

Action Results in Web API 2

02/03/2014 • 3 minutes to read • Contributors      [all](#)

In this article

[void](#)

[HttpResponseMessage](#)

[IHttpActionResult](#)

[Other Return Types](#)

by [Mike Wasson](#)

This topic describes how ASP.NET Web API converts the return value from a controller action into an HTTP response message.

A Web API controller action can return any of the following:

1. `void`
2. `HttpResponseMessage`
3. `IHttpActionResult`
4. Some other type

Depending on which of these is returned, Web API uses a different mechanism to create the HTTP response.

Return type	How Web API creates the response
<code>void</code>	Return empty 204 (No Content)
<code>HttpResponseMessage</code>	Convert directly to an HTTP response message.
<code>IHttpActionResult</code>	Call <code>ExecuteAsync</code> to create an <code>HttpResponseMessage</code> , then convert to an HTTP response message.
Other type	Write the serialized return value into the response body; return 200 (OK).

The rest of this topic describes each option in more detail.


void

If the return type is `void`, Web API simply returns an empty HTTP response with status code 204 (No Content).

Example controller:

C#	 Copy
<pre>public class ValuesController : ApiController { public void Post() { } }</pre>	


HTTP response:

console	 Copy
<pre>HTTP/1.1 204 No Content Server: Microsoft-IIS/8.0 Date: Mon, 27 Jan 2014 02:13:26 GMT</pre>	

HttpResponseMessage

If the action returns an [HttpResponseMessage](#), Web API converts the return value directly into an HTTP response message, using the properties of the **HttpResponseMessage** object to populate the response.

This option gives you a lot of control over the response message. For example, the following controller action sets the Cache-Control header.

C#	 Copy
<pre>public class ValuesController : ApiController { public HttpResponseMessage Get() { HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK, "value"); response.Content = new StringContent("hello", Encoding.Unicode); response.Headers.CacheControl = new CacheControlHeaderValue() {</pre>	

```
        MaxAge = TimeSpan.FromMinutes(20)
    };
    return response;
}
}
```

Response:

console

 Copy

```
HTTP/1.1 200 OK
Cache-Control: max-age=1200
Content-Length: 10
Content-Type: text/plain; charset=utf-16
Server: Microsoft-IIS/8.0
Date: Mon, 27 Jan 2014 08:53:35 GMT

hello
```

If you pass a domain model to the **CreateResponse** method, Web API uses a [media formatter](#) to write the serialized model into the response body.

C#

 Copy

```
public HttpResponseMessage Get()
{
    // Get a list of products from a database.
    IEnumerable<Product> products = GetProductsFromDB();

    // Write the list to the response body.
    HttpResponseMessage response = Request.CreateResponse(HttpStatusCode.OK,
products);
    return response;
}
```


Web API uses the Accept header in the request to choose the formatter. For more information, see [Content Negotiation](#).

IHttpActionResult

The **IHttpActionResult** interface was introduced in Web API 2. Essentially, it defines an **HttpResponseMessage** factory. Here are some advantages of using the **IHttpActionResult** interface:

- Simplifies [unit testing](#) your controllers.
- Moves common logic for creating HTTP responses into separate classes.
- Makes the intent of the controller action clearer, by hiding the low-level details of constructing the response.

IHttpActionResult contains a single method, **ExecuteAsync**, which asynchronously creates an **HttpResponseMessage** instance.

C#	 Copy
<pre>public interface IHttpActionResult { Task<HttpResponseMessage> ExecuteAsync(CancellationToken cancellationTokentoken); }</pre>	

If a controller action returns an **IHttpActionResult**, Web API calls the **ExecuteAsync** method to create an **HttpResponseMessage**. Then it converts the **HttpResponseMessage** into an HTTP response message.

Here is a simple implementaton of **IHttpActionResult** that creates a plain text response:

C#	 Copy
<pre>public class TextResult : IHttpActionResult { string _value; HttpRequestMessage _request; public TextResult(string value, HttpRequestMessage request) { _value = value; _request = request; } public Task<HttpResponseMessage> ExecuteAsync(CancellationToken cancellationTokentoken) { var response = new HttpResponseMessage() { Content = new StringContent(_value), RequestMessage = _request }; return Task.FromResult(response); } }</pre>	

Example controller action:

C#

 Copy

```
public class ValuesController : ApiController
{
    public IHttpActionResult Get()
    {
        return new TextResult("hello", Request);
    }
}
```

Response:

console

 Copy

```
HTTP/1.1 200 OK
Content-Length: 5
Content-Type: text/plain; charset=utf-8
Server: Microsoft-IIS/8.0
Date: Mon, 27 Jan 2014 08:53:35 GMT

hello
```

More often, you will use the **IHttpActionResult** implementations defined in the [System.Web.Http.Results](#) namespace. The **ApiController** class defines helper methods that return these built-in action results.

In the following example, if the request does not match an existing product ID, the controller calls [ApiController.NotFound](#) to create a 404 (Not Found) response. Otherwise, the controller calls [ApiController.OK](#), which creates a 200 (OK) response that contains the product.

C#

 Copy

```
public IHttpActionResult Get (int id)
{
    Product product = _repository.Get (id);
    if (product == null)
    {
        return NotFound(); // Returns a NotFoundResult
    }
    return Ok(product); // Returns an OkNegotiatedContentResult
}
```

Other Return Types

For all other return types, Web API uses a [media formatter](#) to serialize the return value. Web API writes the serialized value into the response body. The response status code is 200 (OK).

C#	 Copy
<pre>public class ProductsController : ApiController { public IEnumerable<Product> Get() { return GetAllProductsFromDB(); } }</pre>	

A disadvantage of this approach is that you cannot directly return an error code, such as 404. However, you can throw an **HttpResponseException** for error codes. For more information, see [Exception Handling in ASP.NET Web API](#).

Web API uses the Accept header in the request to choose the formatter. For more information, see [Content Negotiation](#).

Example request

console	 Copy
<pre>GET http://localhost/api/products HTTP/1.1 User-Agent: Fiddler Host: localhost:24127 Accept: application/json</pre>	

Example response:

console	 Copy
<pre>HTTP/1.1 200 OK Content-Type: application/json; charset=utf-8 Server: Microsoft-IIS/8.0 Date: Mon, 27 Jan 2014 08:53:35 GMT Content-Length: 56 [{"Id":1,"Name":"Yo-yo","Category":"Toys","Price":6.95}]</pre>	

