Code Snippets

- Async Parallel
- AsyncHelper
- Authentik NPM Redirect with Websocket support
- Batching with LINQ
- Build project with SourceGenerator on linux
- Change file encoding with linux
- Docker Buildx Multi-Arch
- Handle Exceptions with Task.WhenAll()
- Left outer join
- List iteration using span
- Load Assembly Dynamically
- macOS Defaults
- Mount host volumes in WSL2
- Optional<T>
- RegEx in multiple files multi-line pattern

Async Parallel

```
static Task ParallelForEachAsync<T> (IEnumerable<T> source,
                                     int degreeofParallelization,
                                     Func<T, Task> body)
{
    async Task AwaitPartition(IEnumerator<T> partition)
    {
        using (partition)
        {
            while (partition.MoveNext())
            {
                await body(partition.Current);
            }
        }
    }

    return Task.WhenAll(Partitioner
                        .Create(source)
                        .GetPartitions(degreeofParallelization)
                        .AsParallel()
                        .Select(AwaitPartition));
}
```

Usage:

```
var tasks = Enumerable.Range(0, 10000).Select(_ => new Func<Task>(() =>
Worker(results))).ToList();

await ParallelForEachAsync(tasks, 1000, async func =>
{
    await func();
});
```

https://www.youtube.com/embed/lHuyl_WTpME

AsyncHelper

```
public static class AsyncHelper
    {
        private static readonly TaskFactory _myTaskFactory = new
TaskFactory(CancellationToken.None,

TaskCreationOptions.None, TaskContinuationOptions.None, TaskScheduler.Default);

        public static TResult RunSync<TResult>(Func<Task<TResult>> func)
        {
            var cultureUi = CultureInfo.CurrentUICulture;
            var culture = CultureInfo.CurrentCulture;
            return _myTaskFactory.StartNew(() =>
            {
                Thread.CurrentThread.CurrentCulture = culture;
                Thread.CurrentThread.CurrentUICulture = cultureUi;
                return func();
            }).Unwrap().GetAwaiter().GetResult();
        }

        public static void RunSync(Func<Task> func)
        {
            var cultureUi = CultureInfo.CurrentUICulture;
            var culture = CultureInfo.CurrentCulture;
            _myTaskFactory.StartNew(() =>
            {
                Thread.CurrentThread.CurrentCulture = culture;
                Thread.CurrentThread.CurrentUICulture = cultureUi;
                return func();
            }).Unwrap().GetAwaiter().GetResult();
        }
    }
```

https://github.com/aspnet/AspNetIdentity/blob/main/src/Microsoft.AspNet.Identity.Core/AsyncHelper.cs

Authentik NPM Redirect with Websocket support

```
# Increase buffer size for large headers
# This is needed only if you get 'upstream sent too big header while reading response
# header from upstream' error when trying to access an application protected by goauthentik
proxy_buffers 8 16k;
proxy_buffer_size 32k;

location / {
    # Put your proxy_pass to your application here
    proxy_pass          $forward_scheme://$server:$port;
    proxy_set_header Host $host;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Connection upgrade;
    proxy_set_header Accept-Encoding gzip;

    # authentik-specific config
    auth_request        /outpost.goauthentik.io/auth/nginx;
    error_page          401 = @goauthentik_proxy_signin;
    auth_request_set $auth_cookie $upstream_http_set_cookie;
    add_header Set-Cookie $auth_cookie;

    # translate headers from the outposts back to the actual upstream
    auth_request_set $authentik_username $upstream_http_x_authentik_username;
    auth_request_set $authentik_groups $upstream_http_x_authentik_groups;
    auth_request_set $authentik_email $upstream_http_x_authentik_email;
    auth_request_set $authentik_name $upstream_http_x_authentik_name;
    auth_request_set $authentik_uid $upstream_http_x_authentik_uid;

    proxy_set_header X-authentik-username $authentik_username;
    proxy_set_header X-authentik-groups $authentik_groups;
    proxy_set_header X-authentik-email $authentik_email;
```

```
    proxy_set_header X-authentik-name $authentik_name;
    proxy_set_header X-authentik-uid $authentik_uid;
}

# all requests to /outpost.goauthentik.io must be accessible without authentication
location /outpost.goauthentik.io {
    proxy_pass          https://authentik-server-1:9443/outpost.goauthentik.io;
    # ensure the host of this vserver matches your external URL you've configured
    # in authentik
    proxy_set_header    Host $host;
    proxy_set_header    X-Original-URL $scheme://$http_host$request_uri;
    add_header          Set-Cookie $auth_cookie;
    auth_request_set    $auth_cookie $upstream_http_set_cookie;

    # required for POST requests to work
    proxy_pass_request_body off;
    proxy_set_header Content-Length "";
}

# Special location for when the /auth endpoint returns a 401,
# redirect to the /start URL which initiates SSO
location @goauthentik_proxy_signin {
    internal;
    add_header Set-Cookie $auth_cookie;
    return 302 /outpost.goauthentik.io/start?rd=$request_uri;
    # For domain level, use the below error_page to redirect to your authentik server with the
full redirect path
    # return 302
https://auth.ultimatecloud.tk/outpost.goauthentik.io/start?rd=$scheme://$http_host$request_uri
;
}
```

Batching with LINQ

```
static IEnumerable<IEnumerable<T>> Batch<T>(this IEnumerable<T> source, int batchSize)
{
    return source.Select((item, inx) => new { item, inx })
    □□□ .GroupBy(x => x.inx / batchSize)
□□□□ .Select(g => g.Select(x => x.item));
}
```

https://stackoverflow.com/questions/13731796/create-batches-in-linq

Build project with SourceGenerator on linux

Add Microsoft.Net.Compilers.Toolset package to the project that references the source generator.

Change file encoding with linux

To get current encoding:

```
file -i {filename}
```

To convert it:

```
iconv -f {sourceEncoding} -t {targetEncoding} {filename}
```

Example:

```
iconv -f ISO_8859-1 -t UTF-8 filename.txt
```

Docker Buildx Multi-Arch

```
sudo docker buildx create --use --name multi-arch-builder
```

Usage:

```
sudo docker buildx build --platform=linux/arm64 -t tag:latest -f ./DockerTest/Dockerfile --load --no-cache .
```

DockerFile:

```
#See https://aka.ms/customizecontainer to learn how to customize your debug container and how Visual Studio uses this Dockerfile to build your images for faster debugging.

FROM mcr.microsoft.com/dotnet/aspnet:8.0-alpine AS base
USER app
WORKDIR /app
EXPOSE 8080
EXPOSE 8081

FROM --platform=$BUILDPLATFORM mcr.microsoft.com/dotnet/sdk:8.0-alpine AS build
ARG BUILDPLATFORM
ARG TARGETARCH
ARG BUILD_CONFIGURATION=Release
RUN echo "Target: $TARGETARCH"
RUN echo "Build: $BUILDPLATFORM"
WORKDIR /src
COPY ["DockerTest/DockerTest.csproj", "DockerTest/"]
RUN dotnet restore "./DockerTest/DockerTest.csproj" -a $TARGETARCH
COPY . .
WORKDIR "/src/DockerTest"
RUN dotnet build "./DockerTest.csproj" -c $BUILD_CONFIGURATION -o /app/build -a $TARGETARCH

FROM build AS publish
ARG TARGETARCH
ARG BUILD_CONFIGURATION=Release
RUN dotnet publish "./DockerTest.csproj" -c $BUILD_CONFIGURATION -o /app/publish
```

```
/p:UseAppHost=false -a $TARGETARCH

FROM base AS final
WORKDIR /app
COPY --from=publish /app/publish .
ENTRYPOINT ["dotnet", "DockerTest.dll"]
```

https://stackoverflow.com/questions/70757791/build-linux-arm64-docker-image-on-linux-amd64-host

https://ruanbekker.medium.com/how-to-create-arm-based-container-images-with-buildx-fe917d186824

Handle Exceptions with Task.WhenAll()

```
var tasks = Enumerable.Range(0, 10).Select(async _ => await Task.Delay(1000));
var whenAll = Task.WhenAll(tasks);
try
{
  await whenAll;
}
catch {}
if (whenAll.Exception is AggregateException ex)
{
  //Handle Exception
}
```

https://www.youtube.com/watch?v=s_NrqRI7Gnc&t=321s

https://www.youtube.com/embed/s_NrqRI7Gnc?si=-zTzIp8LCLxUI7s0start=383

Left outer join

```
from leftTable in context.leftTable
join rightTable in context.rightTable on leftTable.FKRightTable equals rightTable.Id into
rightTableJoin
from joinedRightTable in rightTableJoin.DefaultIfEmpty()
select new { LeftData = leftTable.data, RightData = joinedRightTable?.data }
```

List iteration using span

```
var list = new List<int>();
foreach(var item in CollectionsMarshal.AsSpan(list))
{

}
```

Only usable if the collection is not changed during iteration (read only)

https://www.youtube.com/embed/jUZ3VKFyB-A

Load Assembly Dynamically

In order to Copy NuGet dependecies to the output folder add this to classlibrary.csproj:

```
<CopyLocalLockFileAssemblies>true</CopyLocalLockFileAssemblies>
```

Set the Output directory with

```
<BaseOutputPath>$(SolutionDir)AppName\bin</BaseOutputPath>
```

To load Assembly the best approach is to use

```
Assembly module = Assembly.LoadFrom(filePath);
```

To check if type A can be assigned to a variable with type B:

```
B.IsAssignableFrom(A)
```

macOS Defaults

Auto-Resize Columns in column view

```
defaults write com.apple.finder _FXEnableColumnAutoSizing -bool YES && killall Finder
```

Scroll to Exposé app

```
defaults write com.apple.dock "scroll-to-open" -bool "true" && killall Dock
```

Make Dock icons of hidden applications translucent

```
defaults write com.apple.dock showhidden -bool true && killall Dock
```

https://raw.githubusercontent.com/mathiasbynens/dotfiles/refs/heads/main/.macos

https://macos-defaults.com

Mount host volumes in WSL2

Add the following to `/etc/wsl.conf`

```
[automount]
options = "metadata"
```

Optional<T>

See Attachments

RegEx in multiple files multi-line pattern

```
pcregrep -h -r --include=".*\.cs" -M "LogEvent\([\s\S]*?\)\;" .
```

- -h: Exclude filenames
- -r: recursive
- --include: include file names that matches with the pattern
- -M: multi-line mode
- PATTERN: match all occurrence even if it is multi-line

Example: await LogEvent($"This is a test message that has embedded {property}");

```
pcregrep -h -r --include=".*\.cs" -o1 -M "LogEvent\(([\s\S]*?)\)\;" .
```

- -o: shows only match, not the entire line. -o{n} shows only the {n}th group
- Matches only: $"This is a test message that has embedded {property}"