

Forum:[Hack Forums](#)

Thread: [Pipe server,communication between different processes using pipes](#)

itcorp - April 4, 2024, 11:01 am

I was adding extension handling to my application, so I wrote this piece of code to allow inter-process communication (IPC) between different applications or instances of the same application allowing dynamic interaction and passing of data.I hope you will benefit from it

Code public class NamedPipeServer : IDisposable

```
{
    private string pipeName;
    private NamedPipeServerStream pipeServer;
    private CancellationTokenSource cancellationTokenSource;
    private bool isDisposed;

    public event EventHandler<string[]> ArgumentsReceived;

    public NamedPipeServer(string pipeName)
    {
        if (string.IsNullOrEmpty(pipeName))
            throw new ArgumentException("Pipe name cannot be null or empty.");

        this.pipeName = pipeName;
    }

    public async Task Start()
    {
        if (pipeServer != null && pipeServer.IsConnected)
            throw new InvalidOperationException("The server is already running.");

        cancellationTokenSource = new CancellationTokenSource();
        pipeServer = new NamedPipeServerStream(pipeName, PipeDirection.InOut,
NamedPipeServerStream.MaxAllowedServerInstances, PipeTransmissionMode.Byte,
PipeOptions.Asynchronous);
        await pipeServer.WaitForConnectionAsync(cancellationTokenSource.Token);
        _ = Task.Run(() => HandleIncomingMessages(cancellationTokenSource.Token));
    }

    public void Stop()
    {
        cancellationTokenSource?.Cancel();
        pipeServer?.Dispose();
    }

    private async Task HandleIncomingMessages(CancellationToken cancellationToken)
    {
        using (StreamReader reader = new StreamReader(pipeServer))
        {
            while (!cancellationToken.IsCancellationRequested)
```

```

    {
        try
        {
            string data = await reader.ReadLineAsync();
            if (!string.IsNullOrEmpty(data))
            {
                string[] arguments = data.Split(new char[] { '\n' },
StringSplitOptions.RemoveEmptyEntries);
                ArgumentsReceived?.Invoke(this, arguments);
            }
        }
        catch (IOException ex)
        {
            Console.WriteLine($"Pipe server error: {ex.Message}");
            break;
        }
        catch (Exception ex)
        {
            Console.WriteLine($"Error: {ex.Message}");
        }
    }
}

public void Dispose()
{
    Dispose(true);
    GC.SuppressFinalize(this);
}
protected virtual void Dispose(bool disposing)
{
    if (!IsDisposed)
    {
        if (disposing)
        {
            Stop();
            cancellationTokenSource?.Dispose();
        }

        IsDisposed = true;
    }
}
}
}

```

If you have suggestions for improving the code I'm open minded.
I wish you all the best.

PulseLink - April 4, 2024, 4:42 pm

the only thing you could really add to that IMO is to add specific functions for the arguments received if you are trying to create a c2c server.