مربوطه پی خواهیم برد.

با استفاده از app.Context.Request.RawUrl آدرس اصلی و درخواست شده را یافته و در صورتی که شامل نام صفحه مربوطه بود، با نام صفحه‌ی جدید جابجا می‌کنیم تا اطلاعات به صفحه‌ی جدید پاس شوند ولی در نوار آدرس، هنوز آدرس قبلی یا درخواست شده، قابل مشاهده است.

در خط app.Context.Items["start"] که یک کلاس ارث بری شده از اینترفیس IDictionary است، بر اساس کلید، داده شما را ذخیره و در مواقع لزوم در هر رویداد به شما باز می‌گرداند.

```
public class UrlPath : IHttpModule
    {
        public void Init(HttpApplication app)
        {
            app.BeginRequest+=new EventHandler(_BeginRequest);
            app.EndRequest+=new EventHandler(_EndRequest);
        }

        public void Dispose()
        {

        }

         void _BeginRequest(object sender, EventArgs e)
         {

             HttpApplication app = (HttpApplication) sender;
             app.Context.Items["start"] = DateTime.Now;

             if (app.Context.Request.RawUrl.ToLower().Contains("tours_list.aspx"))
             {

app.Context.RewritePath(app.Context.Request.RawUrl.ToLower().Replace("tours_list.aspx","tours_cat.aspx"
));
             }

         }
         void _EndRequest(object sender, EventArgs e)
         {
             HttpApplication app = (HttpApplication)sender;
             string log = (DateTime.Now -
DateTime.Parse(app.Context.Items["start"].ToString())).ToString();
             Debugger.Log(0,"duration","request took " + log+Environment.NewLine);

         }
```

```
    }
```

حالا باید کلاس نوشته شده را به عنوان یک httpmodule به سیستم معرفی کنیم. به همین منظور وارد web.config شوید و کلاس جدید را معرفی کنید:

```
<httpModules>
     <add name="UrlPath" type="UrlPath"/>
   </httpModules>
```

اگر کلاس شما داخل یک namespace قرار دارد، در قسمت type حتما قبل از نام کلاس، آن را تعریف کنید namspace.ClassName
حالا دیگر کلاس UrlPath به عنوان یک httpmodule به سیستم معرفی شده است. تگ httpmodule را بین تگ <system.web> قرار داده ایم.
در ادامه پروژه را start بزنید تا نتیجه کار را ببینید:
اگر IIS شما، هم نسخه‌ی IIS من باشد، نباید تفاوتی مشاهده کنید و می‌بینید که درخواست‌ها هیچ تغییری نکردند؛ چرا که اصلا httpmodule اجرا نشده است. در واقع در نسخه‌های قدیمی IIS یعنی 6 به قبل، این تعریف درست است ولی از نسخه‌ی 7 به بعد IIS، روش دیگری برای تعریف را قبول دارد و باید تگ httpmodule، بین دو تگ <syste.webserver> قرار بگیرد و نام تگ httpmodule به module تغییر پیدا کند.
پس کد فوق به این صورت تغییر می‌کند:

```
<system.webServer>

    <modules>
      <add name="UrlRewrite" type="UrlRewrite"/>
    </modules>
</system.webServer>
```

حالا اگر قصدا دارید که پروژه‌ی شما در هر دو IIS مورد حمایت قرار گیرد، باید این ماژول را در هر دو جا معرفی کرده و در تگ system.webserver نیاز است تگ زیر تعریف شود که به طوری پیش فرض در webconfig می‌باشد:

```
<validation validateIntegratedModeConfiguration="false"/>
```

در غیر این صورت خطای زیر را دریافت می‌کنید:
HTTP Error 500.22 - Internal Server Error در کل استفاده از این ماژول به شما کمک می‌کند تمامی اطلاعات ارسالی به سیستم را قبل از رسیدن به قسمت پردازش بررسی نمایید و هر نوع تغییری را که میخواهید اعمال کنید و لازم نیست این تغییرات را روی هر بخش، جداگانه انجام دهید یا یک کلاس بنویسید که هر بار در یک جا صدا بزنید و خیلی از موارد دیگر

Global.asax و HttpModule

اگر با global.asax کار کرده باشید حتما می‌پرسید که الان چه تفاوتی با httpmodule دارد؟ در فایل global هم همین‌ها را دارید و دقیقا همین مزایا مهیاست؛ در واقع global.asax یک پیاده سازی از httpapplication هست.
کلاس‌های httpmodule نام دیگری هم دارند به اسم Portable global .asax به معنی یک فایل global.asax قابل حمل یا پرتابل. دلیل این نام گذاری این هست که شما موقعی که یک کد را در فایل global می‌نویسید، برای همیشه آن کد متعلق به همان پروژه هست و قابل انتقال به یک پروژه دیگر نیست ولی شما میتوانید httpmoduleها را در قالب یک پروژه به هر پروژه ای که دوست دارید رفرنس کنید و کد شما قابلیت استفاده مجدد و Reuse پیدا می‌کند و هم اینکه در صورت نیاز می‌توانید آن‌ها را در قالب یک dll منتشر کنید.

نظرات خوانندگان

نویسنده: عباس حجتی
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۰:۲۴

با سلام؛ ممنون بابت مطلب خوبی که ارسال کردید.
من می‌خوام از httpMuodule برای کنترل دسترسی کاربران (Authorization) استفاده کنم و به سشن کاربر نیاز دارم. مشکلی که هست موقع استفاده به این شکل HttpContext.Current.Session پیغام زیر رو میده:
Object not reference to instance of an object
آیا برای استفاده از سشن راه خاصی هست؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۴:۲۲

بله. در جدولی که تهیه کردند این مورد دقیقا ذکر شده:
« **AcquireRequestState** : این قسمت برای مدیریت state می‌باشد مثلا مدیریت **session** ها»
به این معنا که سشن در تمام رویدادگردان‌های آن مهیا نیست. فقط تعدادی از آن‌ها دسترسی به سشن دارند. برای مثال:

```
public class SimpleModule : IHttpModule
    {
        void IHttpModule.Init(HttpApplication application)
        {
            application.BeginRequest += new System.EventHandler(BeginRequest);
            application.AcquireRequestState += new EventHandler(application_AcquireRequestState);
        }

        public void BeginRequest(object sender, EventArgs e)
        {
            // no session here
        }

        void application_AcquireRequestState(object sender, EventArgs e)
        {
            HttpApplication app = sender as HttpApplication;
            app.Session.Add("Message", "hello module");
        }

        public void Dispose()
        {
        }
    }
```

نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۴:۵۰

از رویداد AcquireRequestState استفاده کنید ، همانطور که گفتیم این رویداد برای مدیریت stateهاست ، عموما در رویدادهای ابتدایی sessionها هنوز ایجاد نشدند
کد زیر نمونه ای از دسترسی به session هاست :

```
public void Init(HttpApplication app)
        {
            app.AcquireRequestState+=new EventHandler(_AUTH);
        }

    private void _AUTH(object sender, EventArgs e)
        {
            HttpApplication app = (HttpApplication) sender;
            HttpContext context = app.Context;
            HttpSessionState session = context.Session;

                if(session!=null)
            {
                string text = "session is not exist";
```

```
                if (session["userid"] != null)
                {
                    text = "session is exist";
                }
                    context.Response.Write(text);

            }

        }
```

توجه داشته باشید که خط if(session!=null) بسیار مهم هست و در صورت نبودن خط شرطی بعدی دچار خطایی که شما گرفتید می‌شود

نویسنده: عباس حجتی
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۷:۱۰

ممنون، درست شد.
یک مشکل دیگه اینکه درخواست‌های AJAX دیگه سمت سرور نمیره، می‌تونه به HttpModule ارتباطی داشته باشه؟

ممنون، این مشکل هم با فیلتر کردن درخواست‌های AJAX برطرف شد.

```
if (HttpContext.Current.Request.Headers["X-Requested-With"] != "XMLHttpRequest")
{
}
```

عنوان: **با HttpHandler بیشتر آشنا شوید**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۰۹/۲۷ ۱۵:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IhttpHandler, IhttpHandlerFactory, IHttpAsyncHandler

در مقاله قبل توضیح دادیم که وظیفه httphandler رندر و پردازش خروجی یک درخواست هست؛ حالا در این مقاله قصد داریم که مفهوم httphandler را بیشتر بررسی کنیم.

HttpHandler

برای تهیه‌ی یک httphandler، باید کلاسی را بر اساس اینترفیس IHttpHandler پیاده سازی کنیم و بعدا آن را در web.config برنامه معرفی کنیم. برای پیاده سازی این اینترفیس، به یک متد به اسم ProcessRequest با یک پارامتر از نوع HttpContext و یک پراپرتی به اسم IsReusable نیاز داریم که مقدار برگشتی این پراپرتی را false بگذارید؛ بعدا خواهم گفت چرا اینکار را می‌کنیم.
نحوه‌ی پیاده‌سازی یک httphandler به شکل زیر است:

```
public class MyHttpHandler : IHttpHandler
{
    public void ProcessRequest(HttpContext context)
    {
    }

    public bool IsReusable
    {
        get { return false; }
    }
}
```

با استفاده از شیء context می‌توان به دو شیء httpresponse و httprequest دسترسی داشت. تکه کد زیر مثالی است در مورد نحوه‌ی تغییر در محتوای سایت:

```
public class MyHttpHandler : IHttpHandler
{
    public void ProcessRequest(HttpContext context)
    {
        HttpResponse response = context.Response;
        HttpRequest request = context.Request;

        response.Write("Every Page has a some text like this");
    }

    public bool IsReusable
    {
        get { return false; }
    }
}
```

بگذارید همین کد ساده را در وب کانفیگ معرفی کنیم:

```
<system.web>
  <httpHandlers>
      <add verb="*" path="*.aspx" type="MyHttpHandler"/>
  </httpHandlers>
</system.web>
```

اگر نسخه IIS شما همانند نسخه‌ی من باشد، نباید هیچ تغییری مشاهده کنید؛ زیرا کد بالا فقط در مورد نسخه‌ی IIS6 صدق می‌کند و برای نسخه‌های IIS 7 به بعد باید به شیوه زیر عمل کنید:

```
<configuration>
  <system.web>
    <httpHandlers>

  <add name="myhttphandler" verb="*" path="*.aspx" type="MyHttpHandler"/>
```

```
    </httpHandlers>
  </system.web>
</configuration>
```

خروجی نهایی باید تنها این متن باشد: Every Page has a some text like this

گزینه Type که نام کلاس می‌باشد و اگر کلاس داخل یک فضای نام قرار گرفته باشد، باید اینطور نوشت : namespace.ClassName
گزینه verb شامل مقادیری چون Get,Post,Head,Putو Delete می‌باشد و httphandler را فقط برای این نوع درخواست‌ها اجرا می‌کند و در صورتیکه بخواهید چندتا از آن‌ها را استفاده کنید، با , از هم جدا می‌شوند. مثلا Get,post و درصورتیکه همه‌ی گزینه‌ها را بخواهید علامت * را میتوان استفاده کرد.
گزینه‌ی path این امکان را به شما می‌دهد که مسیر و نوع فایل‌هایی را که قصد دارید روی آن‌ها فقط اجرا شود، مشخص کنید و ما در قطعه کد بالا گفته‌ایم که تنها روی فایل‌هایی با پسوند aspx اجرا شود و چون مسیری هم ذکر نکردیم برای همه‌ی مسیرها قابل اجراست. یکی از مزیت‌های دادن پسوند این است که می‌توانید پسوندهای اختصاصی داشته باشید. مثلا پسوند RSS برای فیدهای وب سایتتان. بسیاری از برنامه نویسان به جای استفاده از صفحات aspx از ashx استفاده می‌کنند که به مراتب سبک‌تر از aspx هست و شامل بخش ui نمی‌شود و نتیجه خروجی آن بر اساس کدی که می‌نویسید مشخص می‌شود که میتواند صفحه متنی یا عکس یا xml یا ... باشد. در اینجا در مورد ساخت صفحات ashx توضیح داده شده است.

IHttpHandlerFactory

کار این اینترفیس پیاده سازی یک کلاس است که خروجی آن یک کلاس از نوع IHttpHandler هست. اگر دقت کنید در مثال‌های قبلی ما برای معرفی یک هندلر در وب کانفیگ یک سری path را به آن میدادیم و برای نمونه aspx.* را معرفی می‌کردیم؛ یعنی این هندلر را بر روی همه‌ی فایل‌های aspx اجرا کن و اگر دو یا چند هندلر در وب کانفیگ معرفی کنیم و برای همه مسیر aspx را قرار بدهیم، یعنی همه این هندلرها باید روی صفحات aspx اجرا گردند ولی در httphandlerfactory، ما چند هندلر داریم و میخواهیم فقط یکی از آن‌ها بر روی صفحات aspx انجام بگیرد، پس ما یک هندلرفکتوری را برای صفحات aspx معرفی می‌کنیم و در حین اجرا تصمیم می‌گیریم که کدام هندلر را ارسال کنیم.

اجازه بدهید نوشتن این نوع کلاس را آغاز کنیم،ابتدا دو هندلر به نام‌های httphandler1 و httphandler2 می‌نویسیم :

```
public class MyHttpHandler1 :IHttpHandler
{

    public void ProcessRequest(HttpContext context)
    {
        HttpResponse response = context.Response;
        response.Write("this is httphandler1");
    }

    public bool IsReusable
    {
        get { return false; }
    }
}

public class MyHttpHandler2 : IHttpHandler
{

    public void ProcessRequest(HttpContext context)
    {
        HttpResponse response = context.Response;
        response.Write("this is httphandler2");
    }

    public bool IsReusable
    {
        get { return false; }
    }
}
```

سپس کلاس MyFactory را بر اساس اینترفیس IHttpFactory پیاده سازی می‌کنیم و باید دو متد برای آن صدا بزنیم؛ یکی که هندلر انتخابی را بر میگرداند و دیگری هم برای رها کردن یا آزادسازی یک هندلر هست که در این مقاله کاری با آن نداریم. عموما

GC دات نت در این زمینه کارآیی خوبی دارد. در قسمت هندلرهای غیرهمزمان به طور مختصر خواهیم گفت که GC چطور آن‌ها را مدیریت می‌کند. کد زیر نمونه کلاسی است که توسط IHttpFactory پیاده سازی شده است:

```
public class MyFactory : IHttpHandlerFactory
{
    public IHttpHandler GetHandler(HttpContext context, string requestType, string url, string
pathTrasnlated)
    {

    }

    public void ReleaseHandler(IHttpHandler handler)
    {

    }
}
```

در متد GetHandler چهار آرگومان وجود دارند که به ترتیب برای موارد زیر به کار می‌روند:

Context	یک شی از کلاس httpcontext که دسترسی ما را برای اشیاء سروری چون response,request,session و... فراهم می‌کند.
RequestType	مشخص می‌کند که درخواست صفحه به چه صورتی است. این گزینه برای مواردی است که verb بیش از یک مورد را حمایت می‌کند. برای مثال دوست دارید یک هندلر را برای درخواست‌های Get ارسال کنید و هندلر دیگر را برای درخواست‌های نوع Post
URL	مسیر مجازی virtual Path صفحه صدا زده شده
PathTranslated	مسیر فیزیکی صفحه درخواست کننده را ارسال می‌کند.

متد GetHandler را بدین شکل می‌نویسیم و میخواهیم همه صفحات aspx هندلر شماره یک را انتخاب کنند و صفحات aspx که نامشان با t شروع می‌شوند، هندلر شماره دو را انتخاب کند:

```
public IHttpHandler GetHandler(HttpContext context, string requestType, string url, string
pathTrasnlated)
    {
        string handlername = "MyHttpHandler1";
        if(url.Substring(url.LastIndexOf("/")+1).StartsWith("t"))
        {
            handlername = "MyHttpHandler2";
        }

        try
        {
            return (IHttpHandler) Activator.CreateInstance(Type.GetType(handlername));
        }
        catch (Exception e)
        {
            throw new HttpException("Error: " + handlername, e);
        }
    }

    public void ReleaseHandler(IHttpHandler handler)
    {

    }
}
```

شی Activator که برای ساخت اشیاء با انتخاب بهترین constructor موجود بر اساس یک نوع Type مشخص به کار می‌رود و خروجی Object را می‌گرداند؛ با یک تبدیل ساده، خروجی به قالب اصلی خود باز میگردد. برای مطالعه بیشتر در مورد این کلاس به اینجا و اینجا مراجعه کنید.

نحوه‌ی تعریف factory در وب کانفیگ مانند قبل است و فقط باید در Type به جای نام هندلر نام فکتوری را نوشت. برنامه را اجرا

کنید تا نتیجه آن را ببینیم:

تصویر زیر نتیجه صدا زده شدن فایل default.aspx است:

تصویر زیر نتیجه صدا زده شدن فایل Tours_List.aspx است:

AsyncHttpHandlers

برای اینکه کار این اینترفیس را درک کنید بهتر هست اینجا را مطالعه کنید. در اینجا به خوبی تفاوت متدهای همزمان و غیرهمزمان توضیح داده شده است.

متن زیر خلاصه‌ترین و بهترین توضیح برای این پرسش است، چرا غیرهمزمان؟

در اعمالی که disk I/O و یا network I/O دارند، پردازش موازی و اعمال async به شدت مقیاس پذیری سیستم را بالا می‌برند. به این ترتیب worker thread جاری (که تعداد آن‌ها محدود است)، سریعتر آزاد شده و به worker pool بازگشت داده می‌شود تا بتواند به یک درخواست دیگر رسیده سرویس دهد. در این حالت می‌توان با منابع کمتری، درخواست‌های بیشتری را پردازش کرد.

موقعی که اینترفیس IHttpAsyncHandler را ارث بری کنید (این اینترفیس نیز از IHttpHandler ارث بری کرده است و دو متد اضافه‌تر دارد)، باید دو متد دیگر را نیز پیاده سازی کنید:

```
public IAsyncResult BeginProcessRequest(HttpContext context, AsyncCallback callback, object obj)
    {

    }

    public void EndProcessRequest(IAsyncResult result)
    {

    }
```

پراپرتی ISResuable هم موقعی که true برگشت بدهد، باعث می‌شود pooling فعال شده و این هندلر در حافظه باقی بماند و تمامی درخواست‌ها از طریق همین یک نمونه اجرا شوند.

به زبان ساده‌تر، این پراپرتی می‌گوید اگر چندین درخواست از طرف کلاینت‌ها برسد، توسط یک نمونه یا instance از هندلر پردازش خواهند شد؛ چون به طور پیش فرض موقعی که تمام درخواست‌های از pipeline بگذرند، هندلرها توسط httpapplication در یک لیست بازیافت قرار گرفته و همه‌ی آن‌ها با null مقداردهی می‌شوند تا از حافظه پاک شوند ولی اگر این پراپرتی true برگرداند، هندلر مربوطه نال **نشده** و برای پاسخگویی به درخواست‌های بعدی در حافظه خواهد ماند.

مهمترین مزیت این گزینه، این می‌باشد که کاآیی سیستم را بالا می‌برد و اشیا کمتری به GC پاس می‌شوند. ولی یک عیب هم دارد

که این تردهایی که ایجاد می‌کند، امنیت کمتری دارند و باید توسط برنامه نویس این امنیت بالاتر رود. این پراپرتی را در مواقعی که با هندلرهای همزمان کار می‌کنید برابر با false بگذارید چون این گزینه بیشتر بر روی هندلرهای غیرهمزمان اثر دارد و هم اینکه بعضی‌ها توصیه می‌کنند که false بگذارید چون GC مدیریت خوبی در مورد هندلرها دارد و هم این که ارزش یافتن باگ در کد را ندارد.

بر میگردیم سراغ کد نویسی هندلر غیر همزمان. در آخرین قطعه کد نوشته شده، ما دو متد دیگر را پیاده سازی کردیم که یکی از آن‌ها BeginProcessRequest است و خروجی آن کلاسی است که از اینترفیس IAsyncResult ارث بری کرده است. پس یک کلاس با ارث بری از این اینترفیس می‌نویسیم و در این کلاس نیاز است که 4 پراپرتی را پیاده سازی کنیم که این کلاس به شکل زیر در خواهد آمد:

```
public class AsynchOperation : IAsyncResult
{
    private bool _completed;
    private Object _state;
    private AsyncCallback _callback;
    private HttpContext _context;

    bool IAsyncResult.IsCompleted { get { return _completed; } }
    WaitHandle IAsyncResult.AsyncWaitHandle { get { return null; } }
    Object IAsyncResult.AsyncState { get { return _state; } }
    bool IAsyncResult.CompletedSynchronously { get { return false; } }
}
```

متدهای private اجباری نیستند؛ ولی برای ذخیره مقادیر get و set نیاز است. همانطور که از اسامی آن‌ها پیداست مشخص است که برای چه کاری ساخته شده اند.

خب اجازه بدهید یک تابع سازنده به آن برای مقداردهی اولیه این متغیرهای خصوصی داشته باشیم:

```
public AsynchOperation(AsyncCallback callback, HttpContext context, Object state)
    {
        _callback = callback;
        _context = context;
        _state = state;
        _completed = false;
    }
```

همانطور که می‌بینید موارد موجود در متد BeginProcessRequest را تحویل می‌گیریم تا اطلاعات درخواستی مربوطه را داشته باشیم و مقدار _Completed را هم برابر با false قرار می‌دهیم. سپس نوبت این می‌رسد که ما درخواست را در صف pool قرار دهیم. برای همین تکه کد زیر را اضافه می‌کنیم:

```
public void StartAsyncWork()
    {
        ThreadPool.QueueUserWorkItem(new WaitCallback(StartAsyncTask),null);
    }
```

با اضافه شدن درخواست به صف، هر موقع درخواست‌های قبلی تمام شوند و callback خودشان را ارسال کنند، نوبت درخواست‌های جدیدتر هم میرسد. StartAsyncTask هم متدی است که وظیفه‌ی اصلی پردازش درخواست را به دوش دارد و موقعی که نوبت درخواست برسد، کدهای این متد اجرا می‌گردد که ما در اینجا مانند مثال اول روی صفحه چیزی نوشتیم:

```
private void StartAsyncTask(Object workItemState)
    {

        _context.Response.Write("<p>Completion IsThreadPoolThread is " +
Thread.CurrentThread.IsThreadPoolThread + "</p>\r\n");

        _context.Response.Write("Hello World from Async Handler!");
        _completed = true;
        _callback(this);
    }
```

دو خط اول اطلاعات را چاپ کرده و در خط سوم متغیر _completed را true کرده و در آخر این درخواست را فراخوانی مجدد

می‌کنیم تا بگوییم که کار این درخواست پایان یافته‌است؛ پس این درخواست را از صف بیرون بکش و درخواست بعدی را اجرا کن.
نهایتا کل این کلاس را در متد BeginProcessRequest صدا بزنید:

```
context.Response.Write("<p>Begin IsThreadPoolThread is " + Thread.CurrentThread.IsThreadPoolThread +
"</p>\r\n");
        AsynchOperation asynch = new AsynchOperation(callback, context, obj);
        asynch.StartAsyncWork();
        return asynch;
```

کل کد مربوطه : *(توجه:کدها از داخل سایت msdn برداشته شده است و اکثر کدهای موجود در نت هم به همین قالب می‌نویسند)*

```
public class MyHttpHandler : IHttpAsyncHandler
{
    public IAsyncResult BeginProcessRequest(HttpContext context, AsyncCallback callback, object obj)
    {
        context.Response.Write("<p>Begin IsThreadPoolThread is " +
Thread.CurrentThread.IsThreadPoolThread + "</p>\r\n");
        AsynchOperation asynch = new AsynchOperation(callback, context, obj);
        asynch.StartAsyncWork();
        return asynch;
    }

    public void EndProcessRequest(IAsyncResult result)
    {

    }
    public void ProcessRequest(HttpContext context)
    {
       throw new InvalidOperationException();

    }

    public bool IsReusable
    {
        get { return false; }
    }
}

public class AsynchOperation : IAsyncResult
{
    private bool _completed;
    private Object _state;
    private AsyncCallback _callback;
    private HttpContext _context;

    bool IAsyncResult.IsCompleted { get { return _completed; } }
    WaitHandle IAsyncResult.AsyncWaitHandle { get { return null; } }
    Object IAsyncResult.AsyncState { get { return _state; } }
    bool IAsyncResult.CompletedSynchronously { get { return false; } }

    public AsynchOperation(AsyncCallback callback, HttpContext context, Object state)
    {
        _callback = callback;
        _context = context;
        _state = state;
        _completed = false;
    }


    public void StartAsyncWork()
    {

        ThreadPool.QueueUserWorkItem(new WaitCallback(StartAsyncTask),null);

    }
    private void StartAsyncTask(Object workItemState)
    {

        _context.Response.Write("<p>Completion IsThreadPoolThread is " +
Thread.CurrentThread.IsThreadPoolThread + "</p>\r\n");

        _context.Response.Write("Hello World from Async Handler!");
        _completed = true;
        _callback(this);
    }
```

آشنایی با فایل ASHX

در مطالب بالاتر به فایل‌های Ashx اشاره کردیم. این فایل به نام Generic Web Handler شناخته می‌شوند و می‌توانید با Add New Item این نوع فایل‌ها را اضافه کنید. این فایل شامل هیچ UI ایی نمی‌باشد و فقط شامل بخش کد می‌باشد. برای همین نسبت به aspx سبک‌تر بوده و شامل یک directive به اسم **@WebHandler** است.

مایکروسافت در MSDN نوشته است که httphandlerها در واقع فرآیندهایی هستند (به این فرایندها بیشتر End Point می‌گویند) که در پاسخ به درخواست‌های رسیده شده توسط asp.net application اجرا می‌شوند و بیشترین درخواست هایی هم که می‌رسد از نوع صفحات Aspx می‌باشد و موقعی که کاربری درخواست صفحه‌ی aspx می‌کند هندلرهای مربوط به page اجرا می‌شوند.

در متن بالا به خوبی روشن هست که ashx به دلیل نداشتن UI، تعداد کمتری از handlerها را در مسیر Pipeline قرار می‌دهند و اجرای آن‌ها سریعتر است. غیر از این دو هندلر aspx و ashx، هندلر توکار دیگری چون asmx که مختص وب سرویس هست و axd مربوط به اعمال trace نیز وجود دارند.

در این لینک که در بالاتر هم درج شده بود یک نمونه هندلر برای نمایش تصویر نوشته است. اگر تصاویرتان را بدین صورت اجرا کنید می‌توان جلوی درخواست‌های رسیده از وب سایت‌های دیگر را سد کرد. برای مثال یک نفر مطالب شما را کپی می‌کند و در داخل وبلاگ یا وب سایتش می‌گذارد و شما در اینجا درخواست‌های رسیده خارج از وب سایت خود را لغو خواهید کرد و تصاویر کپی شده نمایش داده نخواهند شد.

عنوان: **اعمال تزریق وابستگی‌ها به مثال رسمی ASP.NET Identity**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۰۹/۲۹ ۱۳:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Entity framework, MVC, Dependency Injection, ASP.NET Identity

پروژه‌ی ASP.NET Identity که نسل جدید سیستم Authentication و Authorization مخصوص ASP.NET است، دارای دو سری مثال رسمی است:

الف) مثال‌های کدپلکس

ب) مثال نیوگت

در ادامه قصد داریم مثال نیوگت آن‌را که مثال کاملی است از نحوه‌ی استفاده از ASP.NET Identity در ASP.NET MVC، جهت اعمال الگوی واحد کار و تزریق وابستگی‌ها، بازنویسی کنیم.

پیشنیازها

- برای درک مطلب جاری نیاز است ابتدا دوره‌ی مرتبطی را در سایت مطالعه کنید و همچنین با نحوه‌ی پیاده سازی الگوی واحد کار در EF Code First آشنا باشید.
- به علاوه فرض بر این است که یک پروژه‌ی خالی ASP.NET MVC 5 را نیز آغاز کرده‌اید و توسط کنسول پاور شل نیوگت، فایل‌های مثال Microsoft.AspNet.Identity.Samples را به آن افزوده‌اید:

```
PM> Install-Package Microsoft.AspNet.Identity.Samples -Pre
```

ساختار پروژه‌ی تکمیلی

همانند مطلب پیاده سازی الگوی واحد کار در EF Code First ، این پروژه‌ی جدید را با چهار اسمبلی class library دیگر به نام‌های

AspNetIdentityDependencyInjectionSample.DataLayer
AspNetIdentityDependencyInjectionSample.DomainClasses
AspNetIdentityDependencyInjectionSample.IocConfig
AspNetIdentityDependencyInjectionSample.ServiceLayer

تکمیل می‌کنیم.

ساختار پروژه‌ی AspNetIdentityDependencyInjectionSample.DomainClasses

مثال Microsoft.AspNet.Identity.Samples بر مبنای primary key از نوع string است. برای نمونه کلاس کاربران آن‌را به نام ApplicationUser در فایل Models\IdentityModels.cs می‌توانید مشاهده کنید. در مطلب جاری، این نوع پیش فرض، به نوع متداول int تغییر خواهد یافت. به همین جهت نیاز است کلاس‌های ذیل را به پروژه‌ی DomainClasses اضافه کرد:

```
using System.ComponentModel.DataAnnotations.Schema;
using Microsoft.AspNet.Identity.EntityFramework;

namespace AspNetIdentityDependencyInjectionSample.DomainClasses
{
  public class ApplicationUser : IdentityUser<int, CustomUserLogin, CustomUserRole, CustomUserClaim>
  {
   // سایر خواص اضافی در اینجا

   [ForeignKey("AddressId")]
   public virtual Address Address { get; set; }
   public int? AddressId { get; set; }
  }
}

using System.Collections.Generic;
```

```
namespace AspNetIdentityDependencyInjectionSample.DomainClasses
{
  public class Address
  {
   public int Id { get; set; }
   public string City { get; set; }
   public string State { get; set; }

   public virtual ICollection<ApplicationUser> ApplicationUsers { set; get; }
  }
}

using Microsoft.AspNet.Identity.EntityFramework;

namespace AspNetIdentityDependencyInjectionSample.DomainClasses
{
  public class CustomRole : IdentityRole<int, CustomUserRole>
  {
   public CustomRole() { }
   public CustomRole(string name) { Name = name; }


  }
}

using Microsoft.AspNet.Identity.EntityFramework;

namespace AspNetIdentityDependencyInjectionSample.DomainClasses
{
  public class CustomUserClaim : IdentityUserClaim<int>
  {

  }
}

using Microsoft.AspNet.Identity.EntityFramework;

namespace AspNetIdentityDependencyInjectionSample.DomainClasses
{
  public class CustomUserLogin : IdentityUserLogin<int>
  {

  }
}

using Microsoft.AspNet.Identity.EntityFramework;

namespace AspNetIdentityDependencyInjectionSample.DomainClasses
{
  public class CustomUserRole : IdentityUserRole<int>
  {

  }
}
```

در اینجا نحوه‌ی تغییر primary key از نوع string را به نوع int، مشاهده می‌کنید. این تغییر نیاز به اعمال به کلاس‌های کاربران و همچنین نقش‌های آن‌ها نیز دارد. به همین جهت صرفا تغییر کلاس ابتدایی ApplicationUser کافی نیست و باید کلاس‌های فوق را نیز اضافه کرد و تغییر داد.

بدیهی است در اینجا کلاس پایه کاربران را می‌توان سفارشی سازی کرد و خواص دیگری را نیز به آن افزود. برای مثال در اینجا یک کلاس جدید آدرس تعریف شده‌است که ارجاعی از آن در کلاس کاربران نیز قابل مشاهده است.

سایر کلاس‌های مدل‌های اصلی برنامه که جداول بانک اطلاعاتی را تشکیل خواهند داد نیز در آینده به همین اسمبلی DomainClasses اضافه می‌شوند.

ساختار پروژه‌ی AspNetIdentityDependencyInjectionSample.DataLayer جهت اعمال الگوی واحد کار

اگر به همان فایل Models\IdentityModels.cs ابتدایی پروژه که اکنون کلاس ApplicationUser آن‌را به پروژه‌ی DomainClasses منتقل کرده‌ایم، مجددا مراجعه کنید، کلاس DbContext مخصوص ASP.NET Identity نیز در آن تعریف شده‌است:

```
public class ApplicationDbContext : IdentityDbContext<ApplicationUser>
```

این کلاس را به پروژه‌ی DataLayer منتقل می‌کنیم و از آن به عنوان DbContext اصلی برنامه استفاده خواهیم کرد. بنابراین دیگر نیازی نیست چندین DbContext در برنامه داشته باشیم. IdentityDbContext، در اصل از DbContext مشتق شده‌است.
اینترفیس IUnitOfWork برنامه، در پروژه‌ی DataLayer چنین شکلی را دارد که نمونه‌ای از آن‌را در مطلب آشنایی با نحوه‌ی پیاده سازی الگوی واحد کار در EF Code First ، پیشتر ملاحظه کرده‌اید.

```
using System.Collections.Generic;
using System.Data.Entity;

namespace AspNetIdentityDependencyInjectionSample.DataLayer.Context
{
  public interface IUnitOfWork
  {
   IDbSet<TEntity> Set<TEntity>() where TEntity : class;
   int SaveAllChanges();
   void MarkAsChanged<TEntity>(TEntity entity) where TEntity : class;
   IList<T> GetRows<T>(string sql, params object[] parameters) where T : class;
   IEnumerable<TEntity> AddThisRange<TEntity>(IEnumerable<TEntity> entities) where TEntity : class;
   void ForceDatabaseInitialize();
  }
}
```

اکنون کلاس ApplicationDbContext منتقل شده به DataLayer یک چنین امضایی را خواهد یافت:

```
public class ApplicationDbContext :
  IdentityDbContext<ApplicationUser, CustomRole, int, CustomUserLogin, CustomUserRole,
CustomUserClaim>,
  IUnitOfWork
{
  public DbSet<Category> Categories { set; get; }
  public DbSet<Product> Products { set; get; }
  public DbSet<Address> Addresses { set; get; }
```

تعریف آن باید جهت اعمال کلاس‌های سفارشی سازی شده‌ی کاربران و نقش‌های آن‌ها برای استفاده از primary key از نوع int به شکل فوق، تغییر یابد. همچنین در انتهای آن مانند قبل، IUnitOfWork نیز ذکر شده‌است. پیاده سازی کامل این کلاس را از پروژه‌ی پیوست انتهای بحث می‌توانید دریافت کنید.
کار کردن با این کلاس، هیچ تفاوتی با DbContextهای متداول EF Code First ندارد و تمام اصول آن‌ها یکی است.

در ادامه اگر به فایل App_Start\IdentityConfig.cs مراجعه کنید، کلاس ذیل در آن قابل مشاهده‌است:

```
public class ApplicationDbInitializer : DropCreateDatabaseIfModelChanges<ApplicationDbContext>
```

نیازی به این کلاس به این شکل نیست. آن‌را حذف کنید و در پروژه‌ی DataLayer، کلاس جدید ذیل را اضافه نمائید:

```
using System.Data.Entity.Migrations;

namespace AspNetIdentityDependencyInjectionSample.DataLayer.Context
{
  public class Configuration : DbMigrationsConfiguration<ApplicationDbContext>
  {
   public Configuration()
   {
    AutomaticMigrationsEnabled = true;
    AutomaticMigrationDataLossAllowed = true;
   }
  }
}
```

در این مثال، بحث migrations به حالت خودکار تنظیم شده‌است و تمام تغییرات در پروژه‌ی DomainClasses را به صورت خودکار به بانک اطلاعاتی اعمال می‌کند. تا همینجا کار تنظیم DataLayer به پایان می‌رسد.

ساختار پروژه‌ی AspNetIdentityDependencyInjectionSample.ServiceLayer

در ادامه مابقی کلاس‌های موجود در فایل App_Start\IdentityConfig.cs را به لایه سرویس برنامه منتقل خواهیم کرد. همچنین برای آن‌ها یک سری اینترفیس جدید نیز تعریف می‌کنیم، تا تزریق وابستگی‌ها به نحو صحیحی صورت گیرد. اگر به فایل‌های کنترلر این مثال پیش فرض مراجعه کنید (پیش از تغییرات بحث جاری)، هرچند به نظر در کنترلرها، کلاس‌های موجود در فایل App_Start\IdentityConfig.cs تزریق شده‌اند، اما به دلیل عدم استفاده از اینترفیس‌ها، وابستگی کاملی بین جزئیات پیاده سازی این کلاس‌ها و نمونه‌های تزریق شده به کنترلرها وجود دارد و عملا معکوس سازی واقعی وابستگی‌ها رخ نداده‌است. بنابراین نیاز است این مسایل را اصلاح کنیم.

الف) انتقال کلاس ApplicationUserManager به لایه سرویس برنامه

کلاس ApplicationUserManager فایل App_Start\IdentityConfig.c را به لایه سرویس منتقل می‌کنیم:

```
using System;
using System.Security.Claims;
using System.Threading.Tasks;
using AspNetIdentityDependencyInjectionSample.DomainClasses;
using AspNetIdentityDependencyInjectionSample.ServiceLayer.Contracts;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.Owin;
using Microsoft.Owin.Security.Cookies;
using Microsoft.Owin.Security.DataProtection;

namespace AspNetIdentityDependencyInjectionSample.ServiceLayer
{
  public class ApplicationUserManager
   : UserManager<ApplicationUser, int>, IApplicationUserManager
  {
   private readonly IDataProtectionProvider _dataProtectionProvider;
   private readonly IIdentityMessageService _emailService;
   private readonly IApplicationRoleManager _roleManager;
   private readonly IIdentityMessageService _smsService;
   private readonly IUserStore<ApplicationUser, int> _store;

   public ApplicationUserManager(IUserStore<ApplicationUser, int> store,
    IApplicationRoleManager roleManager,
    IDataProtectionProvider dataProtectionProvider,
    IIdentityMessageService smsService,
    IIdentityMessageService emailService)
    : base(store)
   {
    _store = store;
    _roleManager = roleManager;
    _dataProtectionProvider = dataProtectionProvider;
    _smsService = smsService;
    _emailService = emailService;

    createApplicationUserManager();
   }


   public void SeedDatabase()
   {
   }

   private void createApplicationUserManager()
   {
    // Configure validation logic for usernames
    this.UserValidator = new UserValidator<ApplicationUser, int>(this)
    {
      AllowOnlyAlphanumericUserNames = false,
      RequireUniqueEmail = true
    };

    // Configure validation logic for passwords
    this.PasswordValidator = new PasswordValidator
    {
      RequiredLength = 6,
      RequireNonLetterOrDigit = true,
      RequireDigit = true,
      RequireLowercase = true,
      RequireUppercase = true,
    };
```

```
    // Configure user lockout defaults
    this.UserLockoutEnabledByDefault = true;
    this.DefaultAccountLockoutTimeSpan = TimeSpan.FromMinutes(5);
    this.MaxFailedAccessAttemptsBeforeLockout = 5;

    // Register two factor authentication providers. This application uses Phone and Emails as a step
of receiving a code for verifying the user
    // You can write your own provider and plug in here.
    this.RegisterTwoFactorProvider("PhoneCode", new PhoneNumberTokenProvider<ApplicationUser, int>
    {
      MessageFormat = "Your security code is: {0}"
    });
    this.RegisterTwoFactorProvider("EmailCode", new EmailTokenProvider<ApplicationUser, int>
    {
      Subject = "SecurityCode",
      BodyFormat = "Your security code is {0}"
    });
    this.EmailService = _emailService;
    this.SmsService = _smsService;

    if (_dataProtectionProvider != null)
    {
      var dataProtector = _dataProtectionProvider.Create("ASP.NET Identity");
      this.UserTokenProvider = new DataProtectorTokenProvider<ApplicationUser, int>(dataProtector);
    }
   }
  }
}
```

تغییراتی که در اینجا اعمال شده‌اند، به شرح زیر می‌باشند:

- متد استاتیک Create این کلاس حذف و تعاریف آن به سازنده‌ی کلاس منتقل شده‌اند. به این ترتیب با هربار وهله سازی این کلاس توسط IoC Container به صورت خودکار این تنظیمات نیز به کلاس پایه UserManager اعمال می‌شوند.

- اگر به کلاس پایه UserManager دقت کنید، به آرگومان‌های جنریک آن یک int هم اضافه شده‌است. این مورد جهت استفاده از primary key از نوع int ضروری است.

- در کلاس پایه UserManager تعدادی متد وجود دارند. تعاریف آن‌ها را به اینترفیس IApplicationUserManager منتقل خواهیم کرد. نیازی هم به پیاده سازی این متدها در کلاس جدید ApplicationUserManager نیست؛ زیرا کلاس پایه UserManager پیشتر آن‌ها را پیاده سازی کرده‌است. به این ترتیب می‌توان به یک تزریق وابستگی واقعی و بدون وابستگی به پیاده سازی خاص UserManager رسید. کنترلری که با IApplicationUserManager بجای ApplicationUserManager کار می‌کند، قابلیت تعویض پیاده سازی آن‌را جهت آزمون‌های واحد خواهد یافت.

- در کلاس اصلی ApplicationDbInitializer پیش فرض این مثال، متد Seed هم قابل مشاهده‌است. این متد را از کلاس جدید Configuration اضافه شده به DataLayer حذف کرده‌ایم. از این جهت که در آن از متدهای کلاس ApplicationUserManager مستقیما استفاده شده‌است. متد Seed اکنون به کلاس جدید اضافه شده به لایه سرویس منتقل شده و در آغاز برنامه فراخوانی خواهد شد. DataLayer نباید وابستگی به لایه سرویس داشته باشد. لایه سرویس است که از امکانات DataLayer استفاده می‌کند.

- اگر به سازنده‌ی کلاس جدید ApplicationUserManager دقت کنید، چند اینترفیس دیگر نیز به آن تزریق شده‌اند. اینترفیس IApplicationRoleManager را ادامه تعریف خواهیم کرد. سایر اینترفیس‌های تزریق شده مانند IUserStore، IDataProtectionProvider و IIdentityMessageService جزو تعاریف اصلی ASP.NET Identity بوده و نیازی به تعریف مجدد آن‌ها نیست. فقط کلاس‌های EmailService و SmsService فایل App_Start\IdentityConfig.c را نیز به لایه سرویس منتقل کرده‌ایم. این کلاس‌ها بر اساس تنظیمات IoC Container مورد استفاده، در اینجا به صورت خودکار ترزیق خواهند شد. حالت پیش فرض آن، وهله سازی مستقیم است که مطابق کدهای فوق به حالت تزریق وابستگی‌ها بهبود یافته‌است.

ب) انتقال کلاس ApplicationSignInManager به لایه سرویس برنامه

کلاس ApplicationSignInManager فایل App_Start\IdentityConfig.c را نیز به لایه سرویس منتقل می‌کنیم.

```
using AspNetIdentityDependencyInjectionSample.DomainClasses;
using AspNetIdentityDependencyInjectionSample.ServiceLayer.Contracts;
using Microsoft.AspNet.Identity.Owin;
using Microsoft.Owin.Security;

namespace AspNetIdentityDependencyInjectionSample.ServiceLayer
{
  public class ApplicationSignInManager :
```

```
   SignInManager<ApplicationUser, int>, IApplicationSignInManager
  {
   private readonly ApplicationUserManager _userManager;
   private readonly IAuthenticationManager _authenticationManager;

   public ApplicationSignInManager(ApplicationUserManager userManager,
              IAuthenticationManager authenticationManager) :
    base(userManager, authenticationManager)
   {
    _userManager = userManager;
    _authenticationManager = authenticationManager;
   }
  }
}
```

در اینجا نیز اینترفیس جدید IApplicationSignInManager را برای مخفی سازی پیاده سازی کلاس پایه توکار SignInManager، اضافه کرده‌ایم. این اینترفیس دقیقا حاوی تعاریف متدهای کلاس پایه SignInManager است و نیازی به پیاده سازی مجدد در کلاس ApplicationSignInManager نخواهد داشت.

ج) انتقال کلاس ApplicationRoleManager به لایه سرویس برنامه

کلاس ApplicationRoleManager فایل App_Start\IdentityConfig.c را نیز به لایه سرویس منتقل خواهیم کرد:

```
using AspNetIdentityDependencyInjectionSample.DomainClasses;
using AspNetIdentityDependencyInjectionSample.ServiceLayer.Contracts;
using Microsoft.AspNet.Identity;

namespace AspNetIdentityDependencyInjectionSample.ServiceLayer
{
  public class ApplicationRoleManager : RoleManager<CustomRole, int>, IApplicationRoleManager
  {
   private readonly IRoleStore<CustomRole, int> _roleStore;
   public ApplicationRoleManager(IRoleStore<CustomRole, int> roleStore)
    : base(roleStore)
   {
    _roleStore = roleStore;
   }


   public CustomRole FindRoleByName(string roleName)
   {
    return this.FindByName(roleName); // RoleManagerExtensions
   }

   public IdentityResult CreateRole(CustomRole role)
   {
    return this.Create(role); // RoleManagerExtensions
   }
  }
}
```

روش کار نیز در اینجا همانند دو کلاس قبل است. اینترفیس جدید IApplicationRoleManager را که حاوی تعاریف متدهای کلاس پایه توکار RoleManager است، به لایه سرویس اضافه می‌کنیم. کنترلرهای برنامه با این اینترفیس بجای استفاده مستقیم از کلاس ApplicationRoleManager کار خواهند کرد.

تا اینجا کار تنظیمات لایه سرویس برنامه به پایان می‌رسد.

ساختار پروژه‌ی AspNetIdentityDependencyInjectionSample.IocConfig

پروژه‌ی IocConfig جایی است که تنظیمات StructureMap را به آن منتقل کرده‌ایم:

```
using System;
using System.Data.Entity;
using System.Threading;
using System.Web;
using AspNetIdentityDependencyInjectionSample.DataLayer.Context;
```

```
using AspNetIdentityDependencyInjectionSample.DomainClasses;
using AspNetIdentityDependencyInjectionSample.ServiceLayer;
using AspNetIdentityDependencyInjectionSample.ServiceLayer.Contracts;
using Microsoft.AspNet.Identity;
using Microsoft.AspNet.Identity.EntityFramework;
using Microsoft.Owin.Security;
using StructureMap;
using StructureMap.Web;

namespace AspNetIdentityDependencyInjectionSample.IocConfig
{
    public static class SmObjectFactory
    {
        private static readonly Lazy<Container> _containerBuilder =
            new Lazy<Container>(defaultContainer, LazyThreadSafetyMode.ExecutionAndPublication);

        public static IContainer Container
        {
            get { return _containerBuilder.Value; }
        }

        private static Container defaultContainer()
        {
            return new Container(ioc =>
            {
                ioc.For<IUnitOfWork>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use<ApplicationDbContext>();


ioc.For<ApplicationDbContext>().HybridHttpOrThreadLocalScoped().Use<ApplicationDbContext>();
                ioc.For<DbContext>().HybridHttpOrThreadLocalScoped().Use<ApplicationDbContext>();

                ioc.For<IUserStore<ApplicationUser, int>>()
                    .HybridHttpOrThreadLocalScoped()
                    .Use<UserStore<ApplicationUser, CustomRole, int, CustomUserLogin, CustomUserRole,
CustomUserClaim>>();

                ioc.For<IRoleStore<CustomRole, int>>()
                    .HybridHttpOrThreadLocalScoped()
                    .Use<RoleStore<CustomRole, int, CustomUserRole>>();

                ioc.For<IAuthenticationManager>()
                      .Use(() => HttpContext.Current.GetOwinContext().Authentication);

                ioc.For<IApplicationSignInManager>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use<ApplicationSignInManager>();

                ioc.For<IApplicationUserManager>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use<ApplicationUserManager>();

                ioc.For<IApplicationRoleManager>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use<ApplicationRoleManager>();

                ioc.For<IIdentityMessageService>().Use<SmsService>();
                ioc.For<IIdentityMessageService>().Use<EmailService>();
                ioc.For<ICustomRoleStore>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use<CustomRoleStore>();

                ioc.For<ICustomUserStore>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use<CustomUserStore>();

                //config.For<IDataProtectionProvider>().Use(()=> app.GetDataProtectionProvider()); //
In Startup class

                ioc.For<ICategoryService>().Use<EfCategoryService>();
                ioc.For<IProductService>().Use<EfProductService>();
            });
        }
    }
}
```

در اینجا نحوه‌ی اتصال اینترفیس‌های برنامه را به کلاس‌ها و یا نمونه‌هایی که آن‌ها را می‌توانند پیاده سازی کنند، مشاهده می‌کنید.

برای مثال IUnitOfWork به ApplicationDbContext مرتبط شده‌است و یا دوبار تعاریف متناظر با DbContext را مشاهده می‌کنید. از این تعاریف به صورت توکار توسط ASP.NET Identity زمانیکه قرار است UserStore و RoleStore را وهله سازی کند، استفاده می‌شوند و ذکر آن‌ها الزامی است.

در تعاریف فوق یک مورد را به فایل Startup.cs موکول کرده‌ایم. برای مشخص سازی نمونه‌ی پیاده سازی کننده‌ی IDataProtectionProvider نیاز است به IAppBuilder کلاس Startup برنامه دسترسی داشت. این کلاس آغازین Owin اکنون به نحو ذیل بازنویسی شده‌است و در آن، تنظیمات IDataProtectionProvider را به همراه وهله سازی CreatePerOwinContext مشاهده می‌کنید:

```
using System;
using AspNetIdentityDependencyInjectionSample.IocConfig;
using AspNetIdentityDependencyInjectionSample.ServiceLayer.Contracts;
using Microsoft.AspNet.Identity;
using Microsoft.Owin;
using Microsoft.Owin.Security.Cookies;
using Microsoft.Owin.Security.DataProtection;
using Owin;
using StructureMap.Web;

namespace AspNetIdentityDependencyInjectionSample
{
    public class Startup
    {
        public void Configuration(IAppBuilder app)
        {
            configureAuth(app);
        }

        private static void configureAuth(IAppBuilder app)
        {
            SmObjectFactory.Container.Configure(config =>
            {
                config.For<IDataProtectionProvider>()
                      .HybridHttpOrThreadLocalScoped()
                      .Use(()=> app.GetDataProtectionProvider());
            });
            SmObjectFactory.Container.GetInstance<IApplicationUserManager>().SeedDatabase();

            // Configure the db context, user manager and role manager to use a single instance per
request
            app.CreatePerOwinContext(() =>
SmObjectFactory.Container.GetInstance<IApplicationUserManager>());

            // Enable the application to use a cookie to store information for the signed in user
            // and to use a cookie to temporarily store information about a user logging in with a
third party login provider
            // Configure the sign in cookie
            app.UseCookieAuthentication(new CookieAuthenticationOptions
            {
                AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
                LoginPath = new PathString("/Account/Login"),
                Provider = new CookieAuthenticationProvider
                {
                    // Enables the application to validate the security stamp when the user logs in.
                    // This is a security feature which is used when you change a password or add an
external login to your account.
                    OnValidateIdentity =
SmObjectFactory.Container.GetInstance<IApplicationUserManager>().OnValidateIdentity()
                }
            });
            app.UseExternalSignInCookie(DefaultAuthenticationTypes.ExternalCookie);

            // Enables the application to temporarily store user information when they are verifying
the second factor in the two-factor authentication process.
            app.UseTwoFactorSignInCookie(DefaultAuthenticationTypes.TwoFactorCookie,
TimeSpan.FromMinutes(5));

            // Enables the application to remember the second login verification factor such as phone
or email.
            // Once you check this option, your second step of verification during the login process
will be remembered on the device where you logged in from.
            // This is similar to the RememberMe option when you log in.

app.UseTwoFactorRememberBrowserCookie(DefaultAuthenticationTypes.TwoFactorRememberBrowserCookie);
        }

    }
```

```
}
```

این تعاریف از فایل پیش فرض Startup.Auth.cs پوشه‌ی App_Start دریافت و جهت کار با IoC Container برنامه، بازنویسی شده‌اند.

تنظیمات برنامه‌ی اصلی ASP.NET MVC، جهت اعمال تزریق وابستگی‌ها

الف) ابتدا نیاز است فایل Global.asax.cs را به نحو ذیل بازنویسی کنیم:

```
using System;
using System.Data.Entity;
using System.Web;
using System.Web.Mvc;
using System.Web.Optimization;
using System.Web.Routing;
using AspNetIdentityDependencyInjectionSample.DataLayer.Context;
using AspNetIdentityDependencyInjectionSample.IocConfig;
using StructureMap.Web.Pipeline;

namespace AspNetIdentityDependencyInjectionSample
{
    public class MvcApplication : HttpApplication
    {
        protected void Application_Start()
        {
            AreaRegistration.RegisterAllAreas();
            FilterConfig.RegisterGlobalFilters(GlobalFilters.Filters);
            RouteConfig.RegisterRoutes(RouteTable.Routes);
            BundleConfig.RegisterBundles(BundleTable.Bundles);


            setDbInitializer();
            //Set current Controller factory as StructureMapControllerFactory
            ControllerBuilder.Current.SetControllerFactory(new StructureMapControllerFactory());
        }

        protected void Application_EndRequest(object sender, EventArgs e)
        {
            HttpContextLifecycle.DisposeAndClearAll();
        }

        public class StructureMapControllerFactory : DefaultControllerFactory
        {
            protected override IController GetControllerInstance(RequestContext requestContext, Type
controllerType)
            {
                if (controllerType == null)
                    throw new InvalidOperationException(string.Format("Page not found: {0}",
requestContext.HttpContext.Request.RawUrl));
                return SmObjectFactory.Container.GetInstance(controllerType) as Controller;
            }
        }

        private static void setDbInitializer()
        {
            Database.SetInitializer(new MigrateDatabaseToLatestVersion<ApplicationDbContext,
Configuration>());
            SmObjectFactory.Container.GetInstance<IUnitOfWork>().ForceDatabaseInitialize();
        }
    }
}
```

در اینجا در متد setDbInitializer، نحوه‌ی استفاده و تعریف فایل Configuration لایه Data را ملاحظه می‌کنید؛ به همراه متد آغاز بانک اطلاعاتی و اعمال تغییرات لازم به آن در ابتدای کار برنامه. همچنین ControllerFactory برنامه نیز به StructureMapControllerFactory تنظیم شده‌است تا کار تزریق وابستگی‌ها به کنترلرهای برنامه به صورت خودکار میسر شود. در پایان کار هر درخواست نیز منابع Disposable رها می‌شوند.

ب) به پوشه‌ی Models برنامه مراجعه کنید. در اینجا در هر کلاسی که Id از نوع string وجود داشت، باید تبدیل به نوع int

شوند. چون primary key برنامه را به نوع int تغییر داده‌ایم. برای مثال کلاس‌های EditUserViewModel و RoleViewModel باید تغییر کنند.

ج) اصلاح کنترلرهای برنامه جهت اعمال تزریق وابستگی‌ها

اکنون اصلاح کنترلرها جهت اعمال تزریق وابستگی‌ها ساده‌است. در ادامه نحوه‌ی تغییر امضای سازنده‌های این کنترلرها را جهت استفاده از اینترفیس‌های جدید مشاهده می‌کنید:

```
[Authorize]
public class AccountController : Controller
{
    private readonly IAuthenticationManager _authenticationManager;
    private readonly IApplicationSignInManager _signInManager;
    private readonly IApplicationUserManager _userManager;
    public AccountController(IApplicationUserManager userManager,
                             IApplicationSignInManager signInManager,
                             IAuthenticationManager authenticationManager)
    {
        _userManager = userManager;
        _signInManager = signInManager;
        _authenticationManager = authenticationManager;
    }

    [Authorize]
public class ManageController : Controller
{
    // Used for XSRF protection when adding external logins
    private const string XsrfKey = "XsrfId";

    private readonly IAuthenticationManager _authenticationManager;
    private readonly IApplicationUserManager _userManager;
    public ManageController(IApplicationUserManager userManager, IAuthenticationManager
authenticationManager)
    {
        _userManager = userManager;
        _authenticationManager = authenticationManager;
    }

    [Authorize(Roles = "Admin")]
public class RolesAdminController : Controller
{
    private readonly IApplicationRoleManager _roleManager;
    private readonly IApplicationUserManager _userManager;
    public RolesAdminController(IApplicationUserManager userManager,
                                IApplicationRoleManager roleManager)
    {
        _userManager = userManager;
        _roleManager = roleManager;
    }


    [Authorize(Roles = "Admin")]
public class UsersAdminController : Controller
{
    private readonly IApplicationRoleManager _roleManager;
    private readonly IApplicationUserManager _userManager;
    public UsersAdminController(IApplicationUserManager userManager,
                                IApplicationRoleManager roleManager)
    {
        _userManager = userManager;
        _roleManager = roleManager;
    }
```

پس از این تغییرات، فقط کافی است بجای خواص برای مثال RoleManager سابق از فیلدهای تزریق شده در کلاس، مثلا _roleManager جدید استفاده کرد. امضای متدهای یکی است و تنها به یک search و replace نیاز دارد.
البته تعدادی اکشن متد نیز در اینجا وجود دارند که از string id استفاده می‌کنند. این‌ها را باید به int? Id تغییر داد تا با نوع primary key جدید مورد استفاده تطابق پیدا کنند.

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید:

AspNetIdentityDependencyInjectionSample

نظرات خوانندگان

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۱۰/۰۵ ۱۶:۱۱

ممنون از مطلب شما؛
برای تغییر نام جداول تشکیل شده، درون متد OnModelCreating کد زیر را نوشتم اما با بروز رسانی دیتابیس تغییری در اسامی جداول حاصل نشد:

```
modelBuilder.Entity<ApplicationUser>().ToTable("User").Property(p => p.Id).HasColumnName("Id");
            modelBuilder.Entity<CustomUserRole>().ToTable("UserRole").HasKey(p => new { p.RoleId,
p.UserId });
            modelBuilder.Entity<CustomUserLogin>().ToTable("UserLogin").HasKey(p => new {
p.LoginProvider, p.ProviderKey, p.UserId });
            modelBuilder.Entity<CustomUserClaim>().ToTable("UserClaim").HasKey(p => p.Id).Property(p =>
p.Id).HasColumnName("UserClaimId");
            modelBuilder.Entity<CustomRole>().ToTable("Role").Property(p =>
p.Id).HasColumnName("RoleId");
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۵ ۱۸:۵۳

این روش پس از امتحان، جواب داد :

```
protected override void OnModelCreating(DbModelBuilder builder)
        {
            base.OnModelCreating(builder);

            builder.Entity<ApplicationUser>().ToTable("Users");
            builder.Entity<CustomRole>().ToTable("Roles");
            builder.Entity<CustomUserClaim>().ToTable("UserClaims");
            builder.Entity<CustomUserRole>().ToTable("UserRoles");
            builder.Entity<CustomUserLogin>().ToTable("UserLogins");
        }
```

اگر متد base.OnModelCreating(builder) را در انتهای کار قرار دهید، تنظیمات پیش فرض کلاس پایه IdentityDbContext (یعنی همان نام‌های قدیمی) اعمال می‌شوند و تنظیمات شما بازنویسی خواهند شد. این متد باید در ابتدای کار فراخوانی شود.

نویسنده: فخاری
تاریخ: ۱۳۹۳/۱۰/۰۸ ۱۹:۲۱

با سلام
ممنون از مطلب مفید شما. واقعا عالی بود ☻
ولی قسمتی از کد برام نامفهوم بود :

```
public ApplicationDbContext()
            : base("connectionString1")
        {
            //this.Database.Log = data => System.Diagnostics.Debug.WriteLine(data);
            //فقط تعریف شده تا یک برک پوینت در اینجا قرار داده شود برای آزمایش تعداد بار فراخوانی آن
        }

        protected override void Dispose(bool disposing)
        {
            base.Dispose(disposing);
            //فقط تعریف شده تا یک برک پوینت در اینجا قرار داده شود برای آزمایش فراخوانی آن
        }
```

مخصوصا بخش Dispose .

چون قبلا که از الگوی واحد کار استفاده می‌شد این بخش وجود نداشت !
آیا وجود این کدها که در بالا اومده الزامی است؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۸ ۱۹:۲۴

خیر. متد Dispose را حذف کنید (در کلاس پایه هست). وجود سازنده هم صرفا جهت ساده سازی تعریف و انتخاب رشته اتصالی تعریف شده در web.config است.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۱ ۱۳:۲۷

نکته‌ای مهم در مورد مدیریت استراکچرمپ در این مثال

اگر از **Sm** ObjectFactory مطلب فوق استفاده می‌کنید، Container آن با ObjectFactory یکی نیست یا به عبارتی ObjectFactory اطلاعی در مورد تنظیمات کلاس سفارشی **Sm** ObjectFactory ندارد. بنابراین دیگر نباید از ObjectFactory قدیمی استفاده کنید.
در این حالت هرجایی ObjectFactory قدیمی را داشتید، با SmObjectFactory.Container تعویض می‌شود.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۱۰/۱۴ ۱۸:۴۶

نحوه دسترسی به کاربری که لاگین کرده چطوریه؟
من با دستور HttpContext.Current.User کار میکنم
بعضی اوقات نال بر میگردونه

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۴ ۱۹:۱۷

User.Identity را در مثال فوق جستجو کنید:

```
User.Identity.GetUserName()
User.Identity.GetUserId()
```

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۱۰/۱۴ ۱۹:۲۱

برای مشخص کردن نمونه پیاده‌سازی کننده IDataProtectionProvider در یک برنامه کنسول نیز باید از فایل Startup استفاده کرد؟ بیشتر هدفم Seed کردن دیتابیس است (مثلاً ایمپورت تعداد زیادی کاربر از طریق یک فایل و...). اینکار رو در متد SeedDatabase هم انجام دادم ولی هر بار استثنای UserId not found رو در:

```
result = this.SetLockoutEnabled(user.Id, false);
```

صادر میکنه، با گذاشتن Breakpoint متوجه شدم که برای Id صفر رو در نظر میگیره! از این جهت ترجیح دادم برای اینکار از طریق برنامه کنسول ویندوزی هم آن را تست کنم، مثل روشی که در اینجا برای ایجاد کاربر نوشته شده.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۱۰/۱۵ ۱:۱۵

در قسمت تزریق وابستگی کد زیر وجود داره

```
ioc.For<IIdentityMessageService>().Use<SmsService>();
ioc.For<IIdentityMessageService>().Use<EmailService>();
```

در صورت استفاده از اینترفیس مربوطه کدوم سرویس نمونه سازی میشه؟ در صورتی که هر دو سرویس وجود داشته باشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۵ ۱:۵۵

این مشکل اصلاح شد . باید از named instanceها در این حالت خاص استفاده شود.

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۱۰/۱۵ ۶:۶

با انجام اصلاحات فوق، زمانی که کاربر اقدام به تایید شماره تلفن همراه کنه با اینکه PhoneCode انتخاب میشه اما بازم قسمت ارسال ایمیل اجرا میشه.

نویسنده: سیروان عفیفی
تاریخ: ۱۳۹۳/۱۰/۱۵ ۱۱:۴۸

از همان متد SeedDatabase استفاده کردم، مشکل این بود که در متد Create نوع پسورد از این لحاظ که حداقل باید 6 کاراکتر باشه و درست بودن ایمیل و... نیز بررسی میشه، اگر مقادیر معتبر نباشه مقدار user.Id برابر با صفر میشه.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۵ ۱۳:۲۲

نیاز به تنظیمات setter injection هم داشت که اضافه شد .

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۱۰/۱۵ ۲۰:۲۳

با این تنظیمات

```
// map same interface to different concrete classes
                ioc.For<IIdentityMessageService>().Use<SmsService>().Named("smsService");
                ioc.For<IIdentityMessageService>().Use<EmailService>().Named("emailService");
                ioc.For<IApplicationUserManager>().HybridHttpOrThreadLocalScoped()
                    .Use<ApplicationUserManager>()
                    .Ctor<IIdentityMessageService>("smsService").Is<SmsService>()
                    .Ctor<IIdentityMessageService>("emailService").Is<EmailService>()
                    .Setter<IIdentityMessageService>(userManager =>
userManager.SmsService).Is<SmsService>()
                    .Setter<IIdentityMessageService>(userManager =>
userManager.EmailService).Is<EmailService>();
```

و این کد سازنده UserManager

```
public ApplicationUserManager(
            IApplicationUserStore store,
            IApplicationRoleManager roleManager,
            IDataProtectionProvider dataProtectionProvider,
            IIdentityMessageService smsService,
            IIdentityMessageService emailService)
            : base(store as IUserStore<User,int>)
        {
            this.roleManager = roleManager;
            this.dataProtectionProvider = dataProtectionProvider;
            EmailService = emailService;
            SmsService = smsService;
            Debug.WriteLine("Ticks = {0}",DateTime.Now.Ticks);
            Debug.WriteLine(emailService.ToString());
            Debug.WriteLine(smsService.ToString());
            CreateApplicationUserManager();
        }
```

در زمان دیباگ اینجور خروجی توی پنجره دیباگ گرفتم

```
Ticks = 635560859158196052
PWS.ServiceLayer.EF.Identity.EmailService
PWS.ServiceLayer.EF.Identity.SmsService
Ticks = 635560859158246098
PWS.ServiceLayer.EF.Identity.EmailService
PWS.ServiceLayer.EF.Identity.EmailService
```

نمی‌دونم چرا دوبار سازنده فراخوانی شده در بار اول تزریق وابستگی درست انجام شده اما در اخرین بار هم سرویس ایمیل به ایمیل و پیامک هر تزریق شده.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۱۶ ۲:۴۵

جهت مدیریت تک وهله‌ای ApplicationUserManger این تغییرات را اعمال کنید؛ یا از این فایل نهایی استفاده کنید.

نویسنده: نیکناز
تاریخ: ۱۳۹۳/۱۰/۲۳ ۱۲:۳۴

سلام؛ در روش برنامه نویسی لایه ای به این صورتی که شما آموزش دادید امکان دیدن جدول‌های بانک در بخش Server Explorer نیست؟ منظورم قبل از اینزرت اطلاعات در آنهاست؟ چون می‌خوام یکسری از جداول رو دستی مستقیم در بانک پر کنم.
و اینکه بعد از اینکه dbContext را در پوشه App_Data حذف کردم چه جوری اتصال کاملشو از بین ببرم چون برنامه ام ارور پیدا نکردن dbcontext را می‌ده و چه جوری می‌تونم دوباره ReCreate کنمش.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۲۳ ۱۳:۲

- در سازنده‌ی کلاس ApplicationDbContext ، به connectionString1 اشاره شده که تعریف آن در فایل web.config برنامه موجود است. به عبارتی در این مثال چنین رشته‌ی اتصالی تعریف شده‌است:

```
<add name="connectionString1"
connectionString="Data Source=(local);Initial Catalog=TestDbIdentity;Integrated Security = true"
providerName="System.Data.SqlClient" />
```

بنابراین بانک اطلاعاتی پیش فرض آن TestDbIdentity نام دارد (جهت اتصال به آن، برای مشاهده جداول و یا تغییر و ثبت اطلاعات). این رشته اتصالی هم مخصوص SQL Server تنظیم شده‌است که می‌توانید توسط management studio و یا سایر ابزارهای مشابه، همانند قبل به آن متصل شوید.
- «در پوشه App_Data حذف کردم» ... از مثال نهایی کامل شده استفاده کنید (^) و نیازی به تکرار این مراحل نیست تا خطای یافت نشدن dbcontext را دریافت نکنید.
- برای ReCreate فقط کافی هست که بانک اطلاعاتی TestDbIdentity را drop کنید. بعد برنامه را مجددا از نو اجرا کنید. چون مراحل migrations آن به حالت خودکار تنظیم شده‌است، بانک اطلاعاتی را به صورت خودکار ایجاد می‌کند یا تغییرات کلاس‌های دومین برنامه را به صورت خودکار به بانک اطلاعاتی اعمال خواهد کرد.
- برای آشنایی بیشتر با مباحث تعریف رشته اتصالی EF Code First و مباحث Migrations آن، سری EF Code First را در سایت یکبار مطالعه کنید.

نویسنده: اقبال
تاریخ: ۱۳۹۳/۱۰/۲۷ ۱۶:۱۰

ممنون از مطالب شما اگر بخواهیم از IApplicationRoleManager و IApplicationUserManager یا به طور کلی از اینترفیس‌ها در AuthorizeAttribute سفارشی (یا فیلتر هایی) که ایجاد کرده ایم استفاده کنیم ، تنظیمات strucutremap به چه صورت خواهد بود.

```
public class CustomAuthorizeAttribute : AuthorizeAttribute
    {


        public IApplicationRoleManager _roleManager { get; set; }

        public IApplicationUserManager _userManager { get; set; }


        public CustomAuthorizeAttribute(IApplicationUserManager userManager,
                                        IApplicationRoleManager roleManager)
        {
            _userManager = userManager;
            _roleManager = roleManager;
        }
        protected override bool AuthorizeCore(HttpContextBase httpContext)
        {
            // Put your custom logic here, returning true for success and false for failure,
            // or return base.AuthorizeCore(httpContext) to defer to the base implementation
            IPrincipal user = httpContext.User;
            IIdentity identity = user.Identity;

            if (!identity.IsAuthenticated)
            {
                return false;
            }

            bool isAuthorized = true;
            // TODO: perform custom authorization against the App
            var qry = _userManager.Users.Where(u => u.UserName == identity.Name).ToList();

            return isAuthorized;
        }
    }
```

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۲۷ ۱۷:۴۱

« تزریق وابستگی‌ها در فیلترهای ASP.NET MVC »

نویسنده: م سلیمانی
تاریخ: ۱۳۹۳/۱۱/۰۷ ۱۲:۳۳

با سلام؛ پروژه این مثال و دانلود کردم و سپس اجراش کردم - مشکلی نداره ولی بعد چند دقیقه ناگهان با این پیغام مواجه میشم.

لطفا راهنمایی بفرمائید. با تشکر.

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۱/۰۷ ۱۳:۲۳

- لطفا خطاها را تصویری ارسال نکنید. متن موجود در تصاویر، قابلیت جستجو و پیگیری ندارند.
- متن آن واضح است و به عمد تنظیم شده‌است. عنوان کرده‌است که آدرس خاصی را یافت نمی‌کند و این آدرس‌ها قابلیت تزریق وابستگی ندارند. آدرس آن (arterySignalR/ping) مرتبط است به browser link که به این صورت می‌توانید آن‌را غیرفعال کنید:

```
public static void RegisterRoutes(RouteCollection routes)
{
    routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
    routes.IgnoreRoute("{*browserlink}", new { browserlink = @".*/arterySignalR/ping" });
    //...
}
```

و یا

```
<appSettings>
    <add key="vs:EnableBrowserLink" value="false" />
</appSettings>
```

عنوان: **آشنایی با ساختار IIS قسمت سوم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۰۳ ۱:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, IHttpModule

همانطور که در مطلب قبلی گفتم، در این مطلب قرار است به WAS بپردازیم؛ در دنباله متن قبلی گفتیم که دومین وظیفه WWW Service این است: موقعی‌که یک درخواست جدید در صف درخواست‌ها وارد شد، به اطلاع WAS برساند.

WAS یا Windows Process Activation Service

در نسخه 7 به بعد، WAS مدیریت پیکربندی application pool و پروسه‌های کارگر را به جای WWW Service به عهده گرفته است. این مورد شما را قادر می‌سازد تا همان پیکربندی که برای Http در نظر گرفته‌اید، بر روی درخواست هایی که Http نیستند هم اعمال کنید. همچنین موقعی که سایت شما نیازی به درخواست‌های Http ندارد می‌توانید WAS را بدون WWW Service راه اندازی کنید. به عنوان یک مثال فرض کنید شما یک وب سرویس WCF را از طریق WCF Listener Adapter مدیریت می‌کنید و احتیاجی به درخواست‌های نوع Http listener ندارید و http.sys کاری برای انجام ندارد پس نیازی هم به راه اندازی www service نیست.

پیکربندی مدیریتی در WAS

در زمان شروع کار IIS، سرویس WAS اطلاعاتی را از فایل ApplicationHost.config می‌خواند و آن‌ها را به دست listener adapterهای مربوطه می‌رساند و lsitener adapterها ارتباط بین WAS و listenerهای مختلف را در IIS، برقرار می‌کنند.
آداپتورها اطلاعات لازم را از WAS می‌گیرند و به listenerهای مربوطه انتقال می‌دهند تا listenerها بر اساس آن تنظیمات یا پیکربرندی‌ها، به درخواست‌ها گوش فرا دهند.
در مورد WCF ، ابتدا WAS تنظیمات را برای آداپتور WCF که NetTcpActivator نام دارد ارسال کرده و این آداپتور بر اساس آن listener مربوطه را پیکربندی کرده تا به درخواست هایی که از طریق پروتوکل net.tcp می‌رسد گوش فرا دهد.
لیست زیر تعدادی از اطلاعاتی را که از فایل پیکربندی می‌خواند و ارسال می‌کند را بیان کرده است:

Global configuration information

Protocol configuration information for both HTTP and non-HTTP protocols

Application pool configuration, such as the process account information

Site configuration, such as bindings and applications

Application configuration, such as the enabled protocols and the application pools to which the applications belong

نکته پایانی اینکه اگر فایل ApplicationHost.config تغییری کند، WAS یک اعلان دریافت کرده و اطلاعات آداپتورها را به روز می‌کند.

مدیریت پروسه‌ها Process Managment

گفتیم که مدیریت پول و پروسه‌های کارگر جزء وظایف این سرویس به شمار می‌رود. موقعی که یک protocol listener درخواستی را دریافت می‌کند، WAS چک می‌کند که آیا یک پروسه کارگر در حال اجراست یا خیر. اگر application pool پروسه‌ای داشته باشد که در حال سرویس دهی به درخواست هاست، آداپتور درخواست را به پروسه کارگر ارسال می‌کند. در صورتی که پروسه‌ای در application pool در حال اجرا نباشد، WAS یک پروسه جدید را آغاز می‌کند و آداپتور درخواست را به آن پاس می‌کند.
نکته: از آنجایی که WAS هم پروسه‌های http و هم non-http را مدیریت می‌کند، پس میتوانید از یک applicatio pool برای چندین protocol استفاده کنید. به عنوان مثال شما یکی سرویس XML دارید که می‌توانید از آن برای سرویس دهی به پروتوکل‌های Http و net.tcp بهره بگیرید.

ماژول‌ها در IIS

قبلا مقاله ای در مورد moduleها با نام " کمی در مورد httpmoduleها " قرار داده بودیم که بهتر است برای آشنایی بیشتر، به آن رجوع کنید. به غیر از وب کانفیگ که برای معرفی ماژول‌ها استفاده می‌کردیم ، میتوانید به صورت گرافیکی و دستی هم این کار را

انجام بدهید. ابتدا یک پروژه class library ایجاد کرده و ماژول خود را بنویسید و سپس آن را به یک dll تبدیل کنید و dll را در شاخه bin که این شاخه در ریشه وب سایتتان قرار دارد کپی کنید. سپس در IIS قسمت module گزینه Add را انتخاب کنید و در قسمت اول نامی برای آن و در قسمت بعدی دقیقا همان قوانین type که در وب کانفیگ مشخص می‌کردید را مشخص کنید:

Namespace.ClassName

گزینه invoke only for requests to asp.net and manage handlers را هم تیک بزنید. کار تمام است.

ماژول‌های کد ماشین یا native

این ماژول‌ها به صورت پیش فرض به سیستم اضافه شده‌اند و در صورتی که میخواهید جایگزینی به منظور خصوصی سازی انجام دهید آن‌ها را پاک کنید و ماژول جدید را اضافه کنید.

جدول ماژول‌های HTTP

نام ماژول	توضیحات	نام فایل منبع
CustomErrorModule	موقعی که هنگام response، کد خطایی تولید می‌گردد، پیام خطا را پیکربندی و سپس ارسال می‌کند.	Inetsrv\Custerr.dll
HttpRedirectionModule	تنظیمات redirection برای درخواست‌های http را در دسترس قرار می‌دهد.	Inetsrv\Redirect.dll
ProtocolSupportModule	انجام عملیات مربوط به پروتوکل‌ها بر عهده این ماژول است؛ مثل تنظیم کردن قسمت هدر برای response.	Inetsrv\Protsup.dll
RequestFilteringModule	این ماژول از IIS 7.5 به بعد اضافه شد. درخواست‌ها را فیلتر می‌کند تا پروتوکل و رفتار محتوا را کنترل کند.	Inetsrv\modrqflt.dll
WebDAVModule	این ماژول از IIS 7.5 به بعد اضافه شد. امنیت بیشتر در هنگام انتشار محتوا روی HTTP SSL	Inetsrv\WebDAV.dll

ماژول‌های امنیتی

نام ماژول	توضیحات	نام فایل منبع
AnonymousAuthenticationModule	موقعی که هیچ کدام از عملیات authentication با موفقیت روبرو نشود، عملیات Anonymous authentication انجام می‌شود.	Inetsrv\Authanon.dll
BasicAuthenticationModule	عمل ساده و اساسی authentication را انجام می‌دهد.	Inetsrv\Authbas.dll
CertificateMappingAuthenticationModule	انجام عمل Certificate Mapping authentication در Active Directory	Inetsrv\Authcert.dll
DigestAuthenticationModule	Digest authentication	Inetsrv\Authmd5.dll
IISCertificateMappingAuthenticationModule	همان Certificate Mapping authentication ولی اینبار با IIS Certificate .	Inetsrv\Authmap.dll

نام ماژول	توضیحات	نام فایل منبع
RequestFilteringModule	عملیات اسکن URL از قبیل نام صفحات و دایرکتوری‌ها ، توع verb و یا کاراکترهای مشکوک و خطرآفرین	Inetsrv\Modrqflt.dll
UrlAuthorizationModule	عمل URL authorization	Inetsrv\Urlauthz.dll
WindowsAuthenticationModule	عمل NTLM integrated authentication	Inetsrv\Authsspi.dll
IpRestrictionModule	محدود کردن IPهای نسخه 4 لیست شده در IP Security در قسمت پیکربندی	Inetsrv\iprestr.dll

ماژول‌های محتوا

نام ماژول	توضیحات	نام فایل منبع
CgiModule	ایجاد پردازش‌های (Common Gateway Interface (CGI به منظور ایجاد خروجی response	Inetsrv\Cgi.dll
DefaultDocumentModule	تلاش برای ساخت یک سند پیش فرض برای درخواست هایی که دایرکتوری والد ارسال می‌شود	Inetsrv\Defdoc.dll
DirectoryListingModule	لیست کردن محتوای یک دایرکتوری	Inetsrv\dirlist.dll
IsapiModule	میزبانی فایل های ISAPI	Inetsrv\Isapi.dll
IsapiFilterModule	پشتیبانی از فیلتر های ISAPI	Inetsrv\Filter.dll
ServerSideIncludeModule	پردازش کدهای include شده سمت سرور	Inetsrv\Iis_ssi.dll
StaticFileModule	ارائه فایل‌های ایستا	Inetsrv\Static.dll
FastCgiModule	پشتبانی از CGI	Inetsrv\iisfcgi.dll

ماژول‌های فشرده سازی

DynamicCompressionModule	فشرده سازی پاسخ response با gzip	Inetsrv\Compdyn.dll
StaticCompressionModule	فشرده سازی محتوای ایستا	Inetsrv\Compstat.dll

ماژول‌های کش کردن

FileCacheModule	تهیه کش در مد کاربری برای فایل‌ها.	Inetsrv\Cachfile.dll
HTTPCacheModule	تهیه کش مد کاربری و مد کرنل برای http.sys	Inetsrv\Cachhttp.dll
TokenCacheModule	تهیه کش مد کاربری بر اساس جفت نام کاربری و یک token که توسط Windows user principals تولید شده است.	Inetsrv\Cachtokn.dll
UriCacheModule	تهیه یک کش مد کاربری از اطلاعات URL	Inetsrv\Cachuri.dll

ماژول‌های عیب یابی و لاگ کردن

CustomLoggingModule	بارگزاری ماژول‌های خصوصی سازی شده جهت لاگ کردن	Inetsrv\Logcust.dll
FailedRequestsTracingModule	برای ردیابی درخواست‌های ناموفق	Inetsrv\Iisfreb.dll
HttpLoggingModule	دریافت اطلاعات و پردازش وضعیت http.sys برای لاگ کردن	Inetsrv\Loghttp.dll
RequestMonitorModule	ردیابی درخواست هایی که در حال حاضر در پروسه‌های کارگر در حال اجرا هستند و گزارش اطلاعاتی در مورد وضعیت اجرا و کنترل رابط برنامه نویسی کاربردی.	Inetsrv\Iisreqs.dll
TracingModule	گزارش رخدادهای Microsoft Event Tracing for Windows یا به اختصار ETW	Inetsrv\Iisetw.dll

ماژول‌های مدیریتی و نظارتی بر کل ماژول‌ها

ManagedEngine	مدیرتی بر ماژول‌های غیر native که در پایین قرار دارند.	Microsoft.NET\Framework\v2.0.50727\webengine.dll
ConfigurationValidationModule	اعتبارسنجی خطاها، مثل موقعی که برنامه در حالت integrated اجرا شده و ماژول‌ها یا هندلرها در system.web تعریف شده‌اند.	Inetsrv\validcfg.dll

از IIS6 به بعد در حالت integrated و ماقبل، در حالت کلاسیک می‌باشند. اگر مقاله ماژول ها را خوانده باشید می‌دانید که تعریف آن‌ها در وب کانفیگ در بین این دو نسخه متفاوت هست و رویداد سطر آخر در جدول بالا این موقعیت را چک می‌کند و اگر به خاطر داشته باشید با اضافه کردن یک خط اعتبارسنجی آن را قطع می‌کردیم. در مورد هندلرها هم به همین صورت می‌باشد.
به علاوه ماژول‌های native بالا، IIS این امکان را فراهم می‌آورند تا از ماژول‌های کد مدیریت شده (یعنی CLR) برای توسعه توابع و کارکرد IIS بهره مند شوید:

ماژول	توضیحات	منبع
AnonymousIdentification	مدیریت منابع تعیین هویت برای کاربران ناشناس مانند asp.net profile	System.Web.Security.AnonymousIdentificationModule
DefaultAuthentication	اطمینان از وجود شی Authentication در context مربوطه	System.Web.Security.DefaultAuthenticationModule
FileAuthorization	تایید هویت کاربر برای دسترسی به فایل درخواست	System.Web.Security.FileAuthorizationModule
FormsAuthentication	با این قسمت که باید کاملا آشنا باشید؛ برای تایید هویت کاربر	System.Web.Security.FormsAuthentica

ماژول	توضیحات	منبع
		tionModule
OutputCache	مدیریت کش	System.Web.Caching.OutputCacheModule
Profile	مدیریت پروفایل کاربران که تنظیماتش را در یک منبع داده‌ای چون دیتابیس ذخیره و بازیابی می‌کند.	System.Web.Profile.ProfileModule
RoleManager	مدیریت نقش و سمت کاربران	System.Web.Security.RoleManagerModule
Session	مدیریت session ها	System.Web.SessionState.SessionStateModule
UrlAuthorization	آیا کاربر جاری حق دسترسی به URL درخواست را دارد؟	System.Web.Security.UrlAuthorizationModule
UrlMappingsModule	تبدیل یک Url واقعی به یک Url کاربرپسند	System.Web.UrlMappingsModule
WindowsAuthentication	شناسایی و تایید و هویت یک کاربر بر اساس لاگین او به ویندوز	System.Web.Security.WindowsAuthenticationModule

عنوان: **آشنایی با ساختار IIS قسمت چهارم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۰۴ ۱:۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS

پردازش درخواست‌های HTTP در IIS

بگذارید در این قسمت خلاصه‌ای از درخواست‌های نوع HTTP را که تا به الان گفته‌ایم، به همراه شکل بیان کنیم:

موقعی که کلاینت درخواست خود را مبنی بر یکی از منابع سرور ارسال می‌کند، Http.sys این درخواست را می‌گیرد.

http.sys با WAS تماس گرفته و درخواست می‌کند تا اطلاعات پیکربندی یا تنظیمات IIS را برای نحوه‌ی برخورد با درخواست، برایش بفرستد.

WAS هم اطلاعات پیکربندی شده را از محل ذخیره داده‌ها که applicationHost.config هست، می‌خواند.

WWW Service که یک آداپتور برای Http.sys هست، اطلاعات را از WAS دریافت می‌کند. این اطلاعات شامل پیکربندی application pool و سایت می‌باشد.

WWW Service اطلاعات را برای Http.sys میفرستد.

WAS یک پروسه کارگر را در application pool ایجاد می‌کند تا درخواست رسیده مورد پردازش قرار بگیرد.

پروسه‌های کارگر درخواست را پردازش کرده و خروجی یا response مورد نظر را تولید می‌کنند.

Http.sys نتیجه را دریافت و برای کلاینت می‌فرستد.

حال بیایید ببینیم موقعی که درخواست وارد پروسه‌ی کارگر میشود چه اتفاقی می‌افتد؟

در پروسه‌های کارگر، یک درخواست از مراحل لیست شده ای به ترتیب عبور می‌کند. در هسته وب سرور، رویدادهایی را فراخوانی می‌کند که در هر رویداد چندین ماژول native برای کارهایی چون authentication یا events logs دارد و در صورتیکه درخواستی نیاز به یک ماژول مدیریت شده CLR داشته باشد، از ماژول managedEngine native کمک گرفته و یک app domain را ایجاد می‌کند تا ماژول‌های مدیریت شده، عملیات لازم خودشان را انجام دهند. مثل authentication form و ...

موقعی هم که درخواست، از تمامی این رویدادها عبور کند، response برای http.sys ارسال می‌شود تا به کلاینت بازگشت داده شود. شکل زیر نحوه ورود یک درخواست به پروسه کارگر را نشان می‌دهد.

از نسخه 7 به بعد، IIS از یک معماری ماژولار استفاده می‌کند و این ویژگی، سه فایده دارد:
Componentization یا کامپوننت سازی
Extensibility یا توسعه پذیری یا قابل گشترش

ASP.NET Integration

Componentization

همه خصوصیات و ویژگی‌های این وب سرور، توسط کامپوننت‌ها مدیریت می‌شوند که باعث می‌شود شما به راحتی بتوانید کامپوننتی را اضافه، حذف یا جایگزین کنید و این باعث می‌شود که چندین امتیاز از IIS قبلی جلوتر باشد:
باعث کاهش attack surface می‌شود که در نتیجه امنیت سیستم را بالا میبرد. با ویژگی حذف کامپوننت‌ها شما می‌توانید ویژگی‌های غیرقابل استفاده IIS را حذف کنید تا وروردی‌های سیستم کاهش یابد. پس با کاهش ویژگی‌هایی که از آن هرگز استفاده نخواهید کرد، مدخل ورود هکر را از بین برده تا امنیت سرور بالاتر برود.
افزایش کارآیی و کاهش مصرف حافظه. با حذف ویژگی‌هایی که هرگز استفاده نمی‌کنید، در مصرف حافظه و بهینه استفاده شدن منابع سرور صرفه جویی کنید.
با وجود ویژگی افزودن و جایگزینی کامپوننت‌ها، ناخودآگاه ذهن ما به سمت کاستوم سازی یا خصوصی سازی کشیده می‌شود. با این کار شما به راحتی یک custom server ایجاد می‌کنید که این سرور بر اساس علایق شما کارش را انجام می‌دهد و به راحتی امکاناتی چون افزودن third partyها را به توسعه دهنده می‌دهد.

Extensibility

با توجه به موارد بالا، خصوصی سازی باعث گسترش امکانات IIS می‌شود که می‌تواند به دلایل زیر اتفاق بیفتد:
قدرت بخشی به برنامه‌های وب. امکانات و قدرتی که می‌تواند در این حالت به برنامه‌های در حال اجرا داد به مراتب بیشتر از استفاده از لایه‌های داخلی خود برنامه هست. برای اینکار شما می‌توانید کدهای خود را با ASP.Net نوشته یا از کدهای native چون C++ استفاده کنید.
تجربه‌ای از توسعه پذیری ساده‌تر و راحت تر
استفاده از قدرت و تمامی امکانات را به شما می‌دهد و می‌توانید تمام دستورات را برای همه منابع حتی فایل‌های ایستا، ASP ، CGI و دیگر منابع اجرا کنید.

ASP.NET Integration

تمامی موارد گفته شده بالا در این گزینه خلاصه می‌شود : محیط ASP.Net Integration به شما امکان استفاده از تمامی امکانات و منابع را به طور کامل می‌دهد. دانلود ماژول‌های مدیریت شده
دانلود ماژول‌های native

عنوان: **آشنایی با ساختار IIS قسمت پنجم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۰۵ ۹:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS

در مطالب قبلی در مورد ماژولار بودن IIS زیاد صحبت کردیم، ولی اجازه بدهید این مورد را به صورت کاربردی‌تر و موشکافانه‌تر بررسی کنیم. برای اینکه به مشکلی در طول این سری از مطالب برنخورید، IIS را به صورت کامل یعنی full feature نصب نمایید. از بخش control panel>programs & features>Turn Windows features on or off اقدام نمایید و هرچه زیر مجموعه Internet information service هست را برگزینید. در صورتی که از نسخه‌های ویندوز سرور 2008 استفاده می‌کنید از طریق server manager>roles>web server اقدام نمایید.

برای نصب یک ماژول باید دو مرحله را انجام داد:

نصب ماژول

فعال سازی ماژول

نکته ای که در مورد ماژول‌های native وجود دارد این هست که این ماژول‌ها دسترسی بدون محدودیتی به منابع سروری دارند و از این رو حتما باید این نکته را دقت کنید که ماژول native شما از یک منبع مورد اعتماد دریافت شده باشد.

نصب یک native module

برای نصب می‌توانید یکی از سه راه زیر را استفاده کنید:

ویرایش دستی فایل کانفیگ و از نسخه IIS7.5 به بعد هم می‌توانید از configuration editor هم استفاده کنید.

استفاده از محیط گرافیکی IIS

استفاده از خط فرمان با دستور Appcmd

مزیت روش دستی این هست که شما دقیقا می‌دانید در پشت صحنه چه اتفاقی می‌افتد و نتیجه هر کدام از این سه روش، اضافه شدن یک مدخل ورودی به تگ <globalmodules> است. برای اعمال تغییرات، مسیر زیر را بروید:

```
%windir%\system32\inetsrv\config\applicationhost.config
```

کسی که نیاز به دسترسی به این مسیر و انجام تغییرات دارد باید در بالاترین سطح مدیریتی سرور باشد.

اگر فایل را باز کنید و تگ globalmodule را پیدا کنید متوجه می‌شوید که تمامی ماژول‌ها در این قسمت معرفی شده‌اند و برای خود یک مدخل ورودی یا همان تگ add را دارند که در آن مسیر فایل dll هم ذکر شده است:

```
<globalModules>
 <addname="DefaultDocumentModule"image="%windir%\system32\inetsrv\defdoc.dll"/>
 <addname="DirectoryListingModule"image="%windir%\system32\inetsrv\dirlist.dll"/>
<add name="StaticFileModule"image="%windir%\system32\inetsrv\static.dll"/>
...
 </globalModules>
```

برای حذف یا جایگزینی یک ماژول به راحتی می‌توانید مدخل ورودی یک ماژول را به صورت دستی حذف نمایید و برای جایگزینی هم بعد از حذف، ماژول خود را معرفی کنید. ولی توجه داشته باشید که این حذف به معنی حذف این ماژول از تمامی اپلیکیشن‌های موجود بر روی IIS هست و سپس اضافه کردن یک ماژول به این بخش. همچنین اگر قصد شما فقط حذف یک ماژول از روی یکی از اپلیکشن‌ها باشد باید از طریق فایل کانفیگ سایت از مسیر تگ‌های <system.webserver><modules> و با استفاده از تگ‌های add و remove به معرفی یک ماژول مختص این اپلیکیشن و یا حذف یک ماژول خاص اقدام نمایید.

PreConditions

این ویژگی می‌تواند در خط معرفی ماژول، مورد استفاده قرار بگیرد. اگر به فایل نگاه کنید می‌بینید که در بعضی خطوط این ویژگی ذکر شده است. تعریف این ویژگی به هسته IIS می‌گوید که این ماژول در چه مواردی و به چه شیوه ای باید به کار گرفته شود.

```
<add name="ManagedEngine64" image="%windir%\Microsoft.NET\Framework64\v2.0.50727\webengine.dll"
preCondition="integratedMode,runtimeVersionv2.0,bitness64" />

 <add name="ManagedEngine" image="%windir%\Microsoft.NET\Framework\v2.0.50727\webengine.dll"
preCondition="integratedMode,runtimeVersionv2.0,bitness32" />

<add name="ManagedEngineV4.0_32bit" image="%windir%\Microsoft.NET\Framework\v4.0.30319\webengine4.dll"
preCondition="integratedMode,runtimeVersionv4.0,bitness32" />

<add name="ManagedEngineV4.0_64bit"
image="%windir%\Microsoft.NET\Framework64\v4.0.30319\webengine4.dll"
preCondition="integratedMode,runtimeVersionv4.0,bitness64" />
```

مقادیری که precondition میتواند بگیر شامل موارد زیر هستند: *bitness*

این گزینه به دو صورت bitness32 و bitness64 یافت می‌شود. امروزه پردازنده‌های 64 بیتی بسیار متداول شده اند و بسیاری از تولید کنندگان دارند به سمت عرضه ابزارهای 64 بیتی رو می‌آورند و به زودی عرضه‌های 32 بیتی را متوقف می‌کنند و به سمت سیستم عامل‌های 64 بیت سوییچ خواند کرد ولی باز هم هنوز برنامه‌های 32 بیتی زیادی هستند که مورد استفاده قرار می‌گیرند و نمی‌توان آن‌ها را نادیده گرفت. برای همین ویندوزهای 64 بیتی مایکروسافت در کنار محیط 64 بیتی‌شان از یک محیط 32 بیت به اسم WOW64 استفاده می‌کنند. در این حالت این امتیاز به شما داده می‌شود که از پروسه‌های کارگر 32 بیتی در کنار پروسه‌های کارگر 64 بیتی استفاده کنید و PreCondition به bitness32 یا bitness64 تنظیم می‌شود تا از صحت بارگزاری dll در یک محیط درست مطمئن شود. در صورتی که این خصوصیت ذکر نشود یک هندلر 32 بیتی و 64 بیتی و یک module map اجرا می‌شود.

Runtime version

اگر ماژول خاصی برای اجرا به ورژن خاصی از net framwork. نیاز دارد، این ویژگی ذکر می‌شود. در صورتی که ماژولی قصد اجرای بر روی فریم ورک اشتباهی داشته باشد سبب خطا خواهد شد.

Mana gedHandler

با معرفی IIS7 ما با یک مدل توسعه پذیر روبرو شدیم و می‌توانستیم ماژول‌ها و هندلرهای خود را بنویسیم و مستقیما در Pipeline قرار دهیم ولی سوییچ کردن بین دو بخش کدهای مدیریت شده و native یک عمل سنگین برای سیستم به شمار می‌آید و به منظور کاهش این بار گزینه managedhandler قرار داده شده است تا تعیین کند مواقعی که درخواست نیازی به این ماژول ندارد، این ماژول اجرا نگردد. به عنوان مثال فایل‌های ایستا چون jpg یا html و... شامل این ماژول نخواهند شد. واضح‌ترین مثال در این زمینه Forms Authentication می‌باشد و موقعی اجرا می‌شوند که درخواست فایل‌های aspx شده باشد و اگر یک فایل html را درخواست کنید این ماژول امنیتی روی آن اثری ندارد و عملیات شناسایی هویت روی آن اجرا نمی‌شود و اگر می‌خواهید روی همه فایل‌ها، این عملیات شناسایی انجام شود باید خصوصیت "precondition="managedhandler حذف شود.

در صورتی که تگ module را به صورت زیر بنویسید تمامی ماژول‌ها برای تمامی درخواست‌ها اجرا خواهد شد، صرف نظر از اینکه آیا ماژولی به صورت "precondition="managedmodule مقداردهی شده است یا خیر.

```
<modules runAllManagedModulesForAllRequests="true"/>
```

The Mode Precondition

تا به الان گفتیم که چگونه میتوانیم یک ماژول و یا هندلر مدیریت شده‌ای را به Pipeline اضافه کنیم؛ ولی IIS7 به بالا نیاز دارد که تا پروسه‌های کارگر را به روشی خاص به این منظور اجرا کند و فریم ورک دات نت را برای اجرای آن‌ها بارگزاری کند. همچنین به اجرای ماژولی به اسم webengine.dll برای مدیریت ماژولهای مدیریت شده نیازمند است و خود IIS در مورد کدهای مدیریت شده چیزی متوجه نمی‌شود. پس ما برای اینکه IIS را متوجه این موضوع نمائیم، باید Integrated mode را به آن معرفی کنیم.

در نسخه‌های قبلی IIS یک روش قدیمی برای کدهای مدیریت شده وجود داشت که از طریق اینترفیسی به نام ISAPI صورت می‌گرفت. در این حالت ASPNET_ISAPI.DLL مسئول این کار بود و اگر هنوز هم میخواهید از این dll در نسخه‌های جدیدتر IIS کمک بگیرید باید به جای معرفی integration mode ، classic mode را معرفی کنید.

با برابر کردن precondtion به مقدار "intgretedmode" هندلر یا ماژول شما در یک پول با خصوصیت integrated بارگزاری خواهد شد و اگر مقدار آن "classicmode" باشد در یک پول بدون خاصیت integrated بارگزاری می‌شود.

تفاوت بین دو روش کلاسیک و مجتمع integrated بر سر این هست که در روش جدید، ماژول شما به عنوان یک پلاگین برای IIS

دیده نمی‌شود و کد شما را جزئی از کامپوننت‌های خود به شمار می‌آورد. به صورت واضح‌تر در حالت کلاسیک موقعی که درخواستی وارد pipeline میشد ابتدا از کامپوننت‌ها و ماژول‌های داخلی خود IIS عبور داده میشد و بعد فایل ASPNET_ISAPI.DLL جهت پردازش کدهای مدیریت شده صدا زده میشد و با توجه به کدهای شما، بعضی مراحل تکرار میشد؛ مثلا اگر کد شما در مورد Authentication بود و بعد از گذر از مراحل auth داخل خود IIS و بقیه موارد دوباره نوبت کد شما و گذر از مراحل authentication بود. یعنی وجود دو pipeline؛ ولی در حالت مجتمع این دوبار انجام وظیفه از بین رفته است چرا که کدهای شما به طور مستقیم در pipeline قرار دارند و آن‌ها را جزئی از خود می‌داند، نه یک پلاگین که افزون بر فعالیت خودشان، اجرای کدهای شما رو هم بر دوش بکشند.

شکل زیر نمونه ای از حالت کلاسیک را نشان می‌دهد که در آن دو بار عمل auth دارد انجام میگیرد.

شکل زیر هم نمونه ای حالت مجتمع هست:

در کل امروزه دیگر استفاده از روش کلاسیک راهکار درستی نیست و این ویژگی تنها به عنوان یک سازگاری با نمونه کارهای قدیمی است.

تگ‌هایی که از خصوصت precondition استفاده می‌کنند به شرح زیر هستند:

ISAPI filters

globalModules

handlers

modules

در مورد بقیه تگ‌ها در آینده بیشتر بحث می‌کنیم. بهتر هست مطلب را با توضیح precondition جهت ممانعت از طولانی و طومار شدن در اینجا ببندیم و در قسمت آینده دیگر روش‌های نصب ماژولمان را دنبال کنیم.

عنوان: **آشنایی با ساختار IIS قسمت ششم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۰۷ ۰:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS

در مطلب قبلی روش دستی را برای اضافه کردن ماژول‌های خود، نام بردیم. در اینجا به روش‌های دیگر اضافه کردن ماژول‌ها می‌پردازیم.

استفاده از محیط گرافیکی IIS جهت لیست کردن، اضافه و حذف ماژول‌ها

به بخش modules درIIS بروید. در پنل سمت راست همه امکانات جهت افزودن و ویرایش و حذف وجود دارند:

روش معرفی ماژول در خط فرمان با استفاده از دستور Appcmd

```
Appcmd.exe install module   /name:MODULE_NAME /image:PATH_TO_DLL
```

قسمت name که نام ماژول است و قسمت image هم مسیر قرار گرفتن فایل dll هست.

برای نمونه:

```
%windir%\system32\inetsrv\appcmd.exe install module /name:DefaultDocumentModule
/image:%windir%\system32\inetsrv\defdoc.dll
```

در صورتیکه ماژولی که قبلا افزوده شده باشد را بخواهید اضافه کنید، خطای زیر را دریافت خواهید کرد:

```
ERROR ( message:Failed to add duplicate collection element "DefaultDocumentModule". )
```

جهت حذف ماژول دستور زیر را صادر کنید:

```
Appcmd.exe uninstall module MODULE_NAME
```

نمونه:

```
%windir%\system32\inetsrv\appcmd.exe uninstall module DefaultDocumentModule
```

گرفتن کوئری یا لیستی از ماژول‌های فعال برای یک اپلیکیشن یا عمومی:

```
Appcmd.exe list modules [/app.name:APPLICATION_NAME]
```

سوییچ aap.name اختیاری است ولی اگر نام یک اپلیکیشن را به آن بدهید، فقط ماژول‌هایی را که روی این اپلیکیشن اجرا می‌شوند، لیست می‌کند.

نمونه:

```
%windir%\system32\inetsrv\appcmd.exe list modules /app.name:"Default Web Site"
```

کد زیر هم نمونه ای برای لیست کردن تمامی ماژول‌های عمومی که بر روی تمامی اپلیکیشن‌ها اجرا می‌شوند:

```
%windir%\system32\inetsrv\appcmd.exe list modules
```

خط زیر یک ماژول را برای همه اپلیکیشن‌ها یا اپلیکیشن خاصی فعال می‌کند که بستگی دارد سوییچ type چگونه مقداردهی شده باشد:

```
Appcmd.exe add module /name:MODULE_NAME /type:MGD_TYPE
```

برای مثال خط زیر باعث می‌شود ماژول Forms Authentication فقط برای وب اپلیکیشن default web site فعال شود:

```
%windir%\system32\inetsrv\appcmd.exe add module /name:FormsAuthentication
/type:System.Web.Security.FormsAuthenticationModule /app.name:"Default Web Site"
```

یا در پایین آن را به صورت عمومی یا global فعال می‌کند:

```
%windir%\system32\inetsrv\appcmd.exe add module /name:FormsAuthentication
/type:System.Web.Security.FormsAuthenticationModule
```

برای غیرفعال کردن یک ماژول از دستور زیر استفاده می‌شود:

```
Appcmd.exe delete module MODULE_NAME [/app.name:APPLICATION_NAME]
```

اگر غیر فعال کردن یک ماژول در یک اپلیکیشن خاص مدنظر شما باشد دستور زیر نمونه آن است:

```
%windir%\system32\inetsrv\appcmd.exe delete module FormsAuthentication /app.name:"Default Web Site"
```

اگر قصد دارید آن‌را بر روی تمامی اپلیکیشن‌ها غیرفعال کنید، دستور زیر نمونه آن است:

```
%windir%\system32\inetsrv\appcmd.exe delete module FormsAuthentication
```

حفظ کردن یا به خاطر سپردن دستورات بالا ممکن است کار سخت و دشواری باشد، به همین جهت از help کمک بگیرید:

```
Appcmd.exe module /?
```

یا به شکل اختصاصی‌تر برای یک دستور

```
Appcmd.exe install module /?Appcmd add module /?
```

عنوان: ثبت جزئیات استثناهای Entity framework توسط ELMAH
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۸ ۱۱:۲۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Entity framework, MVC, ELMAH

در حین بروز استثناهای Entity framework، می‌توان توسط ابزارهای Logging متنوعی مانند ELMAH ، جزئیات متداول آن‌ها را برای بررسی‌های آتی ذخیره کرد. اما این جزئیات فاقد SQL نهایی تولیدی و همچنین پارامترهای ورودی توسط کاربر یا تنظیم شده توسط برنامه هستند. برای اینکه بتوان این جزئیات را نیز ثبت کرد، می‌توان یک IDbCommandInterceptor جدید را طراحی کرد.

کلاس EfExceptionsInterceptor

در اینجا نمونه‌ای از یک پیاده سازی اینترفیس IDbCommandInterceptor را مشاهده می‌کنید. همچنین طراحی یک متد عمومی که می‌تواند به جزئیات SQL نهایی و پارامترهای آن دسترسی داشته باشد، در اینترفیس IEfExceptionsLogger ذکر شده‌است.

```
public interface IEfExceptionsLogger
{
    void LogException<TResult>(DbCommand command,
        DbCommandInterceptionContext<TResult> interceptionContext);
}

using System.Data.Common;
using System.Data.Entity.Infrastructure.Interception;

namespace ElmahEFLogger
{
    public class EfExceptionsInterceptor : IDbCommandInterceptor
    {
        private readonly IEfExceptionsLogger _efExceptionsLogger;

        public EfExceptionsInterceptor(IEfExceptionsLogger efExceptionsLogger)
        {
            _efExceptionsLogger = efExceptionsLogger;
        }

        public void NonQueryExecuted(DbCommand command, DbCommandInterceptionContext<int>
interceptionContext)
        {
            _efExceptionsLogger.LogException(command, interceptionContext);
        }

        public void NonQueryExecuting(DbCommand command, DbCommandInterceptionContext<int>
interceptionContext)
        {
            _efExceptionsLogger.LogException(command, interceptionContext);
        }

        public void ReaderExecuted(DbCommand command, DbCommandInterceptionContext<DbDataReader>
interceptionContext)
        {
            _efExceptionsLogger.LogException(command, interceptionContext);
        }

        public void ReaderExecuting(DbCommand command, DbCommandInterceptionContext<DbDataReader>
interceptionContext)
        {
            _efExceptionsLogger.LogException(command, interceptionContext);
        }

        public void ScalarExecuted(DbCommand command, DbCommandInterceptionContext<object>
interceptionContext)
        {
            _efExceptionsLogger.LogException(command, interceptionContext);
        }

        public void ScalarExecuting(DbCommand command, DbCommandInterceptionContext<object>
interceptionContext)
        {
            _efExceptionsLogger.LogException(command, interceptionContext);
        }
    }
}
```

تهیه یک پیاده سازی سفارشی از IEfExceptionsLogger توسط ELMAH

اکنون که ساختار کلی IDbCommandInterceptor سفارشی برنامه مشخص شد، می‌توان پیاده سازی خاصی از آن‌را جهت استفاده از ELMAH به نحو ذیل ارائه داد:

```
using System;
using System.Data.Common;
using System.Data.Entity.Infrastructure.Interception;
using Elmah;

namespace ElmahEFLogger.CustomElmahLogger
{
    public class ElmahEfExceptionsLogger : IEfExceptionsLogger
    {
        /// <summary>
        /// Manually log errors using ELMAH
        /// </summary>
        public void LogException<TResult>(DbCommand command,
            DbCommandInterceptionContext<TResult> interceptionContext)
        {
            var ex = interceptionContext.OriginalException;
            if (ex == null)
                return;

            var sqlData = CommandDumper.LogSqlAndParameters(command, interceptionContext);
            var contextualMessage = string.Format("{0}{1}OriginalException:{1}{2} {1}", sqlData,
Environment.NewLine, ex);


            if (!string.IsNullOrWhiteSpace(contextualMessage))
            {
                ex = new Exception(contextualMessage, new ElmahEfInterceptorException(ex.Message));
            }

            try
            {
                ErrorSignal.FromCurrentContext().Raise(ex);
            }
            catch
            {
                ErrorLog.GetDefault(null).Log(new Error(ex));
            }
        }
    }
}
```

در اینجا شیء Command به همراه SQL نهایی تولید و پارامترهای مرتبط است. همچنین interceptionContext.OriginalException جزئیات عمومی استثنای رخ داده را به همراه دارد. می‌توان این اطلاعات را پس از اندکی نظم بخشیدن، به متد Raise مربوط به ELMAH ارسال کرد تا جزئیات بیشتری از استثنای رخ داده شده، در لاگ‌های آن ظاهر شوند.

استفاده از ElmahEfExceptionsLogger جهت طراحی یک Interceptor عمومی

```
public class ElmahEfInterceptor : EfExceptionsInterceptor
    {
        public ElmahEfInterceptor()
            : base(new ElmahEfExceptionsLogger())
        { }
    }
```

برای استفاده از ElmahEfExceptionsLogger و تهیه یک Interceptor عمومی، می‌توان با ارث بری از کلاس Interceptor ابتدای بحث شروع کرد و وهله‌ای از ElmahEfExceptionsLogger را به سازنده‌ی آن تزریق نمود (یکی از چندین روش ممکن). سپس برای

استفاده از آن کافی است به ابتدای متد Application_Start فایل Global.asax.cs مراجعه و در ادامه سطر ذیل را اضافه نمود:

```
DbInteception.Add(new ElmahEfInterceptor());
```

پس از آن جزئیات کلیه استثناهای EF در لاگ‌های نهایی ELMAH به نحو ذیل ظاهر خواهند شد:

کدهای کامل این پروژه را از اینجا می‌توانید دریافت کنید:

ElmahEFLogger

نظرات خوانندگان

نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۳/۱۰/۰۸ ۲۰:۴۷

با سلام و تشکر از مطلب شما

در صورت استفاده از پکیج بالا باید فیلتر کلی که در پست آموزش MVC قسمت 16 توضیح دادین برای ثبت استثناهای elmah حذف بشه؟

نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۰۸ ۲۱:۲۵

خیر. کار این Interceptor صرفا لاگ کردن سراسری و نامرئی ریز جزئیات استثناهای EF در کل برنامه است. استثناهای رخ داده، در ادامه از فیلتر ELMAH اصلی تنظیم شده رد خواهند شد و یکبار هم در آنجا لاگ می‌شوند. این مورد چون منحصر به EF نیست، بنابراین ضرورتی به حذف آن هم نیست.

عنوان: آشنایی با ساختار IIS قسمت هفتم
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۰۸ ۱۳:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, Security

در این قسمت بیشتر یک سری از ماژول‌ها را به شما در قالب جداول گروه بندی شده معرفی خواهیم کرد :

همانطور که در قسمت‌های قبلی گفتیم سرور IIS آماده خصوصی سازی و کار بر اساس علائق شماست؛ ولی توجه داشته باشید حذف تمامی ماژول‌ها ممکن است اثرات جانبی هم داشته باشد. در اینجا ما ماژول هایی را به شما معرفی می‌کنیم که بدانید کار هر ماژول چیست تا مثلا با حذف ماژولی، امنیت وب سایت خود را به خطر نیندازید :

ماژول‌های سودمند یا utility

نام ماژول:	UriCacheModule
توضیح:	این ماژول نوعی کش برای URLها به شمار می‌رود. موقعی که url درخواست می‌شود، اطلاعات در اولین درخواست خوانده شده و کش می‌شود و اگر دوباره همان url درخواست شود، بدون خواندن تنظیمات و بر اساس تنظیمات قبلی، کار url مربوطه را انجام میدهد تا اطلاعات پیکربندی تغییر کند و بر اساس اطلاعات جدید، خود را به روز کند.
تگ قابل پیکربندی:	لازم ندارد
وابستگی:	ندارد
اثرات حذف آن:	کارایی سیستم کاهش می‌یابد و سیستم مجبور است برای هر درخواست فایل پیکربندی را بخواند.
نام ماژول :	FileCacheModule
توضیح :	فایل هندلِ فایل‌هایی که قبلا در سرور باز شده‌اند را کش می‌کند تا در صورت نیاز در دفعات بعدی سریعتر عمل کند.
تگ قابل پیکربندی :	لازم ندارد .
وابستگی :	ندارد.
اثرات حذف آن :	کارایی سیستم کاهش می‌یابد. سیستم در هر اجرای دستور مربوط به فایل‌ها باید فایل هندل را به دست آورد.

نام ماژول :	**TokenCacheModule**
توضیح :	توکن‌های امنیتی ویندوز که پسوردهایی بر اساس authentication schemes هستند را کش می‌کند (anonymous authentication, basic authentication, IIS client certificate authentication)
تگ قابل پیکربندی :	لازم ندارد
وابستگی :	ندارد
اثرات حذف آن :	کارایی سیستم به شدت پایین می‌آید. کاربر باید با هر درخواستی لاگین کند. یکی از اصلی‌ترین ضربه‌ها با حذف این ماژول این است که اگر مثلا یک پسورد از یک فایل html محافظت می‌کند و این صفحه به 50 تصویر ارجاع دارد، 51 بار باید درخواست لاگین اجرا گردد یا شاید هم بدتر

MANAGED ENGINE: ASP.NET INTEGRATION

نام ماژول :	**ManagedEngine**
توضیح :	مدیریت ماژول‌های native و مدیریت شده
تگ قابل پیکربندی :	
وابستگی :	ندارد
اثرات حذف آن :	مشخصا غیرفعال شدن asp.net integrated و غیر فعال شدن تمامی ماژول‌ها و هندلرهای تگ وب کانفیگ یا داخل فایل کانفیگ IIS که در مقالات قبلی به تفصیل بیان کرده‌ایم.

IIS 7 NATIVE MODULES

نام ماژول :	**HttpCacheModule**
توضیح :	مدیریت کش خروجی در htttp.sys بر اساس پیکربندی مثل تعریف سایز کش و ...
تگ قابل پیکربندی :	System.webServer/caching

نام ماژول :	**HttpCacheModule**
وابستگی :	ندارد.
اثرات حذف آن :	محتوا دیگر به صورت کرنل مد، کش نمی‌شود و کش پروفایل هم ندید گرفته می‌شود و احتمالا بر کارآیی و استفاده از منابع هم اثر می‌گذارد.
نام ماژول :	**DynamicCompressionModule**
توضیح :	پیاده سازی in-memory compression در محتوای پویا
تگ قابل پیکربندی :	system.webServer/httpCompression and system.webServer/urlCompression.
وابستگی :	وابستگی ندارد چرا که به طور پیش فرض غیرفعال است.

نام ماژول :	StaticCompressionModule
توضیح :	پیادسازی فشرده سازی در محتوای ایستا و برای فایل‌های سیستمی از نوع in memory
تگ قابل پیکربندی :	system.webServer/httpCompression and system.webServer/urlCompression
وابستگی :	ندارد.
اثرات حذف آن :	در صورت عدم فشرده سازی بر مصرف ترافیک تاثیر گذار است.
نام ماژول :	**DefaultDocumentModule**
توضیح :	پیاده سازی یک لیست سند پیش فرض. درخواست‌ها مدام پشت سر هم می‌آیند و این درخواست‌های پشت سرهم، به سند پیش فرض هدایت می‌شوند. همان پنجره ای که شما به ترتیب فایل‌های index.htm,index.asp,default.aspx و... را تعیین می‌کنید.
تگ قابل پیکربندی :	system.webServer/defaultDocument

نام ماژول :	StaticCompressionModule
وابستگی :	ندارد.
اثرات حذف آن :	درخواست را به ریشه هدایت می‌کند. مثلا برای localhost صفحه 404 باز میگرداند و اگر directoryBrowsing فعال باشد لیستی از دایرکتوری ریشه را باز میگرداند.

نام ماژول :	**DirectoryListingModule**
توضیح :	پیادی سازی لیستی از محتویات یک دایرکتوری
تگ قابل پیکربندی :	system.webServer/directoryBrowse
وابستگی :	ندارد.
اثرات حذف آن :	اگر این ماژول و ماژول قبلی غیرفعال باشند response نهایی خالی است.
نام ماژول :	**ProtocolSupportModule**
توضیح :	پیاده سازی اختصاصی از response header پیاده سازی تنظیمات trace و HTTP verbs. پیاده سازی تنظیمات مربوطه به keep-alive بر اساس پیکربندی
تگ قابل پیکربندی :	system.webServer/httpProtocol
وابستگی :	ندارد.
اثرات حذف آن :	بازگرداندن پیام خطای "Method not allowed 405".

نام ماژول :	**HttpRedirectionModule**
توضیح :	پیاده سازی عملیات انتقال یا redirect

نام ماژول :	**HttpRedirectionModule**
تگ قابل پیکربندی :	system.webServer/httpRedirect
وابستگی :	ندارد.
اثرات حذف آن :	خطر امنیتی: اگر منابعی با redirect کردن محافظت می‌شوند، از این پس در دسترسند.
نام ماژول :	**ServerSideIncludeModule**
توضیح :	حمایت از فایل shtm یا shtml و ...
تگ قابل پیکربندی :	system.webServer/serverSideInclude
وابستگی :	ندارد.
اثرات حذف آن :	این فایل‌ها به صورت متنی نمایش داده می‌شوند

نام ماژول :	**StaticFileModule**
توضیح :	فایل‌های ایستا را به همراه پسوند ارسال می‌کند. مثل jpg,html و نوع محتوا را بر اساس staticContent/mimeMap پیکربندی می‌کند.
تگ قابل پیکربندی :	system.webServer/staticContent
وابستگی :	ندارد.
اثرات حذف آن :	فایل‌های ایستا دیگر ارائه نشده و به جای آن خطای 404 بازگشت داده می‌شود.
نام ماژول :	**AnonymousAuthenticationModule**
توضیح :	پیاده سازی سیستم شناسایی افراد ناشناس. همانطور که میدانید در یک وب سایت حداقل محتوایی برای افرادی بدون داشتن اکانت هم وجود دارد. برای اینکار یک شیء httpuser ایجاد می‌کند.
تگ قابل پیکربندی :	

نام ماژول :	**StaticFileModule**
	system.webServer/security/authentication/anonymousAuthentication
وابستگی :	ندارد.
اثرات حذف آن :	حداقل باید یک سیستم امنیتی برای شناسایی یا authenticate وجود داشته باشد. httpuser یک ساختار داده ای در IIS می‌باشد و در صورت نبودن هیچ سیستم شناسایی وجود نداشته و در نبود شیء httpuser سیستم خطای 401.2 را تولید می‌کند.

نام ماژول :	**CertificateMappingAuthenticationModule**
توضیح :	مجوز SSL را به Active Directory نگاشت می‌کند.
تگ قابل پیکربندی :	system.webServer/security/authentication/clientCertificateMappingAuthentication
وابستگی :	برای اینکه این ماژول وظیفه خود را انجام دهد باید تنظیمات SSL انجام شود و همچنین سیستم IIS جزئی از دامنه Active directory باشد
اثرات حذف آن :	درخواست‌ها، نرمال رسیدگی میشوند انگار SSL وجود ندارد.
نام ماژول :	**BasicAuthenticationModule**
توضیح :	پیاده سازی پایه‌ای و روتین شناسایی کاربران بر اساس آن چیزی که در استانداد زیر آمده است RFC 2617 .
تگ قابل پیکربندی :	system.webServer/security/authentication/basicAuthentication
وابستگی :	None.
اثرات حذف آن :	حداقل باید یک سیستم امنیتی برای شناساسایی یا authenticate وجود داشته باشد. httpuser یک ساختار

نام ماژول :	**CertificateMappingAuthenticationModule**
	داده‌ای در IIS می‌باشد و در صورت نبود، هیچ سیستم شناسایی یافت نشده و نبود شیء httpuser در سیستم، خطای 401.2 را تولید می‌کند.

نام ماژول :	**WindowsAuthenticationModule**
توضیح :	((windows Authentication (NTLM or Negotiate (Kerberos
تگ قابل پیکربندی :	system.webServer/security/authentication/windowsAuthentication
وابستگی :	ندارد.
اثرات حذف آن :	حداقل باید یک سیستم امنیتی برای شناسایی یا authenticate وجود داشته باشد. httpuser یک ساختار داده ای در IIS می‌باشد و در صورت نبود، هیچ سیستم شناسایی یافت نشده و نبود شیء httpuser در سیستم، خطای 401.2 را تولید می‌کند.
نام ماژول :	**DigestAuthenticationModule**
توضیح :	پیاده سازی سیستم شناسایی دیاجست بر اساس RFC 2617 .
تگ قابل پیکربندی :	system.webServer/security/authentication/digestAuthentication
وابستگی :	IIS باید بخشی از دامنه Active Directory باشد.
اثرات حذف آن :	حداقل باید یک سیستم امنیتی برای شناسایی یا authenticate وجود داشته باشد. httpuser یک ساختار داده ای در IIS می‌باشد و در صورت نبود، هیچ سیستم شناسایی یافت نشده و نبود شیء httpuser در سیستم، خطای 401.2 را تولید می‌کند.

نام ماژول :	**IISCertificateMappingAuthenticationModule**
توضیح :	پیاده سازی نگاشت مجوزهای IIS، نگهداری و ذخیره اطلاعات همه نگاشت‌ها و مجوزهای کاربری چون SSL client certificates
تگ قابل پیکربندی :	system.webServer/iisClientCertificateMappingAuthentication
وابستگی :	اطلاعات SSL به همراه دریافت client certificates جهت پیکربندی این ماژول
اثرات حذف آن :	حداقل باید یک سیستم امنیتی برای شناسایی یا authenticate وجود داشته باشد. httpuser یک ساختار داده ای در IIS می‌باشد و در صورت نبود، هیچ سیستم شناسایی یافت نشده و نبود شیء httpuser در سیستم، خطای 401.2 را تولید می‌کند.
نام ماژول :	**UrlAuthorizationModule**
توضیح :	پیاده سازی authorization بر اساس قوانین پیکربندی شده
تگ قابل پیکربندی :	system.webServer/security/authorization
وابستگی :	ندارد.
اثرات حذف آن :	محتواهای محافظت شده توسط authorization دیگر محافظت نمی‌شوند.

نام ماژول :	**IsapiModule**
توضیح :	پیاده سازی ISAPI Extension
تگ قابل پیکربندی :	system.webServer/isapiCgiRestriction
وابستگی :	ندارد.

نام ماژول :	**IsapiModule**
اثرات حذف آن :	هندلرهای معرفی شده در بخش IsapiModule و تگ handlers دیگر اجرا نمی‌شوند
نام ماژول :	**IsapiFilterModule**
توضیح :	پیاده سازی ISAPI filter
تگ قابل پیکربندی :	system.webServer/isapiFilters
وابستگی :	ندارد.
اثرات حذف آن :	اگر برنامه ای از ISAPI filter استفاده می‌کند، در اجرا دچار مشکل خواهد شد.

نام ماژول :	**IpRestrictionModule**
توضیح :	یک سیستم تشخیص دسترسی بر اساس آی پی‌های ورژن4
تگ قابل پیکربندی :	system.webServer/security/ipSecurity
وابستگی :	IPv4 stack باید نصب شود.
اثرات حذف آن :	کلاینت هایی که IP هایشان در IPsecurity لیست شده‌اند ندید گرفته میشوند
نام ماژول :	**RequestFilteringModule**
توضیح :	پیاده سازی یک مجموعه قدرتمند از قوانین امنیتی که درخواست‌های مشکوک را پس می‌زند.
تگ قابل پیکربندی :	system.webServer/security/requestFiltering
وابستگی :	ندارد.
اثرات حذف آن :	دیگر قوانین امنیتی اجرا نخواهند شد و سبب وجود مشکلات امنیتی میشود.

نام ماژول :	**CustomLoggingModule**
توضیح :	پیاده سازی اینترفیس ILogPlugin در سمت IIS، به مشتریان اجازه میدهد تا لاگ‌های خود را توسعه دهند. هر چند این روش توصیه نمی‌شود و توصیه کارشناس مایکروسافت استفاده از یک ماژول دست نویس از نوع RQ_LOG_REQUEST می باشد. Implements the ILogPlugin interface on top of IIS. ILogPlugin is a previous COM implementation that allowed customers to extend IIS logging. We do not not recommend extending IIS using this interface. Instead, customers should write a module and subscribe to the RQ_LOG_REQUEST notification.
تگ قابل پیکربندی :	system.webServer/httpLogging and system.applicationhost/sites/site/logFile/customLogPluginClsid
وابستگی :	ندارد.
اثرات حذف آن :	مسلما پلاگین‌های‌های این اینترفیس از کار می‌افتند که سیستم ODBC Logging هم جز آن است.
نام ماژول :	**CustomErrorModule**
توضیح :	پیاده سازی مدیریت خطاهای ویژه
تگ قابل پیکبرندی :	system.webServer/httpErrors
وابستگی :	None.
اثرات حذف آن :	در صورتی که خطایی از هسته باشد، نتیجه یک صفحه، با توضیح مختصری از خطا خواهد بود. در غیر این صورت اگر خطا از برنامه یا کامپوننتی باشد جزئیات خطا فاش خواهد شد

نام ماژول :	**HttpLoggingModule**
توضیح :	پیاده سازی سیستم logging استاندارد http.sys
تگ قابل پیکربندی :	system.applicationHost/log and

نام ماژول :	**HttpLoggingModule**
	system.webServer/httpLogging
وابستگی :	ندارد.
اثرات حذف آن :	از کار افتادن سیستم لاگ
نام ماژول :	**FailedRequestsTracingModule**
توضیح :	پیاده سازی سیستم ردیابی درخواست‌های ناموفق و اجرای قوانین، طبق پیکربندی
تگ قابل پیکربندی :	system.webServer/tracing and system.webServer/httpTracing
وابستگی :	ندارد.
اثرات حذف آن :	Tracing http requests will no longer work.

نام ماژول :	**RequestMonitorModule**
توضیح :	پیاده سازی IIS Run-time State and Control Interface یا به اختصار RSCA . به کاربران اجازه می‌دهد از اطلاعات، حین اجرا، کوئری بگیرند. مثل درخواست درحال اجرای جاری، آغاز به کار یا توقف وب سایت و دامنه‌های اپلیکیشن در حال اجرای جاری
تگ قابل پیکربندی :	ندارد.
وابستگی :	ندارد.
اثرات حذف آن :	ابزارهای مرتبط با این موضوع از کار می‌افتند
نام ماژول :	**CgiModule**
توضیح :	پیاده سازی CGI در سمت IIS
تگ قابل پیکبرندی :	system.webServer/cgi and system.webServer/isapiCgiRestriction

نام ماژول :	**RequestMonitorModule**
وابستگی :	ندارد.
اثرات حذف آن :	برنامه‌های CGI متوقف می‌شوند

نام ماژول :	**TracingModule**
توضیح :	پیاده سازی سیستم ردیابی ETW
تگ قابل پیکربندی :	system.webServer/httpTracing
وابستگی :	ندارد.
اثرات حذف آن :	باعث از کار افتادن سیستم مربوطه می‌شود
نام ماژول :	**ConfigurationValidationModule**
توضیح :	اعتبارسنجی تنظیمات برنامه ASP.Net که به حالت integrate انتقال یافته است
تگ قابل پیکربندی :	system.webServer/Validation
وابستگی :	ندارد.
اثرات حذف آن :	عدم اعتبارسنجی و در نتیجه عدم نمایش خطاها

MANAGED MODULES:

نام ماژول :	**OutputCache**
توضیح :	پیاده سازی output caching
تگ قابل پیکربندی :	system.web/caching/outputCache
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	عدم اجرای output cache

نام ماژول :	**OutputCache**
نام ماژول :	**Session**
توضیح :	مدیریت سشن ها
تگ قابل پیکربندی :	system.web/sessionState
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	سشن‌ها از دسترس خارج می‌شوند.

نام ماژول :	**WindowsAuthentication**
توضیح :	اینجا
تگ قابل پیکربندی :	system.web/authentication
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	این حالت قابل اجرا نخواهد بود
نام ماژول :	**FormsAuthentication**
توضیح :	اینجا
تگ قابل پیکربندی :	system.web/authentication
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	این حالت قابل اجرا نیست و کاربران مجوز دار هم نمی‌توانند به منابع محافظت شده دسترسی داشته باشند.

نام ماژول :	**DefaultAuthentication**
توضیح :	اطمینان از وجود شی Authentication در context مربوطه
تگ قابل پیکربندی :	system.web/authentication
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	اگر مد Forms authentication انتخاب شده باشد بر روی بعضی از کاربران ناشناس کار نخواهد کرد و رویداد DefaultAuthentication.OnAuthenticate اجرا نخواهد شد.
نام ماژول :	**RoleManager**
توضیح :	اینجا
تگ قابل پیکربندی :	
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	این قابلیت در دسترس نمی‌باشد

نام ماژول :	**UrlAuthorization**
توضیح :	اینجا
تگ قابل پیکربندی :	system.web/authorization.
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	باعث از کار افتادن asp.net authorization و فاش شدن بعضی اطلاعات و همچنین دیگر تهدیدات امنیتی

نام ماژول :	UrlAuthorization
نام ماژول :	AnonymousIdentification
توضیح :	اینجا
تگ قابل پیکربندی :	
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	The anonymous identification feature used by the ASP.NET Profile will not work.

نام ماژول :	Profile
توضیح :	اینجا
تگ قابل پیکربندی :	
وابستگی :	ManagedEngine module must be installed.
اثرات حذف آن :	ASP.Net Profile از کار خواهد افتاد
نام ماژول :	UrlMappingsModule
توضیح :	تبدیل یک Url واقعی به یک Url کاربرپسند
تگ قابل پیکبرندی :	
وابستگی :	نیاز به ManagedEngine .
اثرات حذف آن :	نگاشت Urlها صورت نمی‌گیرد

عنوان: آشنایی با ساختار IIS قسمت هشتم
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۰۹ ۲۳:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, IHandler

پس از بررسی مفاهیم، بهتر هست وارد یک کار عملی شویم. مثال مورد نظر، یک مثال از وب سایت شرکت مایکروسافت است که هنگام نمایش تصاویر، بر حسب پیکربندی موجود، یک پرچسب یا تگی را در گوشه‌ای از تصویر درج می‌کند. البته تصویر را ذخیره نمی‌کنیم و تگ را بر روی تصویر اصلی قرار نمی‌دهیم. تنها هنگام نمایش به کاربر، روی response خروجی آن را درج می‌کنیم.
قبلا ما در این مقاله به بررسی httpandler پرداخته‌ایم، ولی بهتر هست در این مثال کمی حالت پیشرفته‌تر آن‌را بررسی کنیم.

ابتدا اجازه دهید کمی قابلیت‌های فایل کانفیگ IIS را گسترش دهیم.
مسیر زیر را باز کنید:

```
%windir%\system32\inetsrv\config\schema
```

یک فایل xml را با نام imagecopyright.xml ساخته و تگ‌های زیر را داخلش قرار دهید:

احتمال زیاد دسترسی برای ویرایش این دایرکتوری به خاطر مراتب امنیتی با مشکل برخواهید خورد برای ویرایش این نکته امنیتی از اینجا یا به خصوص از اینجا کمک بگیرید.

```
<configSchema>

     <sectionSchema name="system.webServer/imageCopyright">
         <attribute name="enabled" type="bool" defaultValue="false" />
         <attribute name="message" type="string" defaultValue="Your Copyright Message" />
        <attribute name="color" type="string" defaultValue="Red"/>
   </sectionSchema>
 </configSchema>
```

با این کار ما یک شِما یا اسکیما را ایجاد کردیم که دارای سه خصوصیت زیر است:
enabled: آیا این هندلر فعال باشد یا خیر.
message: پیامی که باید به عنوان تگ درج شود.
color: رنگ متن که به طور پیش فرض قرمز رنگ است.

به هر کدام از تگ‌های بالا یک مقدار پیش فرض داده ایم تا اگر مقداردهی نشدند، ماژول طبق مقادیر پیش فرض کار خود را انجام هد.
بعد از نوشتن شما، لازم هست که آن را در فایل applicationhost.config نیز به عنوان یک section جدید در زیر مجموعه system.webserver معرفی کنیم:

```
<configSections>

...
   <sectionGroup name="system.webServer">
        <section name="imageCopyright"  overrideModeDefault="Allow"/>
...
   </sectionGroup>
</configSections>
```

تعریف کد بالا به شما اجازه میدهد تا در زیر مجموعه تگ system.webserver، برای هندلر خود تگ تعریف کنید. در کد بالا، شمای خود را بر اساس نام فایل مشخص می‌کنیم و خصوصیت overrideModeDefault، یک قفل گذار امنیتی برای تغییر محتواست. در صورتی که allow باشد هر کسی در هر مرحله‌ی دسترسی در سیستم و در هر فضای نامی، در فایل‌های وب کانفیگ می‌تواند به مقادیر این section دسترسی یافته و آن‌ها را تغییر دهد. ولی اگر با Deny مقدادهی شده باشد، مقادیر قفل شده و هیچ دسترسی برای تغییر آن‌ها وجود ندارد.
در مثال زیر ما به ماژول windows Authentication اجازه می‌دهیم که هر کاربری در هر سطح دسترسی به این section

دسترسی داشته باشد؛ از تمامی سایت‌ها یا اپلیکشین‌ها یا virtual directories موجود در سیستم و در بعضی موارد این گزینه باعث افزایش ریسک امنیتی می‌گردد.

```
<section name="windowsAuthentication" overrideModeDefault="Allow" />
```

در کد زیر اینبار ما دسترسی را بستیم و در تعاریف دامنه‌های دسترسی، دسترسی را فقط برای سطح مدیریت سایت AdministratorSite باز گذاشته‌ایم:

```
<location path="AdministratorSite" overrideMode="Allow">
   <security>
            <authentication>
                     <providers>
                <windowsAuthentication enabled="false">
                     </providers>
                        <add value="Negotiate" />
                        <add value="NTLM" />
 </location>
                </windowsAuthentication>
            </authentication>
    </security>
```

برای خارج نشدن بیش از اندازه از بحث، به ادامه تعریف هندلر می‌پردازیم. بعد از معرفی یک section برای هندلر خود، میتوانیم به راحتی تگ آن را در قسمت system.webserver تعریف کنیم. این کار می‌تواند از طریق فایل web.config سایت یا applicationhost.config صورت بگیرد یا میتواند از طریق ویرایش دستی یا خط فرمان appcmd معرفی شود؛ ولی در کل باید به صورت زیر تعریف شود:

```
<system.webServer>
     <imageCopyright />
 </system.webServer>
```

در کد بالا این تگ تنها معرفی شده است؛ ولی مقادیر آن پیش فرض می‌باشند. در صورتی که بخواهید مقادیر آن را تغییر دهید کد به شکل زیر تغییر می‌کند:

```
<system.webServer>
 <imageCopyright enabled="true" message="an example of www.dotnettips.info" color="Blue" />
 </system.webServer>
```

در صورتی که میخواهید از خط فرمان کمک بگیرید به این شکل بنویسید:

```
%windir%\system32\inetsrv\appcmd set config -section:system.webServer/imageCopyright /color:yellow
/message:"Dotnettips.info" /enabled:true
```

برای اطمینان از این که دستور شما اجرا شده است یا خیر، یک کوئری یا لیست از تگ مورد نظر در system.webserver بگیرید:

```
%windir%\system32\inetsrv\appcmd list config -section:system.webServer/imageCopyright
```

در این مرحله یک دایرکتوری برای پروژه تصاویر ایجاد کنید و در این مثال ما فقط تصاویر jpg را ذخیره می‌کنیم و در هنگام درج تگ، تصاویر jpg را هندل می‌کنیم؛ برای مثال ما:

```
c:\inetpub\mypictures
```

در این مرحله دایرکتوری ایجاد شده را به عنوان یک application معرفی می‌کنیم:

```
%windir%\system32\inetsrv\appcmd add app -site.name:"Default Web Site" -path:/mypictures -
physicalPath:%systemdrive%\inetpub\mypictures
```

و برای آن ماژول DirectoryBrowse را فعال می‌کنیم. برای اطلاعات بیشتر به مقاله قبلی که به تشریح وظایف ماژول‌ها پرداختیم رجوع کنید. فقط به این نکته اشاره کنم که اگر کاربر آدرس localhost/mypictures را درخواست کند، فایل‌های این قسمت را برای ما لیست می‌کند. برای فعال سازی، کد زیر را فعال می‌کنیم:

```
%windir%\system32\inetsrv\appcmd set config "Default Web Site/mypictures"  -section:directoryBrowse -
enabled:true
```

حال زمان این رسیده است تا کد نوشته و فایل cs آن را در مسیر زیر ذخیره کنیم:

```
c:\inetpub\mypictures\App_Code\imagecopyrighthandler.cs
```

هندل مورد نظر در زبان سی شارپ :

```
#region Using directives
using System;
using System.Web;
using System.Drawing;
using System.Drawing.Imaging;
using Microsoft.Web.Administration;
#endregion

namespace IIS7Demos
{
    public class imageCopyrightHandler : IHttpHandler
    {
        public void ProcessRequest(HttpContext context)
        {
            ConfigurationSection imageCopyrightHandlerSection =
                WebConfigurationManager.GetSection("system.webServer/imageCopyright");

            HandleImage(    context,
                            (bool)imageCopyrightHandlerSection.Attributes["enabled"].Value,
                            (string)imageCopyrightHandlerSection.Attributes["message"].Value,
                            (string)imageCopyrightHandlerSection.Attributes["color"].Value
                        );
        }

        void HandleImage(   HttpContext context,
                            bool enabled,
                            string copyrightText,
                            string color
                        )
        {
            try
            {
                string strPath = context.Request.PhysicalPath;
                if (enabled)
                {
                    Bitmap bitmap = new Bitmap(strPath);
                    // add copyright message
                    Graphics g = Graphics.FromImage(bitmap);
                    Font f = new Font("Arial", 50, GraphicsUnit.Pixel);
                    SolidBrush sb = new SolidBrush(Color.FromName(color));
                    g.DrawString(   copyrightText,
                                    f,
                                    sb,
                                    5,
                                    bitmap.Height - f.Height - 5
                                );
                    f.Dispose();
                    g.Dispose();
                    // slow, but good looking resize for large images
                    context.Response.ContentType = "image/jpeg";
                    bitmap.Save(
                                        context.Response.OutputStream,
                                        System.Drawing.Imaging.ImageFormat.Jpeg
                                    );
                    bitmap.Dispose();
                }
                else
                {
                    context.Response.WriteFile(strPath);
                }
            }
            catch (Exception e)
            {
                context.Response.Write(e.Message);
            }
        }

        public bool IsReusable
        {
            get { return true; }
        }
    }
}
```

در خط WebConfigurationManager.GetSection، در صورتیکه تگ imagecopyright تعریف شده باشد، همه اطلاعات این تگ را از فایل کانفیگ بیرون کشیده و داخل شیء imageCopyrightHandlerSection از نوع ConfigurationSection قرار می‌دهیم.
سپس اطلاعات هر سه گزینه را خوانده و به همراه context (اطلاعات درخواست) به تابع handleimage که ما آن را نوشته ایم ارسال می‌کنیم. کار این تابع درج تگ می‌باشد.
در خطوط اولیه تابع، ما آدرس فیزیکی منبع درخواست شده را به دست آورده و در صورتیکه مقدار گزینه enable با true مقدار دهی شده باشد، آن را به شی bitmap نسبت می‌دهیم و با استفاده از دیگر کلاس‌های گرافیکی، تگ مورد نظر را با متن و رنگ مشخص شده ایجاد می‌کنیم. در نهایت شیء bitmap را ذخیره و نوع خروجی response را از نوع image/jpeg تعریف می‌کنیم تا مرورگر بداند که خروجی ما یک تصویر است. ولی در صورتی که enabled با false مقداردهی شده باشد، همان تصویر اصلی را بدون درج تگ ارسال می‌کنیم.
فضای نام Microsoft.Web.Administration برای اجرای خود نیاز دارد تا اسمبلی آن رفرنس شود. برای اینکار به درون دایرکتوری mypictures رفته و در داخل فایل web.config که بعد از تبدیل این دایرکتوری به اپلیکیشن ایجاد شده بنویسید:

```
<system.web>
     <compilation>
       <assemblies>
         <add assembly="Microsoft.Web.Administration, Version=7.0.0.0,
 Culture=neutral, PublicKeyToken=31bf3856ad364e35, processorArchitecture=MSIL"/>
      </assemblies>
    </compilation>
 </system.web>
```

در صورتی که کلاس خود را کامپایل کنید می‌توانید آن را داخل پوشه‌ی Bin به جای App_Code قرار دهید و نیاز به رفرنس کرده اسمبلی Microsoft.Web.Administration نیز ندارید.
در آخرین مرحله فقط باید به IIS بگویید که تنها فایل‌های jpg را برای این هندلر، هندل کن. این کار را از طریق خط فرمان نجام می‌دهیم:

```
appcmd set config "Default Web Site/mypictures/" -section:handlers
/+[name='JPGimageCopyrightHandler',path='*.jpg',verb='GET',type='IIS7Demos.imageCopyrightHandler']
```

هندلر مورد نظر تنها برای این اپلیکیشن و در مسیر mypicture فعال شده و در قسمت name، یک نام اختیاری بدون فاصله و unique بر می‌گزینیم. در قسمت path نوع فایل‌هایی را که نیاز به هندل هست، مشخص کردیم و در قسمت verb گفته‌ایم که تنها برای درخواست‌های نوع GET، هندلر را اجرا کن و در قسمت type هم که اگر مقاله httphandler را خوانده باشید می‌دانید که به معرفی هندلر می‌پردازیم؛ اولی نام فضای نام هست و بعد از . نام کلاس، که در اینجا می‌شود :

```
'IIS7Demos.imageCopyrightHandler
```

الان همه چیز برای اجرا آماده است و فقط یک مورد برای احتیاط الزامی است و آن هم این است که پروسه‌های کارگر، ممکن است از قبل در حال اجرا بوده باشند و هنوز شمای جدید ما را شناسایی نکرده باشند، برای همین باید آن‌ها را با تنظیمات جدیدمان آشنا کنیم تا احیانا برایمان استثناء صادر نشود:

```
appcmd recycle AppPool DefaultAppPool
```

کارمان تمام شده ، چند تصویر داخل دایرکتوری قرار داده و درخواست تصاویر موجود را بدهید تا تگ را ببینید:

فعلا تا بدین جا کافی است. در قسمت آینده این هندلر را کمی بیشتر توسعه خواهیم داد.

عنوان: **ایجاد کوکی با jcookie**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۰/۲۴ ۹:۵۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, Cookie

همانطور که از نامش پیداست jcookie یک پلاگین jquery است. این پلاگین به شما این اجازه را می‌دهد تا هر نوع داده ای را که مایل هستید از قبیل رشته‌ها، آرایه‌ها و object را در قالب json با رمزگذاری base 64 ذخیره نمایید. استفاده از این رمزگذاری باعث کوچکتر شدن حجم کوکی تا 70 درصد می‌شود. در این مقاله شما یاد می‌گیرید که چطور برای ذخیره و بازیابی کوکی از آن استفاده کرده و چگونه در یک زبان سمت سرور، مثل سی شارپ نیز کوکی مورد نظر را با همان فرمت بخوانید. جهت دانلود فایل jcookie به اینجا مراجعه کنید. **ذخیره کوکی**

برای ساخت یک کوکی به روش زیر اقدام می‌کنیم. استفاده از $.jCookies دو خاصیت به نام‌های نام کوکی و مقدار کوکی را name & Value در دسترس ما می‌گذارد:

```
var d = new Date();

                $.jCookies({
                    name: 'dotnettips.info',
                    value: { Title: 'ساخت کوکی با jcookie', Author: 'علی یگانه مقدم', Seen:
d.getDate(), Favorite: true }
                });
```

همانطور که می‌بینید ذخیره اطلاعات توسط jcookie بسیار ساده و راحت بوده و هر نوع داده ای در آن به راحتی قابل ذخیره سازیست. برای مثال می‌توانید اطلاعات یک کلاس را خیلی راحت و سریع با آن ذخیره کنید. به طور پیش فرض تاریخ انقضای کوکی 27 روز بعد از ایجاد آن می‌باشد. در صورتی که تمایل دارید این تاریخ را تغییر دهید یکی از خاصیت‌های seconds,minutes,hours و days در دسترس شماست و مقادیری که جلوی آن‌ها به کارگرفته می‌شود باید نوع صحیح بوده و در صورتی که مقدار نامعتبر وارد شود خاصیت مورد نظر نادیده گرفته می‌شود.

```
$.jCookies ({ name : 'User', value : { username : 'Bob' , level : 5 }, minutes : 60 });
```

برای تغییر پیش فرض‌های ساخت کوکی مانند انقضای 27 روز به عدد پیش فرض خودتان فایل jcookies.js را باز کرده و تنظیمات پیش فرض آن را تغییر بدهید. برای تغییر دنبال کد زیر بگردید:

```
$.jCookies.defaults =
{
name : '',
value : '',
days : 27
}
```

بازیابی کوکی

برای بازیابی کوکی مجددا از $.jCookies استفاده می‌شود ولی تنها باید یک خاصیت get که نام کوکی هست را بنویسید:

```
var values = $.jCookies ({ get: 'dotnettips.info' });
```

در صورتی که نام کوکی‌ای که درخواست کرده اید وجود نداشته باشد یا اینکه تاریخ انقضای آن سر رسیده است و از سیستم کلاینت حذف شده است یا اینکه هنگام درخواست کوکی با خطایی مواجه شده باشد، مقدا برگشتی false خواهد بود و اگر نیاز دارید که بدانید آیا نوع برگشتی false به خاطر خطا بوده است یا خیر یک خاصیت نوع بول به نام error هم اضافه می‌شود:

```
var values = $.jCookies({ get: 'Rutabaga', error: true });
```

در صورتی که خطایی داده شود response مقدار values در مرورگر کروم به شکل زیر خواهد بود. در هر مرورگر نحوه نمایش خطا می‌تواند متفاوت باشد.

```
Error : {
            arguments : undefined,
            message : "Invalid base64 data",
            stack : "–",
```

```
            type : undefined
        }
```

بازیابی همه کوکی ها

در صورتی که به خاصیت get مقدار * را بدهید تمامی کوکی‌ها برگشت داده خواهند شد و به صورت آرایه ای از نام کوکی‌ها در دسترسی خواهند بود:

```
var values = $.jCookies({ get: '*' });
                alert(values["dotnettips.info"].Title);
                alert(values["data2"].Title);
```

حذف کوکی

نحوه کدنویسی حذف کوکی هم دقیقا مشابه خواندن کوکی است؛ با این تفاوت که به جای استفاده از خاصیت get از خاصیت erase استفاده می‌کنیم و با دادن نام کوکی به این خاصیت، کوکی حذف خواهد شد:

```
var value = $.jCookies({ erase: 'dotnettips.info' });
```

در صورتی که کوکی وجود داشته باشد، آن را حذف کرده و مقدار true را برگشت خواهد زد و در صورتی که کوکی وجود نداشته باشد مقدار false را بر میگرداند.

بازیابی کوکی در سمت سرور با سی شارپ

در این روش ما ابتدا با همان دستور معمولی دات نت یعنی page.request.cookie درخواست دریافت کوکی را می‌دهیم ولی از آنجا که در jcookie دو عمل روی داده‌ها صورت گرفته است باید دو کار اضافه‌تر را انجام داد:

برگشت داده‌ها از حالت رمزگذاری base64

داده‌ها در فرمت json هستند و باید به حالتی قابل استفاده در محیط شی گرا تبدیل شوند.

برای بازگردانی از حالت base64 از کلاس و متد Convert.FromBase64String در فضای نام system.convert استفاده می‌کنیم که آرایه ای از نوع بایت را بر میگرداند و از Encoding.UTF8.GetString هم برای decode کردن آرایه به نوع رشته استفاده می‌کنیم. تا به اینجای کار داده‌های ما به صورت یک json خوانا با فرمت string درآمده است. برای دسترسی به داده‌های موجود در این فرمت باید آن‌ها را Deserialize کنیم که این کار را از طریق کلاس JavaScriptSerializer در فضای نام System.Web.Script.Serialization انجام می‌دهیم و از کلاس دیکشنری برای ذخیره داده‌های برگشتی استفاده می‌کنیم که نوع string برای نام خاصیت و نوع آبجکت برای ذخیره مقدار خاصیت خواهد بود. یعنی برای بازگردانی اولین مثال بالا باید داده‌های در نوع دیکشنری به صورت زیر لیست شوند

Title	ایجاد کوکی با jcookie
Author	علی یگانه مقدم
Seen	2015/1/14
Favorite	true

```
byte[] from64 = Convert.FromBase64String(Page.Request.Cookies["dotnettips.info"].Value);
            string json = Encoding.UTF8.GetString(from64);
            Dictionary<string, object> article =new
JavaScriptSerializer().Deserialize<Dictionary<string, object>>(json);
            Page.Response.Write("Title: "+ (string)article["Title"]);
```

پشتیبانی از یونیکد

موقعی که من اولین مثال بالا را نوشتم و مقادیر را به صورت فارسی وارد کردم متوجه شدم که رشته‌های یونیکد را انکود نمی‌کند و در نتیجه زبان فارسی در آن پشتیبانی نمی‌شود. برای همین تغییراتی در فایل js ایجاد کرده و عبارت value قبل از تبدیل به base64 را به صورت utf-16 انکود کردم و در هنگام خواندن کوکی هم به صورت utf-16 دیکود کردم و مشکل زبان فارسی هم در این حالت حل شد. البته کدی که اضافه کردم قابلیت‌های انکودینگ بیشتری هم دارد.

فقط تنها مورد این هست که برای خواندن کوکی در سمت سرور باید یک تغییر کوچک یک کلمه ای بدهیم؛ باید کلمه UTF8 را به

Unicode که می‌شود همان UTF-16 در کد تغییر دهیم، که به کد زیر تغییر خواهد یافت:

```
byte[] from64 = Convert.FromBase64String(Page.Request.Cookies["dotnettips.info"].Value);
            string json = Encoding.Unicode.GetString(from64);
            Dictionary<string, object> article =new
JavaScriptSerializer().Deserialize<Dictionary<string, object>>(json);
            Page.Response.Write("Title: "+ (string)article["Title"]);
```

برای دریافت jcookie با پشتیبانی از زبان فارسی به اینجا مراجعه کنید.

کدهای بالا در فایل زیر قرار گرفته اند. jcookie.zip

عنوان: **کار با وب سرویس جاوایی تشخیص ایمیل‌های موقتی در دات نت**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۳/۱۰/۲۷ ۱۲:۴۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, Security, WCF, xml, Validation, Java

یکی از وب سرویس‌های سایت name api ، امکان تشخیص موقتی بودن ایمیل مورد استفاده‌ی جهت ثبت نام در یک سایت را فراهم می‌کند. آدرس WSDL آن نیز در اینجا قرار دارد. اگر مطابق معمول استفاده از سرویس‌های وب در دات نت، بر روی ارجاعات پروژه کلیک راست کرده و گزینه‌ی Add service refrence را انتخاب کنیم و سپس آدرس WSDL یاد شده را به آن معرفی کنیم، بدون مشکل ساختار این وب سرویس دریافت و برای استفاده‌ی از آن به یک چنین کدی خواهیم رسید:

```
var client = new SoapDisposableEmailAddressDetectorClient();

var context = new soapContext
{
    //todo: get your API key here: http://www.nameapi.org/en/register/
    apiKey = "test"
};
var result = client.isDisposable(context, "DaDiDoo@mailinator.com");

if (result.disposable.ToString() == "YES")
{
    Console.WriteLine("YES! It's Disposable!");
}
```

متد isDisposable ارائه شده‌ی توسط این وب سرویس، دو پارامتر context که در آن باید API Key خود را مشخص کرد و همچنین آدرس ایمیل مورد بررسی را دریافت می‌کند. اگر به همین ترتیب این پروژه را اجرا کنید، با خطای Bad request از طرف سرور متوقف خواهید شد:

```
Additional information: The remote server returned an unexpected response: (400) Bad Request.
```

اگر به خروجی این وب سرویس در فیدلر مراجعه کنیم، چنین شکلی را خواهد داشت:

```
<html><head><title>Bad Request</title></head><body><h1>Bad Request</h1><p>No api-key
provided!</p></body></html>
```

عنوان کرده‌است که api-key را، در درخواست وب خود ذکر نکرده‌ایم.

اگر همین وب سرویس را توسط امکانات سایت http://wsdlbrowser.com بررسی کنید، بدون مشکل کار می‌کند. اما تفاوت در کجاست؟

خروجی ارسالی به سرور، توسط سایت http://wsdlbrowser.com به این شکل است:

```
<SOAP-ENV:Envelope xmlns:SOAP-ENV="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:ns1="http://disposableemailaddressdetector.email.services.v4_0.soap.server.nameapi.org/">
  <SOAP-ENV:Body>
    <ns1:isDisposable>
      <context>
        <apiKey>test</apiKey>
      </context>
      <emailAddress>sdsdg@site.com</emailAddress>
    </ns1:isDisposable>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

و نمونه‌ی تولید شده‌ی توسط WCF (امکان Add service reference در حقیقت یک WCF Client را ایجاد می‌کند) به صورت زیر می‌باشد:

```
<s:Envelope xmlns:s="http://schemas.xmlsoap.org/soap/envelope/">
  <s:Body xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
     <isDisposable
```

```
xmlns="http://disposableemailaddressdetector.email.services.v4_0.soap.server.nameapi.org/">
          <context xmlns=""><apiKey>test</apiKey></context>
          <emailAddress xmlns="">DaDiDoo@mailinator.com</emailAddress>
      </isDisposable>
  </s:Body>
</s:Envelope>
```

از لحاظ اصول XML، خروجی تولیدی توسط WCF هیچ ایرادی ندارد. از این جهت که نام فضای نام مرتبط با http://schemas.xmlsoap.org/soap/envelope/ را به s تنظیم کرده‌است و سپس با استفاده از این نام، Envelope را تشکیل داده‌است. اما ... این وب سرور جاوایی دقیقا با نام SOAP-ENV کار می‌کند و فضای نام ns1 بعدی آن. کاری هم به اصول XML ندارد که باید بر اساس نام xmlns ذکر شده، کار Parse ورودی دریافتی صورت گیرد و نه بر اساس یک رشته‌ی ثابت از پیش تعیین شده. بنابراین باید راهی را پیدا کنیم تا بتوان این s را تبدیل به SOAP-ENV کرد.

برای این منظور به سه کلاس ذیل خواهیم رسید:

```
public class EndpointBehavior : IEndpointBehavior
{
    public void AddBindingParameters(ServiceEndpoint endpoint, BindingParameterCollection
bindingParameters)
    { }

    public void ApplyDispatchBehavior(ServiceEndpoint endpoint, EndpointDispatcher endpointDispatcher)
    { }

    public void Validate(ServiceEndpoint endpoint)
    { }

    public void ApplyClientBehavior(ServiceEndpoint endpoint, ClientRuntime clientRuntime)
    {
        clientRuntime.MessageInspectors.Add(new ClientMessageInspector());
    }
}

public class ClientMessageInspector : IClientMessageInspector
{
    public void AfterReceiveReply(ref Message reply, object correlationState)
    { }

    public object BeforeSendRequest(ref Message request, System.ServiceModel.IClientChannel channel)
    {
        request = new MyCustomMessage(request);
        return request;
    }
}

/// <summary>
/// To customize WCF envelope and namespace prefix
/// </summary>
public class MyCustomMessage : Message
{
    private readonly Message _message;

    public MyCustomMessage(Message message)
    {
        _message = message;
    }

    public override MessageHeaders Headers
    {
        get { return _message.Headers; }
    }

    public override MessageProperties Properties
    {
        get { return _message.Properties; }
    }

    public override MessageVersion Version
    {
        get { return _message.Version; }
    }

    protected override void OnWriteStartBody(XmlDictionaryWriter writer)
    {
```

```
        writer.WriteStartElement("Body", "http://schemas.xmlsoap.org/soap/envelope/");
    }

    protected override void OnWriteBodyContents(XmlDictionaryWriter writer)
    {
        _message.WriteBodyContents(writer);
    }

    protected override void OnWriteStartEnvelope(XmlDictionaryWriter writer)
    {
        writer.WriteStartElement("SOAP-ENV", "Envelope", "http://schemas.xmlsoap.org/soap/envelope/");
        writer.WriteAttributeString("xmlns", "ns1", null, value:
"http://disposableemailaddressdetector.email.services.v4_0.soap.server.nameapi.org/");
    }
}
```

که پس از تعریف client به نحو ذیل معرفی می‌شوند:

```
var client = new SoapDisposableEmailAddressDetectorClient();
client.Endpoint.Behaviors.Add(new EndpointBehavior());
```

توسط EndpointBehavior سفارشی، می‌توان به متد **Envelope** OnWriteStart دسترسی یافت و سپس s آن‌را با SOAP-ENV درخواستی این وب سرویس جایگزین کرد. اکنون اگر برنامه را اجرا کنید، بدون مشکل کار خواهد کرد و دیگر پیام یافت نشدن API-Key را صادر نمی‌کند.

کدهای کامل این مثال را از اینجا می‌توانید دریافت کنید.

عنوان: **کار با اسکنر در برنامه های تحت وب (قسمت اول)**
نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۳/۱۱/۱۷ ۰:۱۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, WCF, TWAIN, Cross-Origin Request

در اکثر برنامه‌های سازمانی، مثل برنامه‌های مدیریت آرشیو اسناد، همواره این نیاز جزو خواسته‌های کاربران بوده که بتوانند به صورت مستقیم و از طریق تنها یک کلیک، تصویر مورد نظر را اسکن کرده و به صورت خودکار در برنامه وارد کنند؛ یعنی بدون اینکه نیاز باشد با استفاده از یک برنامه دیگر ابتدا تصویر را اسکن کرده و سپس در فرم وب، فایل اسکن شده را Browse کنند.
این نیاز اساسا به معنی دسترسی به سخت افزار کاربر از طریق مرورگر می‌باشد که به دلایل متعددی امکان پذیر نیست! مشکلات امنیتی ایجاد شده در این راه بسیار جدی است. اما خوشبختانه راههایی برای رسیدن به این هدف وجود دارند:

1- **ActiveX** : که به صورت native فقط در IE پشتیبانی می‌شود (در نسخه‌های جدیدتر IE نیاز به بروز رسانی پلاگین ActiveX controls می‌باشد) و برای استفاده‌ی از آن در مرورگرهای Firefox و Chrome هم باید از پلاگین‌های جانبی روی هر مرورگر استفاده کرد که مثلا برای اجرای بر روی Firefox، باید افزونه‌ی IE Tab را نصب کرد که تنها کارش این است که سایت را از طریق موتور IE در پنجره‌ی فایرفاکس اجرا کند! که البته این مورد مثل این می‌ماند که سایت در IE باز شده باشد که ممکن است زیاد خوشایند نباشد؛ در غیر اینصورت باید پروژه را از اول بر مبنای اجرای بر روی IE طراحی و پیاده سازی کرد. در ضمن از مشکلات اجرای پلاگین نوشته شده توسط برنامه نویس در IE و نسخه‌های مختلف آن هم چشم پوشی می‌کنیم. یکی از مزیت‌های این روش نسبت به بقیه این هست که دانلود و نصب پلاگین مورد نیاز به صورت خودکار و توسط مرورگر انجام می‌شود.

2- **استفاده از یک کامپوننت جانبی** : مثل کامپوننت‌های LEADTOOLS که ابزارهای متنوع و SDKهای بسیار قدرتمندی را برای اینکار و کارهای دیگر، مانند کار با اسکن تصاویر مغزی، پردازش تصویر، بارکد خوان‌ها، مدیریت اسناد و ... فراهم کرده است. قیمت این ابزار بسیار زیاد است و در برخی موارد امکانات فراهم آورده بسیار بیشتر است از خواسته‌ی ما. این ابزار، هم در HTML & Javascript و هم در DotNet قابل استفاده است و مستندات نسبتا خوبی هم در این زمینه ارائه کرده است. همچنین کامپوننت Dynamsoft که باز هم غیر رایگان هست و قیمت بالایی نیز دارد.

اگر روال کار کامپوننت‌های بالا را مورد بررسی قرار دهید (از طریق اجرای Demo ها، اینجا و اینجا) متوجه خواهید شد که هر دو نیاز به نصب یک سرویس یا App سمت کلاینت جهت اجرای دستورات خود دارند. پس می‌شود اینطور نتیجه گرفت که انجام اینکار بدون اینکه چیزی سمت کاربر نصب شود، ممکن نیست و در هر دو، لینک نصب فایل exe سرویس برای دانلود قرار داده شده است. بر این اساس به راه حل سومی خواهیم رسید که خودمان این سرویس را جهت تعامل با اسکنر سمت کاربر طراحی و پیاده سازی نماییم.

اما چطور ممکن است که با اجرای یک فایل exe سمت کاربر (با این فرض که کاربر در یک دامین مطمئن قرار دارد و می‌شود درخواست نصب سرویس را نمود) این امکان را برای کاربر فراهم نمود که با یک کلیک در مرورگر، اسکنر به صورت خودکار اسکن را آغاز کرده و سپس تصویر حاصل را به یکی از کنترلرهای ما در سمت سرور ارسال نماید؟

برای اینکار ما با دو صورت مساله مواجه هستیم؛ اول اینکه چطور تصویر را سمت کاربر اسکن کنیم و دوم اینکه چطور این تصویر را به سرور ارسال نماییم!
برای مساله‌ی اول از کتابخانه (Windows Image Acquisition (WIA استفاده خواهیم نمود که این کتابخانه به ما این امکان را میدهد تا با سخت افزارهایی که از TWAIN پشتیبانی می‌کنند، بتوانیم ارتباط برقرار نماییم.

برای مساله‌ی دوم هم نیاز به پیاده سازی یک WCF Service و اجرای آن (هاست کردن) در سمت کلاینت داریم. در واقع با صدا زدن متدهای این سرویس، از کتابخانه‌ی بالا استفاده کرده و اسکن را انجام می‌دهیم.

ادامه دارد...

عنوان: **آشنایی با ساختار IIS قسمت نهم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۱/۱۷ ۱:۴۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, Strong Name, UI, GAC

در قسمت قبلی ما یک هندلر ایجاد کردیم و درخواست‌هایی را که برای فایل jpg و به صورت GET ارسال میشد، هندل می‌کردیم و تگی را در گوشه‌ی تصویر درج و آن را در خروجی نمایش میدادیم. در این مقاله قصد داریم که کمی هندلر مورد نظر را توسعه دهیم و برای آن یک UI یا یک رابط کاربری ایجاد نماییم. برای توسعه دادن ماژولها و هندلرها ما یک dll نوشته و باید آن را در GAC که مخفف عبارت Global Assembly Cache ریجستر کنیم.

جهت اینکار یک پروژه از نوع class library ایجاد کنید. فایل class1.cs را که به طور پیش فرض ایجاد می‌شود، حذف کنید و رفرنس‌های Microsoft.Web.Management.dll و Microsoft.Web.Administration.dll را از مسیر زیر اضافه کنید:

```
\Windows\system32\inetsrv
```

اولین رفرنس شامل کلاس‌هایی است که جهت ساخت ماژول‌ها برای کنسول IIS مورد نیاز است و دومی هم برای خواندن پیکربندی‌های نوشته شده مورد استفاده قرار می‌گیرد.
برای طراحی UI بر پایه winform باید رفرنس‌های System.Windows.Forms.dll و System.Web.dll را از سری اسمبلی‌های دات نت نیز اضافه کنیم و در مرحله‌ی بعدی جهت ایجاد امضاء یا strong name (^ و ^) به خاطر ثبت در GAC پروژه را انتخاب و وارد Properties پروژه شوید. در تب signing گزینه sign the assembly را تیک زده و در لیست باز شده گزینه new را انتخاب نمایید و نام imageCopyrightUI را به آن نسبت داده و گزینه تعیین کلمه عبور را غیرفعال کنید و تایید و تمام. الان باید یک فایل snk مخفف strong name key ایجاد شده باشد تا بعدا با استفاده از این کلید dll ایجاد شده را در GAC ریجستر کنیم.

در مرحله بعدی در تب Build Events کد زیر را در بخش Post-build event command line اضافه کنید. این کد باعث می‌شود بعد از هر بار کامپایل پروژه، به طور خودکار در GAC ثبت شود:

```
call "%VS80COMNTOOLS%\vsvars32.bat" > NULL
gacutil.exe /if "$(TargetPath)"
```

نکته:در صورتی که از VS2005 استفاده می‌کنید در تب Debug در قسمت Start External Program مسیر زیر را قرار بدهید. اینکار برای تست و دیباگینگ پروژه به شما کمک خواهد کرد. این تنظیم شامل نسخه‌های اکسپرس نمی‌شود.

```
\windows\system32\inetsrv\inetmgr.exe
```

بعد از پایان اینکار پروژه را Rebuild کنید. با اینکار dll در GAC ثبت می‌شود. استفاده از سوییچ‌های if به طور همزمان در درستور gacutil به معنی این هست که اگر اولین بار است نصب می‌شود، پس با سوییچ i نصب کن. ولی اگر قبلا نصب شده است نسخه جدید را به هر صورتی هست جایگزین قبلی کن یا همان reinstall کن.

ساخت یک Module Provider

رابط‌های کاربری IIS همانند هسته و کل سیستمش، ماژولار و قابل خصوصی سازی است. رابط کاربری، مجموعه‌ای از ماژول هایی است که میتوان آن‌ها را حذف یا جایگزین کرد. تگ ورودی یا معرفی برای هر UI یک module provider است. خیلی خودمانی، تگ ماژول پروایدر به معرفی یک UI در IIS می‌پردازد. لیستی از module providerها را می‌توان در فایل زیر در تگ بخش <modules> پیدا کرد.

```
%windir%\system32\inetsrv\Administration.config
```

در اولین گام یک کلاس را به اسم imageCopyrightUIModuleProvider.cs ایجاد کرده و سپس آن‌را به کد زیر، تغییر می‌دهیم. کد زیر با استفاده از ModuleDefinition یک نام به تگ Module Provider داده و کلاس imageCopyrightUI را که بعدا تعریف می‌کنیم، به عنوان مدخل entry رابط کاربری معرفی کرده:

```
using System;
using System.Security;
using Microsoft.Web.Management.Server;

namespace IIS7Demos
{
    class imageCopyrightUIProvider : ModuleProvider
    {
        public override Type ServiceType
```

```
        {
            get { return null; }
        }

        public override ModuleDefinition GetModuleDefinition(IManagementContext context)
        {
            return new ModuleDefinition(Name, typeof(imageCopyrightUI).AssemblyQualifiedName);
        }

        public override bool SupportsScope(ManagementScope scope)
        {
            return true;
        }
    }
}
```

با ارث بری از کلاس module provider، سه متد بازنویسی می‌شوند که یکی از آن ها SupportsScope هست که میدان عمل پروایدر را مشخص می‌کند، مانند اینکه این پروایدر در چه میدانی باید کار کند که می‌تواند سه گزینه‌ی server,site,application باشد. در کد زیر مثلا میدان عمل application انتخاب شده است ولی در کد بالا با برگشت مستقیم true، همه‌ی میدان را جهت پشتیبانی از این پروایدر اعلام کردیم.

```
public override bool SupportsScope(ManagementScope scope)
        {
            return (scope == ManagementScope.Application) ;
        }
```

حالا که پروایدر (معرف رابط کاربری به IIS) تامین شده، نیاز است قلب کار یعنی ماژول معرفی گردد. اصلی‌ترین متدی که باید از اینترفیس ماژول پیاده سازی شود متد initialize است. این متد جایی است که تمام عملیات در آن رخ می‌دهد. در کلاس زیر imageCopyrightUI ما به معرفی مدخل entry رابط کاربری می‌پردازیم. در سازنده‌های این متد، پارامترهای نام، صفحه رابط کاربری وتوضیحی در مورد آن است. تصویر کوچک و بزرگ جهت آیکن سازی (در صورت عدم تعریف آیکن، چرخ دنده نمایش داده می‌شود) و توصیف‌های بلندتر را نیز شامل می‌شود.

```
internal class imageCopyrightUI : Module
    {
        protected override void Initialize(IServiceProvider serviceProvider, ModuleInfo moduleInfo)
        {
            base.Initialize(serviceProvider, moduleInfo);
            IControlPanel controlPanel = (IControlPanel)GetService(typeof(IControlPanel));
            ModulePageInfo modulePageInfo = new ModulePageInfo(this, typeof(imageCopyrightUIPage),
"Image Copyright", "Image Copyright",Resource1.Visual_Studio_2012,Resource1.Visual_Studio_2012);
            controlPanel.RegisterPage(modulePageInfo);
        }
    }
```

شیء ControlPanel مکانی است که قرار است آیکن ماژول نمایش داده شود. شکل زیر به خوبی نام همه قسمت‌ها را بر اساس نام کلاس و اینترفیس آن‌ها دسته بندی کرده است:

پس با تعریف این کلاس جدید ما روی صفحه‌ی کنترل پنل IIS، یک آیکن ساخته و صفحه‌ی رابط کاربری را به نام imageCopyrightUIPage، در آن ریجستر می‌کنیم. این کلاس را پایینتر شرح داده‌ایم. ولی قبل از آن اجازه بدهید تا انواع کلاس هایی را که برای ساخت صفحه کاربرد دارند، بررسی نماییم. در این مثال ما با استفاده از پایه‌ای‌ترین کلاس، ساده‌ترین نوع صفحه ممکن را خواهیم ساخت. 4 کلاس برای ساخت یک صفحه وجود دارند که بسته به سناریوی کاری، شما یکی را انتخاب می‌کنید.

ModulePage	شامل اساسی‌ترین متدها و سورس‌ها شده و هیچگونه رابط کاری ویژه‌ای را در اختیار شما قرار نمی‌دهد. تنها یک صفحه‌ی خام به شما می‌دهد که می‌توانید از آن استفاده کرده یا حتی با ارث بری از آن، کلاس‌های جدیدتری را برای ساخت صفحات مختلف و ویژه‌تر بسازید. در حال حاضر که هیچ کدام از ویژگی‌های IIS فعلی از این کلاس برای ساخت رابط کاربری استفاده نکرده‌اند.
ModuleDialogPage	یک صفحه شبیه به دیالوگ را ایجاد می‌کند و شامل دکمه‌های Apply و Cancel میشود به همراه یک سری متدهای اضافی‌تر که اجازه‌ی override کردن آنها را دارید. همچنین یک سری از کارهایی چون refresh و از این دست عملیات خودکار را نیز انجام میدهد. از نمونه رابطهایی که از این صفحات استفاده می‌کنند میتوان machine key و management service را اسم برد.

ModulePage	شامل اساسی‌ترین متدها و سورس‌ها شده و هیچگونه رابط کاری ویژه‌ای را در اختیار شما قرار نمی‌دهد. تنها یک صفحه‌ی خام به شما می‌دهد که می‌توانید از آن استفاده کرده یا حتی با ارث بری از آن، کلاس‌های جدیدتری را برای ساخت صفحات مختلف و ویژه‌تر بسازید. در حال حاضر که هیچ کدام از ویژگی‌های IIS فعلی از این کلاس برای ساخت رابط کاربری استفاده نکرده‌اند.
ModulePropertiesPage	این صفحه یک رابط کاربری را شبیه پنجره property که در ویژوال استادیو وجود دارد، در دسترس شما قرار میدهد. تمام عناصر آن در یک حالت گرید grid لیست می‌شوند. از نمونه‌های موجود میتوان به CGI,ASP.Net Compilation اشاره کرد.
ModuleListPage	این کلاس برای مواقعی کاربرد دارد که شما قرار است لیستی از آیتم‌ها را نشان دهید. در این صفحه شما یک ListView دارید که میتوانید عملیات جست و جو، گروه بندی و نحوه‌ی نمایش لیست را روی آن اعمال کنید.

در این مثال ما از اولین کلاس نامبرده که پایه‌ی همه کلاس هاست استفاده می‌کنیم. کد زیر را در کلاسی به اسم imageCopyrightUIPage می‌نویسیم:

```
public sealed class imageCopyrightUIPage : ModulePage
    {
        public string message;
        public bool featureenabled;
        public string color;

        ComboBox _colCombo = new ComboBox();
        TextBox _msgTB = new TextBox();
        CheckBox _enabledCB = new CheckBox();

        public imageCopyrightUIPage()
        {
            this.Initialize();
        }


        void Initialize()
        {
            Label crlabel = new Label();
            crlabel.Left = 50;
            crlabel.Top = 100;
            crlabel.AutoSize = true;
            crlabel.Text = "Enable Image Copyright:";
            _enabledCB.Text = "";
            _enabledCB.Left = 200;
            _enabledCB.Top = 100;
            _enabledCB.AutoSize = true;

            Label msglabel = new Label();
            msglabel.Left = 150;
            msglabel.Top = 130;
            msglabel.AutoSize = true;
            msglabel.Text = "Message:";
            _msgTB.Left = 200;
            _msgTB.Top = 130;
            _msgTB.Width = 200;
            _msgTB.Height = 50;

            Label collabel = new Label();
            collabel.Left = 160;
            collabel.Top = 160;
            collabel.AutoSize = true;
            collabel.Text = "Color:";
            _colCombo.Left = 200;
```

```
        _colCombo.Top = 160;
        _colCombo.Width = 50;
        _colCombo.Height = 90;
        _colCombo.Items.Add((object)"Yellow");
        _colCombo.Items.Add((object)"Blue");
        _colCombo.Items.Add((object)"Red");
        _colCombo.Items.Add((object)"White");

        Button apply = new Button();
        apply.Text = "Apply";
        apply.Click += new EventHandler(this.applyClick);
        apply.Left = 200;
        apply.AutoSize = true;
        apply.Top = 250;

        Controls.Add(crlabel);
        Controls.Add(_enabledCB);
        Controls.Add(collabel);
        Controls.Add(_colCombo);
        Controls.Add(msglabel);
        Controls.Add(_msgTB);
        Controls.Add(apply);
    }

    public void ReadConfig()
    {
        try
        {
            ServerManager mgr;
            ConfigurationSection section;
            mgr = new ServerManager();
            Configuration config =
            mgr.GetWebConfiguration(
                   Connection.ConfigurationPath.SiteName,
                   Connection.ConfigurationPath.ApplicationPath +
                   Connection.ConfigurationPath.FolderPath);

            section = config.GetSection("system.webServer/imageCopyright");
            color = (string)section.GetAttribute("color").Value;
            message = (string)section.GetAttribute("message").Value;
            featureenabled = (bool)section.GetAttribute("enabled").Value;

        }

        catch
        { }

    }

    void UpdateUI()
    {
        _enabledCB.Checked = featureenabled;
        int n = _colCombo.FindString(color, 0);
        _colCombo.SelectedIndex = n;
        _msgTB.Text = message;
    }


    protected override void OnActivated(bool initialActivation)
    {
        base.OnActivated(initialActivation);
        if (initialActivation)
        {
            ReadConfig();
            UpdateUI();
        }
    }



    private void applyClick(Object sender, EventArgs e)
    {
        try
        {
            UpdateVariables();
            ServerManager mgr;
            ConfigurationSection section;
            mgr = new ServerManager();
            Configuration config =
            mgr.GetWebConfiguration
            (
```

```
                        Connection.ConfigurationPath.SiteName,
                        Connection.ConfigurationPath.ApplicationPath +
                        Connection.ConfigurationPath.FolderPath
                );

                section = config.GetSection("system.webServer/imageCopyright");
                section.GetAttribute("color").Value = (object)color;
                section.GetAttribute("message").Value = (object)message;
                section.GetAttribute("enabled").Value = (object)featureenabled;

                mgr.CommitChanges();

            }

            catch
            { }

        }

        public void UpdateVariables()
        {
            featureenabled = _enabledCB.Checked;
            color = _colCombo.Text;
            message = _msgTB.Text;
        }
    }
```

اولین چیزی که در کلاس بالا صدا زده می‌شود، سازنده‌ی کلاس هست که ما در آن یک تابع تعریف کردیم به اسم **initialize** که به آماده سازی اینترفیس یا رابط کاربری می‌پردازد و کنترل‌ها را روی صفحه می‌چیند. این سه کنترل، یکی *Combox* برای تعیین رنگ، یک *Checkbox* برای فعال بودن ماژول و دیگری هم یک *textbox* جهت نوشتن متن است. مابقی هم که سه *label* برای نامگذاری اشیاست. بعد از اینکه کنترل‌ها روی صفحه درج شدند، لازم است که تنظیمات پیش فرض یا قبلی روی کنترل‌ها نمایش یابند که اینکار را به وسیله تابع **readConfig** انجام می‌دهیم و تنظیمات خوانده شده را در متغیرهای عمومی قرار داده و با استفاده از تابع **UpdateUI** این اطلاعات را روی کنترل‌ها ست می‌کنیم و به این ترتیب *UI* به روز می‌شود. این دو تابع را به ترتیب پشت سر هم در یک متد به اسم **OnActivated** که *override* کرده‌ایم صدا میزنیم. در واقع این متد یک جورایی همانند رویداد *Load* می‌باشد؛ اگر *true* برگرداند اولین فعال سازی رابط کاربری بعد از باز شدن IIS است و در غیر این صورت *false* بر میگرداند.

در صورتی که کاربر مقادیر را تغییر دهد و روی گزینه *apply* کلیک کند تابع applyClick اجرا شده و ابتدا به تابع UpdateVariables ارجاع داده میشود که در آن مقادیر خوانده شده و در متغیرهای *Global* قرار می‌گیرند و سپس با استفاده از دو شیء از نوع *serverManger* و *ConfigSection* جایگذاری یا ذخیره می‌شوند.
استفاده از دو کلاس Servermanager و Configsection در دو قسمت خواندن و نوشتن مقادیر به کار رفته‌اند. کلاس servermanager به ما اجازه دسترسی به تنظیمات IIS و قابلیت‌های آن را میدهد. در تابع ReadConfig مسیر وب سایتی را که در لیست IIS انتخاب شده است، دریافت کرده و به وب کانفیگ آن وب سایت رجوع نموده و تگ imageCopyright آن را که در تگ system.webserver قرار گرفته است، میخواند (در صورتی که این تگ در آن وب کانفیگ موجود نباشد، خواندن و سپس ذخیره مجدد آن روی تگ داخل فایل applicationHost.config اتفاق میفتد که نتیجتا برای همه‌ی وب سایت هایی که این تگ را ندارند یا مقدارهای پیش فرض آن را تغییر نداده‌اند رخ میدهد) عملیات نوشتن هم مشابه خواندن است. تنها باید خط زیر را در آخر برای اعمال تغییرات نوشت؛ مثل EF با گزینه Context.SaveChanges:

```
mgr.CommitChanges();
```

وقت آن است که رابط کاربری را به IIS اضافه کنیم: پروژه را Rebuild کنید. بعد از آن با خطوطی که قبلا در Post-Build Command نوشتیم باید dll ما در GAC ریجستر شود. برای همین آدرس زیر را در cmd تایپ کنید:

```
%vs110comntools%\vsvars32.bat
```

عبارت اول که مسیر ویژوال استودیوی شماست و عدد 110 یعنی نسخه‌ی 11. هر نسخه‌ای را که استفاده میکنید، یک صفر جلویش بگذارید و جایگزین عدد بالا کنید. مثلا نسخه 8 می‌شود 80 و فایل بچ بالا هم دستورات visual studio را برای شما آزاد می‌کند.

سپس دستور زیر را وارد کنید:

```
GACUTIL /l ClassLibrary1
```

کلمه classLibrary1 نام پروژه‌ی ما بود که در GAC ریجستر شده است. با سوییچ l تمامی اطلاعات اسمبلی‌هایی که در GAC ریجستر شده‌اند، نمایش می‌یابند. ولی اگر اسم آن اسمبلی را جلویش بنویسید، فقط اطلاعات آن اسمبلی نمایش میابد. با اجرای خط فوق میتوانیم کلید عمومی public key اسمبلی خود را بدانیم که در شکل زیر مشخص شده است:

پس اگر کلید را دریافت کرده‌اید، خط زیر را به فایل administration.config در تگ <ModuleProviders> اضافه کنید:

```
<add name="imageCopyrightUI" type="ClassLibrary1.imageCopyrightUIProvider,   ClassLibrary1,
Version=1.0.0.0, Culture=neutral, PublicKeyToken=d0b3b3b2aa8ea14b"/>
```

عبارت ClassLibrary1.imageCopyrightUIProvider به کلاس imageCopyrightUIProvider اشاره می‌کند که در این کلاس UI معرفی می‌شود. مابقی عبارت هم کاملا مشخص است و در لینک‌های بالا در مورد Strong name توضیح داده شده اند.

فایل administration.config در مسیر زیر قرار دارد:

```
%windir%\system32\inetsrv\config\administration.config
```

حالا تنها کاری که نیاز است، باز کردن IIS است. به بخش وب سایت‌ها رفته و اپلیکیشنی که قبلا با نام mypictures را ایجاد کرده بودیم، انتخاب کنید. در سمت راست، آخر لیست، بخش others باید ماژول ما دیده شود. بازش کنید و تنظیمات آن را تغییر دهید و حالا یک تصویر را از اپلیکیشن mypictures، روی مرورگر درخواست کنید تا تغییرات را روی تگ مشاهده کنید:

حالا دیگر باید ماژول نویسی برای IIS را فراگرفته باشیم. این ماژول‌ها می‌توانند از یک مورد ساده تا یک کلاس مهم و امنیتی باشند که روی سرور شما برای همه یا بعضی از وب سایت‌ها در حال اجرا هستند و در صورت لزوم و اجازه شما، برنامه نویس‌ها میتوانند مثل همه‌ی تگ‌های موجود در وب کانفیگ سایتی را که مینویسند، تگ ماژول شما و تنظیمات آن را با استفاده از attribute یا خصوصیت‌های تعریف شده، بر اساس سلایق و نیازهایشان تغییر دهند و روی سرور شما آپلود کنند. الان شما یک سرور خصوصی سازی شده دارید.

از آنجا که این مقاله طولانی شده است، باقی موارد ویرایشی روی این UI را در مقاله بعدی بررسی خواهیم کرد.

عنوان: **آشنایی با ساختار IIS قسمت دهم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۱/۱۸ ۱:۱۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS

در دو مقاله پیشین (^ و ^) در مورد اینکه چگونه یک httpmodule یا httphandler را به عنوان یک ماژول جدید به IIS اضافه کنیم صحبت کردیم و الان قصد داریم که در این بخش این مبحث را ببندیم. آخرین بار توانستیم که یک UI را به IIS نسبت داده و از آن استفاده کنیم و الان قصد داریم آن را تکمیل‌تر کرده و کمی آن را شکیل‌تر کنیم. اولین نکته‌ای که توجه ما را جلب میکند این است که ماژول ما یک آیکن پیش فرض (چرخ دنده) دارد که بهتر است به آیکن دلخواه ما تغییر کند. به این منظور در پروژه، یک فایل Resource ایجاد کنید و یک تصویر را به این فایل Resource اضافه کنید و در کلاس imageCopyrightUI کد را به صورت زیر تغییر دهید:

```
internal class imageCopyrightUI : Module
    {
        protected override void Initialize(IServiceProvider serviceProvider, ModuleInfo moduleInfo)
        {

            base.Initialize(serviceProvider, moduleInfo);
            IControlPanel controlPanel = (IControlPanel)GetService(typeof(IControlPanel));
            ModulePageInfo modulePageInfo = new ModulePageInfo(this, typeof(imageCopyrightUIPage),
"Image Copyright", "Image Copyright",Resource1.Visual_Studio_2012,Resource1.Visual_Studio_2012);
            controlPanel.RegisterPage(modulePageInfo);
        }
    }
```

حال پروژه را Rebuild کنید و IIS را مجددا باز کنید تا ببینید که کامپوننت جدید شما آیکن جدید را دارد. دومین نکته‌ای که به چشم می‌آید این است که اگر دقت کنید سایر ماژول‌ها یا کامپوننت‌ها هر کدام در گروهی جداگانه جای گرفته‌اند و افزونه‌ی شما در یک گروه به اسم others، که زیاد جالب نیست و شاید دوست داشته باشید که ماژول شما هم در یکی از همین دسته‌ها جای بگیرد یا حتی اینکه خودتان یک گروه جدید بسازید. برای اینکه خوب قسمت‌ها و نام کلاس‌های آن‌ها را یاد بگیرید به شکلی که در قسمت قبلی هم گذاشته بودیم خوب دقت کنید.

نکته ای که باید توجه داشته باشید این است که گروه بندی در IIS به سه شیوه صورت می‌گیرید : یکی به شیوه **ناحیه Area** مثل *ASP.Net* و *management* و *IIS* ، **دسته بندی Category** مثل *ServerFeatures* , *Application Development* و **بدون گروه بندی** که می‌توانید از طریق لیست Group By در بالای پنجره IIS یکی از این سه حالت را انتخاب نمایید

برای اینکه ماژول در یک ناحیه یا دسته‌ای مشخص قرار بگیرد کد زیر را بنویسید:

```
internal class imageCopyrightUI : Module
    {
        protected override void Initialize(IServiceProvider serviceProvider, ModuleInfo moduleInfo)
        {
            base.Initialize(serviceProvider, moduleInfo);
            IControlPanel controlPanel = (IControlPanel)GetService(typeof(IControlPanel));
            ModulePageInfo modulePageInfo = new ModulePageInfo(this, typeof(imageCopyrightUIPage),
"Image Copyright", "Image Copyright", Resource1.Visual_Studio_2012, Resource1.Visual_Studio_2012);
            controlPanel.RegisterPage(ControlPanelCategoryInfo.AspNet,modulePageInfo);
        }
    }
```

اگر خوب دقت کنید می‌پینید که تنها تغییر کدها در متد regiterpage بوده است که با استفاده از کلاس ControlPanelCategoryInfo ناحیه مورد نظر را مشخص کردیم و اگر حالا پروژه را Rebuild کنید می‌بینید که در ناحیه ASP.Net قرار گرفته است. گروه‌های مختلف دیگری چون Application Development,Server,Performance و ... هم در این کلاس وجود دارند که جمعا 9 گروه می‌شوند.

حال این سوال پیش می‌آید اگر بخواهم گروه اختصاصی ایجاد کنم، چه کاری باید انجام شود. پس کد را به شکل زیر تغییر دهید:

```
internal class imageCopyrightUI : Module
    {
        protected override void Initialize(IServiceProvider serviceProvider, ModuleInfo moduleInfo)
        {
            base.Initialize(serviceProvider, moduleInfo);
            IControlPanel controlPanel = (IControlPanel)GetService(typeof(IControlPanel));
            ModulePageInfo modulePageInfo = new ModulePageInfo(this, typeof(imageCopyrightUIPage),
"Image Copyright", "Image Copyright", Resource1.Visual_Studio_2012, Resource1.Visual_Studio_2012);

                        ControlPanelCategorization areaCategorization = null;
            foreach (ControlPanelCategorization categorization in controlPanel.Categorizations) {
                if (categorization.Key == ControlPanelCategorization.AreaCategorization) {
                    areaCategorization = categorization;
                    break;
                }
            }
           ControlPanelCategoryInfo cat=new ControlPanelCategoryInfo("dotnettips","Dot Net Tips","This
is a Tutorial",areaCategorization);
            controlPanel.RegisterCategory(cat);
            controlPanel.RegisterPage(cat.Name,modulePageInfo);
        }
    }
```

در خطوط foreach ما به دنبال نوع گروهی که قرار است ماژول ما در آن قرار بگیرد می‌گردیم و علاقمندیم که گروه بندی ما در نوع Area باشد. برای همین این نوع را یافته و در متغیری از نوع ControlPanelCategorization قرار می‌دهیم. سپس توسط کلاس ControlPanelCategoryInfo یک نوع گروه بندی جدید ایجاد کرده ایم که به ترتیب نام، عنوان، توضیح و نهایتا نوع گروه بندی را در آن لحاظ می‌کنیم و سپس با دستور controlPanel.RegisterCategory گروه جدید خود را که در area قرار دارد، در IIS ثبت می‌کنیم و موقع افزودن صفحه مستقیما نام گروه را در آن ذکر می‌کنیم. پروژه را Rebuild و IIS را مجددا اجرا کنید. اگر Group by

شما روی Area باشد که به طور پیش فرض چنین است شما باید گروه Dot Net Tips را ببینید؛ حال از طریق لیست Group By گزینه‌ی Category را انتخاب نمایید. همانطور که مشاهده می‌کنید دوباره ماژول شما در دسته‌ی Others قرار می‌گیرد؛ چرا که ما برای Area گروه بندی را لحاظ کرده بودیم. برای اینکه بتوانیم در دو حالت، دسته بندی داشته باشیم، کد را به شکل زیر تغییر می‌دهیم:

```
ControlPanelCategorization areaCategorization = null;
                        ControlPanelCategorization CategoryCategorization = null;
            foreach (ControlPanelCategorization categorization in controlPanel.Categorizations) {
                if (categorization.Key == ControlPanelCategorization.AreaCategorization) {
                    areaCategorization = categorization;
                }
                if (categorization.Key == ControlPanelCategorization.CategoryCategorization)
                {
                    CategoryCategorization = categorization;
                }
            }
           ControlPanelCategoryInfo cat1=new ControlPanelCategoryInfo("dotnettipsarea","Dot Net Tips
Area","This is a Tutorial",areaCategorization);
            controlPanel.RegisterCategory(cat1);
            controlPanel.RegisterPage(cat1.Name,modulePageInfo);

            ControlPanelCategoryInfo cat2 = new ControlPanelCategoryInfo("dotnettipscat", "Dot Net Tips
Category", "This is a Tutorial", CategoryCategorization);
            controlPanel.RegisterCategory(cat2);
            controlPanel.RegisterPage(cat2.Name, modulePageInfo);
```

فکر کنم اتفاق بالا با توجه به مواردی که قبلا یاد گرفتید با وضوح کامل مشخص باشد که اینبار دو حالت گروه بندی را ذخیره و هر کدام را جداگانه در کنترل پنل IIS ثبت کردیم.

بین سایر گزینه‌های کنترل پنل گزینه controlpanel.regiterhomepage هم به چشم می‌خورد که اگر رابط کاربری را با این گزینه رجیستر کنیم صفحه اصلی IIS پریده و رابط کاربری ما جایگزینش می‌شود که البته این قسمت را باید با احتیاط با آن برخورد کرد وگرنه اگر بخواهیم همین رابط کاربری را به عنوان صفحه‌ی خانگی رجیستر کنیم، دسترسی ما به دیگر ماژول‌ها قطع خواهد شد.

سورس پروژه

تا به اینجا این مبحث از سری آموزشی ما بسته می‌شود. در مقالات آینده موارد دیگری از IIS را مورد بررسی قرار خواهیم داد.

عنوان: **آشنایی با ساختار IIS قسمت یازدهم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۱/۱۹ ۱:۲۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS

در این مطلب و همینطور مطلب بعدی قرار است به مبحث لاگ فایل‌ها Logfile بپردازیم. همانطور که می‌دانید سیستم IIS مثل هر سیستم دیگری لاگ هایی دارد که به مرور زمان این لاگ‌ها میتوانند مقدار زیادی از ظرفیت دیسک سخت را به خود اختصاص بدهند و این عمل میتواند موجب بروز مشکلاتی در سرور شود. به خوبی به یاد دارم که برای یکی از مشتریانم VPS تهیه نموده بودیم و بعد از یک سال با من تماس گرفت که سایت بالا نمی‌آید و وقتی بررسی شد، دیدم که از فضای دیسک سخت چند گیگابایتی، تنها چند مگابایت به طور ناچیز فضا برایش باقی مانده است و باعث شده است سرور از کار بیفتد. پس از بررسی متوجه شدیم تمام این فضاها توسط لاگ فایل‌ها پر شده است و از آنجا که سرویس دهنده تا مبلغی را به عنوان مدیریت سرور، ماهانه دریافت نکند، مدیریت این سرور مجازی را به عهده نداشته اند که البته بعدها با انتقال به یک سرور دیگر از یک سرویس دهنده دیگر مشکلات ما در مورد سرور برای همیشه حل شد.

پس این داستان به خوبی روشن می‌کند که مدیریت این لاگ‌ها چقدر میتواند مهم و حیاتی باشد. آقای تیم وَن از تیم تحریریه مایکروسافت در بخش IIS موارد زیر را برای مدیریت لاگ‌ها بر می‌شمارد:

فعال سازی فشرده سازی
انتقال لاگ‌ها به یک سیستم راه دور
حذف لاگ فایل‌های قدیمی از طریق اسکریپت نویسی
حذف لاگ فایل‌های قدیمی توسط IIS Log File Cleaner

فشرده سازی دایرکتوری لاگ فایل ها

مسیر ذخیره لاگ فایل‌ها در آدرس زیر می‌باشد. به این آدرس رفته و Properties دایرکتوری مورد نظر را باز کنید و در برگه‌ی General آن، گزینه‌ی Advanced را زده تا در کادر جدیدی که باز می‌شود، گزینه‌ی Compress contents to save disk space را انتخاب کنید تا محتویات در هنگام ذخیره روی دیسک سخت فشرده شوند.

```
%SystemDrive%\inetpub\logs\LogFiles
```

این روش ساده‌ترین روش موجود برای مدیریت لاگ هاست ولی روش نهایی نیست و باز به مرور زمان این روش هم کارایی خودش را از دست خواهد داد. این روش بیشتر شبیه خرید زمان می‌باشد تا اینکه یک راه حل نهایی برای حل مشکل باشد. البته این را هم باید مدنظر داشت که موقع تیک زدن گزینه بالا عملیات فشرده سازی باعث کند شدن سرعت کامپیوتر در حین آغاز عمل ذخیره سازی لاگ فایل‌ها هم خواهد شد. پس اگر قصد چنین کاری ذا دارید در ساعاتی که سرور کمترین فشار از طرف کاربران را دارد یا اصطلاحا پیک کاری آن پایین است انجامش دهید.

انتقال لاگ فایل‌ها به یک سیستم راه دور

همانطور که در بالا اشاره کردیم محل پیش فرض ذخیره سازی لاگ‌ها درمسیر

```
%SystemDrive%\inetpub\logs\LogFiles
```

قرار دارد و این محل ذخیره سازی برای هر سرور یا حتی یک وب سایت خاص در صفحه تنظیمات Logging مشخص شده است و شما در میتوانید این لاگ‌ها را حتی برای کل سرور یا مربوط به یک سایت خاص، به سروری دیگر انتقال دهید. این امکان می‌تواند به امنیت سیستم هم کمک فراوانی کند تا اگر دیسک محلی Local Disk هم دچار مشکل شد، باز خواندن لاگ فایل‌ها میسر باشد و با استفاده از ابزارهای تحلیل لاگ فایل ها ، آن‌ها را مورد بررسی قرار دهیم. برای تغییر محل ذخیره سازی لاگ‌ها به یک سیستم راه دور، راه حل زیر را طی کنید.

در IIS وب سایتی را که میخواهید لاگ آن انتقال یابد، انتخاب کنید؛ یا اگر لاگ کل سیستم IIS را میخواهید انتقال بدهید نام سرور را در لیست درختی انتخاب کنید و از ماژول‌های سمت راست، ماژول Logging را انتخاب کنید و در قسمت Directory که محل ذخیره سازی فعلی لاگ‌ها را نوشته شده است، به صورت UNC آدرس دهی کنید. در آدرس زیر اولی نام سرور است Contoso-\\server1 و دومی هم Logs نام پوشه‌ای که به اشتراک گذاشته شده است.

حذف لاگ فایل‌های قدیمی با استفاده از اسکریپت

با این روش میتوانید لاگ فایل هایی را که بعد از مدتی معین که دلخواه شما هست، از سیستم حذف نمایید و اگر این اسکریپت را زمان بندی خودکار نمایید، می‌توانید از مراقبت مداوم و ثابت این کار نیز رها شوید.

با ستفاده از VBScript بررسی می‌کنیم که اگر مثلا عمر لاگ فایل به 30 روز رسیده است، باید حذف شوند. خط دوم کد زیر نهایت عمر یک لاگ فایل را مشخص می‌کند:

```
sLogFolder = "c:\inetpub\logs\LogFiles"
iMaxAge = 30   'in days
Set objFSO = CreateObject("Scripting.FileSystemObject")
set colFolder = objFSO.GetFolder(sLogFolder)
For Each colSubfolder in colFolder.SubFolders
        Set objFolder = objFSO.GetFolder(colSubfolder.Path)
        Set colFiles = objFolder.Files
        For Each objFile in colFiles
                iFileAge = now-objFile.DateCreated
                if iFileAge > (iMaxAge+1)  then
                        objFSO.deletefile objFile, True
                end if
        Next
Next
```

اسکریپت بالا تمامی subfolderها را برای همه سایت‌ها بررسی کرده و لاگ‌های آنان را حذف می‌کند. ولی اگر دوست دارید این عملیات را تنها به یک وب سایت محدود کنید، باید مسیر را در خط اول دقیق‌تر مشخص کنید.

برای اجرای دستی اسکریپت در cmd تایپ کنید:

```
cscript.exe c:\scripts\retentionscript.vbs
```

ولی اگر میخواهید این اسکریپت در هر دوره‌ی زمانی خاص اجرا شود، یا زمان بندی Scheduling گردد، دیگر مجبور نیستید هر بار به فکر نگهداری از لاگ‌ها باشید.

زمان بندی اجرای اسکریپت

server manager (*قابل تست در ویندوزهای سرور*) را باز کرده و از منوی Tools گزینه Task Scheduler را انتخاب کنید و در قسمت Actions گزینه Create Task را انتخاب نمایید. در کادر باز شده نام "Delete Log Files" را برای مثال برگزینید و در قسمت Security هم کاربری که اجازه اجرای اسکریپت را دارد مشخص کنید.

برگه Triggers را انتخاب کرده و گزینه New را انتخاب کنید و عملیات زمان بندی را تنظیم کنید و حتما بعد از زمان بندی مطمئن باشید که تیک Enabled فعال است.

در برگه Actions هم گزینه New را انتخاب کنید؛ در کادر باز شده از لیست Start a program را انتخاب کرده و در قسمت Program\script، دستور cscript را ذکر نمایید و به عنوان آرگومان ورودی Add arguments هم مسیر اسکریپت خود را ذکر نمایید و کادر را تایید کنید.

برای آغاز زمان بندی در لیست وظیفه‌های فعال active task pane، وظیفه ای که الان ساخته اید را اجرا کرده و به مسیر ذخیره لاگ‌ها رفته و می‌بینید که لاگ‌های مورد نظر حذف شده‌اند؛ پس از صحت اجرای اسکریپت مطمئن می‌شویم. دوباره به لیست وظایف رفته و گزینه End را بزنید تا وظیفه، در حالت Ready قرار گیرد تا از همین الان فرایند زمان بندی اجرای اسکریپت آغاز شود.

حذف لاگ فایل‌ها با استفاده از IIS Log Cleaner Tools

ساده‌ترین ابزار برای مدیریت حذف لاگ فایل هاست که هر یک ساعت یکبار اجرا شده و لاگ فایل‌های تاریخ گذشته را که زمانش را شما تعیین می‌کنید، به سمت سطل زباله که البته درستش بازیافت است Recycle Bin انتقال میدهد تا از ضرر از دست دادن لاگ‌ها جلوگیری کند که بعدا شما میتوانید آن‌ها را به صورت دستی حذف کنید. همچنین عملیات خودکار حذف را نیز می‌توان متوقف نمود.

ابتدا برنامه را از اینجا دانلود کنید. موقعیکه برنامه را اجرا کنید، در نوتیفیکیشن taskbar می‌نشیند و برنامه با یک پیغام به شما اعلام می‌کند، این اولین بار است که برنامه را باز کرده‌اید. پس یک سر به setting آن بزنید؛ با انتخاب گزینه‌ی settings برنامه بسته شده و فایل Settings.txt برای شما باز می‌شود که مدت زمان عمر لاگ فایل و مسیر ذخیره آن‌ها، از شما پرسیده می‌شود که مقدار عمر هر لاگ فایل به طور پیش فرض 30 روز و مسیر ذخیره‌ی لاگ‌ها همان مسیر پیش فرض IIS است که اگر شما دستی آن را تغییر داده اید، با پرسیدن آن، از محل لاگ‌ها اطمینان کسب می‌کند. در صورتی که قصد تغییری را در فایل، دارید آن را تغییر داده و ذخیره کنید و برنامه را مجددا اجرا کنید.

نکات نهایی در مورد این برنامه :

اگر از ابزار IIS Cleaner Tool استفاده می‌کنید باید دستی سطل بازیافت را هم پاک کنید و هم اینکه میتوانید یک محدودیت حجمی برای Recycle Bin قرار دهید که اگر به یک حدی رسید، خودکار پاک کند تا مشکلی برای سیستم عامل ایجاد نشود که البته به طور پیش فرض چنین است.

برنامه بالا به طور پیش فرض ریشه‌ی لاگ‌ها را حذف می‌کند. پس اگر میخواهید فقط سایت خاصی را مد نظر داشته باشد، آدرس دایرکتوری آن را اضافه کنید. البته چون این برنامه فقط روی یک دایرکتوری کار می‌کند و شما چند وب سایت دارید و مثلا میخواهید سه تای آن‌ها را پاکسازی کنید، چاره‌ی جز استفاده از اسکریپت‌های با زمان بندی ندارید.

برنامه‌ی بالا فقط فایل هایی با پسوند log را به سطل بازیافت انتقال می‌دهد.

برنامه‌ی بالا یک سرویس نیست و باید به طور دستی توسط کاربر اجرا گردد. پس اگر ریست هم شد باید دستی اجرا شود یا آن را به داخل پوشه startup بکشید.

برنامه برای اجرایش نیاز به لاگین کاربر و مجوز نوشتن در آن پوشه را دارد تا به درستی کار کند.

در قسمت بعدی مبحث لاگ‌ها را ادامه خواهیم داد و با ماژول Logging در IIS و تنظیماتش آشنا خواهیم شد.

عنوان: **آشنایی با ساختار IIS قسمت دوازدهم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۱/۲۰ ۱۲:۰
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, log, Logfile, LogParser

پیکربندی قسمت لاگ‌ها، میتواند برای یک سرور و یا وب سایت خاص از طریق فایل کانفیگ یا از طریق خود IIS انجام گیرد. برای اینکه به بیشتر این قابلیت‌ها در IIS دسترسی داشت، باید یکی از نسخه‌های ویندوز سرور 2012 و ویندوز 8 را نصب کرده باشید.
لاگ‌ها به ثبت خطاها و درخواست‌های HTTP می‌پردازند و با تحلیل آن‌ها میتوان عملیات بهینه سازی را بر روی سرو اجرا کرد.
تمامی ثبت لاگ‌ها توسط Http.sys انجام می‌گیرد.

نحوه‌ی ذخیره سازی لاگها

در این بخش نحوه‌ی ذخیره سازی و فرمت ذخیره‌ی لاگ‌ها را در دو سطح سایت و سرور به طور جداگانه بررسی می‌کنیم. در IIS ماژول Logging را باز کنید و در لیست One log file per می‌توانید مشخص کنید که لاگ‌ها در چه سطحی اجرا شوند. اگر گزینه‌ی server باشد، تمامی خطاها و درخواست‌های رسیده به سرور در یک فایل لاگ ثبت می‌شوند. ولی اگر سطح سایت باشد، برای هر سایت بر روی IIS لاگ‌ها، جداگانه بررسی می‌شوند. به طور پیش فرض سطح سایت انتخاب شده است.

سطح سایت

موقعی‌که در لیست، سایت را انتخاب کنید، در لیست format می‌توانید تعیین کنید که لاگ‌ها به چه صورتی باید ذخیره شوند.
مواردی که در این حالت لیست می‌شوند گزینه‌های W3C,IIS,NCSA,Custom می‌باشند که در زیر یکایک آن‌ها را بررسی می‌کنیم:

فرمت IIS: این فرمت توسط مایکروسافت ارائه شده و در این حالت لاگ‌های همه‌ی وب سایت‌ها ذخیره می‌شوند. به این فرمت Fixed ASCII Based Text نیز می‌گویند؛ چرا که اجازه‌ی خصوصی سازی ندارد و نمی‌توانید بگویید چه فیلدهایی در لاگ قرار داشته باشند. لاگ فایل‌های این فرمت با ، (کاما) از هم جدا می‌شوند و مقدار زمانی که برای هر فیلد ثبت می‌شود، به صورت محلی local Time می‌باشد.
فیلدهایی که در لاگ این نوع فرمت خواهند آمد، به شرح زیر است:

Client IP address

User name

Date

Time

Service and instance

Server name

Server IP address

Time taken

Client bytes sent

Server bytes sent

Service status code (A value of 200 indicates that the request was fulfilled successfully)

.Windows status code (A value of 0 indicates that the request was fulfilled successfully)

Request type

Target of operation

Parameters (the parameters that are passed to a script)

احتمال این وجود دارد که بعضی از فیلدها در بعضی رکوردها، شامل اطلاعاتی نباشند که به جای مقدار آن علامت - ثبت می‌گردد و برای کاراکترهایی که قابل نمایش نیستند یا کاراکتر نمایشی ندارند، از علامت + استفاده می‌شود. دلیل اینکار هم این است که ممکن است یک کاربر مهاجم، به ارسال اطلاعات کلیدهای کنترلی چون Carriage return اختصارا CR یا Line Feed به اختصار LF کند، که باعث شکسته شدن خط لاگ فایل می‌شود و در نتیجه از استاندارد خارج خواهد شد و هنگام خواندن آن هم با خطا روبرو می‌شویم؛ در نتیجه با جایگزینی چنین کاراکترهایی با + از این اتفاق جلوگیری می‌شود.
شکل زیر نمونه ای از یک خط لاگ در این فرمت است:

```
192.168.114.201, -, 03/20/01, 7:55:20, W3SVC2, SERVER, 172.21.13.45, 4502, 163, 3223, 200, 0, GET,
/DeptLogo.gif, -,
```

نام فیلد	نوع حالت مقداردهی	توضیح اتفاقات افتاده
Client IP address	192.168.114.201	آی پی کلاینت
User name	-	کاربر ناشناس است
Date	03/20/01	تاریخ فعالیت
Time	7:55:20	ساعت فعالیت
	Service and instance	W3SVC2
لاگی که مربوط به سایت خاصی می‌شود به صورت W3SVC# نمایش داده می‌شود که علامت # شماره سایت می‌باشد که در اینجا این لاگ مربوط به سایت شماره 2 است	**Server name**	SERVER
نام سرور	**Server IP**	172.21.13.45
آی پی سرور	**Time taken**	4502
چقدر انجام عملیات این درخواست به طول انجامیده است که بر حسب میلی ثانیه است.	**Client bytes sent**	163
تعداد بایت هایی که از طرف کلاینت به سرور ارسال شده است	**Server bytes sent**	3223
تعداد بایت هایی که از طرف سرور به سمت کلاینت ارسال شده است	**Service status code**	200
درخواست کاملا موفقیت آمیز بوده است	**Windows status code**	0
درخواست کاملا موفقیت آمیز بوده است	**Request type**	GET
نوع درخواست کاربر	**Target of operation**	/DeptLogo.gif
کاربر قصد دانلود یک فایل تصویری GIF	**Parameters**	-

نام فیلد	نوع حالت مقداردهی	توضیح اتفاقات افتاده
داشته است که نامش Deptlogo است		

فرمت NCSA: این فرمت توسط مرکز علمی کاربردهای ابرمحاسباتی National Center for Supercomputing Applications ایجاد شده و دقیقا مانند قبلی نمیتوان در آن نوع فیلدها را مشخص کرد و برای جدا سازی، از فاصله space استفاده می‌کند و ثبت مقدار زمان در آن هم به صورت محلی و هم UTC می‌باشد.
این فیلدها در لاگ آن نمایش داده می‌شوند:

Remote host address

Remote log name (This value is always a hyphen)

User name

Date, time, and Greenwich mean time (GMT) offset

Request and protocol version

Service status code (A value of 200 indicates that the request was fulfilled successfully)

Bytes sen

نمونه ای از یک لاگ ثبت شده:

```
172.21.13.45 - Microsoft\JohnDoe [08/Apr/2001:17:39:04 -0800] "GET /scripts/iisadmin/ism.dll?http/serv
HTTP/1.0" 200 3401
```

نام فیلد	مقدار ثبت شده	توضیح اتفاق افتاده
Remote host address	172.21.13.45	آی پی کلاینت
Remote log name	-	نامی وجود ندارد
User name	Microsoft\JohnDoe	نام کاربری
Date, time, and GMT offset	[08/Apr/2001:17:39:04 -0800]	تاریخ و ساعت فعالیت به صورت محلی که 8 ساعت از مبدا گرینویچ بیشتر است
Request and protocol version	GET /scripts/iisadmin/ism.dll?http/serv HTTP/1.0	کاربر با متد GET و Http نسخه‌ی یک، درخواست فایل ism.dll را کرده است.
Service status code	200	عملیات کاملا موفقیت آمیز بود.
Bytes sent	3401	تعداد بایت‌های ارسال شده به سمت کاربر

امنیت در برابر کاربران مهاجم مانند همان فرمت قبلی صورت گرفته است.

فرمت W3C : توسط W3C توسط کنسرسیوم جهانی وب ارائه شده است و یک فرمت customizable ASCII text-based است. به این معنی که میتوان فیلدهایی که در گزارش نهایی می‌آید را خودتان مشخص کنید، که برای اینکار در کنار لیست، دکمه‌ی Select

وجود دارد که میتوانید هر کدام از فیلدهایی را که خواستید، انتخاب کنید تا به ترتیب در خط لاگ ظاهر شوند. تاریخ ثبت به صورت UTC است.

نام فیلد	توضیح	به طور پیش فرض انتخاب شده است
Date	تاریخ رخ دادن فعالیت	بله
Time	ساعت زخ دادن فعالیت بر اساس UTC	بله
Client IP Addres s	آی پی کلاینت	بله
User Name	نام کاربری که هویت آن تایید شده و در صورتی که هویت تایید شده نباشد و کاربر ناشناس باشد، جای آن - قرار می‌گیرد	بله
Service Name and Instance Number	نام و شماره سایتی که درخواست در آن صورت گرفته است	خیر
Server Name	نام سروری که لاگ روی آن ثبت می‌شود	خیر
Server IP Address	آی پی سرور که لاگ روی آن ثبت می‌شود	بله
Server Port	شمره پورتی که سرویس مورد نظر روی آن پورت اعمال می‌شود.	بله
Method	متد درخواست مثل GET	بله
URI Stem	هدف درخواست یا Target مثل index.htm	بله
URI Query	کوئری ارسال شده برای صفحات داینامیک	بله
HTTP Status	کد وضیعینی HTTP status	بله
Win32 Status	کد وضعیتی ویندوز	خیر
Bytes Sent	تعداد بایت‌های ارسال شده به سمت کلاینت	خیر
Bytes Received	تعداد بایت‌های دریافت شده از سمت کلاینت	خیر
Time Taken	زمان به طول انجامیدن درخواست بر حسب میلی ثانیه	خیر
Protocol Version	درخواست با چه نسخه‌ای از پروتکل http یا ftp ارسال شده است	خیر
Host	اگر در هدر درخواست ارسالی این گزینه بوده باشد، نوشته خواهد شد.	خیر
User Agent	اطلاعات را از هدر درخواست می‌گیرد.	بله
Cookie	اگر کوکی رد و بدل شده باشد، محتویات کوکی ارسالی یا دریافت شده	خیر
Referrer	کاربر از چه سایتی به سمت سایت ما آمده است.	خیر
Protocol Substatus		بله

نام فیلد	توضیح	به طور پیش فرض انتخاب شده است
	در صورت رخ دادن خطا در IIS ، کد خطا بازگردانده میشود. در IIS به منظور امنیت بیشتر و کاهش حملات، محتوای خطاهای رخ داده در IIS به صورت متنی نمایش داده نمی‌شوند و شامل کد خطایی به اسم Substatus Code هستند تا مدیران شبکه با ردیابی لاگ‌ها پی به دلیل خطا و درخواست‌های ناموفق ببرند. برای مثال Error 404.2 به این معنی است که فایل درخواستی به دلیل قوانین محدود کنده، قفل شده و قابل ارائه نیست. ولی هکر تنها با خطای 404 یعنی وجود نداشتن فایل روبرو می‌شود. در حالت substatus code، کد شماره 2 را هم خواهید داشت که در لاگ ثبت می‌شود. هر شخصی که در سرور توانایی دسترسی به لاگ‌ها را داشته باشد، می‌تواند کد دوم خطا را نیز مشاهده کند. برای مثال مدیر سرور متوجه میشود که یکی از فایل‌های مورد نظر به کاربران، خطای 404 نمایش میدهد و با بررسی لاگ‌ها متوجه می‌شود که کد خطا 404.9 هست. از آنجا که ما همه‌ی کدها را حفظ نیستیم به این صفحه رجوع می‌کنیم و متوجه میشویم تعداد کاربرانی که برای این فایل، اتصال connection ایجاد کرده‌اند بیش از مقدار مجاز است و مدیر میتواند این وضع را کنترل کند. برای مثال تعداد اتصالات مجاز را نامحدود unlimited تعیین کند.	

حروف - و + برای موارد بالا هم صدق می‌کند. در ضمن گزینه‌های زیر در حالتی که درخواست از پروتکل FTP باشد مقداری نخواهند گرفت:

uri-query

host

(User-Agent)

Cookie

Referrer

substatus

گزینه Custom : موقعی که شما این گزینه را انتخاب کنید ماژول logging غیرفعال خواهد شد. زیرا این امکان در IIS قابل پیکربندی نیست و نوشتن ماژول آن بر عهده شما خواهد بود؛ با استفاده از اینترفیس های ILogPluginEx ، ILogPlugin و ILogUIPlugin آن را پیاده سازی کنید.

ذخیره اطلاعات به انکدینگ UTF-8 و موضوع امنیت

در صورتی که شما از سایتی با زبانی غیر از انگلیسی و لاتین و فراتر از ANSI استفاده می‌کنید، این گزینه حتما باید انتخاب شده باشد تا درخواست را بهتر لاگ کند. حتی برای وب سایت‌های انگلیسی زبان هم انتخاب این گزینه بسیار خوب است؛ چرا که اگر به سمت سرور کاراکترهای خاصی در URL ارسال شوند، نمی‌تواند با کدپیج موجود آن‌ها را درست تبدیل کند.

ادامه‌ی تنظیمات

موارد بعدی که در تنظیمات لاگ‌ها کاملا مشخص و واضح است، عملیات زمان بندی است که برای ساخت یک فایل لاگ جدید به کار می‌رود؛ برای مثال هر ساعت یک لاگ فایل جدید بسازد و فعالیت‌های موجود در هر ساعت در یک لاگ ذخیره می‌شوند.

گزینه‌ی بعدی حداکثر حجم هر فایل لاگ است که به صورت بایت مشخص می‌شود. اگر مقداری که تعیین میکنید کمتر از 1048576 بایت باشد، خودش به طور پیش فرض همان 1048576 بایت را در نظر خواهد گرفت.

گزینه بعدی do not create a new logfile بدین معناست که همه‌ی لاگ‌ها در یک فایل ذخیره می‌شوند و فایل جدیدی برای لاگ‌ها ایجاد نمی‌شود.

گزینه آخری به اسم use local time for filenaming and rollover است که اگر انتخاب شود، نامگذاری هر فایل لاگ بر اساس زمان محلی ساخت فایل لاگ خواهد بود. در صورتیکه انتخاب نشود، نامگذاری با زمان UTC درج خواهد شد.

سطح سرور

لاگ‌ها فقط در سمت سرور انجام می‌گیرد و لاگ هر سایت در یک فایل لاگ ثبت می‌شود. اگر بخواهید لاگ‌ها را در سطح سرور انجام دهید، گزینه‌ی binary هم اضافه خواهد شد.

Binary: در این گزینه دیگر از قالب بندی یا فرمت بندی لاگ‌ها خبری نیست و لاگ هر وب سایت به صورت اختصاصی صورت نمی‌گیرد. عملیات ذخیره سازی و ثبت هر لاگ می‌تواند از منابع یک سرور از قبیل حافظه و CPU و ... استفاده کند و اگر تعداد این وب سایت‌ها بالا باشد، باقی روش‌ها باعث فشار به سرور می‌شوند. برای همین ایجاد یک فایل خام از لاگ‌ها در این مواقع می‌تواند راهگشا باشد. برای همه یک فایل لاگ ایجاد شده و بدون قالب بندی ذخیره می‌کند. پسوند این نوع لاگ‌ها ibl است که مخفف Internet Binary Log می‌باشد. دلیل این تغییر پسوند این است که اطمینان کسب شود کاربر، با برنامه‌های متنی چون notepad یا امثال آن که به Text Utilities معروفند فایل را باز نمی‌کند. برای خواندن این فایل‌های میتوان از برنامه‌ی Log parser استفاده کرد. پروتکل‌های FTP,NNTP و SMTP در این حالت لاگشان ثبت نمی‌شود.

عنوان: **کار با اسکنر در برنامه های تحت وب (قسمت دوم و آخر)**
نویسنده: محمد رعیت پیشه
تاریخ: ۱۳۹۳/۱۱/۲۰ ۲۳:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, MVC, WCF, TWAIN, Cross-Origin Request, Scanning

در قسمت قبل « کار با اسکنر در برنامه‌های تحت وب (قسمت اول) » دیدی از کاری که قرار است انجام دهیم، رسیدیم. حالا سراغ یک پروژه‌ی عملی و پیاده سازی مطالب مطرح شده می‌رویم.

ابتدا پروژه‌ی WCF را شروع می‌کنیم. ویژوال استودیو را باز کرده و از قسمت New Project > Visual C# > WCF یک پروژه‌ی WCF Service Application جدید را مثلا با نام "WcfServiceScanner" ایجاد نمایید. پس از ایجاد، دو فایل IService1.cs و Service1.scv موجود را به IScannerService و ScannerService تغییر نام دهید. سپس ابتدا محتویات کلاس اینترفیس IScannerService را به صورت زیر تعریف نمایید :

```
[ServiceContract]
    public interface IScannerService
    {
        [OperationContract]
        [WebInvoke(Method = "GET",
           BodyStyle = WebMessageBodyStyle.Wrapped,
           RequestFormat = WebMessageFormat.Json,
           ResponseFormat = WebMessageFormat.Json,
           UriTemplate = "GetScan")]
        string GetScan();
    }
```

در اینجا ما فقط اعلان متدهای مورد نیاز خود را ایجاد کرده‌ایم. علت استفاده از Attribute ایی با نام WebInvoke ، مشخص نمودن نوع خروجی به صورت Json است و همچنین عنوان آدرس مناسبی برای صدا زدن متد. پس از آن کلاس ScannerService را مطابق کدهای زیر تغییر دهید:

```
public class ScannerService : IScannerService
    {
        public string GetScan()
        {
            // TODO Add code here
        }
    }
```

تا اینجا فقط یک WCF Service معمولی ساخته‌ایم .در ادامه به سراغ کلاس WIA برای ارتباط با اسکنر می‌رویم.
بر روی پروژه‌ی خود راست کلیک کرده و Add Reference را انتخاب نموده و سپس در قسمت COM، گزینه‌ی Microsoft Windows Image Acquisition Library v2.0 را به پروژه‌ی خود اضافه نمایید.
با اضافه شدن این ارجاع به پروژه، دسترسی به فضای نام WIA برای ما امکان پذیر می‌شود که ارجاعی از آن را در کلاس ScannerService قرار می‌دهیم.

```
using WIA;
```

اکنون متد GetScan را مطابق زیر اصلاح می‌نماییم:

```
public string GetScan()
        {
            var imgResult = String.Empty;
            var dialog = new CommonDialogClass();
            try
            {
                // نمایش فرم پیشفرض اسکنر
                var image = dialog.ShowAcquireImage(WiaDeviceType.ScannerDeviceType);

                // ذخیره تصویر در یک فایل موقت
                var filename = Path.GetTempFileName();
```

```
            image.SaveFile(filename);
            var img = Image.FromFile(filename);

            // img تبدیل تصویر به Base64 جهت ارسال سمت کاربر و نمایش در تگ
            imgResult = ImageHelper.ImageToBase64(img, ImageFormat.Jpeg);
        }
        catch
        {
            // از آنجاییه که امکان نمایش خطا وجود ندارد در صورت بروز خطا رشته خالی
            // بازگردانده می‌شود که به معنای نبود تصویر می‌باشد
        }

        return imgResult;
    }
```

دقت داشته باشید که کدها را در زمان توسعه بین Try..Catch قرار ندهید چون ممکن‌است در این زمان به خطاهایی برخورد کنید که نیاز باشد در مرورگر آنها را دیده و رفع خطا نمایید.

CommonDialogClass کلاس اصلی در اینجا جهت نمایش فرم کار با اسکنر می‌باشد و متدهای مختلفی را جهت ارتباط با اسکنر در اختیار ما قرار می‌دهد که بسته به نیاز خود می‌توانید از آنها استفاده کنید. برای نمونه در مثال ما نیز متد اصلی که مورد استفاده قرار گرفته، ShowAcquireImage می‌باشد که این متد، فرم پیش فرض دریافت اسکنر را به کاربر نمایش می‌دهد و کاربر از طریق آن می‌تواند قبل از شروع اسکن، یکسری تنظیمات را انجام دهد.

این متد ابتدا به صورت خودکار فرم تعیین دستگاه اسکنر ورودی را نمایش داده :

و سپس فرم پیش فرض اسکنرهای TWAIN را جهت تعیین تنظیمات اسکن نمایش می‌دهد که این امکان نیز در این فرم فراهم است تا دستگاه‌های Feeder یا Flated انتخاب گردند.

خروجی این متد همان عکس اسکن شده است که از نوع WIA.ImageFile می‌باشد و ما پس از دریافتش، ابتدا آن را در یک فایل موقت ذخیره نموده و سپس با استفاده از یک متد کمکی آن را به فرمت Base64 برای درخواست کننده اسکن ارسال می‌نماییم.

کدهای کلاس کمکی ImageHelper:

```
public static string ImageToBase64(Image image, System.Drawing.Imaging.ImageFormat format)
        {
            if (image != null)
            {
                using (MemoryStream ms = new MemoryStream())
                {
                    // Convert Image to byte[]
                    image.Save(ms, format);
                    byte[] imageBytes = ms.ToArray();

                    // Convert byte[] to Base64 String
                    string base64String = Convert.ToBase64String(imageBytes);
                    return base64String;
                }
            }
            return String.Empty;
        }
```

توجه داشته باشید که خروجی این متد قرار است توسط callBack یک متد جاوا اسکریپتی مورد استفاده قرار گرفته و احیانا عکس مورد نظر در صفحه نمایش داده شود. پس بهتر است که از قالب تصویر به شکل Base64 استفاده گردد. ضمن اینکه پلاگین‌های Jquery مرتبط با ویرایش تصویر هم از این قالب پشتیبانی می‌کنند. (اینجا)
این مثال به ساده‌ترین شکل نوشته شد. کلاس دیگری هم در اینجا وجود دارد و در صورتیکه از اسکنر نوع Feeder استفاده می‌کنید، می‌توانید از کدهای آن استفاده کنید.

کار ما تا اینجا در پروژه‌ی WCF Service تقریبا تمام است. اگر پروژه را یکبار Build نمایید برای اولین بار احتمالا پیغام خطاهای زیر ظاهر خواهند شد:

جهت رفع این خطا، در قسمت Referenceهای پروژه خود، WIA را انتخاب نموده و از Propertiesهای آن خصوصیت Embed Interop Types را به False تغییر دهید؛ مشکل حل می‌شود.

به سراغ پروژه‌ی ویندوز فرم جهت هاست کردن این WCF سرویس می‌رویم. می‌توانید این سرویس را بر روی یک Console App یا Windows Service هم هاست کنید که در اینجا برای سادگی مثال، از WinForm استفاده می‌کنیم.
یک پروژه‌ی WinForm جدید را ایجاد کنید و سپس از قسمت Solution > Add Reference به مسیر پروژه‌ی قبلی رفته و dllهای آن را به پروژه خود اضافه نمایید.
Form1.cs را باز کرده و ابتدا دو متغیر زیر را در آن به صورت عمومی تعریف نمایید:

```
private readonly Uri _baseAddress = new Uri("http://localhost:6019");
        private ServiceHost _host;
```

برای استفاده از کلاس ServiceHost لازم است تا ارجاعی به فضای نام System.ServiceModel داده شود. متغیر _baseAddress نگه دارنده‌ی آدرس ثابت سرویس اسکنر در سمت کلاینت می‌باشد و به این ترتیب ما دقیقا می‌دانیم باید سرویس را با کدام آدرس در کدهای جاوا اسکریپتی خود فراخوانی نماییم.
حال در رویداد Form_Load برنامه، کدهای زیر را جهت هاست کردن سرویس اضافه می‌نماییم:

```
private void Form1_Load(object sender, EventArgs e)
        {
            _host = new ServiceHost(typeof(WcfServiceScanner.ScannerService), _baseAddress);
            _host.Open();
        }

        private void Form1_FormClosing(object sender, FormClosingEventArgs e)
        {
            _host.Close();
        }
```

همین چند خط برای هاست کردن سرویس روی آدرس localhost و پورت 8010 کامپیوتر کلاینت کافی است. اما یکسری تنظیمات مربوط به خود سرویس هم وجود دارد که باید در زمان پیاده سازی سرویس، در خود پروژه‌ی سرویس، ایجاد می‌گردید. اما از آنجا که ما قرار است سرویس را در یک پروژه‌ی دیگر هاست کنیم، بنابراین این تنظیمات را باید در همین پروژه‌ی WinForm قرار دهیم.
فایل App.Config پروژه‌ی WinForm را باز کرده و کدهای آنرا مطابق زیر تغییر دهید:

```
<?xml version="1.0" encoding="utf-8" ?>
```

```
<configuration>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.5" />
    </startup>

  <system.serviceModel>
    <behaviors>
      <serviceBehaviors>
        <behavior name="BehaviourMetaData">
          <serviceMetadata httpGetEnabled="true" />
        </behavior>
      </serviceBehaviors>
    </behaviors>
    <services>
      <service name="WcfServiceScanner.ScannerService"
               behaviorConfiguration="BehaviourMetaData">
        <endpoint address=""
                  binding="basicHttpBinding"
                  contract="WcfServiceScanner.IScannerService" />
      </service>
    </services>
  </system.serviceModel>

</configuration>
```

اکنون پروژه‌ی هاست آماده اجرا می‌باشد. اگر آنرا اجرا کنید و در مرورگر خود آدرس ذکر شده را وارد کنید، صفحه‌ی زیر را مشاهده خواهد کرد که به معنی صحت اجرای سرویس اسکنر می‌باشد.

ScannerService Service

You have created a service.

To test this service, you will need to create a client and use it to call the service. You can do this using the svcutil.exe tool from the command line with the following syntax:

```
svcutil.exe http://localhost:6019/?wsdl
```

You can also access the service description as a single file:

```
http://localhost:6019/?singleWsdl
```

This will generate a configuration file and a code file that contains the client class. Add the two files to your client application and use the generated client class to call the Service. For example:

C#

```
class Test
{
    static void Main()
    {
        ScannerServiceClient client = new ScannerServiceClient();

        // Use the 'client' variable to call operations on the service.

        // Always close the client.
        client.Close();
    }
}
```

اگر موفق به اجرا نشدید و احیانا با خطای زیر مواجه شدید، اطمینان حاصل کنید که ویژوال استودیو Run as Administrator باشد. مشکل حل خواهد شد.

به سراغ پروژه‌ی بعدی، یعنی وب سایت خود می‌رویم. یک پروژه‌ی MVC جدید ایجاد نمایید و در View مورد نظر خود، کدهای زیر را جهت صدا زدن متد GetScan اضافه می‌کنیم.

(از آنجا که کدها به صورت جاوا اسکریپت می‌باشد، پس مهم نیست که حتما پروژه MVC باشد؛ یک صفحه‌ی HTML ساده هم کافی است).

```
<a href="#" id="get-scan">Get Scan</a>
<img src="" id="img-scanned" />
<script>
    $("#get-scan").click(function () {
        var url = 'http://localhost:6019/';
        $.get(url, function (data) {
            $("#img-scanned").attr("src","  "+ data.GetScanResult);
        });
    });
</script>
```

دقت کنید در هنگام دریافت اطلاعات از سرویس، نتیجه به شکل GetScanResult خواهد بود. الان اگر پروژه را اجرا نمایید و روی لینک کلیک کنید، اسکنر شروع به دریافت اسکن خواهد کرد اما نتیجه‌ای بازگشت داده نخواهد شد و علت هم مشکل امنیتی CORS می‌باشد که به دلیل دریافت اطلاعات از یک دامین دیگر رخ می‌دهد و اگر با Firebug درخواست را بررسی کنید متوجه خطا به شکل زیر خواهید شد.

راه حل‌های زیادی برای این مشکل ارائه شده است، و متاسفانه بسیاری از آنها در شرایط پروژه‌ی ما جوابگو نمی‌باشد (به دلیل هاست روی یک پروژه ویندوزی). تنها راه حل مطمئن (تست شده) استفاده از یک کلاس سفارشی در پروژه‌ی WCF Service می‌باشد که مثال آن در اینجا آورده شده است.

برای رفع مشکل به پروژه WcfServiceScanner بازگشته و کلاس جدیدی را به نام CORSSupport ایجاد کرده و کدهای زیر را به آن اضافه کنید:

```
public class CORSSupport : IDispatchMessageInspector
    {
        Dictionary<string, string> requiredHeaders;
        public CORSSupport(Dictionary<string, string> headers)
        {
            requiredHeaders = headers ?? new Dictionary<string, string>();
        }

        public object AfterReceiveRequest(ref System.ServiceModel.Channels.Message request,
System.ServiceModel.IClientChannel channel, System.ServiceModel.InstanceContext instanceContext)
        {
            var httpRequest = request.Properties["httpRequest"] as HttpRequestMessageProperty;
            if (httpRequest.Method.ToLower() == "options")
                instanceContext.Abort();
            return httpRequest;
        }

        public void BeforeSendReply(ref System.ServiceModel.Channels.Message reply, object
correlationState)
        {
            var httpResponse = reply.Properties["httpResponse"] as HttpResponseMessageProperty;
            var httpRequest = correlationState as HttpRequestMessageProperty;

            foreach (var item in requiredHeaders)
            {
                httpResponse.Headers.Add(item.Key, item.Value);
            }
            var origin = httpRequest.Headers["origin"];
            if (origin != null)
                httpResponse.Headers.Add("Access-Control-Allow-Origin", origin);

            var method = httpRequest.Method;
            if (method.ToLower() == "options")
                httpResponse.StatusCode = System.Net.HttpStatusCode.NoContent;

        }
    }

    // Simply apply this attribute to a DataService-derived class to get
    // CORS support in that service
    [AttributeUsage(AttributeTargets.Class)]
    public class CORSSupportBehaviorAttribute : Attribute, IServiceBehavior
    {
        #region IServiceBehavior Members

        void IServiceBehavior.AddBindingParameters(ServiceDescription serviceDescription,
ServiceHostBase serviceHostBase, System.Collections.ObjectModel.Collection<ServiceEndpoint> endpoints,
BindingParameterCollection bindingParameters)
        {
        }

        void IServiceBehavior.ApplyDispatchBehavior(ServiceDescription serviceDescription,
ServiceHostBase serviceHostBase)
        {
            var requiredHeaders = new Dictionary<string, string>();

            //Chrome doesn't accept wildcards when authorization flag is true
```

```
            //requiredHeaders.Add("Access-Control-Allow-Origin", "*");
            requiredHeaders.Add("Access-Control-Request-Method", "POST,GET,PUT,DELETE,OPTIONS");
            requiredHeaders.Add("Access-Control-Allow-Headers", "Accept, Origin, Authorization, X-
Requested-With,Content-Type");
            requiredHeaders.Add("Access-Control-Allow-Credentials", "true");
            foreach (ChannelDispatcher cd in serviceHostBase.ChannelDispatchers)
            {
                foreach (EndpointDispatcher ed in cd.Endpoints)
                {
                    ed.DispatchRuntime.MessageInspectors.Add(new CORSSupport(requiredHeaders));
                }
            }
        }

        void IServiceBehavior.Validate(ServiceDescription serviceDescription, ServiceHostBase
serviceHostBase)
        {
        }

        #endregion
    }
```

فضاهای نام لازم برای این کلاس به شرح زیر می‌باشد:

```
using System.ServiceModel;
using System.ServiceModel.Channels;
using System.ServiceModel.Description;
using System.ServiceModel.Dispatcher;
```

کلاس ScannerService را باز کرده و آنرا به ویژگی

```
[CORSSupportBehavior]
    public class ScannerService : IScannerService
    {
```

مزین نمایید.

کار تمام است، یکبار دیگر ابتدا پروژه‌ی WcfServiecScanner و سپس پروژه هاست را Build کرده و برنامه‌ی هاست را اجرا کنید. اکنون مشاهده می‌کنید که با زدن دکمه‌ی اسکن، اسکنر فرم تنظیمات اسکن را نمایش می‌دهد که پس از زدن دکمه‌ی Scan، پروسه آغاز شده و پس از اتمام، تصویر اسکن شده در صفحه‌ی وب سایت نمایش داده می‌شود.

نظرات خوانندگان

نویسنده: امیر بختیاری
تاریخ: ۱۳۹۳/۱۱/۲۱ ۱:۴۸

با سلام و تشکر خدمت شما دوست گرامی

من تمام مراحل را به دقت ایجاد کردم ولی بازم همون خطایی که خودتون هم عنوان کردید را میده

```
Cross-Origin Request Blocked: The Same Origin Policy disallows reading the remote resource at
http://localhost:13748/. This can be fixed by moving the resource to the same domain or enabling CORS.
```

فایل پروژه را به طور کامل میزارم ببینید علت چیه WcfServiceScanner.zip

با تشکر مجدد

نویسنده: شروین ایرانی
تاریخ: ۱۳۹۳/۱۱/۲۱ ۹:۵

با سلام؛ آیا امکانش هست که سورس پروژه‌ی مثال رو هم قرار بدید؟ تشکر

نویسنده: امیر بختیاری
تاریخ: ۱۳۹۳/۱۱/۲۱ ۱۰:۳۱

سورس پروژه را از نظر قبلی می‌تونید دریافت کنید

عنوان: آشنایی با ساختار IIS قسمت سیزدهم
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۳/۱۱/۲۱ ۱۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: ASP.Net, IIS, Logfile, ftplogging

در مبحث قبلی گفتیم که ویرایش تنظیمات لاگ‌ها از طریق IIS یا ویرایش مستقیم فایل‌های کانفیگ میسر است. در این مقاله که قسمت پایانی مبحث لاگ هاست، در مورد ویرایش فایلهای کانفیگ صحبت می‌کنیم؛ همچنین استفاده از دستورات appcmd برای ویرایش و نهایتا کد نویسی در زبان سی شارپ و جاوااسکریپت.
تنظیمات لاگ سایت‌ها در فایل applicationhost در آدرس زیر قرار دارد:

```
C:\Windows\System32\inetsrv\config\applicationHost.config
```

برای هر تگ سایت، یک تگ <logfile> وجود دارد که ویژگی‌های Attributes آن، نوع ثبت لاگ را مشخص می‌کنند و می‌توانید مستقیما در اینجا به ویرایش بپردازید. البته ویرایش فایل کانفیگ از طریق IIS به طور مستقیم هم امکان پذیر است. برای این منظور در IIS سرور را انتخاب و از بین ماژول‌های قسمت management گزینه‌ی Configuration Editor را انتخاب کنید. در قسمت Section گزینه‌ی System.applicationhost را باز کرده و از زیر مجموعه‌های آن گزینه‌ی Site را برگزینید. در تنظیمات باز شده، گزینه collection را انتخاب کنید تا در انتهای سطر، دکمه‌ی ... پدیدار گردد. روی آن کلیک کنید تا محیطی ویرایشی باز گردد که به شما اجازه‌ی افزودن و ویرایش خصوصیت‌ها را می‌دهد. برای ویرایش لاگ‌ها باید خصوصیت logfile را باز کنید. اگر قسمت قبلی را مطالعه کرده باشید، باید بسیاری از این خصوصیت‌ها و مقادیر را بشناسید.
خصوصیات دیگری را هم مشاهده خواهید کرد که شاید قبلا ندیده‌اید که البته بستگی به ورژن IIS شما دارد؛ مثلا خصوصیت‌های maxLogLineLength و flushByEntryCountW3Clog از IIS8.5 اضافه شده اند.

جدول خصوصیت ها

خصوصیت	توضیح
customLogPluginClsid	یک پارامتر رشته‌ای اختیاری که در آن، آی دی کلاس یا کلاس‌هایی نوشته می‌شود که برای custom logging نوشته شده‌اند و این گزینه ترتیب اجرای آن‌ها را تعیین می‌کند.
directory	اختیاری است. محل ذخیره‌ی لاگ فایل‌ها را مشخص می‌کند و در صورتیکه ذکر نشود، همان مسیر پیش فرض است.
enabled	اختیاری است. فعال بودن سیستم لاگ برای آن سایت را مشخص می‌کند. مقدار پیش فرض آن true است.
flushByEntryCountW3CLog	این مقدار مشخص می‌کند چند رخداد باید اتفاق بیفتد تا عمل ذخیره سازی لاگ صورت گیرد. اگر بعد از هر رخداد عمل ثبت لاگ انجام شود، سرعت ثبت لاگ‌ها بالا می‌رود؛ ولی باعث استفاده‌ی مداوم از منابع و همچنین درخواست ثبت اطلاعات را روی دیسک خواهد داد و تاوان آن با زیاد شدن عملیات روی دیسک، پرداخته خواهد شد. ولی در حالتیکه چند رخداد را نگهداری سپس دسته‌ای ثبت کند، باعث افزایش کارآیی و راندمان سرور خواهد شد. در صورتیکه سرور به مشکلات لحظه‌ای برخورد می‌کند مقدار آن را کاهش دهید. مقدار پیش فرض 0 است. یعنی اینکه ثبت، بعد از 64000 لاگ خواهد بود.
localTimeRollover	نحوه‌ی نامگذاری فایل‌های لاگ را مشخص می‌کند که مقدار بولین گرفته و اختیاری است. به طور پیش فرض مقدار false دارد.

خصوصیت	توضیح
logExtFileFlags	این گزینه در حالتی به کارتان می‌آید که فرمت W3C را برای ثبت لاگ‌ها انتخاب کرده باشید و در اینجا مشخص می‌کنید که چه فیلدهایی باید در لاگ باشند و اگر بیش از یکی بود میتوان با ، (کاما) از هم جدایشان کرد.
logFormat	نوع فرمت ذخیره سازی لاگ‌ها
logSiteId	اختیاری است و مقدار پیش فرض آن true است. بدین معنا که کد یا شماره‌ی سایت هم در لاگ خواهد بود و این در حالتی است که گزارش در سطح سرور باشد. در غیر این صورت اگر هر سایت، جداگانه لاگی برای خود داشته باشد، ذکر نمی‌گردد.
logTargetW3C	اختیاری است و و مقدار file و * ETW را می‌گیرد که به طور پیش فرض روی File تنظیم است. در این حالت فایل لاگ‌ها در یک فایل متنی توسط http.sys ذخیره می‌شود. ولی موقعیکه از ETW استفاده می‌شود، http.sys با استفاده از iislogprovider داده‌ها را به سمت ETW ارسال میکند که منجر به اجرای سرویس Logsvc شده که از داده‌ها کوئری گرفته و آن‌ها را مستقیما از پروسه‌های کارگر جمع آوری و به سمت فایل لاگ ارسال می‌کند. همچنین انتخاب این دو گزینه نیز ممکن است.
maxLogLineLength	حداکثر تعداد خطی که یک لاگ میتواند داشته باشد تا اینکه بتوانید در مصرف دیسک سخت صرفه جویی کنید و بیشتر کاربرد آن برای لاگ‌های کاستوم است. این عدد باید از نوع Uint باشد و اختیاری است و از 2 تا 65536 مقدار میپذیرد که مقدار پیش فرض آن 65536 می‌باشد.
period	همان مبحث زمان بندی در مورد ایجاد فایل‌های لاگ است که در مقاله‌ی پیشین برسی کردیم و مقادیر Dialy,Hourly,monthly وweekly را می‌پذیرد. همچنین maxsize هم هست؛ موقعی که لاگ به نهایت حجمی که برای آن تعیین کردیم میرسد.
truncateSize	اختیاری است و مقدار آن از نوع int64 است. حداکثر حجم یک فایل لاگ را مشخص می‌کند تا اگر period روی maxsize تنظیم شده بود، حداکثر حجم را میتوان از اینجا تعیین نمود. در مقاله پیشین در این باره صحبت کردیم؛ حداقل عدد برای آن 1,048,576 است و اگر کمتر از آن بنویسید، سیستم همین عدد 1,048,576 را در نظر خواهد گرفت. مقدار پیش فرض آن 20971520 می باشد.

* ETW یا Event Tracing Windows ، سیستم و یا نرم افزاری برای عیب یابی و نظارت برای کامپوننت‌های ویندوزی است و یکی از استفاده کننده‌هایش IIS است که از ویندوز 2000 به بعد اضافه شده‌است. برای قطع کردن این ماژول در IIS هم میتوانید قسمت هفتم را بررسی نمایید و دنبال ماژول TracingModule بگردید. این ماژول به صورت Real time به ثبت رخدادهای IIS می‌پردازد.

به غیر از خصوصات بالا، خصوصیت customFields نیز از IIS 8.5 (به بعد) در دسترس است. اگر قصد دارید به غیر از فیلدهای W3c فیلدهای اختصاصی دیگری نیز داشته باشید، میتوان از این گزینه استفاده کرد. این فیلدهای کاستوم می‌توانند اطلاعاتشان را از response header ، request header و server variables دریافت کنند. این ویژگی تنها در فرمت W3C و در سطح سایت قابل انجام است. موقعی که یک فایل لاگ شامل فیلدهای اختصاصی شود، به انتها نام فایل X_ اضافه میگردد تا نشان دهد شامل یک فیلد

اختصاصی یا کاستوم است. نحوه تعریف آن در فایل applicationhost به شکل زیر است:

```
<system.applicationHost>
   <sites>
      <siteDefaults>
         <logFile logFormat="W3C"
            directory="%SystemDrive%\inetpub\logs\LogFiles"
            enabled="true">
            <customFields>
               <clear/>
               <add logFieldName="ContosoField" sourceName="ContosoSource"
                  sourceType="ServerVariable" />
            </customFields>
         </logFile>
      </siteDefaults>
   </sites>
</system.applicationHost>
```

تغییر تنظیمات لاگ با Appcmd

```
appcmd.exe set config -section:system.applicationHost/sites /siteDefaults.logFile.enabled:"True"
/commit:apphost
appcmd.exe set config -section:system.applicationHost/sites /siteDefaults.logFile.logFormat:"W3C"
/commit:apphost
appcmd.exe set config -section:system.applicationHost/sites
/siteDefaults.logFile.directory:"%SystemDrive%\inetpub\logs\LogFiles" /commit:apphost
```

تنظیمات تگ لاگ با برنامه نویسی و اسکریپت نویسی هچنین با رفرنس Microsoft.web.administration در پروژه‌های دات نتی خود میتوانید امکان ویرایش تنظیمات را در برنامه‌های خود نیز داشته باشید:

```
using System;
using System.Text;
using Microsoft.Web.Administration;

internal static class Sample
{
   private static void Main()
   {
      using (ServerManager serverManager = new ServerManager())
      {
         Configuration config = serverManager.GetApplicationHostConfiguration();
         ConfigurationSection sitesSection = config.GetSection("system.applicationHost/sites");
         ConfigurationElement siteDefaultsElement = sitesSection.GetChildElement("siteDefaults");

         ConfigurationElement logFileElement = siteDefaultsElement.GetChildElement("logFile");
         logFileElement["logFormat"] = @"W3C";
         logFileElement["directory"] = @"%SystemDrive%\inetpub\logs\LogFiles";
         logFileElement["enabled"] = true;

         serverManager.CommitChanges();
      }
   }
}
```

با استفاده از اسکریپت نویسی توسط جاوااسکریپت و وی بی اسکریپت هم نیز این امکان مهیاست:

```
var adminManager = new ActiveXObject('Microsoft.ApplicationHost.WritableAdminManager');
adminManager.CommitPath = "MACHINE/WEBROOT/APPHOST";
var sitesSection = adminManager.GetAdminSection("system.applicationHost/sites",
"MACHINE/WEBROOT/APPHOST");
var siteDefaultsElement = sitesSection.ChildElements.Item("siteDefaults");

var logFileElement = siteDefaultsElement.ChildElements.Item("logFile");
logFileElement.Properties.Item("logFormat").Value = "W3C";
logFileElement.Properties.Item("directory").Value = "%SystemDrive%\\inetpub\\logs\\LogFiles";
logFileElement.Properties.Item("enabled").Value = true;

adminManager.CommitChanges();
```

FTP Logging

برای اطمینان از نصب Ftp logging موقع نصب، باید از مورد زیر مطمئن باشید:

IIS را باز کنید و در لیست درختی، سرور را انتخاب کنید. در قسمت FTP میتوانید گزینه‌ی Ftp logging را ببینید. تنظیمات این قسمت هم دقیقا همانند قسمت logging میباشد و همان موارد برای آن هم صدق می‌کند.

بررسی تگ آن در applicationhost

تگ این نوع لاگ در فایل applicationhost در زیر مجموعه‌ی تگ <site> به شکل زیر نوشته می‌شود:

```
<system.ftpServer>
   <log centralLogFileMode="Central">
      <centralLogFile enabled="true" />
   </log>
</system.ftpServer>
```

گزینه **centralLogFileMode** دو مقدار central و site را می‌پذیرد. اگر گزینه‌ی central انتخاب شود، یعنی همه‌ی لاگ‌ها را داخل یک فایل در سطح سرور ثبت کن ولی اگر گزینه‌ی site انتخاب شده باشد، لاگ هر سایت در یک فایل ثبت خواهد شد.

گزینه‌ی **logInUTF8** یک خصوصیت اختیاری است که مقدار پیش فرض آن true میباشد. در این حالت باید تمامی رشته‌ها به انکدینگ UTF-8 تبدیل شوند.

همانطور که می‌بینید تگ log در بالا یک تگ فرزند هم به اسم centralLogFile دارد که همان خصوصیات جدول بالا در آن مهیاست.

دسترسی به تنظیمات این قسمت توسط دستور Appcmd:

```
appcmd.exe set config -section:system.ftpServer/log /centralLogFileMode:"Central" /commit:apphost

appcmd.exe set config -section:system.ftpServer/log /centralLogFile.enabled:"True" /commit:apphost
```

دسترسی به تنظیمات این قسمت توسط دات نت:

```
using System;
using System.Text;
using Microsoft.Web.Administration;

internal static class Sample
{
   private static void Main()
   {
      using (ServerManager serverManager = new ServerManager())
      {
         Configuration config = serverManager.GetApplicationHostConfiguration();

         ConfigurationSection logSection = config.GetSection("system.ftpServer/log");
         logSection["centralLogFileMode"] = @"Central";

         ConfigurationElement centralLogFileElement = logSection.GetChildElement("centralLogFile");
         centralLogFileElement["enabled"] = true;

         serverManager.CommitChanges();
      }
   }
}
```

دسترسی به تنظیمات این قسمت توسط Javascript:

```
var adminManager = new ActiveXObject('Microsoft.ApplicationHost.WritableAdminManager');
adminManager.CommitPath = "MACHINE/WEBROOT/APPHOST";

var logSection = adminManager.GetAdminSection("system.ftpServer/log", "MACHINE/WEBROOT/APPHOST");
logSection.Properties.Item("centralLogFileMode").Value = "Central";

var centralLogFileElement = logSection.ChildElements.Item("centralLogFile");
centralLogFileElement.Properties.Item("enabled").Value = true;

adminManager.CommitChanges();
```