14,408,832 members Sign in

articles quick answers discussions features community help

Search for articles, questions, tips

Articles » Web Development » ASP.NET » General

The Two Interceptors: HttpModule and HttpHandlers

Shivprasad koirala

4 Mar 2009 CPOL

Rate this: ★★★★★ 4.91 (113 votes)

The two interceptors: HttpModule and HttpHandlers

Download source code and demo - 12.02 KB

Table of Contents

Introduction

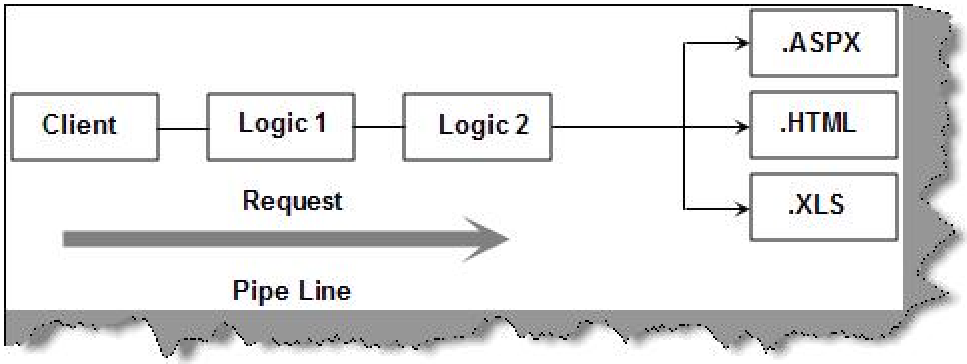
Many times we want to implement pre-processing logic before a request hits the IIS resources. For instance you would like to apply security mechanism, URL rewriting, filter something in the request, etc. ASP.NET has provided two types of interception `HttpModule` and `HttpHandler`. This article walks through it.

For the last few days, I have been writing and recording videos in design patterns, UML, FPA, Enterprise blocks and lot more. You can watch the videos here.

You can download my 400 .NET FAQ EBook from here.

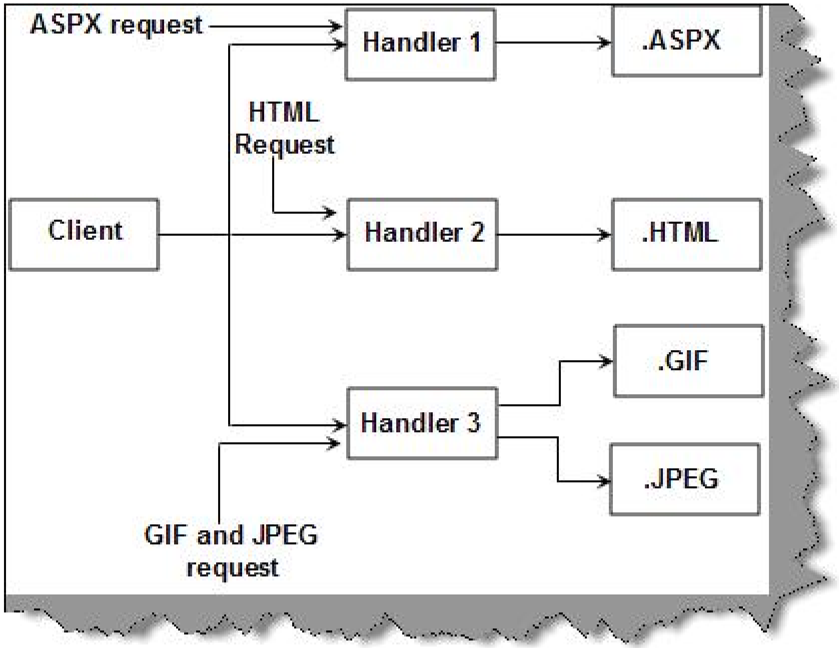
The Problem

Many times we need to inject some kind of logic before the page is requested. Some of the commonly used pre-processing logics are stat counters, URL rewriting, authentication / authorization and many more. We can do this in the code behind but then that can lead to lot of complication and tangled code. The code behind will not solve the purpose because in some implementations like authorization, we want the logic to execute before it reaches the resource. ASP.NET provides two ways of injecting logic in the request pipeline `HttpHandlers` and `HttpModules`.



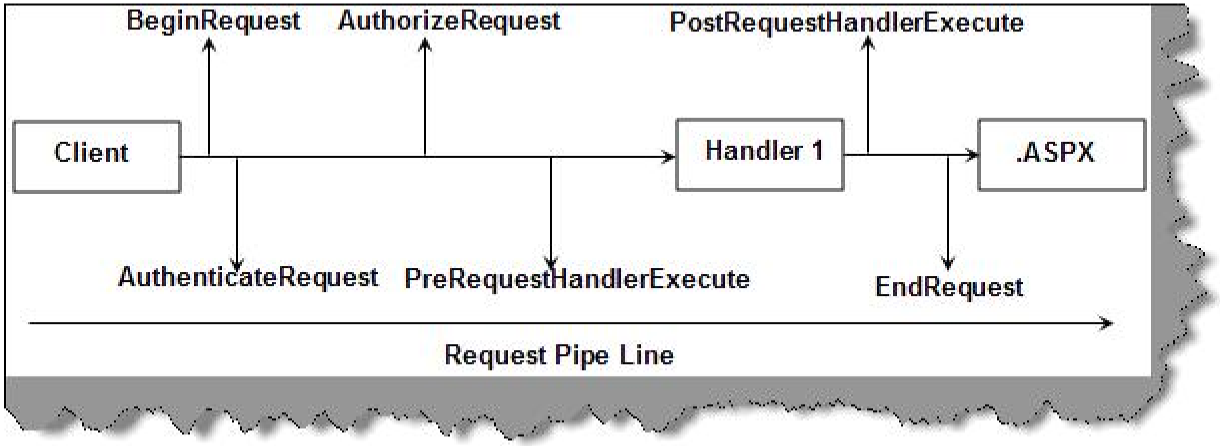
HttpHandler - The Extension Based Preprocessor

`HttpHandler` help us to inject pre-processing logic based on the extension of the file name requested. So when a page is requested, `HttpHandler` executes on the base of extension file names and on the base of verbs. For instance, you can visualize from the figure below how we have different handlers mapped to file extension. We can also map one handler to multiple file extensions. For instance, when any client requests for file with extension 'GIF' and 'JPEG', `handler3` pre-processing logic executes.



HttpModule - The Event Based Preprocessor

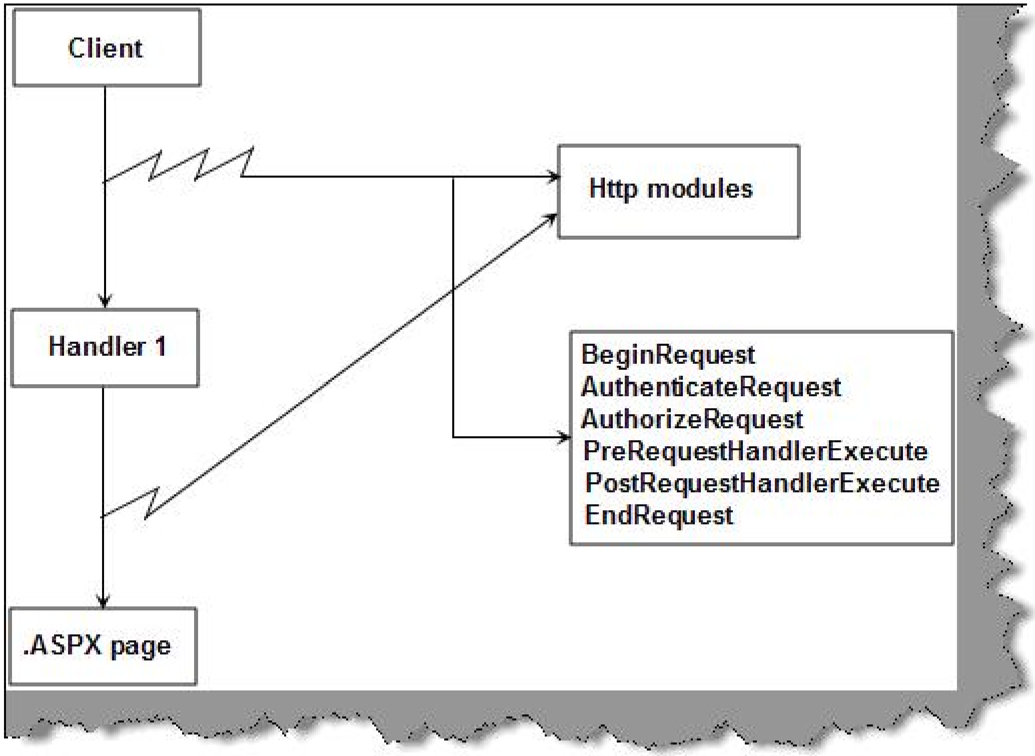
`HttpModule` is an event based methodology to inject pre-processing logic before any resource is requested. When any client sends a request for a resource, the request pipeline emits a lot of events as shown in the figure below:



Below is a detailed explanation of the events. We have just pasted this from here.

- `BeginRequest`: Request has been started. If you need to do something at the beginning of a request (for example, display advertisement banners at the top of each page), synchronize this event.
- `AuthenticateRequest`: If you want to plug in your own custom authentication scheme (for example, look up a user against a database to validate the password), build a module that synchronizes this event and authenticates the user in a way that you want to.
- `AuthorizeRequest`: This event is used internally to implement authorization mechanisms (for example, to store your access control lists (ACLs) in a database rather than in the file system). Although you can override this event, there are not many good reasons to do so.
- `PreRequestHandlerExecute`: This event occurs before the HTTP handler is executed.
- `PostRequestHandlerExecute`: This event occurs after the HTTP handler is executed.
- `EndRequest`: Request has been completed. You may want to build a debugging module that gathers information throughout the request and then writes the information to the page.

We can register these events with the `HttpModules`. So when the request pipe line executes depending on the event registered, the logic from the modules is processed.



BeginRequest
AuthenticateRequest
AuthorizeRequest
PreRequestHandlerExecute
PostRequestHandlerExecute
EndRequest

The Overall Picture of Handler and Modules

Now that we have gone through the basics, let's understand what is the Microsoft definition for handler and modules to get the overall picture.

Reference: INFO: ASP.NET HTTP Modules and HTTP Handlers Overview

“Modules are called before and after the handler executes. Modules enable developers to intercept, participate in, or modify each individual request. Handlers are used to process individual endpoint requests. Handlers enable the ASP.NET Framework to process individual HTTP URLs or groups of URL extensions within an application. Unlike modules, only one handler is used to process a request”.

Image 5

Steps to Implement HttpHandlers

Step 1

`HttpHandlers` are nothing but classes which have pre-processing logic implemented. So the first thing is to create a class project and reference `System.Web` namespace and implement the `IHttpHandler` interface as shown in the below code snippet. `IHttpHandler` interface has two methods which needs to be implemented; one is the `ProcessRequest` and the other is the `IsResuable`. In the `ProcessRequest` method, we are just entering the URL into the file and displaying the same into the browser. We have manipulated the context response object to send the display to the browser.

Hide Copy Code

```
using System;
using System.Web;
using System.IO;
namespace MyPipeLine
{
public class clsMyHandler : IHttpHandler
{
public void ProcessRequest(System.Web.HttpContext context)
{
context.Response.Write("The page request is " + context.Request.RawUrl.ToString());
StreamWriter sw = new StreamWriter(@"C:\requestLog.txt",true);
sw.WriteLine("Page requested at " + DateTime.Now.ToString() +
            context.Request.RawUrl); sw.Close();
}
public bool IsReusable
{
get
{
return true;
}
}
}
```

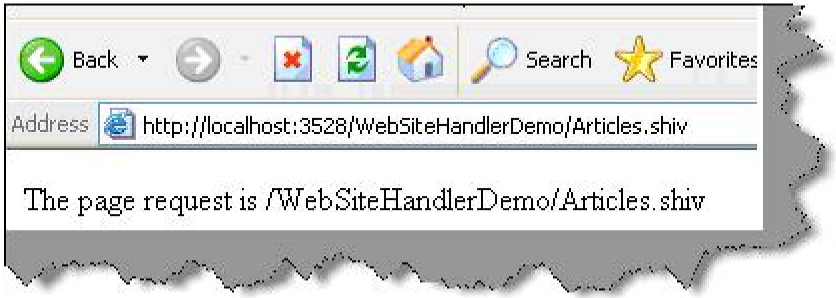
Step 2

In step 2, we need to make an entry of `HttpHandlers` tag. In the tag, we need to specify which kind of extension requested will invoke our class.

Hide Copy Code

```
<system.web>
<httpHandlers>
<add verb="*" path="*.Shiv,*.Koirala" type="MyPipeLine.clsMyHandler, MyPipeLine"/>
</httpHandlers>
</system.web>
```

Once done, request for page name with extension 'Shiv' and you should see a display as shown below. So what has happened is when the IIS sees that request is for a '*.shiv*' page extension, it just calls the `clsMyHandler` class pre-processing logic.



http://localhost:3528/WebSiteHandlerDemo/Articles.shiv

The page request is /WebSiteHandlerDemo/Articles.shiv

Steps to Implement HttpModule

Step 1

As discussed previously, `HttpModule` is an event pre-processor. So the first thing is to implement the `IHttpModule` and register the necessary events which this module should subscribe. For instance, we have registered in this sample for `BeginRequest` and `EndRequest` events. In those events, we have just written an entry on to the log file.

Hide Copy Code

```
public class clsMyModule : IHttpModule
{
public clsMyModule()
{}
public void Init(HttpApplication objApplication)
{
// Register event handler of the pipe line
objApplication.BeginRequest += new EventHandler(this.context_BeginRequest);
objApplication.EndRequest += new EventHandler(this.context_EndRequest);
}
public void Dispose()
{
}
public void context_EndRequest(object sender, EventArgs e)
{
StreamWriter sw = new StreamWriter(@"C:\requestLog.txt",true);
sw.WriteLine("End Request called at " + DateTime.Now.ToString()); sw.Close();
}
public void context_BeginRequest(object sender, EventArgs e)
{
StreamWriter sw = new StreamWriter(@"C:\requestLog.txt",true);
sw.WriteLine("Begin request called at " + DateTime.Now.ToString()); sw.Close();
}
}
```

Step 2

We need to enter those module entries into the `HttpModule` tag as shown in the below code snippet:

Hide Copy Code

```
<httpModules>
<add name="clsMyModule" type="MyPipeLine.clsMyModule, MyPipeLine"/>
</httpModules>
```

The Final Output

If you run the code, you should see something like this in the *RequestLog.txt*. The above example is not so practical. But it will help us understand the fundamentals.

Hide Copy Code

```
Begin request called at 11/12/2008 6:32:00 PM
End Request called at 11/12/2008 6:32:00 PM
Begin request called at 11/12/2008 6:32:03 PM
End Request called at 11/12/2008 6:32:03 PM
Begin request called at 11/12/2008 6:32:06 PM
End Request called at 11/12/2008 6:32:06 PM
Begin request called at 11/12/2008 8:36:04 PM
End Request called at 11/12/2008 8:36:04 PM
Begin request called at 11/12/2008 8:37:06 PM
End Request called at 11/12/2008 8:37:06 PM
Begin request called at 11/12/2008 8:37:09 PM
End Request called at 11/12/2008 8:37:09 PM
Begin request called at 11/12/2008 8:37:38 PM
Page requested at 11/12/2008 8:37:38 PM/WebSiteHandlerDemo/Articles.shiv
End Request called at 11/12/2008 8:37:38 PM
```

Reference

- INFO: ASP.NET HTTP Modules and HTTP Handlers Overview

License

Share

About the Author

Shivprasad koirala

Architect https://www.questpond.com
India 🇮🇳

Do not forget to watch my Learn step by step video series.

Learn MVC 5 step by step in 16 hours
Learn MVC Core step by step

Learn Angular tutorials step by step for beginners
Learn Azure Step by Step
Learn Data Science Step by Step
Learn Python Step by Step
Step by Step Mathematics for Data Science
Learn MSBI in 32 hours
Learn Power BI Mobile Step by Step
Lea...

show more

Comments and Discussions

You must Sign In to use this message board.

Search Comments

Spacing Relaxed ▼ Layout Normal ▼ Per page 25 ▼ Update

First Prev Next

Good but need improvements	**Dheeraj Kumar Kesri**	**23-May-16 19:32**
I can't get your demo working	**ratamoa**	**10-Apr-16 19:32**
Two Thumbs Up	**Member 7916451**	**17-Jan-16 4:45**
Good article	**Prasanna Devarajan**	**11-Nov-15 8:23**
Good article	**ajayendra2707**	**14-Apr-15 22:45**
Simple yet clear	**RaviKumarMvs**	**16-Feb-15 0:35**
Httphandler for Webdav	**kkpraveen88**	**24-Dec-14 10:48**
My vote of 3	**H.J**	**25-Feb-14 2:04**

Awesome article	**Suresh Dammannapeta**	**30-Jan-14 23:50**
My vote of 5	**Azziet**	**11-Mar-13 3:52**
HttpHandlers are not Interceptors	**rajesh_chan**	**28-Jan-13 18:38**
My vote of 5	**shinnapongk**	**9-Dec-12 15:52**
Which one is best?	**Vitaly Tomilov**	**5-Oct-12 2:40**
Getting Error	**Bitla Phanindra**	**28-Sep-12 3:40**
Re: Getting Error	Amol_27101982, India	12-Oct-12 1:37
Re: Getting Error	Bitla Phanindra	12-Oct-12 3:23
Re: Getting Error	Amol_27101982, India	15-Oct-12 5:46
Re: Getting Error	Amol_27101982, India	15-Oct-12 5:48
Re: Getting Error	Bitla Phanindra	15-Oct-12 22:29
My vote of 5	**saanj**	**12-Sep-12 0:15**
My vote of 5	**sumanvidiyala**	**6-Sep-12 17:03**
My vote of 5	**Gerard Castelló Viader**	**24-Aug-12 0:03**
excellent	**thinkindia**	**4-Aug-12 6:43**
Hat's off to you sir..!	**Shubham Bhave**	**3-Aug-12 4:28**
Very Nice Explanation	**Member 8238408**	**18-Jun-12 21:57**

Last Visit: 4-Jan-20 13:35 Last Update: 4-Jan-20 13:36 Refresh 1 2 3 Next »

General News Suggestion Question Bug Answer Joke Praise Rant Admin

Use Ctrl+Left/Right to switch messages, Ctrl+Up/Down to switch threads, Ctrl+Shift+Left/Right to switch pages.

Permalink
Advertise
Privacy
Cookies
Terms of Use

Layout: fixed | fluid

Web01 2.8.191213.1