

Forum: [Hack Forums](#)

Thread: [HttpClient / HttpRequestMessage adding white spaces to headers](#)

vaiperboy - May 12, 2019, 9:42 am

<https://stackoverflow.com/questions/55833095/httpclient-httprequestmessage-adding-white-spaces-to-headers>

I'm the creator of that request and seeking support here
Please suggest me stuff to do as I'm out of ideas, also newbies don't suggest stuff.

Spoiler ([Click to View](#))

I just got into .net core due to the features it has to offer for HttpClient (example is setting the keep-alive header capped down). I'm using 2.1 net core and C# 7.3

My issue

My issue is specifically about setting the "Accept" and "Accept-Language" headers. The site I'm trying to reach parses these two headers as is, meaning it must have the same exact format (including spaces).

Using fiddler (and some other tools) to grab my request.

I'm capturing the "Accept" header as the following: (incorrect format)
text/html, application/xhtml+xml, application/xml; q=0.9, image/webp, image/apng, */*; q=0.8, application/signed-exchange; v=b3
instead of: (correct format)

text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3

"Accept-Language" header as the following: (incorrect format)

en-US, en; q=0.9, ar; q=0.8, und; q=0.7

Instead of: (correct format)

en-US,en;q=0.9,ar;q=0.8,und;q=0.7

I'm simulating a GET request using HttpClient with the help of HttpRequestMessage.

What I have tried

Tried both settings my headers in the HttpClient instance and firing off a HttpRequestMessage with it's own headers, none are working.

Tried lots of solutions online which didn't work, including this

Splitting the header through multiple .Headers.TryAddWithoutValidation

Tried HttpWebRequests as well. No luck. One thing to note, due to my project I also like to stick with the HttpClient.

You'll find below my current code

```
Code var client = new HttpClient(new HttpClientHandler() { UseCookies = false,
    AllowAutoRedirect = false,
    Proxy = new WebProxy("localhost:8888") });
```

```
var httpReq = new HttpRequestMessage(HttpMethod.Get, $"myurl");
```

```
    httpReq.Headers.TryAddWithoutValidation("Host", "myurl");
    httpReq.Headers.TryAddWithoutValidation("Connection", "keep-alive");
    httpReq.Headers.TryAddWithoutValidation("Cookie", $"cookie");
    httpReq.Headers.TryAddWithoutValidation("Upgrade-Insecure-Requests", "1");
    httpReq.Headers.TryAddWithoutValidation("User-Agent", Data.user_agent);
    httpReq.Headers.TryAddWithoutValidation("Accept-Encoding", "gzip, deflate, br");
    httpReq.Headers.TryAddWithoutValidation("Accept",
"text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/
signed-exchange;v=b3");
    httpReq.Headers.TryAddWithoutValidation("Referer", Data.homeUrl);
    httpReq.Headers.TryAddWithoutValidation("Accept-Language",
"en-US,en;q=0.9,ar;q=0.8,und;q=0.7");
```

```
req = await client.SendAsync(httpReq);
```

As noted before, all the other headers are working fine. Just these two Accept and Accept-Language are being inputted differently.

JUSTIN - May 12, 2019, 10:09 am

Take a look at section 14 of RFC2616, aka IETF's official documentation for HTTP.

<https://www.w3.org/Protocols/rfc2616/rfc2616-sec14.html#sec14>

The format that is actually being sent isn't wrong. Spacing shouldn't matter. In reality this is a problem with the site owner.

Regardless, I just tested Accept/Accept-Language headers via the WebClient class and they weren't modified before being sent.

If it's really a big problem you can try implementing that instead.

Since somebody on the Stackoverflow thread pointed out that this is hardcoded in CoreFX (<https://github.com/dotnet/corefx/blob/a10890f4ffe0fadf090c922578ba0e606ebdd16c/src/System.Net.Http/src/System.Net/Http/Headers/NameValuePair.cs#L183>) you could also override the ToString method in the NameValuePair class.

Meterpreter - May 18, 2019, 1:34 am

I would personally recommend using WebClient.

For HttpClient and similar often make it difficult to set Host, Accept, and Content headers. WebClient is much easier. Also Justin is correct, spacing really shouldn't matter, the semi colons specific the break anyway not the space.

Webclient is literally 100 times easier:

```
Webclient wc = new WebClient();
```

```
wc.Headers[HttpRequestHeaders.Host] = "test";  
wc.Headers[HttpRequestHeaders.Accept] = "";  
And if you want to use a custom header its undoubtedly the easiest to use to do so:  
wc.Headers["Custom-Header"] = "custom value";
```