

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: ۲۲:۳۰ ۱۳۹۱/۰۴/۲۵

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: ۲۲:۳۵ ۱۳۹۱/۰۴/۲۵

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: ۲۲:۴۳ ۱۳۹۱/۰۴/۲۵

محدودیت زمانی 15 دقیقه را برای همین منظور اعمال می‌کنند.

نویسنده: مجید مقصودی پور
تاریخ: ۱۲:۱۶ ۱۳۹۱/۰۴/۲۶

باسلام:

میشه گفت یک راه برای جلوگیری از هکرها میباشد هکرها میتونه هم انسان باشه و یا هم یک برنامه کاربردی (روبوته) 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: ۱۲:۱۸ ۱۳۹۱/۰۴/۲۶

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 ارزش استفاده میکنه تغییر بده.
راه‌های رسیدن به خدا زیاده.

نویسنده: وحید نصیری
تاریخ: ۱۶:۵۹ ۱۳۹۱/۰۴/۲۶

از این نمونه‌ها من در این سایت زیاد دارم. کسانی که مرتباً 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
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: ۱۷:۲ ۱۳۹۱/۰۴/۲۶

شما برای جلوگیری از این IP ها کاری میکنید؟

نویسنده: وحید نصیری
تاریخ: ۱۷:۵ ۱۳۹۱/۰۴/۲۶

بله. یک فیلتر آنتی داس درست کردم که شبانه روز دسترسی این نوع مزاحم ها رو حذف می کنه. تا الان هر روز به طور میانگین حداقل 2 بار در زمان های مختلف به این سایت حمله شده.

نویسنده: مجید مقصودی پور
تاریخ: ۱۷:۷ ۱۳۹۱/۰۴/۲۶

اگر در قالب یک مقاله ارائه بدید خیلی مفید واقع میشه. باتشکر.

نویسنده: وحید نصیری
تاریخ: ۱۰:۲۲ ۱۳۹۱/۰۴/۲۷

تقریباً [بر همین اساس](#) پیاده سازی شده. اطلاعات در کش سیستم ذخیره می شن و به این ترتیب IP های مهاجم رو میشه تشخیص داد.

نویسنده: امیر
تاریخ: ۱۴:۲۶ ۱۳۹۱/۰۴/۲۷

با درود و عرض ادب

استاد نصیری

سایت ما هر چند وقت یک بار با ip های مختلفی هک می شود

همه جور محدودیت و enjection هم اعمال کردم.

به نظرتون این مشکل رو چی کار باید کرد.؟

ممون از راهنمایی تون

نویسنده: وحید نصیری
تاریخ: ۱۴:۲۹ ۱۳۹۱/۰۴/۲۷

نمیشه همینطوری ندیده اظهار نظر کرد. نیاز به [code review](#) داره.

نویسنده: صابر فتح اللهی
تاریخ: ۱۵:۱۹ ۱۳۹۱/۰۴/۲۷

سلام

البته توی این [آدرس](#) به فارسی توضیح داده شده همون روش

نویسنده: صابر فتح الهی
تاریخ: ۱۰:۲۳ ۱۳۹۱/۰۵/۱۶

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 درسته؟
یعنی دیگه نمی‌تونن بیان توی سایت اما این امکان گذاشتم که بعدا اونهارو از لیست سیاه در بیارم

نویسنده: وحید نصیری
تاریخ: ۱۳:۵۷ ۱۳۹۱/۰۵/۱۶

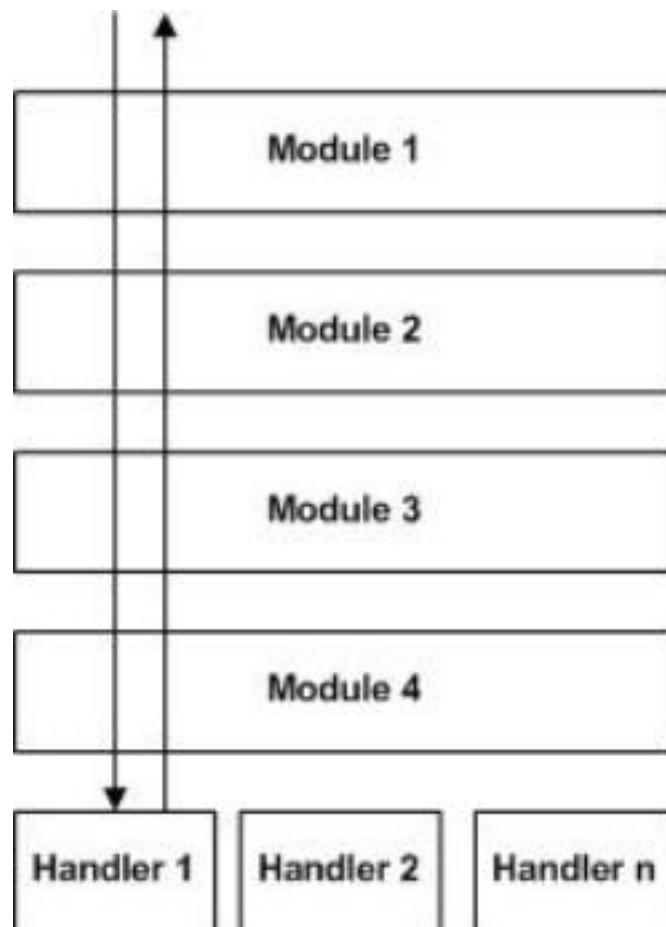
نه. IP های کاربران در ایران عموماً ثابت نیست. نصف روز بسته شود کافی است.

قبل از اینکه به httpmodule ها بپردازیم، اجازه بدید کمی 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. httpmodule ها اسمبلی یا ماژول‌هایی هستند 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 بخش‌هاست. با هر درخواستی که سرور ارسال می‌شود، httpmodule ها صدا زده می‌شوند و به برنامه نویسی 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 state ها و Authentication می‌باشند.

تصویر زیر نحوه‌ی ارسال و بازگشت یک درخواست را به سمت httphandler نشان می‌دهد



برنامه نویسی هم میتواند با استفاده از اینترفیسهای IHttpHandler و IHttpModule در درخواستها دخالت نماید. برای شروع یک کلاس ایجاد کنید که اینترفیس IHttpModule را پیاده سازی می کند. 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 شیء صدا زده می شود. متد Init یک آرگومان از نوع httpapplication دارد که مانند رسم نامگذاری متغیرها، بیشتر به اسم context یا app نام گذاری می شوند:

```
public void Init(HttpApplication app)
{
    app.BeginRequest += new EventHandler(OnBeginRequest);
}

public void Dispose(){ }
```

همانطور که می بینید 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 می باشند:

این رویداد اولین رویدادی است که اجرا می شود، هر نوع عملی که میخواهید در ابتدای ارسال درخواست انجام دهید، باید در این قسمت قرار بگیرد؛ مثلاً قرار دادن یک بنر بالای صفحه	BeginRequest
خود دانت از یک سیستم امنیتی توکار بهره مند است و اگر می خواهید در مورد آن خصوصی سازی انجام بدهید، این رویداد می تواند کمکتان کند	AuthenticateRequest
بعد از رویداد بالا، این رویداد برای شناسایی انجام می شود. مثلاً دسترسی ها؛ دسترسی به قسمت هایی خاصی از منابع به او داده شود و قسمت هایی بعضی از منابع از او گرفته شود.	AuthorizeRequest
این رویداد برای کش کردن اطلاعات استفاده می شود. خود دانت تمامی این رویدادها را به صورت تو کار فراهم آورده است؛ ولی اگر باز خصوصی سازی خاصی مد نظر شماست می توانید در این قسمت ها، تغییراتی را اعمال کنید. مثلاً ایجاد file caching به جای memory cache و ...	ResolveRequestCache
این قسمت برای مدیریت state می باشد مثلاً مدیریت session ها	AcquireRequestState
این رویداد قبل از httphandler اجرا می شود.	PreRequestHandlerExecute
این رویداد بعد از httphandler اجرا می شود.	PostRequestHandlerExecute
این رویداد برای این صدا زده می شود که به شما بگوید عملیات درخواست پایان یافته است و باید state های ایجاد شده را release یا رها کنید.	ReleaseRequestState
برای خصوصی سازی output cache بکار می رود.	UpdateRequestCache
عملیات درخواست پایان یافته است. در صورتیکه قصد نوشتن دیباگری در طی تمامی عملیات دارید، میتواند به شما کمک کند.	EndRequest
این رویداد قبل از ارسال اطلاعات هدر هست. اگر قصد اضافه کردن اطلاعاتی به هدر دارید، این رویداد را به کار ببرید.	PreSendRequestHeaders
این رویداد موقعی صدا زده می شود که متد response.flush فراخوانی شود. اگر می خواهید به محتوا چیزی اضافه کنید، از اینجا کمک بگیرید.	PreSendRequestContent

این رویداد اولین رویدادی است که اجرا می‌شود، هر نوع عملی که می‌خواهید در ابتدای ارسال درخواست انجام دهید، باید در این قسمت قرار بگیرید؛ مثلاً قرار دادن یک بنر بالای صفحه	BeginRequest
این رویداد موقعی رخ می‌دهد که یک استثنای مدیریت نشده رخ بدهد. برای نوشتن سیستم خطایابی خصوصی از این قسمت عمل کنید.	Error
این رویداد موقعی صدا زده می‌شود که درخواست، بنا به هر دلیلی پایان یافته است. برای عملیات پاکسازی و .. می‌شود از آن استفاده کرد. مثلاً یک جور rollback برای کارهای انجام گرفته.	Disposed

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 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 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 namespace.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 <system.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 قصد دارید که پروژه‌ی شما در هر دو IIS مورد حمایت قرار گیرد، باید این مازول 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 مازول 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

HttpModule و Global.asax

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: ۱۰:۲۴ ۱۳۹۳/۰۹/۲۹

با سلام؛ ممنون بابت مطلب خوبی که ارسال کردید.
من می‌خواهم از httpModule 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: ۱۴:۲۲ ۱۳۹۳/۰۹/۲۹

بله. در جدولی که تهیه کردند این مورد دقیقاً ذکر شده:
» **AcquireRequestState** : این قسمت برای مدیریت state می‌باشد مثلاً 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: ۱۴:۵۰ ۱۳۹۳/۰۹/۲۹

از رویداد AcquireRequestState استفاده کنید ، همانطور که گفتیم این رویداد برای مدیریت stateهاست ، عموماً در رویدادهای ابتدایی sessionها هنوز ایجاد نشده‌اند
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: ۱۷:۱۰ ۱۳۹۳/۰۹/۲۹

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")
{
}
```

همانطور که در مطلب قبلی گفتیم، در این مطلب قرار است به WAS بپردازیم؛ در دنباله متن قبلی گفتیم که دومین وظیفه WWW Service این است: موقعی که 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 درخواست‌هایی 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 listener adapter ارتباط بین WAS و listenerهای مختلف را در IIS، برقرار می‌کند. آداپتورها اطلاعات لازم را از WAS می‌گیرند و به listenerهای مربوطه انتقال می‌دهند تا listenerها بر اساس آن تنظیمات یا پیکربندی‌ها، 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 درخواست‌هایی که از طریق پروتوکول 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 تغییر می‌کند، 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 درخواست‌هاست، 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 application pool برای چندین protocol استفاده کنید. به عنوان مثال شما یکی سرویس XML دارید که می‌توانید از آن برای سرویس دهی به پروتوکول‌های Http و net.tcp بهره بگیرید.

ماژول‌ها در IIS

قبلاً 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 دقیقاً 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 پروتوکول‌ها 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 پروتوکول 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 در authentication 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، نوع 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 response به منظور ایجاد خروجی	Inetsrv\Cgi.dll
DefaultDocumentModule	تلاش برای ساخت یک سند پیش فرض برای درخواست‌هایی که دایرکتوری والد ارسال می‌شود	Inetsrv\Defdoc.dll
DirectoryListingModule	لیست کردن محتوای یک دایرکتوری	Inetsrv\dirlist.dll
IsapiModule	میزبانی فایل‌های ISAPI	Inetsrv\Isapi.dll
IsapiFilterModule	پشتیبانی از فیلترهای ISAPI	Inetsrv\Filter.dll
ServerSideIncludeModule	پردازش کدهای include شده سمت سرور	Inetsrv\Iis_ssi.dll
StaticFileModule	ارائه فایل‌های ایستا	Inetsrv\Static.dll
FastCgiModule	پشتیبانی 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 عیب‌یابی و لاگ کردن

Inetsrv\Logcust.dll	بارگزاری ماژول‌های خصوصی سازی شده جهت لاگ کردن	CustomLoggingModule
Inetsrv\Iisfrieb.dll	برای ردیابی درخواست‌های ناموفق	FailedRequestsTracingModule
Inetsrv\Loghttp.dll	دریافت اطلاعات و پردازش وضعیت http.sys برای لاگ کردن	HttpLoggingModule
Inetsrv\Iisreqs.dll	ردیابی درخواست‌هایی که در حال حاضر در پروسه‌های کارگر در حال اجرا هستند و گزارش اطلاعاتی در مورد وضعیت اجرا و کنترل رابط برنامه نویسی کاربردی.	RequestMonitorModule
Inetsrv\Iisetw.dll	گزارش رخدادهای Microsoft Event Tracing for Windows یا به اختصار ETW	TracingModule

ماژول‌های مدیریتی و نظارتی بر کل ماژول‌ها

Microsoft.NET\Framework\v2.0.50727\webengine.dll	مدیریتی بر ماژول‌های غیر native که در پایین قرار دارند.	ManagedEngine
Inetsrv\validcfg.dll	اعتبارسنجی خطاها، مثل موقعی که برنامه در حالت integrated اجرا شده و ماژول‌ها یا هندلرها در system.web تعریف شده‌اند.	ConfigurationValidationModule

از IIS6 به بعد در حالت integrated و ماقبل، در حالت کلاسیک می‌باشند. اگر [مقاله ماژول‌ها](#) 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 می‌آورد 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 کاملاً آشنا باشید؛ برای تایید هویت کاربر	System.Web.Security.FormsAuthenticationModule

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