HttpResponse 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 HTTP-response information from an ASP.NET operation.

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

Inheritance Hierarchy

System.Object
  System.Web.HttpResponse

Syntax

C#

```
public sealed class HttpResponse
```

Constructors

	Name	Description
◆	HttpResponse (TextWriter)	This API supports the product infrastructure and is not intended to be used directly from your code. Initializes a new instance of the HttpResponse class.

Properties

	Name	Description

	Buffer	Gets or sets a value indicating whether to buffer output and send it after the complete response is finished processing.
	BufferOutput	Gets or sets a value indicating whether to buffer output and send it after the complete page is finished processing.
	Cache	Gets the caching policy (such as expiration time, privacy settings, and vary clauses) of a Web page.
	CacheControl	Gets or sets the **Cache-Control** HTTP header that matches one of the HttpCacheability enumeration values.
	Charset	Gets or sets the HTTP character set of the output stream.
	ClientDisconnectedToken	Gets a CancellationToken object that is tripped when the client disconnects.
	ContentEncoding	Gets or sets the HTTP character set of the output stream.
	ContentType	Gets or sets the HTTP MIME type of the output stream.
	Cookies	Gets the response cookie collection.
	Expires	Gets or sets the number of minutes before a page cached on a browser expires. If the user returns to the same page before it expires, the cached version is displayed. Expires is provided for compatibility with earlier versions of ASP.
	ExpiresAbsolute	Gets or sets the absolute date and time at which to remove cached information from the cache. ExpiresAbsolute is provided for compatibility with earlier versions of ASP.
	Filter	Gets or sets a wrapping filter object that is used to modify the HTTP entity body before transmission.
	HeaderEncoding	Gets or sets an Encoding object that represents the encoding for the current header output stream.
	Headers	Gets the collection of response headers.
	HeadersWritten	[Supported in the .NET Framework 4.5.2 and later versions] Gets a value indicating whether the response headers have been written.
	IsClientConnected	Gets a value indicating whether the client is still connected to the server.
	IsRequestBeingRedirected	Gets a Boolean value indicating whether the client is being transferred to a new location.
	Output	Enables output of text to the outgoing HTTP response stream.

☞	OutputStream	Enables binary output to the outgoing HTTP content body.
☞	RedirectLocation	Gets or sets the value of the HTTP **Location** header.
☞	Status	Sets the **Status** line that is returned to the client.
☞	StatusCode	Gets or sets the HTTP status code of the output returned to the client.
☞	StatusDescription	Gets or sets the HTTP status string of the output returned to the client.
☞	SubStatusCode	Gets or sets a value qualifying the status code of the response.
☞	SupportsAsyncFlush	Gets a value that indicates whether the connection supports asynchronous flush operations.
☞	SuppressContent	Gets or sets a value indicating whether to send HTTP content to the client.
☞	SuppressDefaultCacheControlHeader	[Supported in the .NET Framework 4.5.2 and later versions] Gets or sets a value indicating whether to suppress the default `Cache Control: private` header for the current HTTP response.
☞	SuppressFormsAuthenticationRedirect	Gets or sets a value that specifies whether forms authentication redirection to the login page should be suppressed.
☞	TrySkipIisCustomErrors	Gets or sets a value that specifies whether IIS 7.0 custom errors are disabled.

Methods

	Name	Description
◆	AddCacheDependency(CacheDependency[])	Associates a set of cache dependencies with the response to facilitate invalidation of the response if it is stored in the output cache and the specified dependencies change.
◆	AddCacheItemDependencies(ArrayList)	Makes the validity of a cached response dependent on other items in the cache.
◆	AddCacheItemDependencies(String[])	Makes the validity of a cached item dependent on another item in the cache.

◆	AddCacheItemDependency(String)	Makes the validity of a cached response dependent on another item in the cache.
◆	AddFileDependencies(ArrayList)	Adds a group of file names to the collection of file names on which the current response is dependent.
◆	AddFileDependencies(String[])	Adds an array of file names to the collection of file names on which the current response is dependent.
◆	AddFileDependency(String)	Adds a single file name to the collection of file names on which the current response is dependent.
◆	AddHeader(String, String)	Adds an HTTP header to the output stream. AddHeader is provided for compatibility with earlier versions of ASP.
◆	AddOnSendingHeaders(Action<HttpContext>)	[Supported in the .NET Framework 4.5.2 and later versions] Registers a callback that the ASP.NET runtime will invoke immediately before response headers are sent for this request.
◆	AppendCookie(HttpCookie)	This API supports the product infrastructure and is not intended to be used directly from your code. Adds an HTTP cookie to the intrinsic cookie collection.
◆	AppendHeader(String, String)	Adds an HTTP header to the output stream.
◆	AppendToLog(String)	Adds custom log information to the Internet Information Services (IIS) log file.
◆	ApplyAppPathModifier(String)	Adds a session ID to the virtual path if the session is using Cookieless session state and returns the combined path. If Cookieless session state is not used, ApplyAppPathModifier returns the original virtual path.
◆	BeginFlush(AsyncCallback, Object)	Sends the currently buffered response to the client.
◆	BinaryWrite(Byte[])	Writes a string of binary characters to the HTTP output stream.
◆	Clear()	Clears all content output from the buffer stream.
◆	ClearContent()	Clears all content output from the buffer stream.
◆	ClearHeaders()	Clears all headers from the buffer stream.
◆	Close()	Closes the socket connection to a client.
◆	DisableKernelCache()	Disables kernel caching for the current response.
◆	DisableUserCache()	Disables IIS user-mode caching for this response.
◆	End()	Sends all currently buffered output to the client, stops execution of the page, and raises the EndRequest event.

◆	EndFlush(IAsyncResult)	Completes an asynchronous flush operation.
◆	Equals(Object)	Determines whether the specified object is equal to the current object.(Inherited from Object.)
◆	Flush()	Sends all currently buffered output to the client.
◆	FlushAsync()	Asynchronously sends all currently buffered output to the client.
◆	GetHashCode()	Serves as the default hash function. (Inherited from Object.)
◆	GetType()	Gets the Type of the current instance.(Inherited from Object.)
◆	Pics(String)	Appends a HTTP **PICS-Label** header to the output stream.
◆	PushPromise(String)	This API is to support applications sending push promises to HTTP 2.0 clients. You can find more details regarding to Http2 server push at HTTP/2 Specification Section 8.2: Server Push.
◆	PushPromise(String, String, NameValueCollection)	This API is to support applications sending push promises to HTTP 2.0 clients. You can find more details regarding to Http2 server push at HTTP/2 Specification Section 8.2: Server Push.
◆	Redirect(String)	Redirects a request to a new URL and specifies the new URL.
◆	Redirect(String, Boolean)	Redirects a client to a new URL. Specifies the new URL and whether execution of the current page should terminate.
◆	RedirectPermanent(String)	Performs a permanent redirection from the requested URL to the specified URL.
◆	RedirectPermanent(String, Boolean)	Performs a permanent redirection from the requested URL to the specified URL, and provides the option to complete the response.
◆	RedirectToRoute(Object)	Redirects a request to a new URL by using route parameter values.
◆	RedirectToRoute(RouteValueDictionary)	Redirects a request to a new URL by using route parameter values.
◆	RedirectToRoute(String)	Redirects a request to a new URL by using a route name.
◆	RedirectToRoute(String, Object)	Redirects a request to a new URL by using route parameter values and a route name.
◆	RedirectToRoute(String,	Redirects a request to a new URL by using route parameter values and a route name.

	RouteValueDictionary)	
◆	RedirectToRoutePermanent(Object)	Performs a permanent redirection from a requested URL to a new URL by using route parameter values.
◆	RedirectToRoutePermanent(RouteValueDictionary)	Performs a permanent redirection from a requested URL to a new URL by using route parameter values.
◆	RedirectToRoutePermanent(String)	Performs a permanent redirection from a requested URL to a new URL by using a route name.
◆	RedirectToRoutePermanent(String, Object)	Performs a permanent redirection from a requested URL to a new URL by using the route parameter values and the name of the route that correspond to the new URL.
◆	RedirectToRoutePermanent(String, RouteValueDictionary)	Performs a permanent redirection from a requested URL to a new URL by using route parameter values and a route name.
◆ S	RemoveOutputCacheItem(String)	Removes from the cache all cached items that are associated with the default output-cache provider. This method is static.
◆ S	RemoveOutputCacheItem(String, String)	Uses the specified output-cache provider to remove all output-cache items that are associated with the specified path.
◆	SetCookie(HttpCookie)	This API supports the product infrastructure and is not intended to be used directly from your code. Updates an existing cookie in the cookie collection.
◆	ToString()	Returns a string that represents the current object.(Inherited from Object.)
◆	TransmitFile(String)	Writes the specified file directly to an HTTP response output stream, without buffering it in memory.
◆	TransmitFile(String, Int64, Int64)	Writes the specified part of a file directly to an HTTP response output stream without buffering it in memory.
◆	Write(Char)	Writes a character to an HTTP response output stream.
◆	Write(Char[], Int32, Int32)	Writes an array of characters to an HTTP response output stream.
◆	Write(Object)	Writes an Object to an HTTP response stream.
◆	Write(String)	Writes a string to an HTTP response output stream.
◆	WriteFile(IntPtr, Int64, Int64)	Writes the specified file directly to an HTTP response output stream.
◆	WriteFile(String)	Writes the contents of the specified file directly to an HTTP response output

		stream as a file block.
◆	WriteFile(String, Boolean)	Writes the contents of the specified file directly to an HTTP response output stream as a memory block.
◆	WriteFile(String, Int64, Int64)	Writes the specified file directly to an HTTP response output stream.
◆	WriteSubstitution(HttpResponseSubstitutionCallback)	Allows insertion of response substitution blocks into the response, which allows dynamic generation of specified response regions for output cached responses.

Remarks

The methods and properties of the HttpResponse class are exposed through the Response property of the HttpApplication, HttpContext, Page, and UserControl classes.

The following methods of the HttpResponse class are supported only in post back scenarios and not in asynchronous post back scenarios:

- BinaryWrite
- Clear
- ClearContent
- ClearHeaders
- Close
- End
- Flush
- TransmitFile
- Write
- WriteFile
- WriteSubstitution

Partial-page updates are enabled when you use UpdatePanel controls to update selected regions of a page instead of updating the whole page with a post back. For more information, see UpdatePanel Control Overview and Partial-Page Rendering Overview.

Examples

The following example draws three overlapping rectangles when the page is requested. The code begins by setting the ContentType property to image/jpeg, so that the entire page will be rendered as a JPEG image. The code then calls the Clear method to ensure that no extraneous content is sent with this response. Next, the code sets the BufferOutput property to true so that the page is completely processed before it is sent to the requesting client. Two objects used to draw the rectangles are then created: a Bitmap and a Graphics object. The variables created in the page are used as coordinates to draw the rectangles and a string that appears inside the largest rectangle.

When the three rectangles and the string that appears inside them are drawn, the Bitmap is saved to the Stream object that is associated with the OutputStream property and its format is set to JPEG. The code calls the Dispose and Dispose methods to release the resources used by the two drawing objects. Lastly, the code calls the Flush method to send the buffered response to the requesting client.

Note

In code, the HttpResponse object is referred to by the keyword **Response**. For example, `Response.Clear()` refers to the HttpResponse.Clear method. The Page class has a property that is named Response that exposes the current instance of HttpResponse.

C#

```
<%@ Page Language="C#" %>
<%@ import Namespace="System.Drawing" %>
<%@ import Namespace="System.Drawing.Imaging" %>
<%@ import Namespace="System.Drawing.Drawing2D" %>

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

    private void Page_Load(object sender, EventArgs e)
    {
        // Set the page's content type to JPEG files
        // and clears all content output from the buffer stream.
        Response.ContentType = "image/jpeg";
        Response.Clear();

        // Buffer response so that page is sent
        // after processing is complete.
        Response.BufferOutput = true;

        // Create a font style.
        Font rectangleFont = new Font(
            "Arial", 10, FontStyle.Bold);

        // Create integer variables.
        int height = 100;
        int width = 200;

        // Create a random number generator and create
        // variable values based on it.
        Random r = new Random();
        int x = r.Next(75);
        int a = r.Next(155);
        int x1 = r.Next(100);
        // Create a bitmap and use it to create a
        // Graphics object.
        Bitmap bmp = new Bitmap(
            width, height, PixelFormat.Format24bppRgb);
        Graphics g = Graphics.FromImage(bmp);

        g.SmoothingMode = SmoothingMode.AntiAlias;
        g.Clear(Color.LightGray);

        // Use the Graphics object to draw three rectangles.
        g.DrawRectangle(Pens.White, 1, 1, width-3, height-3);
        g.DrawRectangle(Pens.Aquamarine, 2, 2, width-3, height-3);
        g.DrawRectangle(Pens.Black, 0, 0, width, height);

        // Use the Graphics object to write a string
        // on the rectangles.
        g.DrawString(
            "ASP.NET Samples", rectangleFont,
            SystemBrushes.WindowText, new PointF(10, 40));

        // Apply color to two of the rectangles.
        g.FillRectangle(
            new SolidBrush(
                Color.FromArgb(a, 255, 128, 255)),
            x, 20, 100, 50);

        g.FillRectangle(
            new LinearGradientBrush(
                new Point(x, 10),
                new Point(x1 + 75, 50 + 30),
                Color.FromArgb(128, 0, 0, 128),
                Color.FromArgb(255, 255, 255, 240)),
            x1, 50, 75, 30);

        // Save the bitmap to the response stream and
        // convert it to JPEG format.
        bmp.Save(Response.OutputStream, ImageFormat.Jpeg);

        // Release memory used by the Graphics object
        // and the bitmap.
        g.Dispose();
        bmp.Dispose();

        // Send the output to the client.
        Response.Flush();
    }

</script>
<html xmlns="http://www.w3.org/1999/xhtml" >
<head>
    <title>ASP.NET Example</title>
</head>
<body>
    <form id="form1" runat="server">
    </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

System.Web Namespace

Return to top