

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 httpHandler پرداخته‌ایم، ولی بهتر هست در این مثال کمی حالت پیشرفته‌تر آن را 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 برخورد 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 شما یا اسکیمای 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 دهد.

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، شما خود را بر اساس نام فایل مشخص می‌کنیم و خصوصیت overrideModeDefault، یک قفل گذار امنیتی برای تغییر محتوای section است. 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 مقادیری 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" />
        </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 صورت گیرد؛ 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` 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 شیء `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 انجام می‌دهیم:

```
appcmd set config "Default Web Site/mypictures/" -section:handlers
/+[name='JPGImageCopyrightHandler',path='*.jpg',verb='GET',type='IIS7Demos.imageCopyrightHandler']
```

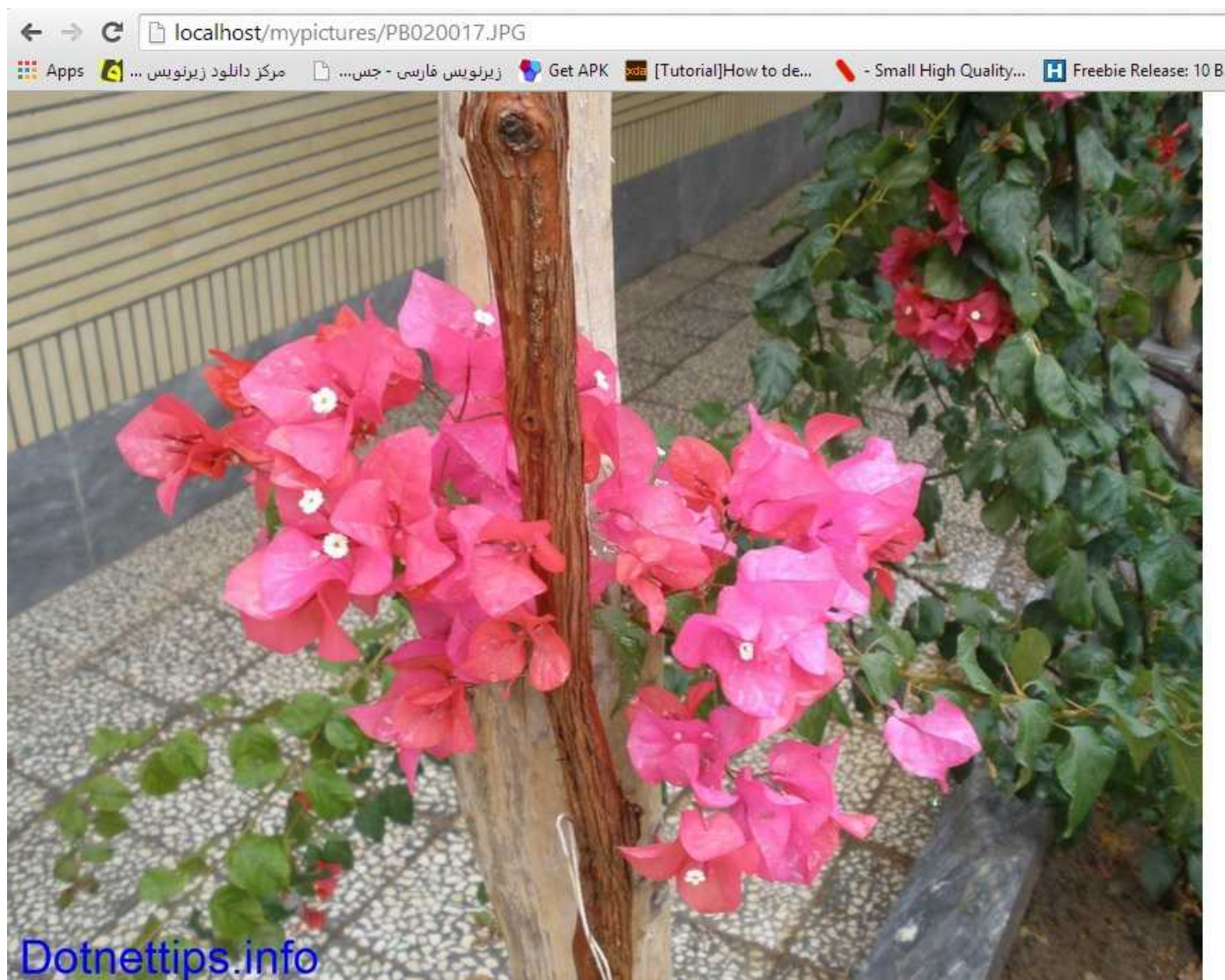
هندلر مورد نظر تنها برای این اپلیکیشن و در مسیر `mypicture` فعال شده و در قسمت `name`، یک نام اختیاری بدون فاصله و `unique` بر می‌گزینیم. در قسمت `path` نوع فایل‌هایی را که نیاز به هندلر هست، مشخص کردیم و در قسمت `verb` گفته‌ایم که تنها برای درخواست‌های نوع `GET`، هندلر را اجرا کن و در قسمت `type` هم 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 احیاناً 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.