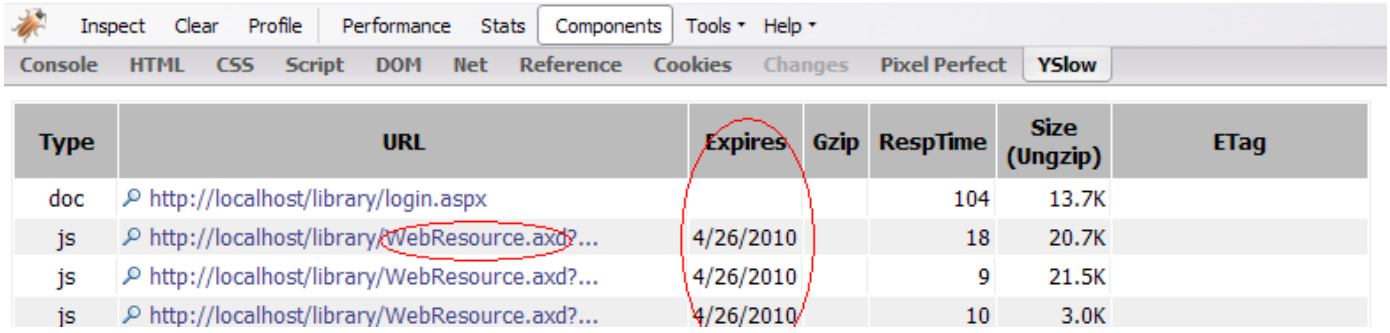
عنوان: **کش کردن اطلاعات غیر پویا در ASP.Net - قسمت دوم**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۲/۰۶ ۱۹:۲۲:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: ASP.Net

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.

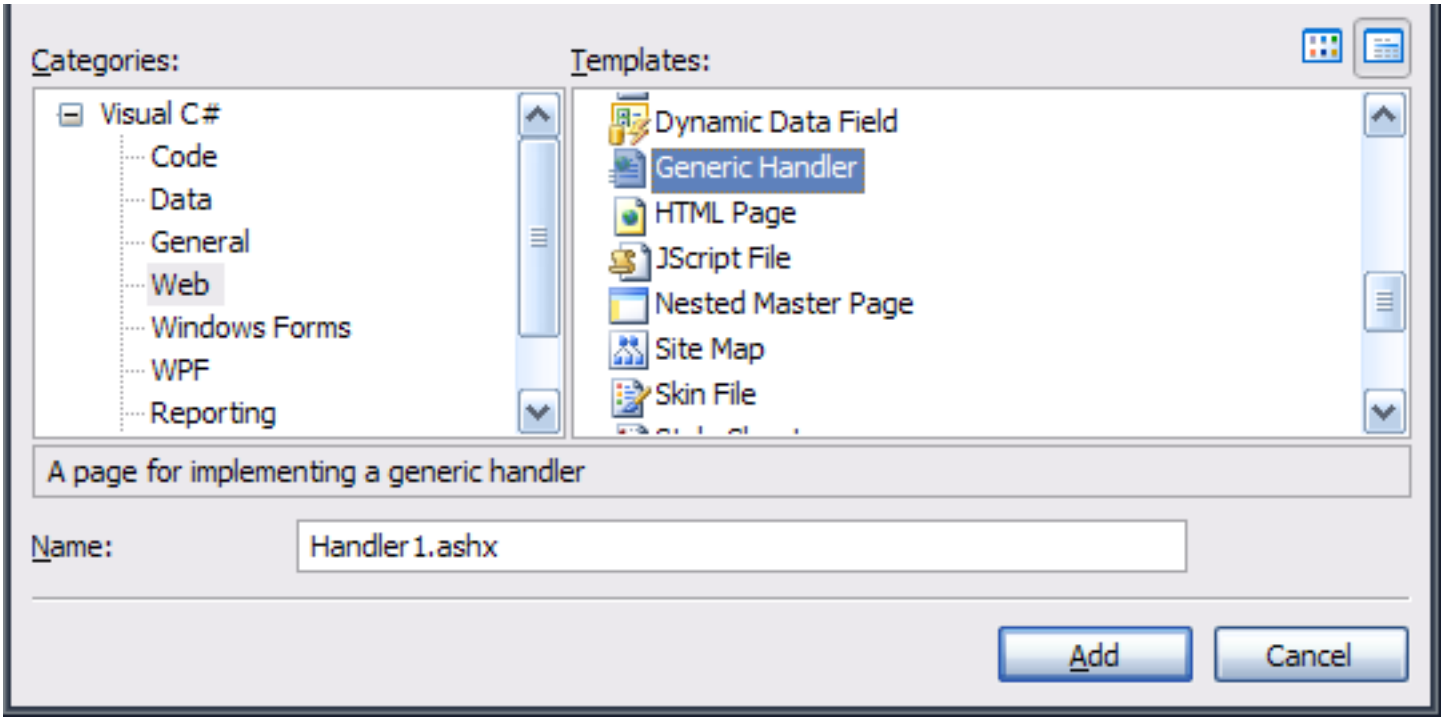


Type	URL	Expires	Gzip	RespTime	Size (Ungzip)	ETag
doc	http://localhost/library/login.aspx			104	13.7K	
js	http://localhost/library/WebResource.axd?...	4/26/2010		18	20.7K	
js	http://localhost/library/WebResource.axd?...	4/26/2010		9	21.5K	
js	http://localhost/library/WebResource.axd?...	4/26/2010		10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

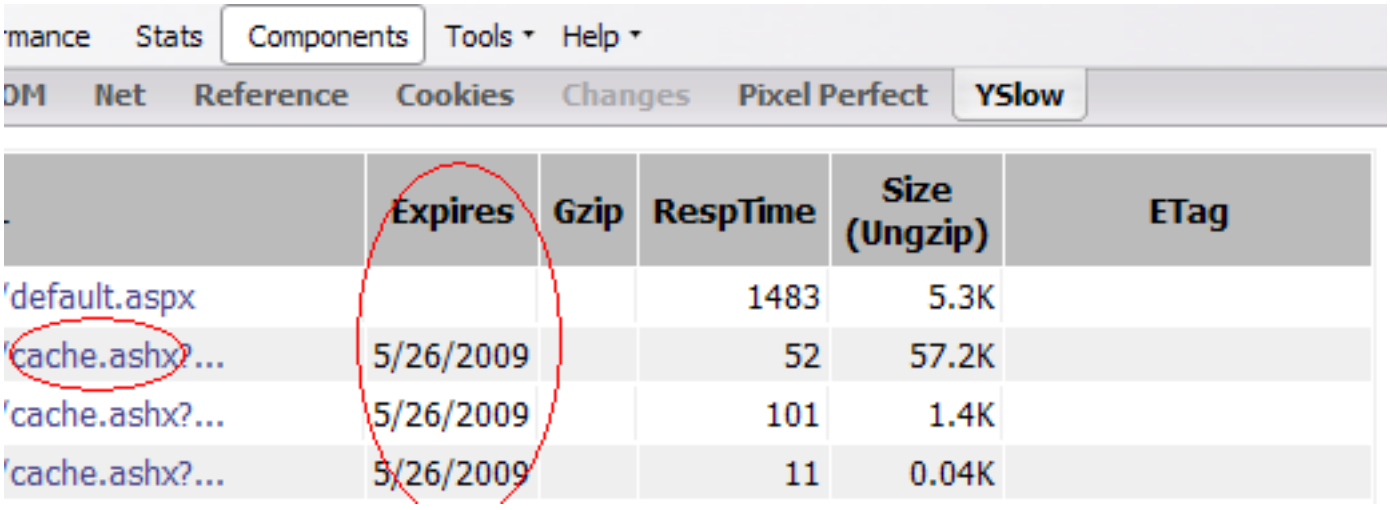
```
<link type="text/css" href="cache.ashx?v=1&file=site.css&contetType=text/css" rel="Stylesheet" />
```

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: ۱۳۸۸/۰۴/۲۵ ۲۱:۲۴:۵۴

خیلی عالی بود. ممنون

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

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

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

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: ۱۳۸۹/۰۲/۱۲ ۱۴:۰۲:۱۷

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: ۱۳۸۹/۰۲/۱۲ ۱۴:۲۸:۲۵

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