.NET Cancellation Token

Canceling the operation is done by raising an exception within the method.

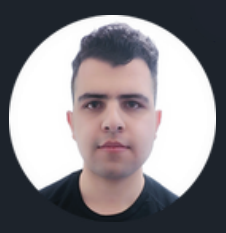
```
var cancellationTokenSource = new CancellationTokenSource();
var cancellationToken = cancellationTokenSource.Token;

Task longRunningTask = Task.Run(() =>
{
    // Simulate a long-running operation
    for (int i = 0; i < 1000000; i++)
    {
        // Check if cancellation is requested
        cancellationToken.ThrowIfCancellationRequested();

        // Simulate some work
        Console.WriteLine(i);
    }
}, cancellationToken);

// Simulate user input to cancel the operation after 1 second
await Task.Delay(1000, cancellationToken);
await cancellationTokenSource.CancelAsync();

try
{
    await longRunningTask;
    Console.WriteLine("Operation completed successfully.");
}
catch (OperationCanceledException)
{
    Console.WriteLine("Operation was cancelled.");
}
```



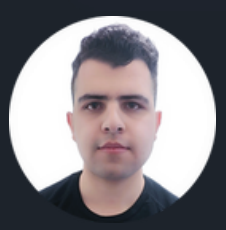
Elliot One

I/O Operation Example

Using cancellation tokens for I/O operations is a good practice for managing resources efficiently.

```
class Program
{
    static async Task Main(string[] args)
    {
        var cancellationTokenSource = new CancellationTokenSource();
        var cancellationToken = cancellationTokenSource.Token;
        try
        {
            // Start a task to call a web service asynchronously with the cancellation token
            var webServiceTask = CallWebServiceAsync("https://google.com", cancellationToken);
            // Simulate cancellation after 0.1 second
            await Task.Delay(100, cancellationToken);
            await cancellationTokenSource.CancelAsync();
            // Wait for the web service call to complete or be cancelled
            var responseData = await webServiceTask;
            Console.WriteLine("Response from web service:");
            Console.WriteLine(responseData);
        }
        catch (OperationCanceledException)
        {
            Console.WriteLine("Web service call was cancelled.");
        }
    }

    static async Task<string> CallWebServiceAsync(
        string url, CancellationToken cancellationToken)
    {
        Console.WriteLine("Web service call started.");
        using var httpClient = new HttpClient();
        // Make a GET request to the web service asynchronously
        var response = await httpClient.GetAsync(url, cancellationToken);
        response.EnsureSuccessStatusCode();
        // Read the response content asynchronously
        var responseData = await response.Content.ReadAsStringAsync(cancellationToken);
        Console.WriteLine("Web service call completed.");
        return responseData;
    }
}
```



Elliot One

Elliot One

Enjoyed Reading This?

Reshare and Spread Knowledge.