Deep Dive into Async Programming

by Salih Cantekin

TASK

```
ThreadPool.QueueUserWorkItem(doSomethingObj);
```

```
Task task = Task.Run(doSometing);
```

```
task.Wait();
```

```
_ = task.ContinueWith((t) => { });
```

```
0 references
async void Foo()
```

```
_ = task.Result;
```

```
_ = new Thread(doSometing);
```

```
_ = task.ConfigureAwait(false);
```

```
_ = task.WaitAsync(cancellationToken);
```

```
Task doSomethingAsync()
{
    //await doSometingElseAsync();
    return doSometingElseAsync();
}
```

```
task.GetAwaiter().GetResult();
```

Salih Cantekin

Lead Developer

salihcantekin

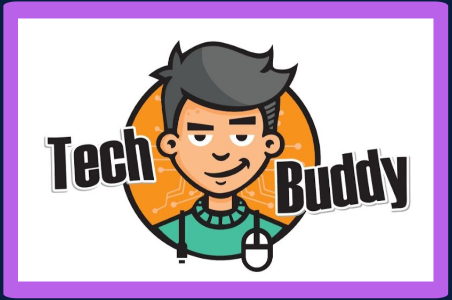
salihcantekin

salihcantekin

salih_sc

TechBuddyTR

salihcantekin



Microsoft MVP 2023 Most Valuable Professional

NEXT

Ne anlatacak bu?

01 Terminoloji

02 Temel Kavramlar

03 Hikayeye Giriş

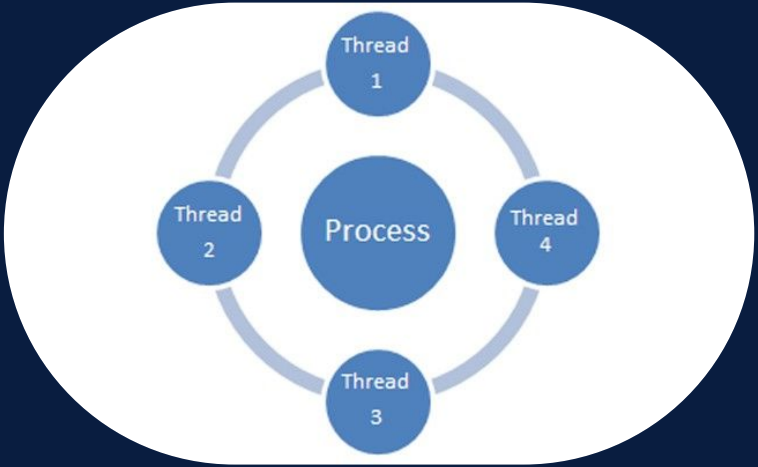
04 Alıştırmalar

05 Soru - Cevap

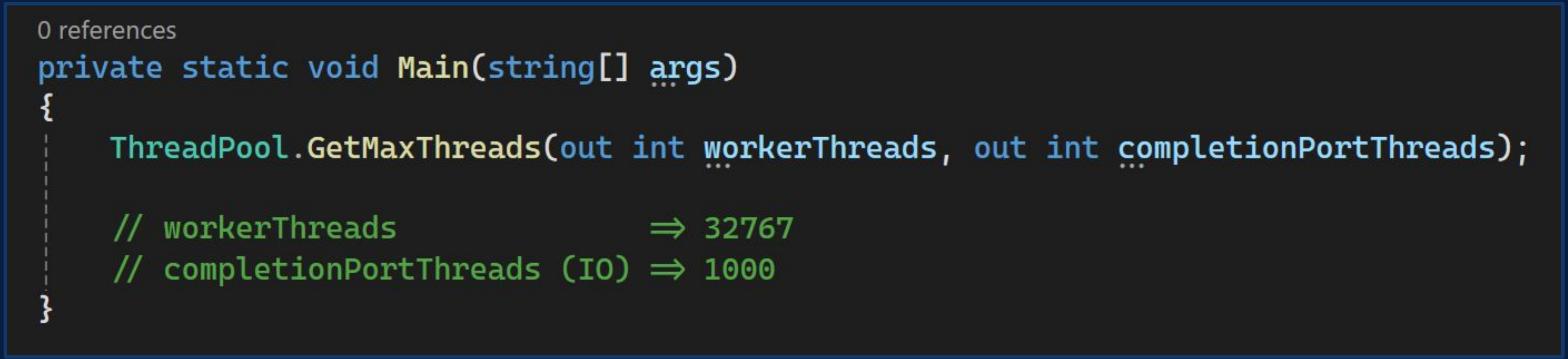
BU İŞİN TEMELİ

THREAD

Thread vs Task?



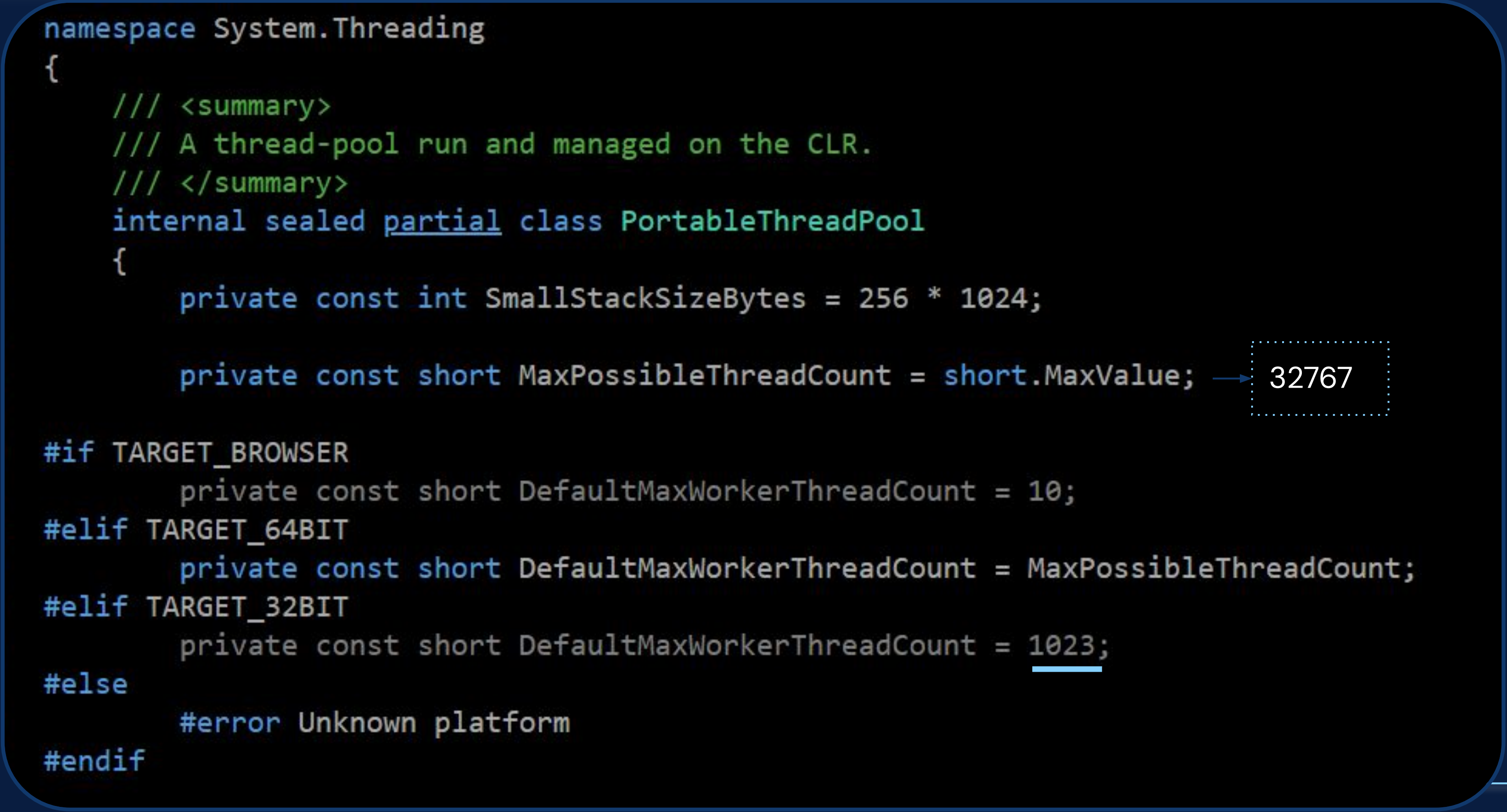
THEADPOOL



```
0 references
private static void Main(string[] args)
{
    ThreadPool.GetMaxThreads(out int workerThreads, out int completionPortThreads);

    // workerThreads                ⇒ 32767
    // completionPortThreads (IO) ⇒ 1000
}
```

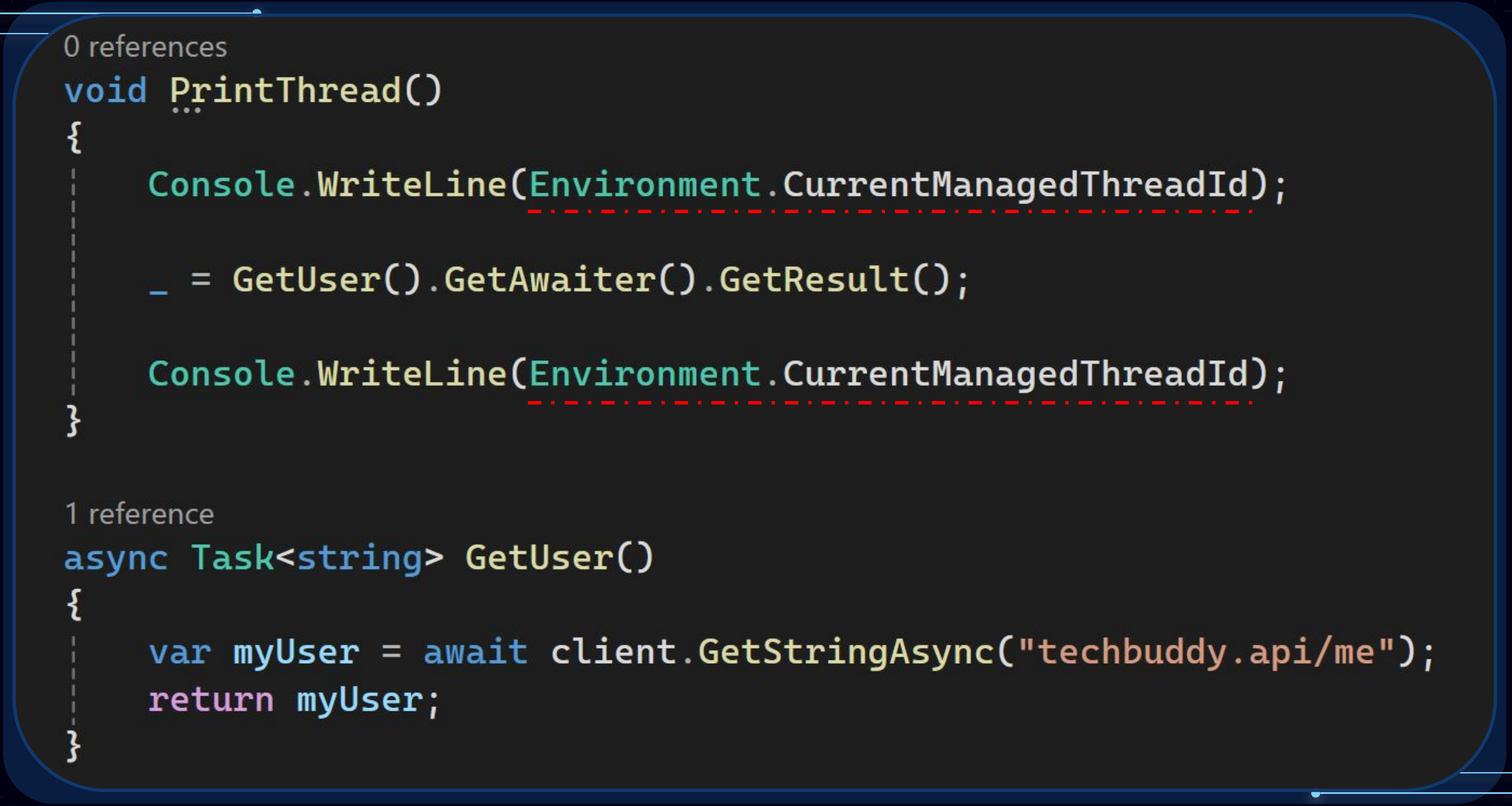
learn.microsoft

THEADPOOL



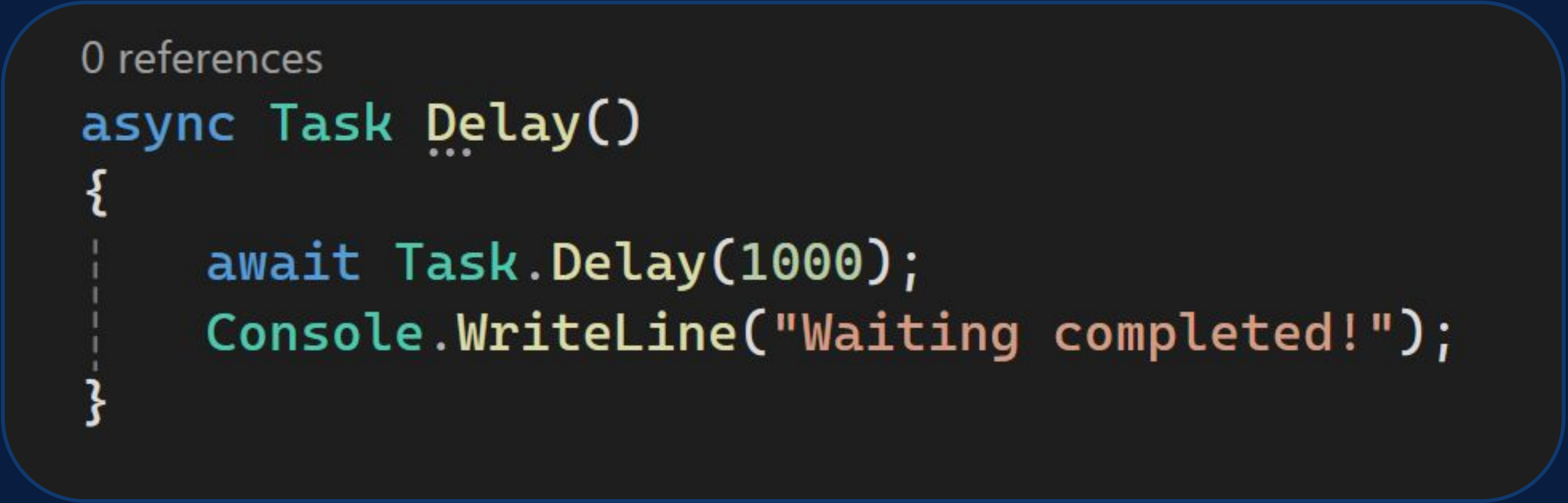
```
namespace System.Threading
{
    /// <summary>
    /// A thread-pool run and managed on the CLR.
    /// </summary>
    internal sealed partial class PortableThreadPool
    {
        private const int SmallStackSizeBytes = 256 * 1024;

        private const short MaxPossibleThreadCount = short.MaxValue;

#if TARGET_BROWSER
        private const short DefaultMaxWorkerThreadCount = 10;
#elif TARGET_64BIT
        private const short DefaultMaxWorkerThreadCount = MaxPossibleThreadCount;
#elif TARGET_32BIT
        private const short DefaultMaxWorkerThreadCount = 1023;
#else
        #error Unknown platform
#endif
```

$\rightarrow$ 32767



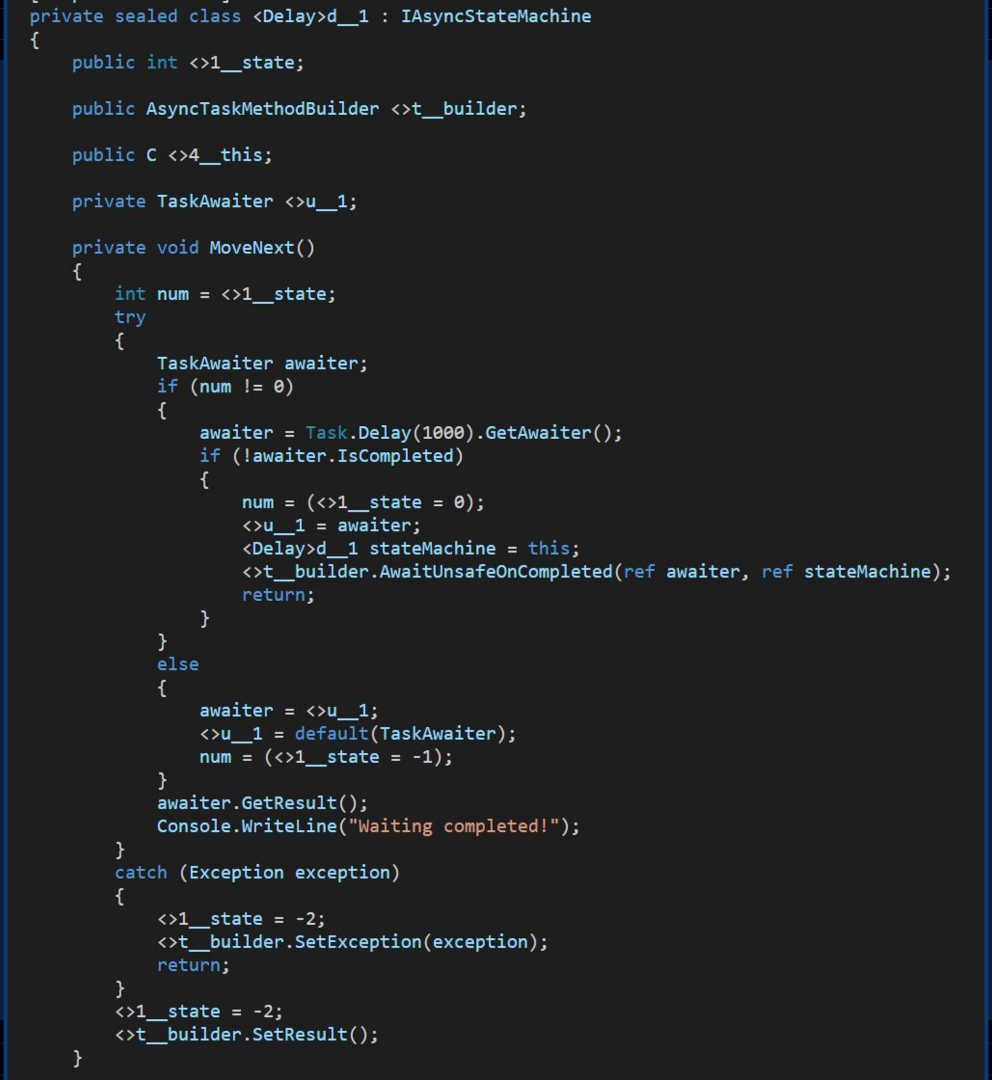
```
0 references
void PrintThread()
{
    Console.WriteLine(Environment.CurrentManagedThreadId);

    _ = GetUser().GetAwaiter().GetResult();

    Console.WriteLine(Environment.CurrentManagedThreadId);
}

1 reference
async Task<string> GetUser()
{
    var myUser = await client.GetStringAsync("techbuddy.api/me");
    return myUser;
}
```

En basitten başlayalım - Ne görüyoruz?



```
0 references
async Task Delay()
{
    await Task.Delay(1000);
    Console.WriteLine("Waiting completed!");
}
```

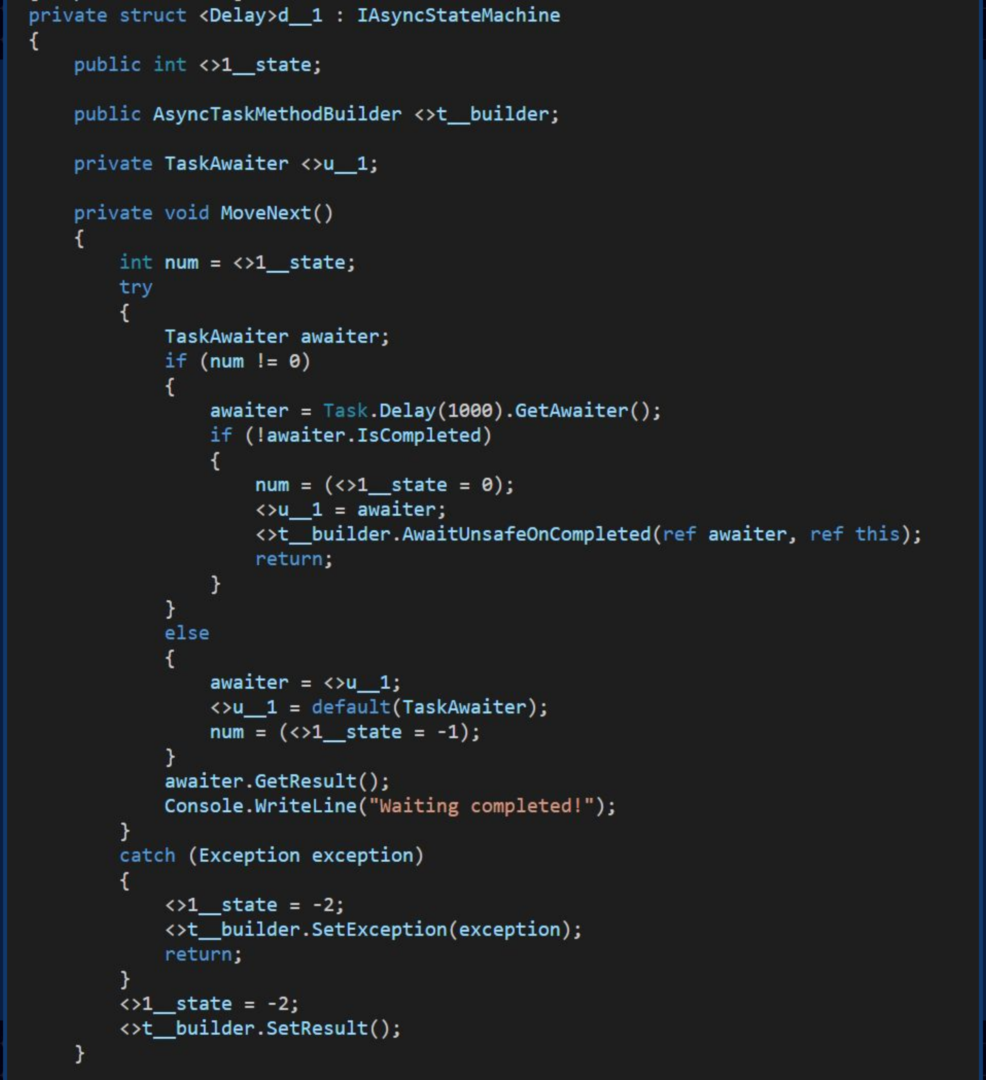
STATE MACHINE DEBUG



```
private sealed class <Delay>d__1 : IAsyncStateMachine
{
    public int <>1__state;

    public AsyncTaskMethodBuilder <>t__builder;

    public C <>4__this;

    private TaskAwaiter <>u__1;

    private void MoveNext()
    {
        int num = <>1__state;
        try
        {
            TaskAwaiter awaiter;
            if (num != 0)
            {
                awaiter = Task.Delay(1000).GetAwaiter();
                if (!awaiter.IsCompleted)
                {
                    num = (<>1__state = 0);
                    <>u__1 = awaiter;
                    <Delay>d__1 stateMachine = this;
                    <>t__builder.AwaitUnsafeOnCompleted(ref awaiter, ref stateMachine);
                    return;
                }
            }
            else
            {
                awaiter = <>u__1;
                <>u__1 = default(TaskAwaiter);
                num = (<>1__state = -1);
            }
            awaiter.GetResult();
            Console.WriteLine("Waiting completed!");
        }
        catch (Exception exception)
        {
            <>1__state = -2;
            <>t__builder.SetException(exception);
            return;
        }
        <>1__state = -2;
        <>t__builder.SetResult();
    }
```

STATE MACHINE DEBUG

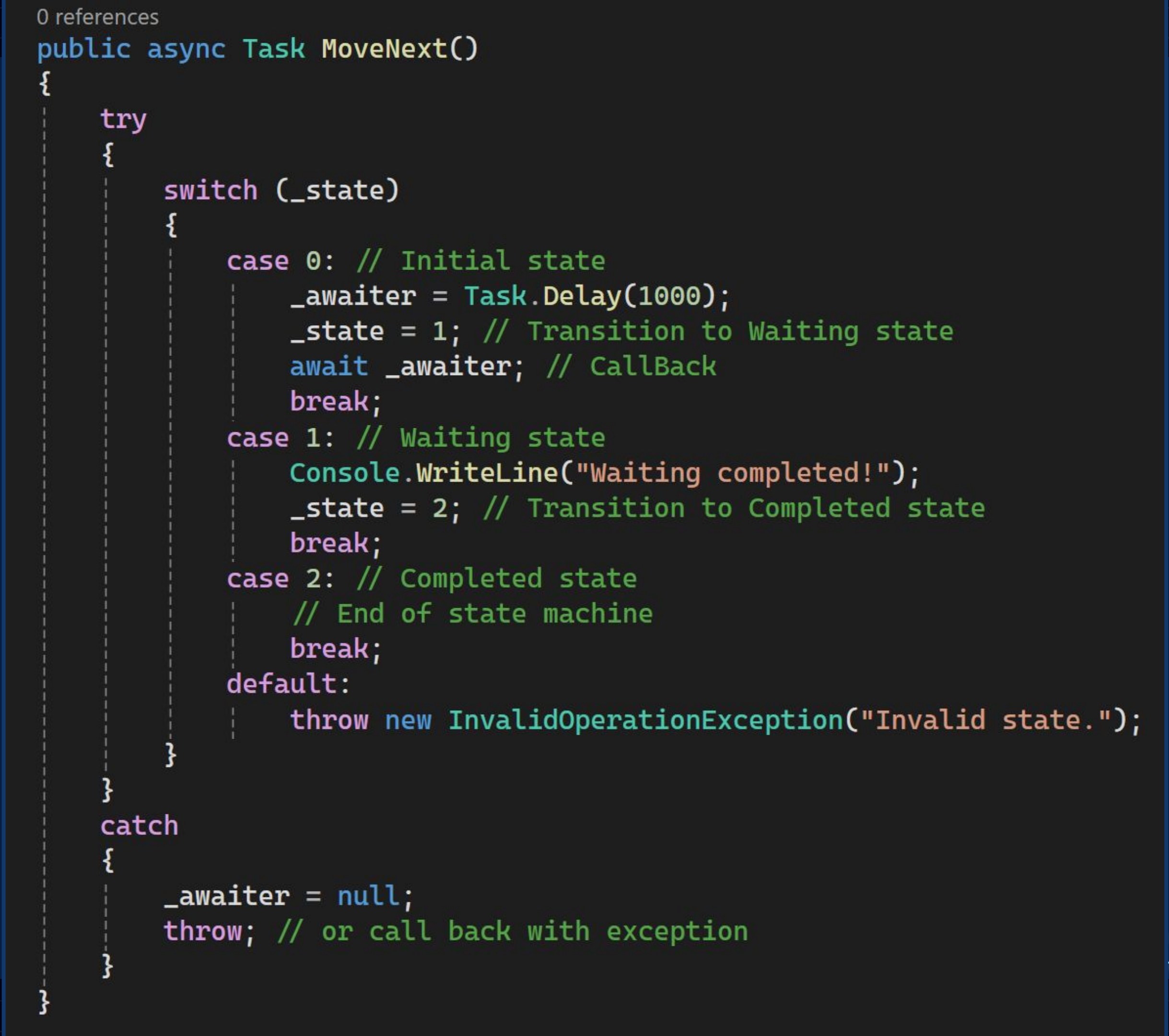
STATE MACHINE RELEASE



```
private struct <Delay>d__1 : IAsyncStateMachine
{
    public int <>1__state;

    public AsyncTaskMethodBuilder <>t__builder;

    private TaskAwaiter <>u__1;

    private void MoveNext()
    {
        int num = <>1__state;
        try
        {
            TaskAwaiter awaiter;
            if (num != 0)
            {
                awaiter = Task.Delay(1000).GetAwaiter();
                if (!awaiter.IsCompleted)
                {
                    num = (<>1__state = 0);
                    <>u__1 = awaiter;
                    <>t__builder.AwaitUnsafeOnCompleted(ref awaiter, ref this);
                    return;
                }
            }
            else
            {
                awaiter = <>u__1;
                <>u__1 = default(TaskAwaiter);
                num = (<>1__state = -1);
            }
            awaiter.GetResult();
            Console.WriteLine("Waiting completed!");
        }
        catch (Exception exception)
        {
            <>1__state = -2;
            <>t__builder.SetException(exception);
            return;
        }
        <>1__state = -2;
        <>t__builder.SetResult();
    }
```

STATE MACHINE RELEASE

SIMPLIFIED STATE MACHINE

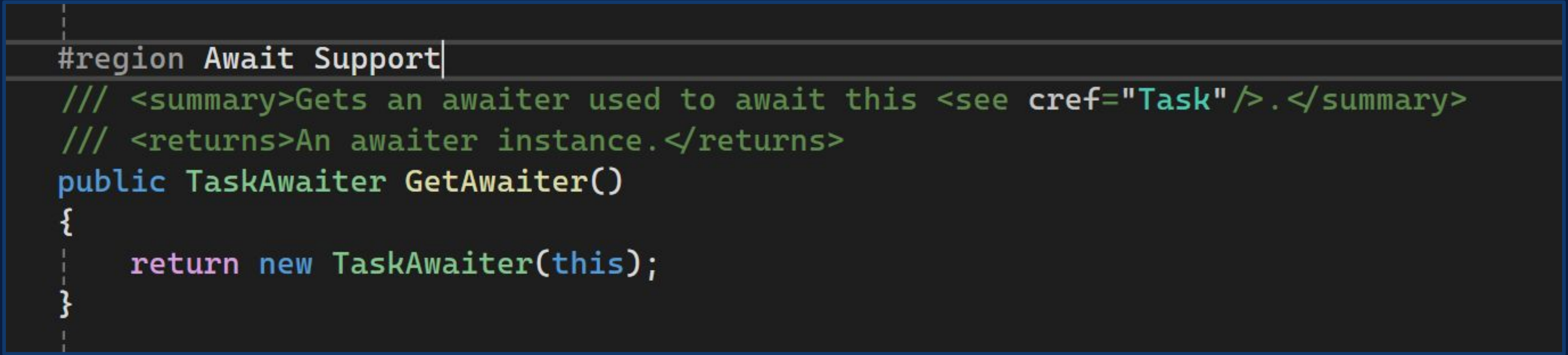


```
0 references
public async Task MoveNext()
{
    try
    {
        switch (_state)
        {
            case 0: // Initial state
                _awaiter = Task.Delay(1000);
                _state = 1; // Transition to Waiting state
                await _awaiter; // CallBack
                break;
            case 1: // Waiting state
                Console.WriteLine("Waiting completed!");
                _state = 2; // Transition to Completed state
                break;
            case 2: // Completed state
                // End of state machine
                break;
            default:
                throw new InvalidOperationException("Invalid state.");
        }
    }
    catch
    {
        _awaiter = null;
        throw; // or call back with exception
    }
}
```

SIMPLIFIED STATE MACHINE

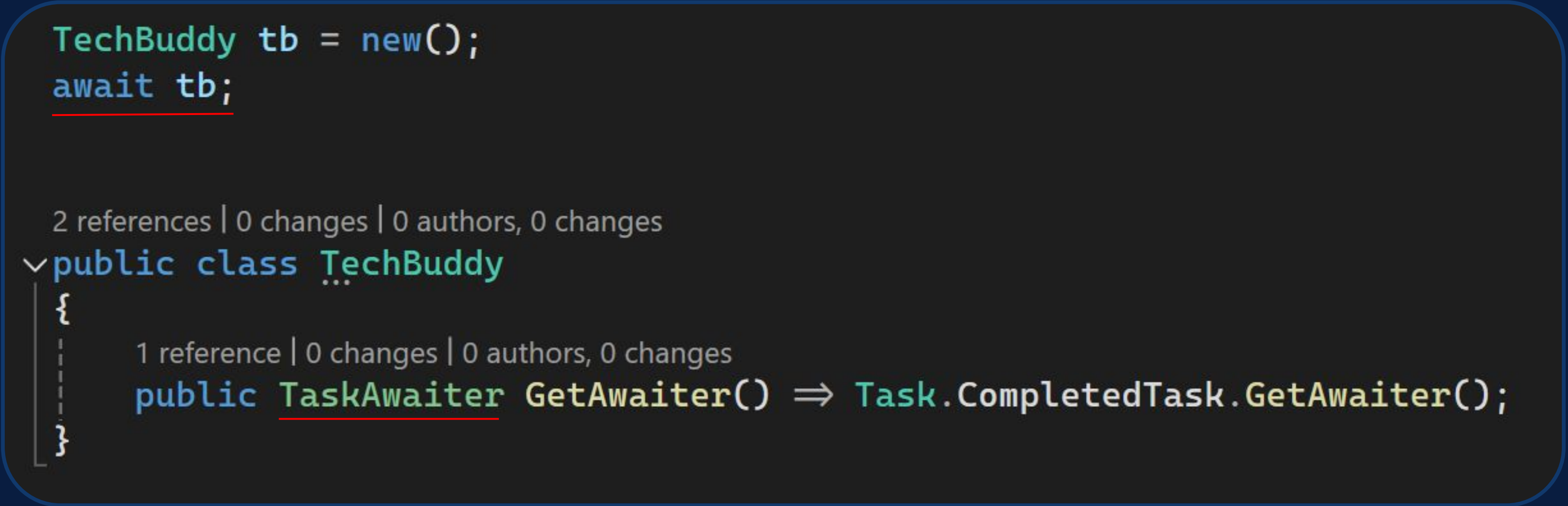
Awaiter()

Task



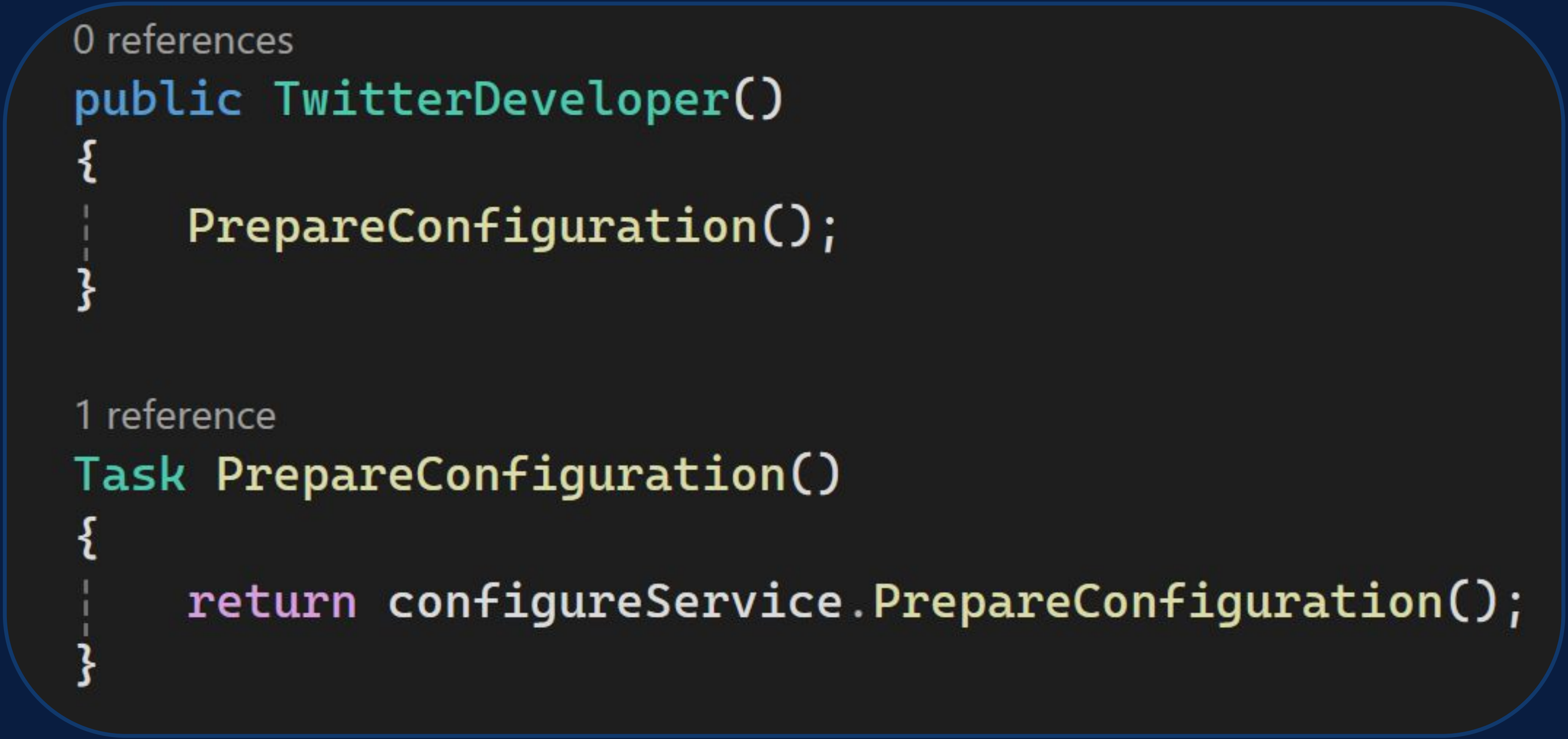
```
#region Await Support
/// <summary>Gets an awaiter used to await this <see cref="Task"/>.</summary>
/// <returns>An awaiter instance.</returns>
public TaskAwaiter GetAwaiter()
{
    return new TaskAwaiter(this);
}
```

TechBuddy



```
TechBuddy tb = new();
await tb;


2 references | 0 changes | 0 authors, 0 changes
public class TechBuddy
{
    1 reference | 0 changes | 0 authors, 0 changes
    public TaskAwaiter GetAwaiter() => Task.CompletedTask.GetAwaiter();
}
```

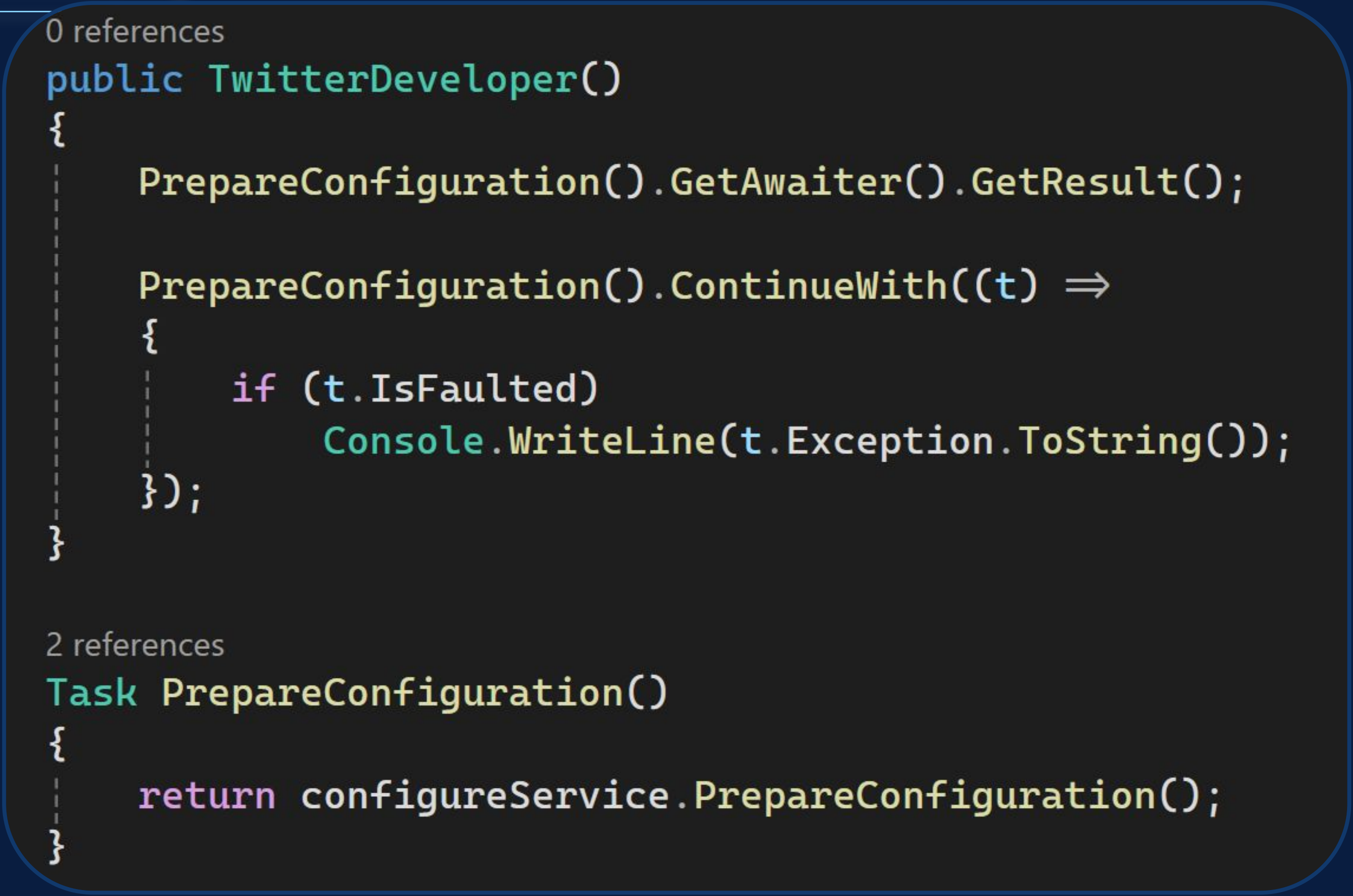
İyileştirmeler



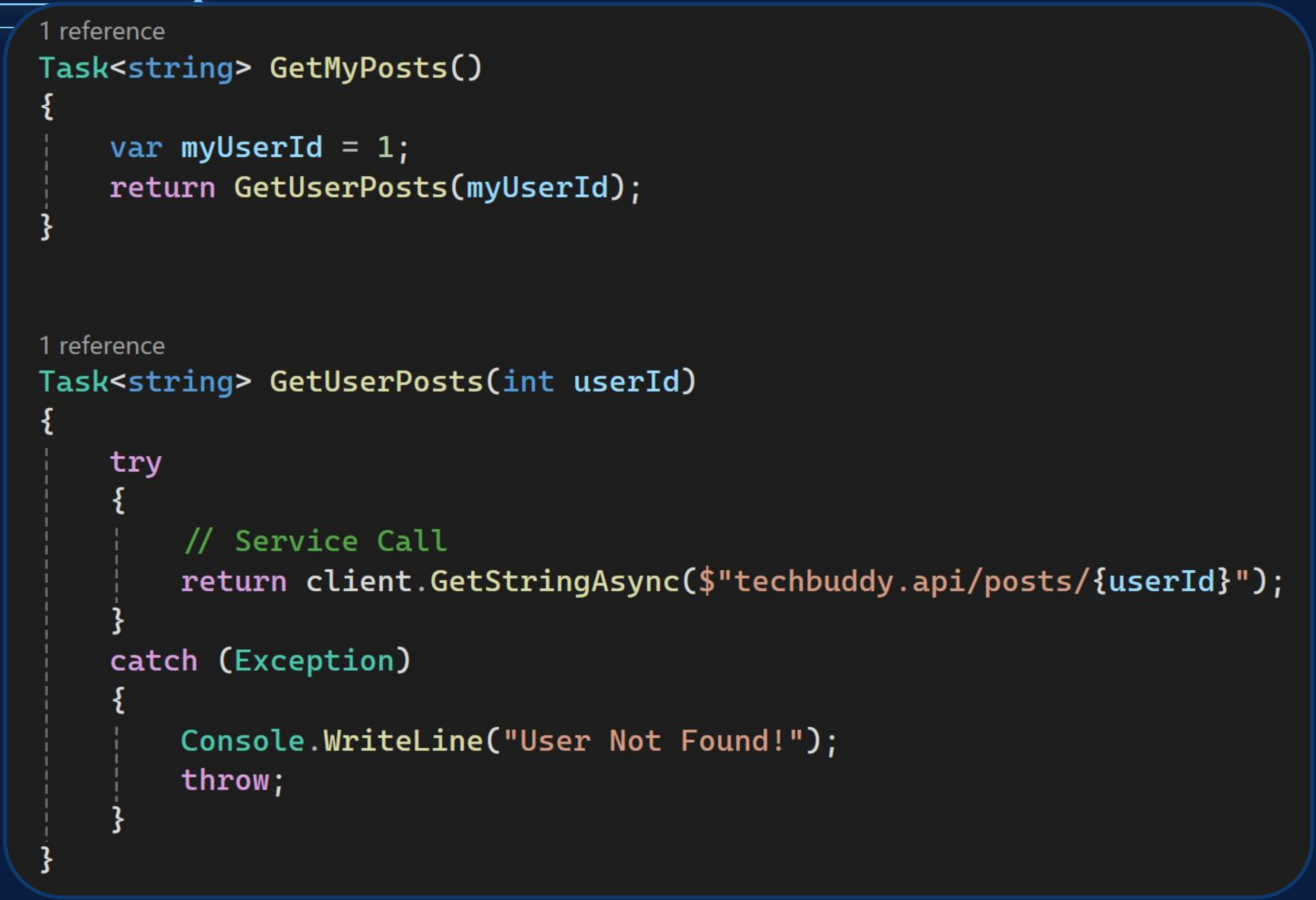
```
0 references
public TwitterDeveloper()
{
    PrepareConfiguration();
}

1 reference
Task PrepareConfiguration()
{
    return configureService.PrepareConfiguration();
}
```

SORU

CEVAP

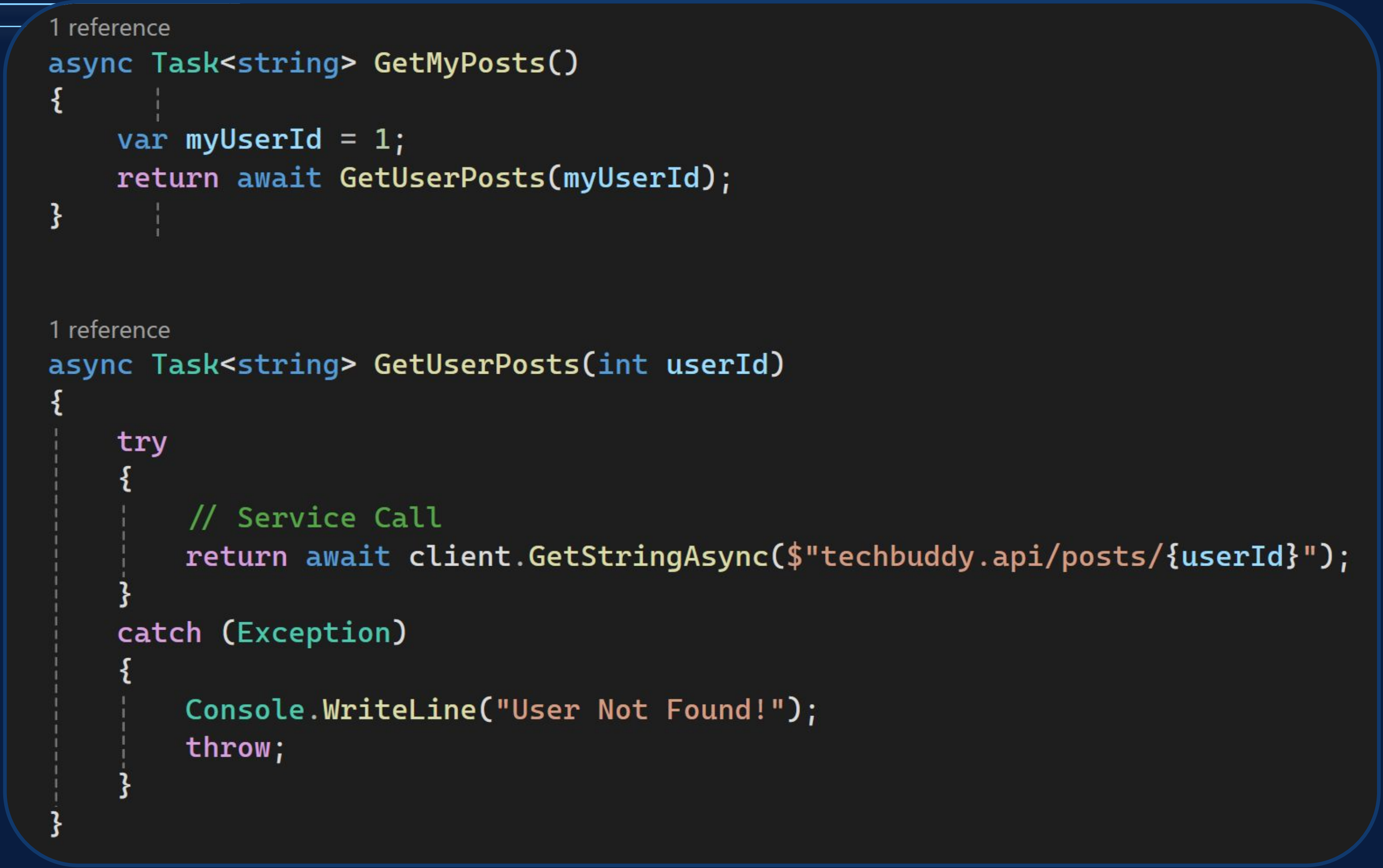


```
0 references
public TwitterDeveloper()
{
    PrepareConfiguration().GetAwaiter().GetResult();

    PrepareConfiguration().ContinueWith((t) =>
    {
        if (t.IsFaulted)
            Console.WriteLine(t.Exception.ToString());
    });
}

2 references
Task PrepareConfiguration()
{
    return configureService.PrepareConfiguration();
}
```



```
1 reference
Task<string> GetMyPosts()
{
    var myUserId = 1;
    return GetUserPosts(myUserId);
}


1 reference
Task<string> GetUserPosts(int userId)
{
    try
    {
        // Service Call
        return client.GetStringAsync($"techbuddy.api/posts/{userId}");
    }
    catch (Exception)
    {
        Console.WriteLine("User Not Found!");
        throw;
    }
}
```

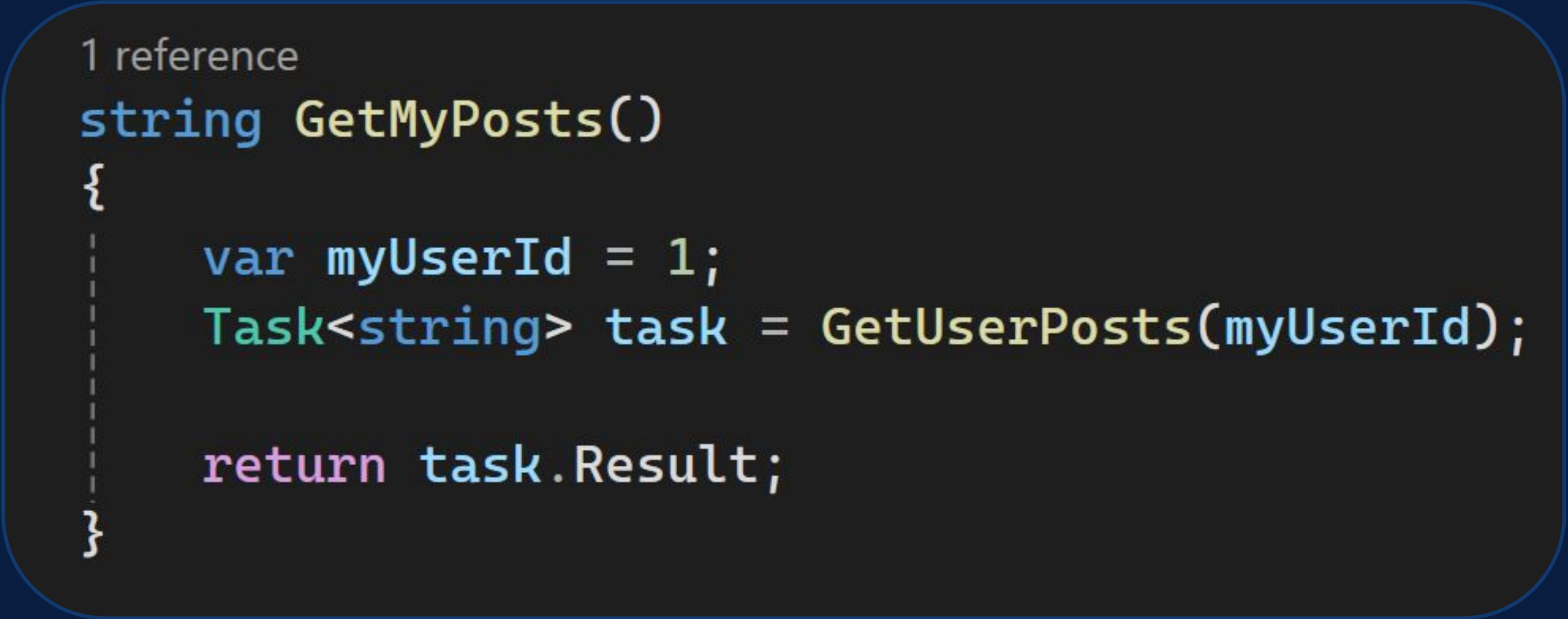
SORU



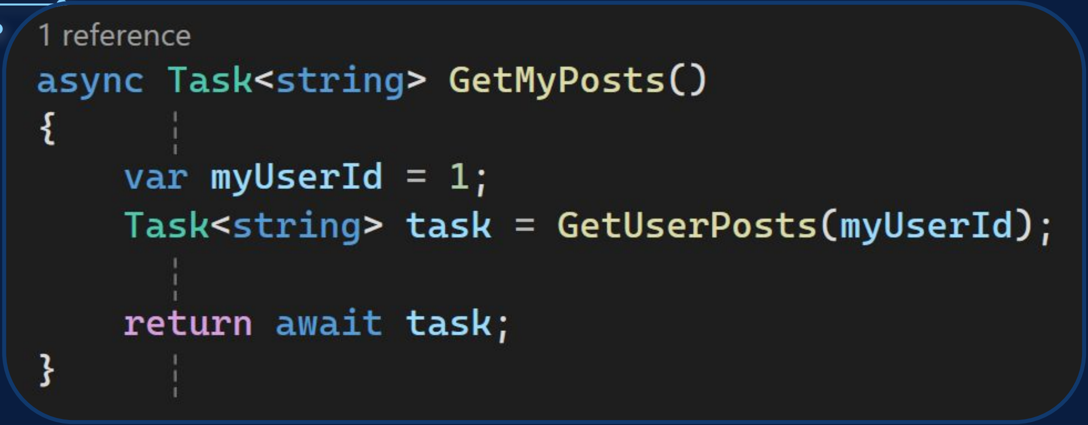
```
1 reference
async Task<string> GetMyPosts()
{
    var myUserId = 1;
    return await GetUserPosts(myUserId);
}


1 reference
async Task<string> GetUserPosts(int userId)
{
    try
    {
        // Service Call
        return await client.GetStringAsync($"techbuddy.api/posts/{userId}");
    }
    catch (Exception)
    {
        Console.WriteLine("User Not Found!");
        throw;
    }
}
```

CEVAP

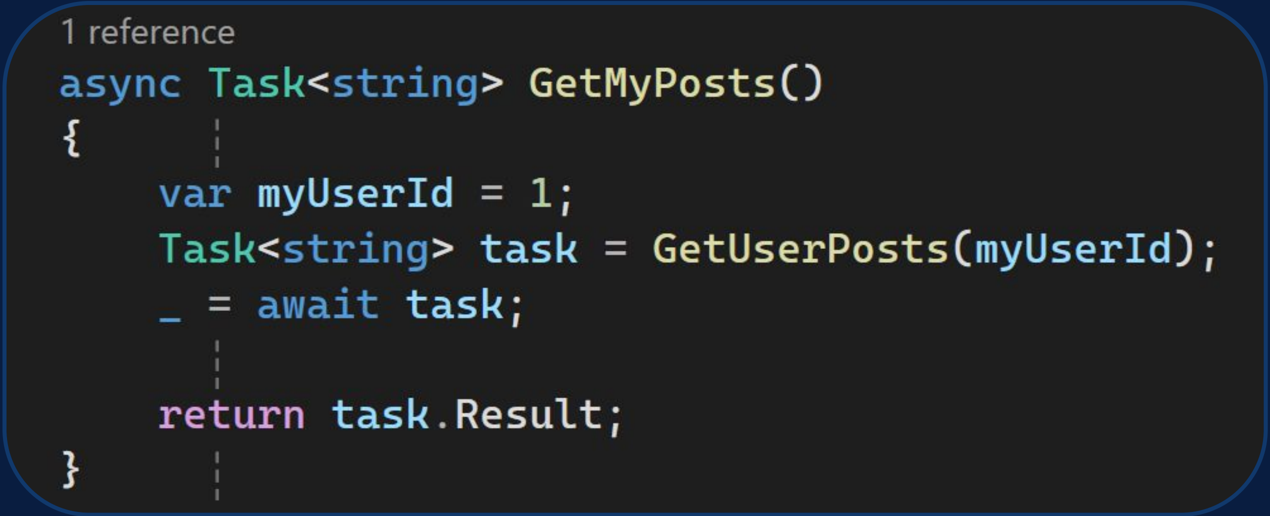
SORU



```
1 reference
string GetMyPosts()
{
    var myUserId = 1;
    Task<string> task = GetUserPosts(myUserId);

    return task.Result;
}
```



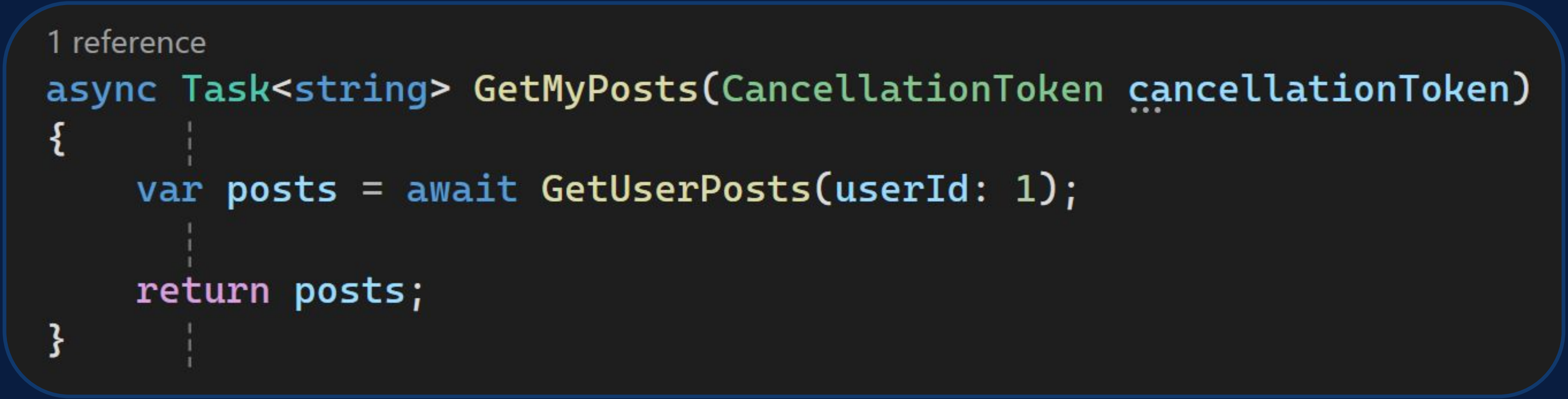
```
1 reference
async Task<string> GetMyPosts()
{
    var myUserId = 1;
    Task<string> task = GetUserPosts(myUserId);

    return await task;
}
```



```
1 reference
async Task<string> GetMyPosts()
{
    var myUserId = 1;
    Task<string> task = GetUserPosts(myUserId);
    _ = await task;

    return task.Result;
}
```

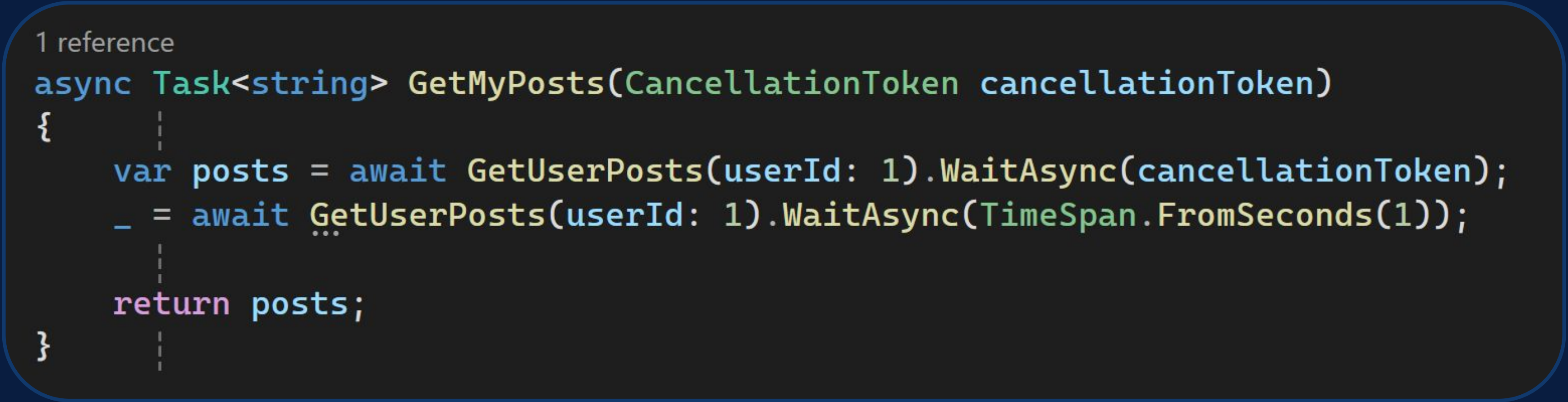
CEVAP

SORU



```
1 reference
async Task<string> GetMyPosts(CancellationToken cancellationToken)
{
    var posts = await GetUserPosts(userId: 1);

    return posts;
}
```

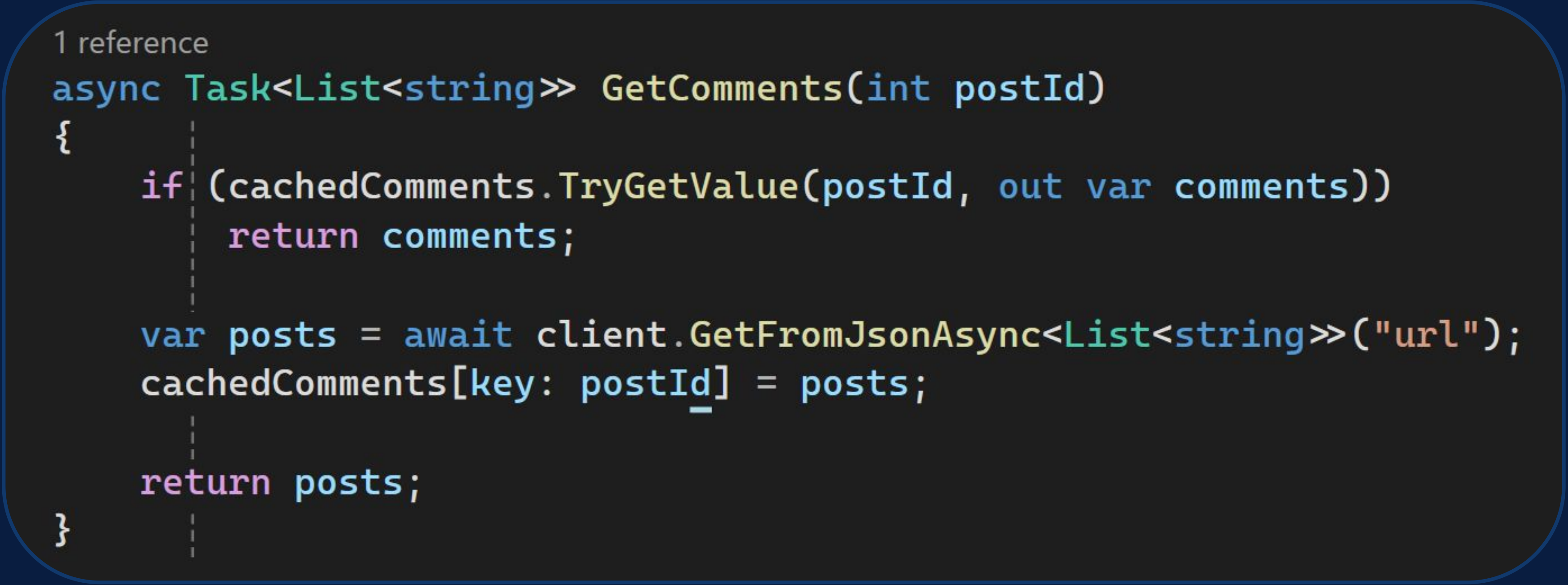
CEVAP



1 reference

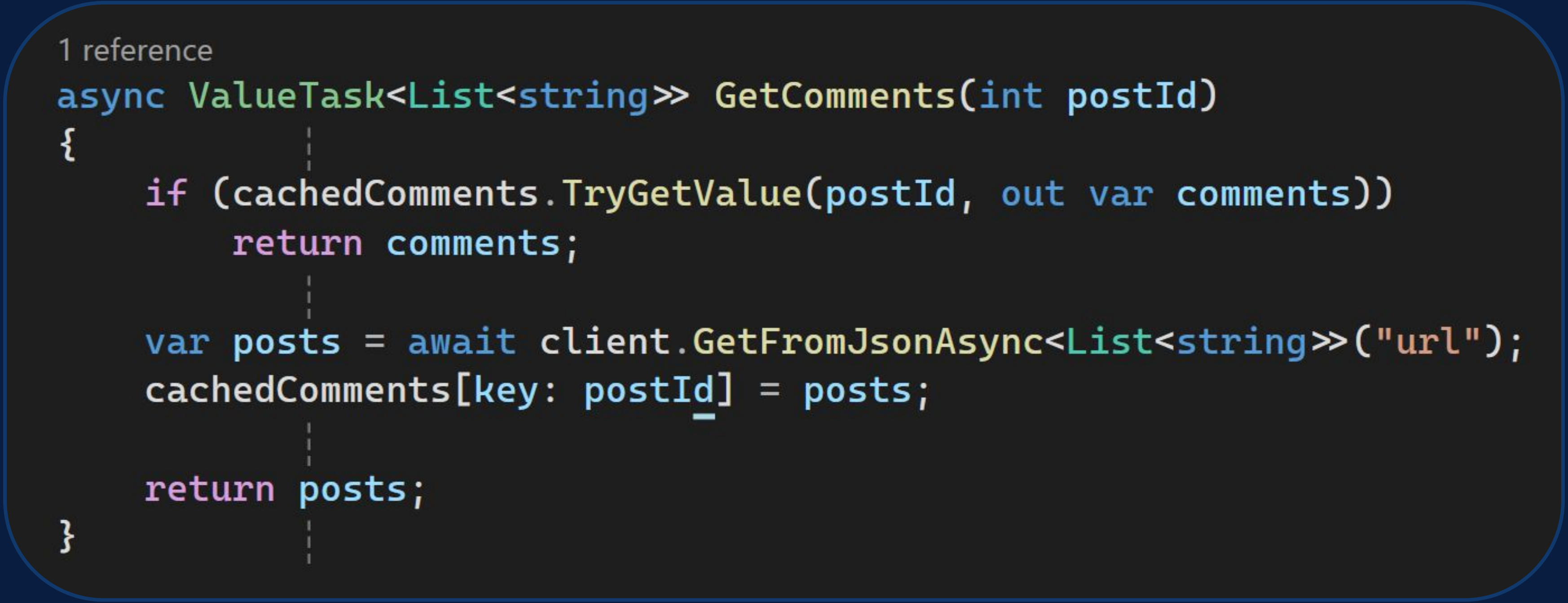
```
async Task<string> GetMyPosts(CancellationToken cancellationToken)
{
    var posts = await GetUserPosts(userId: 1).WaitAsync(cancellationToken);
    _ = await GetUserPosts(userId: 1).WaitAsync(TimeSpan.FromSeconds(1));

    return posts;
}
```

SORU

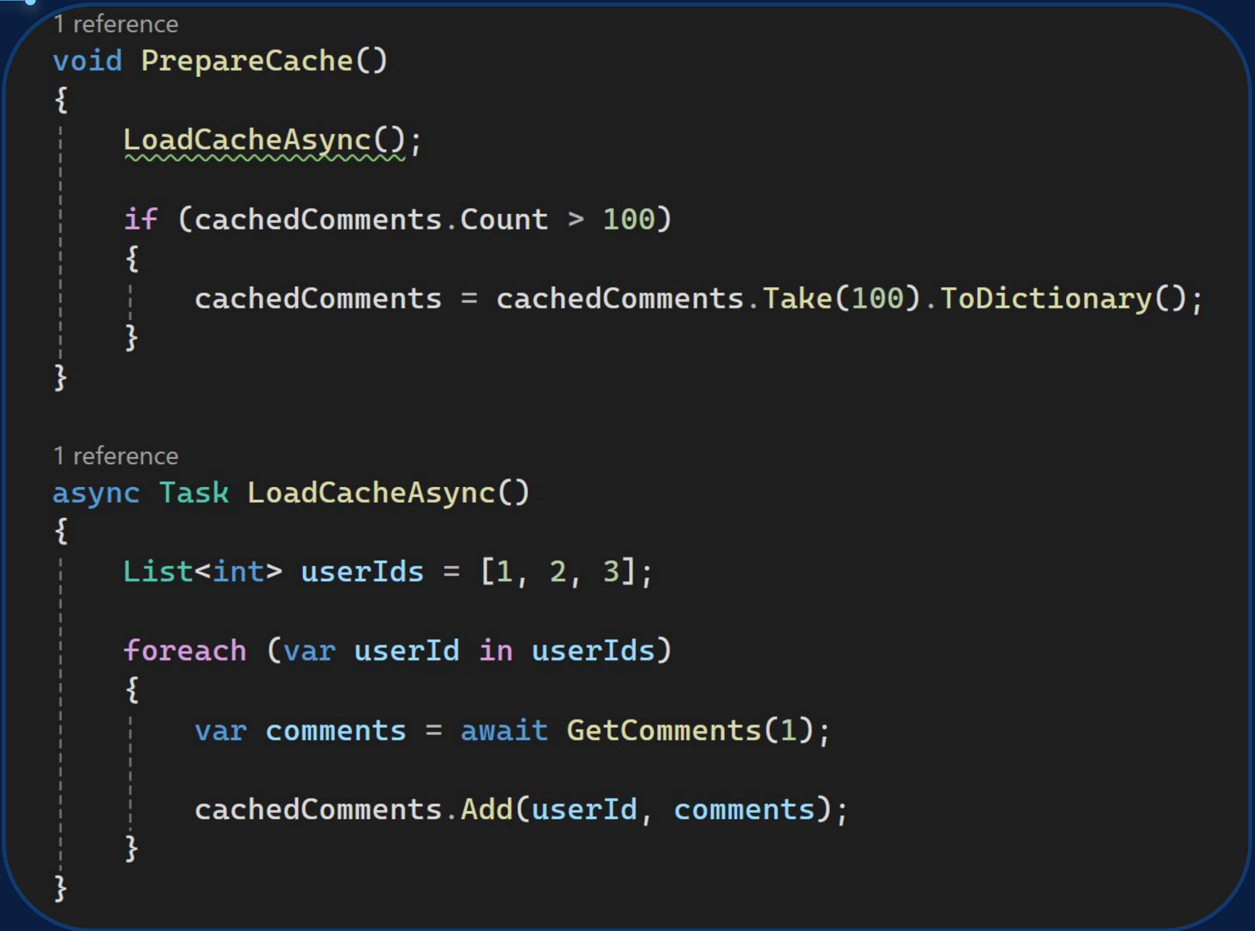


```
1 reference
async Task<List<string>> GetComments(int postId)
{
    if (cachedComments.TryGetValue(postId, out var comments))
        return comments;

    var posts = await client.GetFromJsonAsync<List<string>>("url");
    cachedComments[key: postId] = posts;

    return posts;
}
```

CEVAP

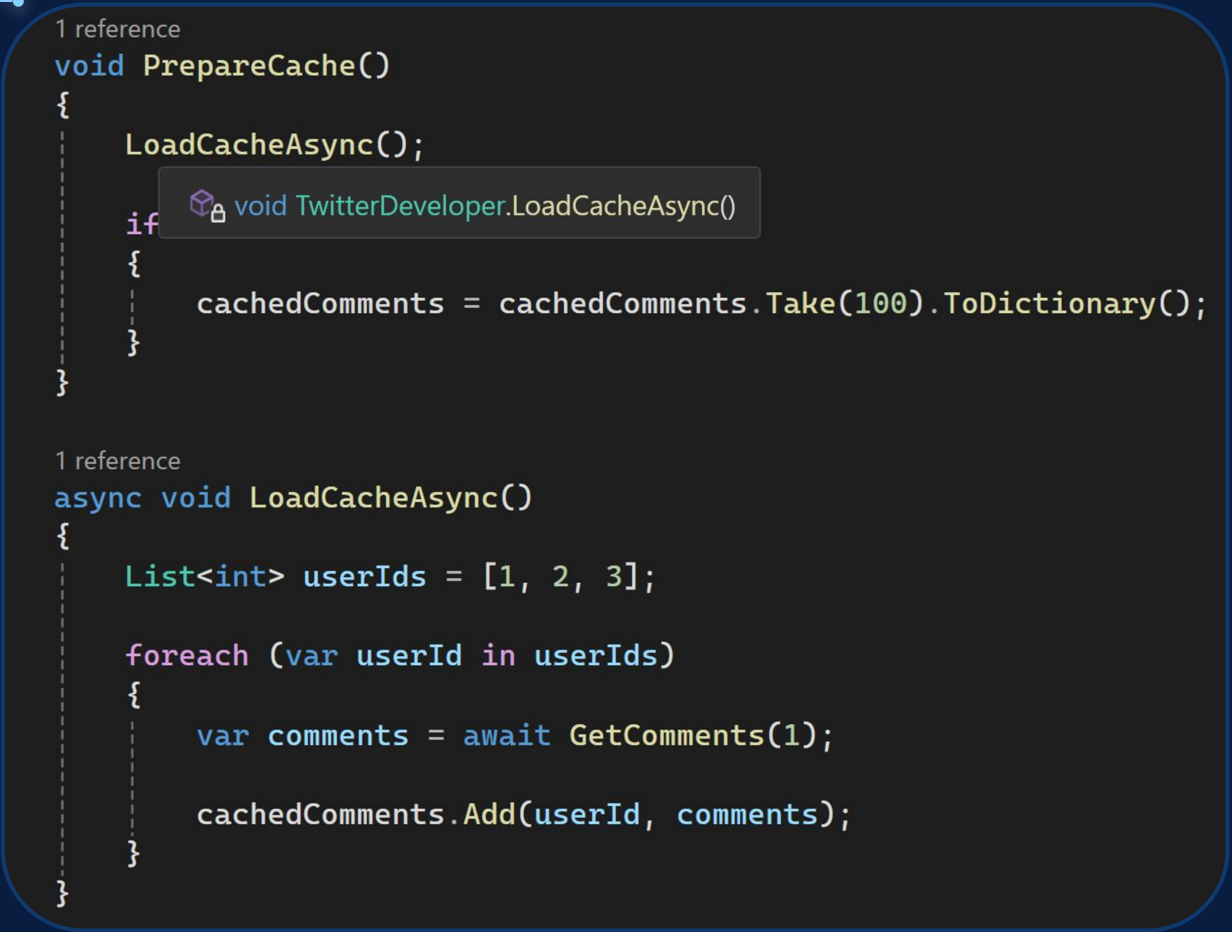


```
1 reference
async ValueTask<List<string>> GetComments(int postId)
{
    if (cachedComments.TryGetValue(postId, out var comments))
        return comments;

    var posts = await client.GetFromJsonAsync<List<string>>("url");
    cachedComments[key: postId] = posts;

    return posts;
}
```



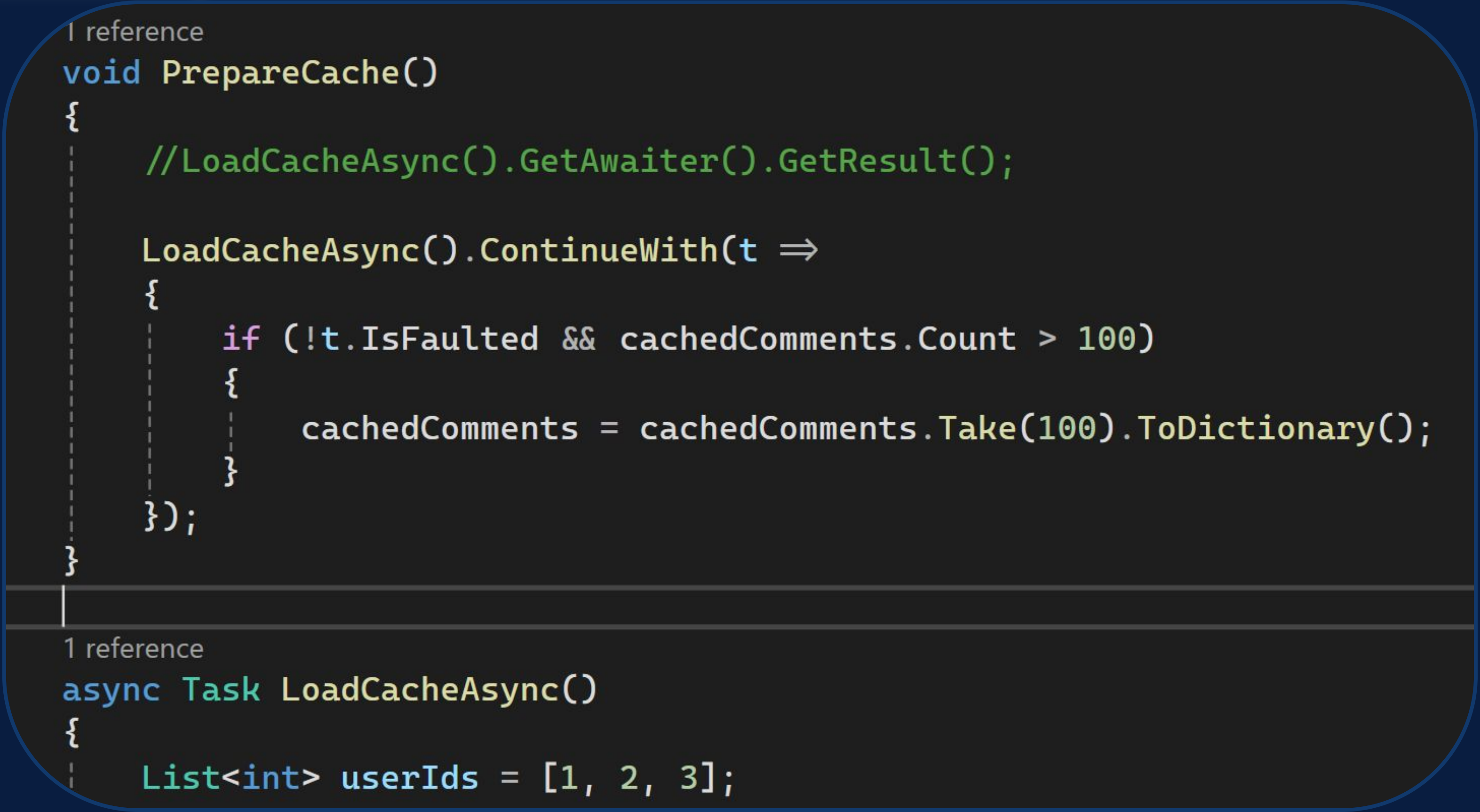
```
1 reference
void PrepareCache()
{
    LoadCacheAsync();

    if (cachedComments.Count > 100)
    {
        cachedComments = cachedComments.Take(100).ToDictionary();
    }
}

1 reference
async Task LoadCacheAsync()
{
    List<int> userIds = [1, 2, 3];

    foreach (var userId in userIds)
    {
        var comments = await GetComments(1);

        cachedComments.Add(userId, comments);
    }
}
```

SORU V1



```
1 reference
void PrepareCache()
{
    LoadCacheAsync();
                  void TwitterDeveloper.LoadCacheAsync()
    if
    {
        cachedComments = cachedComments.Take(100).ToDictionary();
    }
}

1 reference
async void LoadCacheAsync()
{
    List<int> userIds = [1, 2, 3];

    foreach (var userId in userIds)
    {
        var comments = await GetComments(1);

        cachedComments.Add(userId, comments);
    }
}
```

SORU V2



```
1 reference
void PrepareCache()
{
    //LoadCacheAsync().GetAwaiter().GetResult();

    LoadCacheAsync().ContinueWith(t =>
    {
        if (!t.IsFaulted && cachedComments.Count > 100)
        {
            cachedComments = cachedComments.Take(100).ToDictionary();
        }
    });
}

1 reference
async Task LoadCacheAsync()
{
    List<int> userIds = [1, 2, 3];
```

CEVAP

Konuşamadıklarımız...

Task.WhenAll()

IAsyncEnumerable

Task.WhenAny()

IAsyncDisposable

Task.WaitAll()

Task.WaitAny()

Teşekkürler

Herhangi bir soru için

salihcantekin@gmail.com

@TechBuddyTR

Download Slide

SCAN ME