Here's a **complete, safe, and reusable C# example** to compute the **MD5 checksum** from a given `Stream` (e.g., `FileStream`, `MemoryStream`, `NetworkStream`).

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;

public static class Md5Helper
{
    /// <summary>
    /// Computes the MD5 checksum of the provided stream.
    /// The stream position will be reset to its original position after computation.
    /// </summary>
    /// <param name="inputStream">The input stream (must be readable).</param>
    /// <returns>Hexadecimal string representation of the MD5 hash.</returns>
    /// <exception cref="ArgumentNullException">Thrown if inputStream is null.</exception>
    /// <exception cref="ArgumentException">Thrown if stream is not readable.</exception>
    public static string ComputeMd5FromStream(Stream inputStream)
    {
        if (inputStream == null)
            throw new ArgumentNullException(nameof(inputStream), "Stream cannot be null.");

        if (!inputStream.CanRead)
            throw new ArgumentException("Stream must be readable.", nameof(inputStream));

        long originalPosition = 0;
```

```
        if (inputStream.CanSeek)
            originalPosition = inputStream.Position;

        try
        {
            using (var md5 = MD5.Create())
            {
                byte[] hashBytes = md5.ComputeHash(inputStream);

                // Convert byte array to hex string
                StringBuilder sb = new StringBuilder(hashBytes.Length * 2);
                foreach (byte b in hashBytes)
                    sb.Append(b.ToString("x2"));

                return sb.ToString();
            }
        }
        finally
        {
            // Reset stream position if possible
            if (inputStream.CanSeek)
                inputStream.Position = originalPosition;
        }
    }
}

// Example usage
```

```
class Program
{
    static void Main()
    {
        try
        {
            using (FileStream fs = File.OpenRead("example.txt"))
            {
                string md5sum = Md5Helper.ComputeMd5FromStream(fs);
                Console.WriteLine($"MD5: {md5sum}");
            }
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error computing MD5: {ex.Message}");
        }
    }
}
```

## Key Points:

- **Safe for any stream**: Works with `FileStream`, `MemoryStream`, etc.
- **Position preservation**: Resets the stream position after hashing if it's seekable.
- **Hex output**: Returns lowercase hexadecimal string (common for MD5).
- **Error handling**: Throws clear exceptions for null or unreadable streams.

If you want, I can also give you a **fully async version** so you can compute the MD5 without blocking the main thread — useful for large files or network streams.

Do you want me to provide that?