HttpContext Class

.NET Framework (current version)

> **Note**
>
> The .NET API Reference documentation has a new home. Visit the .NET API Browser on docs.microsoft.com to see the new experience.

Encapsulates all HTTP-specific information about an individual HTTP request.

Namespace: System.Web
Assembly: System.Web (in System.Web.dll)

Inheritance Hierarchy

System.Object
  System.Web.HttpContext

Syntax

C#

```
public sealed class HttpContext : IServiceProvider
```

Constructors

	Name	Description
◆	HttpContext(HttpRequest, HttpResponse)	Initializes a new instance of the HttpContext class by using the specified request and response objects.
◆	HttpContext(HttpWorkerRequest)	Initializes a new instance of the HttpContext class that uses the specified worker-request object.

Properties

	Name	Description
	AllErrors	Gets an array of errors accumulated while processing an HTTP request.
	AllowAsyncDuringSyncStages	Gets or sets a value that indicates whether asynchronous operations are allowed during parts of ASP.NET request processing when they are not expected.
	Application	Gets the HttpApplicationState object for the current HTTP request.
	ApplicationInstance	Gets or sets the HttpApplication object for the current HTTP request.
	AsyncPreloadMode	Gets or sets an object that contains flags that pertain to asynchronous preload mode.
	Cache	Gets the Cache object for the current application domain.
S	Current	Gets or sets the HttpContext object for the current HTTP request.
	CurrentHandler	Gets the IHttpHandler object that represents the currently executing handler.
	CurrentNotification	Gets a RequestNotification value that indicates the current HttpApplication event that is processing.
	Error	Gets the first error (if any) accumulated during HTTP request processing.
	Handler	Gets or sets the IHttpHandler object responsible for processing the HTTP request.
	IsCustomErrorEnabled	Gets a value indicating whether custom errors are enabled for the current HTTP request.
	IsDebuggingEnabled	Gets a value indicating whether the current HTTP request is in debug mode.
	IsPostNotification	Gets a value that is the current processing point in the ASP.NET pipeline just after an HttpApplication event has finished processing.
	IsWebSocketRequest	Gets a value that indicates whether the request is an AspNetWebSocket request.
	IsWebSocketRequestUpgrading	Gets a value that indicates whether the connection is upgrading from an HTTP connection to an AspNetWebSocket connection.
	Items	Gets a key/value collection that can be used to organize and share data between an IHttpModule interface and an IHttpHandler interface during an HTTP request.
	PageInstrumentation	Gets a reference to the page-instrumentation service instance for this request.
	PreviousHandle	Gets the IHttpHandler object for the parent handler.

	r	
	Profile	Gets the ProfileBase object for the current user profile.
	Request	Gets the HttpRequest object for the current HTTP request.
	Response	Gets the HttpResponse object for the current HTTP response.
	Server	Gets the HttpServerUtility object that provides methods used in processing Web requests.
	Session	Gets the HttpSessionState object for the current HTTP request.
	SkipAuthorization	Gets or sets a value that specifies whether the UrlAuthorizationModule object should skip the authorization check for the current request.
	ThreadAbortOnTimeout	Gets or sets a value that specifies whether the ASP.NET runtime should call Thread.Abort on the thread that is servicing this request when the request times out.
	Timestamp	Gets the initial timestamp of the current HTTP request.
	Trace	Gets the TraceContext object for the current HTTP response.
	User	Gets or sets security information for the current HTTP request.
	WebSocketNegotiatedProtocol	Gets the negotiated protocol that was sent from the server to the client for an AspNetWebSocket connection.
	WebSocketRequestedProtocols	Gets the ordered list of protocols requested by the client.

Methods

	Name	Description
	AcceptWebSocketRequest(Func<AspNetWebSocketContext, Task>)	Accepts an AspNetWebSocket request using the specified user function.
	AcceptWebSocketRequest(Func<AspNetWebSocketContext, Task>, AspNetWebSocketOptions)	Accepts an AspNetWebSocket request using the specified user function and options object.
	AddError(Exception)	Adds an exception to the exception collection for the current HTTP request.
	AddOnRequestCompleted(Action<	Raises a virtual event that occurs when the HTTP part of the

	HttpContext>)	request is ending.
◆	ClearError()	Clears all errors for the current HTTP request.
◆	DisposeOnPipelineCompleted(IDisposable)	Enables an object's Dispose method to be called when the AspNetWebSocket connection part of this request is completed.
◆	Equals(Object)	Determines whether the specified object is equal to the current object.(Inherited from Object.)
◆ S	GetAppConfig(String)	**Obsolete.** Returns requested configuration information for the current application.
◆	GetConfig(String)	**Obsolete.** Returns requested configuration information for the current HTTP request.
◆ S	GetGlobalResourceObject(String, String)	Gets an application-level resource object based on the specified ClassKey and ResourceKey properties.
◆ S	GetGlobalResourceObject(String, String, CultureInfo)	Gets an application-level resource object based on the specified ClassKey and ResourceKey properties, and on the CultureInfo object.
◆	GetHashCode()	Serves as the default hash function. (Inherited from Object.)
◆ S	GetLocalResourceObject(String, String)	Gets a page-level resource object based on the specified VirtualPath and ResourceKey properties.
◆ S	GetLocalResourceObject(String, String, CultureInfo)	Gets a page-level resource object based on the specified VirtualPath and ResourceKey properties, and on the CultureInfo object.
◆	GetSection(String)	Gets a specified configuration section for the current application's default configuration.
◆	GetType()	Gets the Type of the current instance.(Inherited from Object.)
◆	RemapHandler(IHttpHandler)	Enables you to specify a handler for the request.
◆	RewritePath(String)	Rewrites the URL using the given path.
◆	RewritePath(String, Boolean)	Rewrites the URL using the given path and a Boolean value that specifies whether the virtual path for server resources is modified.
◆	RewritePath(String, String, String)	Rewrites the URL by using the given path, path information, and query string information.
◆	RewritePath(String, String, String, Boolean)	Rewrites the URL using the given virtual path, path information, query string information, and a Boolean value that specifies whether the client file path is set to the rewrite path.

◆	SetSessionStateBehavior(SessionStateBehavior)	Sets the type of session state behavior that is required in order to support an HTTP request.
◆	ToString()	Returns a string that represents the current object.(Inherited from Object.)

Explicit Interface Implementations

	Name	Description
◆	IServiceProvider.GetService(Type)	This API supports the product infrastructure and is not intended to be used directly from your code. Returns an object for the current service type.

Remarks

Classes that inherit the IHttpModule and IHttpHandler interfaces are provided a reference to an HttpContext object for the current HTTP request. The object provides access to the intrinsic Request, Response, and Server properties for the request.

Examples

A Visual Studio Web site project with source code is available to accompany this topic: Download.

The following example demonstrates how to access and display properties of the HttpContext object. The context of the current HTTP request is accessed by using the Context property of the Page object.

C#

```
<%@ Page Language="C#" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<script runat="server">

    protected void Page_Load(object sender, EventArgs e)
    {
        // The HttpContext associated with the page can be accessed by the Context property.
        System.Text.StringBuilder sb = new System.Text.StringBuilder();
        // Use the current HttpContext object to determine if custom errors are enabled.
        sb.Append("Is custom errors enabled: " +
            Context.IsCustomErrorEnabled.ToString() + "<br/>");

        // Use the current HttpContext object to determine if debugging is enabled.
        sb.Append("Is debugging enabled: " +
            Context.IsDebuggingEnabled.ToString() + "<br/>");
        // Use the current HttpContext object to access the current TraceContext object.
        sb.Append("Trace Enabled: " +
            Context.Trace.IsEnabled.ToString() + "<br/>");

        // Use the current HttpContext object to access the current HttpApplicationState
object.
        sb.Append("Number of items in Application state: " +
            Context.Application.Count.ToString() + "<br/>");

        // Use the current HttpContext object to access the current HttpSessionState object.
        // Session state may not be configured.
        try
        {
            sb.Append("Number of items in Session state: " +
                Context.Session.Count.ToString() + "<br/>");
        }
        catch
        {
            sb.Append("Session state not enabled. <br/>");
        }

        // Use the current HttpContext object to access the current Cache object.
        sb.Append("Number of items in the cache: " +
            Context.Cache.Count.ToString() + "<br/>");

        // Use the current HttpContext object to determine the timestamp for the current HTTP
Request.
        sb.Append("Timestamp for the HTTP request: " +
            Context.Timestamp.ToString() + "<br/>");

        // Assign StringBuilder object to output label.
        OutputLabel.Text = sb.ToString();
    }
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head runat="server">
    <title>HttpContext Example</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
       Using the current HttpContext to get information about the current page.
       <br />
       <asp:Label id="OutputLabel" runat="server"></asp:Label>
    </div>
    </form>
</body>
</html>
```

Version Information

.NET Framework
Available since 1.1

Thread Safety

Any public static (**Shared** in Visual Basic) members of this type are thread safe. Any instance members are not guaranteed to be thread safe.

See Also

IHttpModule
IHttpHandler
System.Web Namespace
A Matter of Context

Return to top