**Разобрался с авторизацией:**var registerModel = new  
{  
UserName = userName,  
Password = password,  
};  
string json = JsonConvert.SerializeObject(registerModel);  
var inputMessage = new HttpRequestMessage  
{  
Content = new StringContent(json, Encoding.UTF8, "application/json-patch+json")  
};  
inputMessage.Headers.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

using (var client = new HttpClient())  
{  
var response = client.PostAsync(APP\_PATH + "/api/Account/Login", inputMessage.Content).Result;  
return response.ToString();  
}

**Может кому пригодится, вот рабочий вариант авторизации и получение**

HttpResponseMessage после авторизации:

var registerModel = new  
{  
UserName = userName,  
Password = password,  
};  
string json = JsonConvert.SerializeObject(registerModel);  
var inputMessage = new HttpRequestMessage  
{  
Content = new StringContent(json, Encoding.UTF8, "application/json-patch+json")  
};  
inputMessage.Headers.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));

var client = new HttpClient();  
var response = client.PostAsync(APP\_PATH + "/api/Account/Login", inputMessage.Content).Result;

HttpRequestMessage request = new HttpRequestMessage(HttpMethod.Get, APP\_PATH + "/api/Account/GetProfile");  
request.Headers.Accept.Add(new MediaTypeWithQualityHeaderValue("application/json"));  
HttpResponseMessage responseNew = client.SendAsync(request).Result;  
return responseNew.Content.ReadAsStringAsync().Result;

C# HttpClient SendAsync(System.Net.Http.HttpRequestMessage request, System.Threading.CancellationToken cancellationToken)

[Previous](https://www.demo2s.com/csharp/csharp-httpclient-sendasync-system-net-http-httprequestmessage-request-sy.html)[Next](https://www.demo2s.com/csharp/csharp-httpclient-sendasync-system-net-http-httprequestmessage-reques-enxm.html)

C# HttpClient **SendAsync**(System.Net.Http.HttpRequestMessage request, System.Threading.CancellationToken cancellationToken) Send an HTTP request as an asynchronous operation.

From Type:

Copy

System.Net.Http.HttpClient

**SendAsync**() is a method.

**Syntax**

SendAsync is defined as:

Copy

**public** **override** System.Threading.Tasks.Task<System.Net.Http.HttpResponseMessage> SendAsync (System.Net.Http.HttpRequestMessage request, System.Threading.CancellationToken cancellationToken);

**Parameters:**

C# HttpClient **SendAsync**() has the following parameters:

* *request* - The HTTP request message to send.
* *cancellationToken* - The cancellation token to cancel operation.

**Return**

The task object representing the asynchronous operation.

Example

The following examples show how to use C# HttpClient.**SendAsync**(System.Net.Http.HttpRequestMessage request, System.Threading.CancellationToken cancellationToken).

Example 1

Copy

**using** System;/\* w w w . d e m o 2 s. c o m \*/

**using** System.Net.Http;

**using** System.Net.Http.Headers;

**using** System.Threading;

**using** System.Threading.Tasks;

**using** Microsoft.AspNetCore.Authentication;

**using** Microsoft.AspNetCore.Authorization;

**using** Microsoft.AspNetCore.Mvc;

**namespace** Mvc.Client.Controllers {

**public** **class** HomeController : Controller {

[HttpGet(**"~/"**)]

**public** ActionResult Index() {

**return** View(**"Home"**);

}

[Authorize, HttpPost(**"~/"**)]

**public** async Task<ActionResult> Index(CancellationToken cancellationToken) {

**using** (**var** client = **new** HttpClient()) {

**var** token = await HttpContext.Authentication.GetTokenAsync(**"access\_token"**);

**if** (**string**.IsNullOrEmpty(token)) {

**throw** **new** InvalidOperationException(**"The access token cannot be found in the authentication ticket. "** +

**"Make sure that SaveTokens is set to true in the OIDC options."**);

}

**var** request = **new** HttpRequestMessage(HttpMethod.Get, **"http://localhost:54540/api/message"**);

request.Headers.Authorization = **new** AuthenticationHeaderValue(**"Bearer"**, token);

**var** response = await client.SendAsync(request, cancellationToken);

response.EnsureSuccessStatusCode();

**return** View(**"Home"**, model: await response.Content.ReadAsStringAsync());

}

}

}

}

Example 2

Copy

#region Apache 2 License// w w w . d e m o 2 s . c o m

#endregion

**using** System.Net.Http;

**using** System.Threading;

**using** System.Threading.Tasks;

**namespace** System.Reactive.IronMQ

{

**public** **static** **class** Extensions

{

**public** **static** **string** toString(**this** Cloud cloud)

{

**switch** (cloud)

{

**case** Cloud.AWS: **return** **"mq-aws-us-east-1"**;

**case** Cloud.RackSpace: **return** **"mq-rackspace-dfw"**;

**default**: **return** **null**;

}

}

**public** **static** Task<HttpResponseMessage> DeleteAsync(**this** HttpClient client, **string** requestUri, HttpContent content, CancellationToken token)

{

**var** message = **new** HttpRequestMessage(HttpMethod.Delete, requestUri);

message.Content = content;

**return** client.SendAsync(message, token);

}

**public** **static** Task<HttpResponseMessage> DeleteAsync(**this** HttpClient client, **string** requestUri, HttpContent content)

{

**return** client.DeleteAsync(requestUri, content, CancellationToken.None);

}

}

}

Example 3

Copy

**using** System;// w w w . d e mo 2 s. c o m

**using** System.Collections.Generic;

**using** System.Net.Http;

**using** System.Net.Http.Headers;

**using** System.Text;

**using** System.Threading;

**using** System.Threading.Tasks;

**using** Newtonsoft.Json;

**namespace** Tailspin.Surveys.Common

{

**public** **static** **class** HttpClientExtensions

{

**public** **static** async Task<HttpResponseMessage> SendRequestWithBearerTokenAsync(**this** HttpClient httpClient, HttpMethod method, **string** path, **object** requestBody, **string** accessToken, CancellationToken ct)

{

**var** request = **new** HttpRequestMessage(method, path);

**if** (requestBody != **null**)

{

**var** json = JsonConvert.SerializeObject(requestBody, Formatting.None);

**var** content = **new** StringContent(json, Encoding.UTF8, **"application/json"**);

request.Content = content;

}

request.Headers.Authorization = **new** AuthenticationHeaderValue(**"Bearer"**, accessToken);

request.Headers.Accept.Add(**new** MediaTypeWithQualityHeaderValue(**"application/json"**));

**var** response = await httpClient.SendAsync(request, ct);

**return** response;

}

}

}