WaveFileReader | **WaveFileWriter** | **Mp3FileReader**

IEnumerable<IAudioStream>

+ IEnumerator<IAudioStream>
GetEnumerator(): IEnumerator<IAudioStream>
- IEnumerable.GetEnumerator(): IEnumerator

ExtensionsIAudioStream

+ TrimWavFile(this IAudioStream stream, string outPath, TimeSpan cutFromStart, TimeSpan cutFromEnd)
- TrimWavFile(WaveFileReader reader, WaveFileWriter writer, int startPos, int endPos)
+ Split(this IAudioStream stream, TimeSpan[] marks): IAudioCollection
+ Mp3ToWav(this IAudioStream stream, string outputFile)

IAudioCollection

+ Collection: List<IAudioStream>
+ FileNames: string[]
+ Add(IAudioStream item);
+ Remove(IAudioStream item);
+ Play();
+ Stop();
+ Pause();
+ SortByNames();

IPlayList

+ PlayListName: string
+ Current: IAudioStream
+ Collection: List<IAudioStream>
+ FileNames: string[]
+ Add(IAudioStream item);
+ Remove(IAudioStream item);
+ Play();
+ Stop();
+ Pause();
+ Reverse();
+ SortByNames();
+ SortByDurations();
+ MoveCurrentTo(int index);

DataManager

+ OpenAudiofile(string fileName): AudioStream
+ SavePlaylist(string playListName): List<string>
+ OpenPlaylist(string playListName): PlayList
+ SaveCollection(AudioCollection collection): List<IAudioStream>

IAudioStream

+ Stream: AudioFileReader
+ Format: WaveFormat (зачем это мне??)
+ FileName: string
+ Type: string
+ Volume: float
+ CurrentTime: TimeSpan
+ TotalTime: TimeSpan
+ CanRead: bool
+ Play();
+ Stop();
+ Pause();
+ Dispose();
+ SetPlayMark(TimeSpan playMark);

AudioCollection

- stream; IAudioStream
- outputDevice: WaveOutEvent
- collection: List<IAudioStream>
- fileNames: string[]
+ Collection: List<IAudioStream>
+ FileNames: string[]

+ Add(IAudioStream item);
+ Remove(IAudioStream item);
+ Play();
+ Stop();
+ Pause();
+ SortByNames();
+ IEnumerator<IAudioStream> GetEnumerator(): IEnumerator<IAudioStream>
- IEnumerable.GetEnumerator(): IEnumerator

PlayList

- outputDevice: WaveOutEvent
- collection: List<IAudioStream>
+ PlayListName: string
+ Current: IAudioStream
+ Collection: List<IAudioStream>
+ FileNames: string[]

+ Add(IAudioStream item);
+ Remove(IAudioStream item);
+ Play();
+ Stop();
+ Pause();
+ SortByNames();
+ SortByDurations();
+ MoveCurrentTo(int index);
+ IEnumerator<IAudioStream> GetEnumerator(): IEnumerator<IAudioStream>
- IEnumerable.GetEnumerator(): IEnumerator

PlayListProcessor

+ GetNewPlayList(string playListName): IPLayList
+ OpenPlayList(string playListName): IPLayList
+ SavePlayList(string playListName): List<string>
+ AddAudioInPlayList(string fileName, IPlayList list): IPlayList
+ RemoveAudioInPlayList(IAudioStream item, IPlayList list): IPlayList
+ SortPlayListByName(IPlayList list): IPlayList
+ SortPlayListByDuration(IPlayList list): IPlayList

AudioStream

- outputDevice: WaveOutEvent
+ Stream: AudioFileReader
+ Format: WaveFormat
+ FileName: string
+ Type: string
+ Volume: float
+ CurrentTime: TimeSpan
+ TotalTime: TimeSpan
+ CanRead: bool

+ Play();
+ Stop();
+ Pause();
+ Dispose();
+ SetPlayMark(TimeSpan playMark);
+ CompareTo();

CustomWaveViewer

+ PenColor: Color
+ PenWidth: float
+ WaveStream: AudioStream
+ SamplesPerPixel: int
+ StartPosition: long
- System.ComponentModel.Container components
- waveStream: AudioStream;
- samplesPerPixel: int
- startPosition: long
- bytesPerSample: int

+ FitToScreen();
+ Zoom(int leftSample, int rightSample);
protected override OnMouseDown(MouseEventArgs e);
protected override OnMouseMove(MouseEventArgs e);
protected override OnMouseUp(MouseEventArgs e);
protected override OnResize(EventArgs e);
- DrawVerticalLine(int x);
protected override Dispose(bool disposing);
 protected override OnPaint(PaintEventArgs e);

AudioFileReader | **WaveOutEven**