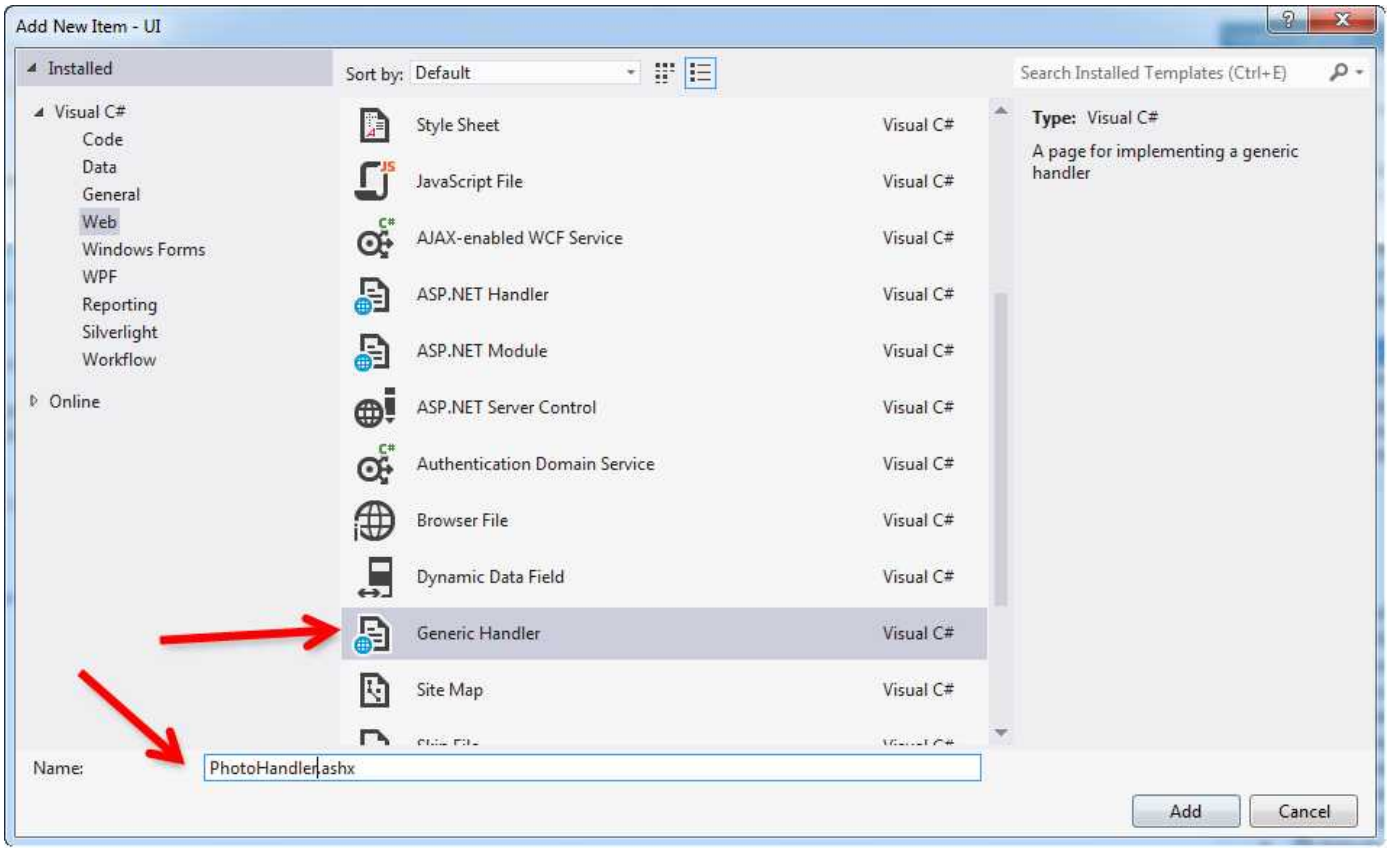
عنوان: **تغییر اندازه تصاویر #2**
نویسنده: صابر فتح الهی
تاریخ: ۱۳۹۱/۱۲/۱۱ ۱۲:۳۵
آدرس: www.dotnettips.info
گروه‌ها: #ASP.Net, C

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).

-2بازیابی 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

تغییر اندازه تصاویر #2

```
<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 : visulstudio2013 , mvc5

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 »