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>
```

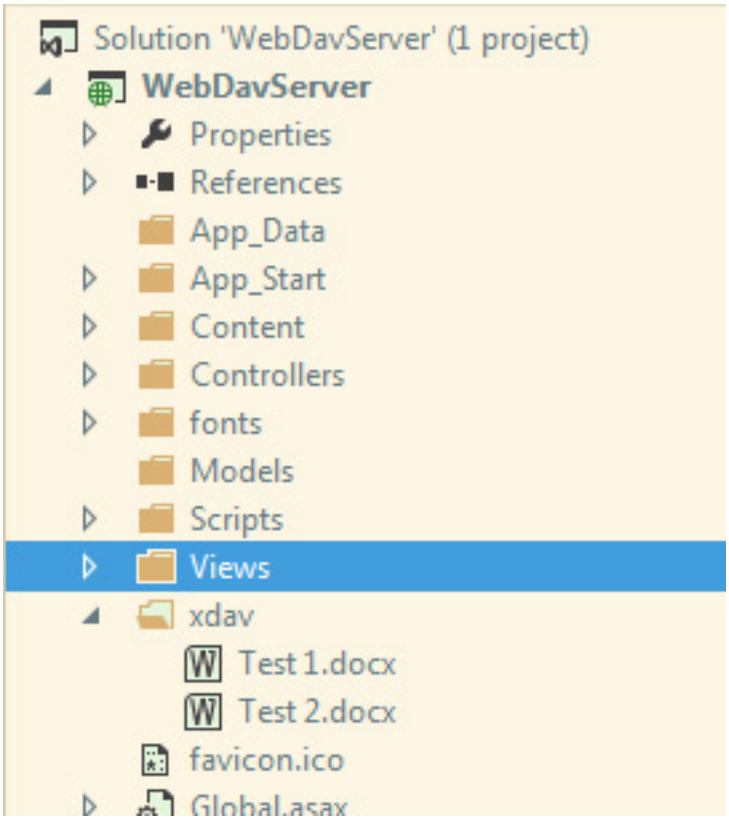
خاصیت Name برای xDav نشانگر درخواست‌هایی است که باید توسط این ماژول اجرا شوند. در اینجا یعنی درخواست‌هایی که آدرس آن‌ها شامل "/xdav/" باشد، توسط این ماژول Handle می‌شوند. عبارت بعد از مقدار Name در URL هم طبیعتا نام فایل مورد نظر شماست.

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);
```

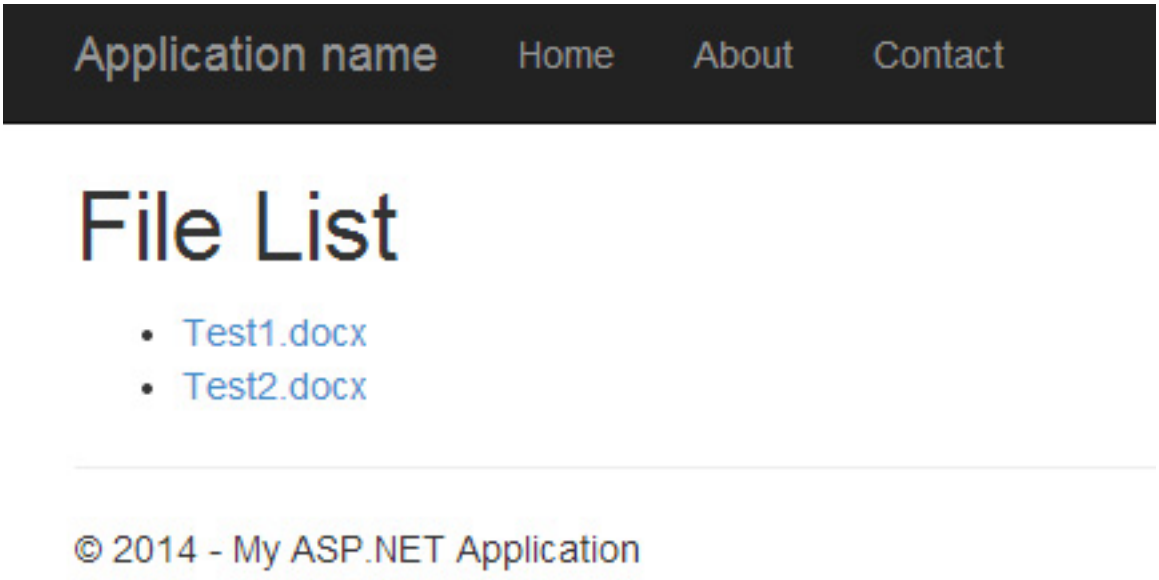
یک لیست از فایل هایی که در پوشه‌ی webDav قرار دارند تهیه می‌کنیم و به View ارسال می‌کنیم. View را هم مثل زیر بازنویسی می‌کنیم.

```
@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.