using System;

using System.Threading;

namespace Viscont.Core.Service.ImageDataTransmission.Data;

public sealed class LifeTimeImage : IDisposable

{

#region Data

private readonly Guid ImageId;

private readonly Func<Guid, bool> \_imageAction;

private readonly Timer \_timer;

/// <summary> В секундах </summary>

public static int TimeTick = 10;

#endregion

#region Prop

public ImageEntry ImageMetadataEntry { get; }

#endregion

#region .ctor

public LifeTimeImage(

Guid imageId,

ImageEntry imageMetadataEntry,

Func<Guid, bool> imageAction)

{

ImageMetadataEntry = imageMetadataEntry;

ImageId = imageId;

\_imageAction = imageAction ?? throw new ArgumentNullException(nameof(imageAction));

\_timer = new Timer(OnTimerTick, null, Timeout.Infinite, Timeout.Infinite);

}

public void Start()

{

\_timer.Change(TimeSpan.FromSeconds(TimeTick), Timeout.InfiniteTimeSpan);

}

public void Dispose()

{

\_timer.Dispose();

ImageMetadataEntry.MemoryMappedFile.Dispose();

}

#endregion

#region Methods

private void OnTimerTick(object obj)

=> \_imageAction(ImageId);

#endregion

}

using System.IO.MemoryMappedFiles;

using Viscont.Core.Framework.ImageDataTransmission;

namespace Viscont.Core.Service.ImageDataTransmission.Data;

public sealed record class ImageEntry(

ImageMetadata ImageMetadata,

MemoryMappedFile MemoryMappedFile);