Piotr Gankiewicz

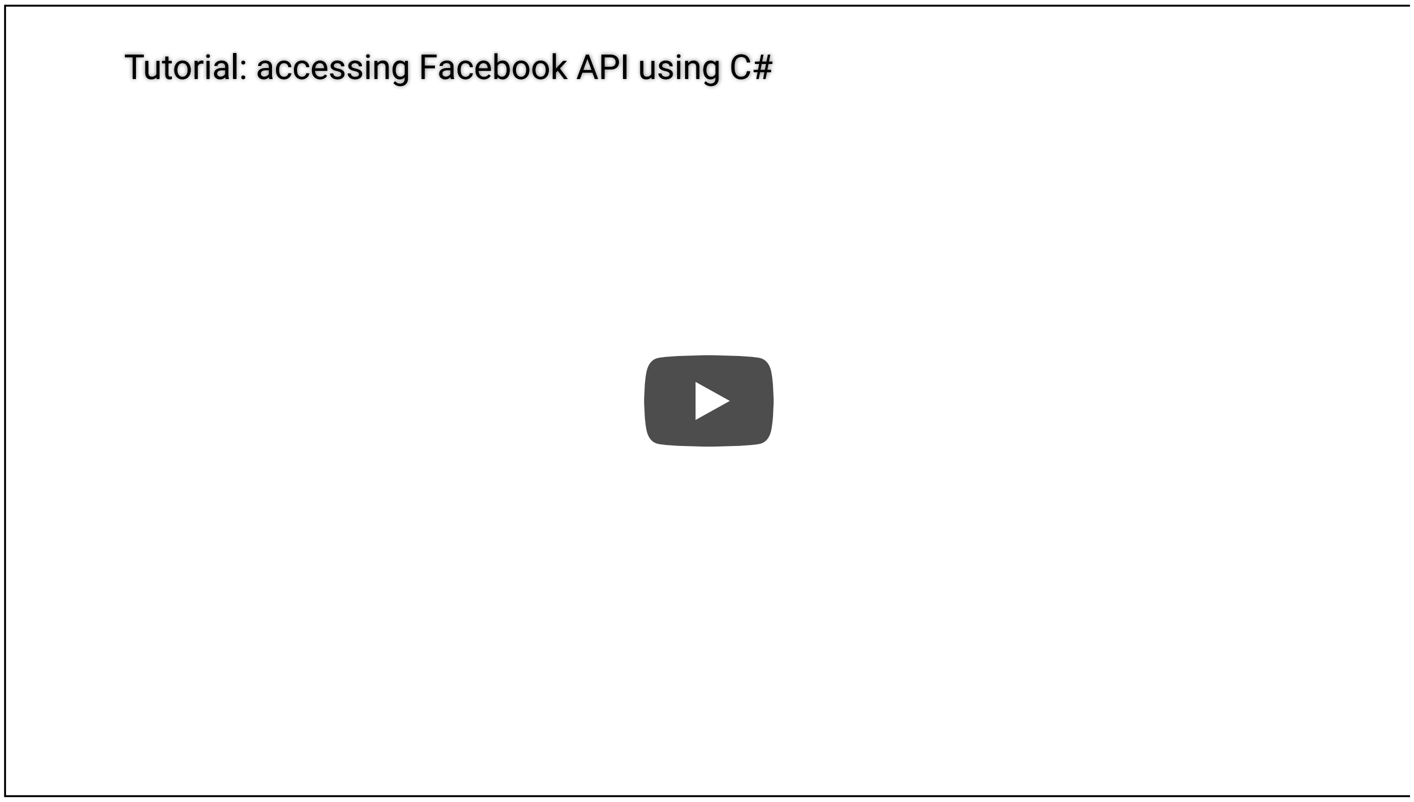
.NET CORE C# TUTORIALS

Accessing Facebook API using C#

6 February 2017 23 Comments Piotr Gankiewicz 4 min read

SHARE

Another quick video tutorial from me. Here, we will focus on implementing our own "SDK" responsible for handling the Facebook Graph API using C# and .NET Core (of course you can achieve the same result on the full .NET platform).



If you're just like me targeting the .NET Core, most likely you realized that **Facebook SDK for .NET** is not really being supported, at least currently, based on the latest date of the *master* branch update. And of course the same does apply to the NuGet **packages** .

Thus, we're kinda on our own, however, it turns out that accessing the **Graph API** is quite easy. All we need to do is to make use of the *HttpClient* or any other type that is responsible for handling the HTTP requests. Let's jump right into the code:

```
public class Account
{
    public string Id { get; set; }
    public string Name { get; set; }
    public string Email { get; set; }
    public string Locale { get; set; }
    public string UserName { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string Gender { get; set; }
}
```

The *Account* basically represents the user account in the Facebook. Please note that you may include there as many more properties as you wish.

```
public interface IFacebookClient
{
    Task<T> GetAsync<T>(string accessToken, string endpoint, string args = null);
    Task PostAsync(string accessToken, string endpoint, object data, string args = null);
}

public class FacebookClient : IFacebookClient
{
    private readonly HttpClient _httpClient;

    public FacebookClient()
    {
        _httpClient = new HttpClient
        {
            BaseAddress = new Uri("https://graph.facebook.com/v2.8/")
        };
        _httpClient.DefaultRequestHeaders
            .Accept
            .Add(new MediaTypeWithQualityHeaderValue("application/json"));
    }

    public async Task<T> GetAsync<T>(string accessToken, string endpoint, string args = null)
    {
        var response = await _httpClient.GetAsync($"{endpoint}?access_token={accessToken}&{ar
        if (!response.IsSuccessStatusCode)
            return default(T);

        var result = await response.Content.ReadAsStringAsync();

        return JsonConvert.DeserializeObject<T>(result);
    }

    public async Task PostAsync(string accessToken, string endpoint, object data, string args
    {
        var payload = GetPayload(data);
        await _httpClient.PostAsync($"{endpoint}?access_token={accessToken}&{args}", payload)
    }

    private static StringContent GetPayload(object data)
    {
        var json = JsonConvert.SerializeObject(data);

        return new StringContent(json, Encoding.UTF8, "application/json");
    }
}
```

The *FacebookClient* is responsible for handling the POST and GET requests to the Facebook API. Quite straightforward code, no magic here, just keep in mind to add a reference to the `Newtonsoft.Json` package.

```
public interface IFacebookService
{
    Task<Account> GetAccountAsync(string accessToken);
    Task PostOnWallAsync(string accessToken, string message);
}

public class FacebookService : IFacebookService
{
    private readonly IFacebookClient _facebookClient;

    public FacebookService(IFacebookClient facebookClient)
    {
        _facebookClient = facebookClient;
    }

    public async Task<Account> GetAccountAsync(string accessToken)
    {
        var result = await _facebookClient.GetAsync<dynamic>(
            accessToken, "me", "fields=id,name,email,first_name,last_name,age_range,birthday,

        if (result == null)
        {
            return new Account();
        }

        var account = new Account
        {
            Id = result.id,
            Email = result.email,
            Name = result.name,
            UserName = result.username,
            FirstName = result.first_name,
            LastName = result.last_name,
            Locale = result.locale
        };

        return account;
    }

    public async Task PostOnWallAsync(string accessToken, string message)
        => await _facebookClient.PostAsync(accessToken, "me/feed", new {message});
}
```

Our *FacebookService* is where the true “business logic” lays in. It does make use of the underlying *FacebookClient* in order to execute the HTTP requests to the specified endpoints using some additional arguments.

```
public class Program
{
    public static void Main(string[] args)
    {
        var accessToken = "YOUR_GRAPH_API_ACCESS_TOKEN";
        var facebookClient = new FacebookClient();
        var facebookService = new FacebookService(facebookClient);
        var getAccountTask = facebookService.GetAccountAsync(accessToken);

        Task.WaitAll(getAccountTask);
        var account = getAccountTask.Result;
        Console.WriteLine($"{account.Id} {account.Name}");

        var postOnWallTask = facebookService.PostOnWallAsync(accessToken, "Hello from C# .NET
        Task.WaitAll(postOnWallTask);
    }
}
```

And finally, our actual application makes use of the *FacebookService* in order to get our Facebook account and post a message onto our wall. Just remember to get your access token **here** and mark the needed permissions. And as you may have already noticed, it's a trivial console application, therefore the *Task.WaitAll()* is being invoked.

It wasn't that difficult after all, was it? You can download the source code of this sample application by clicking **here** .

WRITTEN BY

Piotr Gankiewicz

Microsoft MVP, Bottega Trainer, Software Engineer & Architect, Noordwind co-founder, Open source contributor, Speaker, Motorcyclist, Athlete

NEXT READING

BECOMING A SOFTWARE DEVELOPER COURSES

Becoming a software developer – episode II

2 February 2017 17 Comments Piotr Gankiewicz 7 min read

23 Comments

Pingback: Accessing Facebook API using C# (video tutorial + article) - How to Code .NET

Pingback: Dew Drop - February 6, 2017 (#2415) - Morning Dew

Pingback: Compelling Sunday – 16 Posts on Programming and QA

Pingback: Dew Drop – February 6, 2017 (#2415) (by Alvin A.) - ugurak.net

JESPER URBAN

Hi Piotr,

Nice to meet you and thanks for the nice blog. We are soon starting to make an facebook integration and I was wondering whether you would be interested in helping us on this task on a project basis?

I am looking forward to hearing back from you.

All the best
Jesper Urban
+ 45 31 41 11 07

REPLY ↓

PHAM DAT

Hi, thanks for the tutor.
Can you tell me how to add this source code into 1 project 😀 😀 😀

REPLY ↓

MIZGIN ERKEK

hı
in this code you post on wall but L want to post to facebook page how L can do ?
could you help me

REPLY ↓

QUOC DANH

Hi Piotr,
Thanks for the blog. Can you tell me about endpoint in method _facebookClient.GetAsync.

REPLY ↓

Pingback: Accessing Number of likes and comments per post on a fb page using C# and Facebook Graph API [on hold] - ExceptionsHub

MAREK

Drugi problem i znów ty znasz rozwiązanie.
Po raz kolejny dzięki.

REPLY ↓

GARIK GEVORGYAN

Hi Piotr, thanks for the tutor.
I have question.Why You use FaceBook 7.0.6 NuGet package and why. In whis app you never use it.
I uninstall this package and app continued to work properly.

Thanks in advance

REPLY ↓

JOY

Thanks for your video, I can post text to my wall now.
But do you know how to send messages to friends for fan page people?

Thanks in advanced.

REPLY ↓

JOY

*or fan page people

REPLY ↓

Pingback: 2017 summary | Piotr Gankiewicz

DINGUYEN

Hi Bro,

Thank you so much for your great tutorial. It gives me the perfect basic step in Facebook Graph API.

REPLY ↓

PIOTR GANKIEWICZ

You're welcome :).

REPLY ↓

MOHAMMAD

Great POST 🙂
Could you please advise for a C# book to be read to understand your code bes?

REPLY ↓

TOMMY

Thank you.

I hope you can help.

I want to create a photo album and upload photos to it.

So far I have been able to create the photo album with this

```
public async Task PostOnWallAsync(string accessToken, string message)
=> await _facebookClient.PostAsync(accessToken, "me/albums", new {name="Test album"});
```

But now I am stuck, how do I upload photos to the album I created?

REPLY ↓

LINDSAY

Hi,
Great blog and code. All implements into Xamarin proj seamlessly.
One issue now … once it gets to this:

var response = await _httpClient.GetAsync($”{endpoint}?access_token={accessToken}”);

It waits and waits and waits…

Do you know if anything has change on the GB graph endpoint?
Anything else to add / amend?

I implemented as-is and already retrieve the access_token. Just want account details.

Thx

REPLY ↓

BRENT

change the method signature to async

public async Task GetAsync(string accessToken, string endpoint, string args = null)

REPLY ↓

OMER

thank you.

I tried with VS .Net platform (not Core)
I didn’t arrange project.json
error : Your project.json doesn’t have a runtimes section. You should add ‘”runtimes”: { “win”: { } }’

Can you help me?

```
“frameworks”: {
“net461”: {
“runtimes”: { “win”: {} }
}
}
```

REPLY

SHUBHANG SHARMA

Hey Piotr,

I have an app(website) registered for me and have app access token for it. Using graph explorer, I can create user access token, however as per facebook documentation, we shouldn’t hard code the user access token and should always go through their SDK or so.
Any idea, if we can generate user access token programmatically using app access token?

REPLY

RAFAEL VETRONE

Hello Piotr, your tutorial is good. But I’m having some trouble making it work. I created a ASPNET MVC 4.5.2 application. But HttpClient GetAsync never return! I read at stackoverflow some people having the same issues with MVC 4.5.2. I do these calls in a MvcController Index method. Perhaps I will change to using WebRequests for the REST calls.

REPLY

Leave A Comment

COMMENT

NAME *

EMAIL *

WEBSITE

Post Comment