Urządzenia peryferyjne laboratorium	
Temat:	Karta muzyczna
Data zajęć:	03.11.2021
Grupa zajęciowa:	TN środa 11:15
Członkowie zespołu:	Aleksandra Ciężka 252701 Mateusz Chalik 252735
Grupa	C
Prowadzący:	dr inż. Tomasz Walkowiak

1. Wstęp

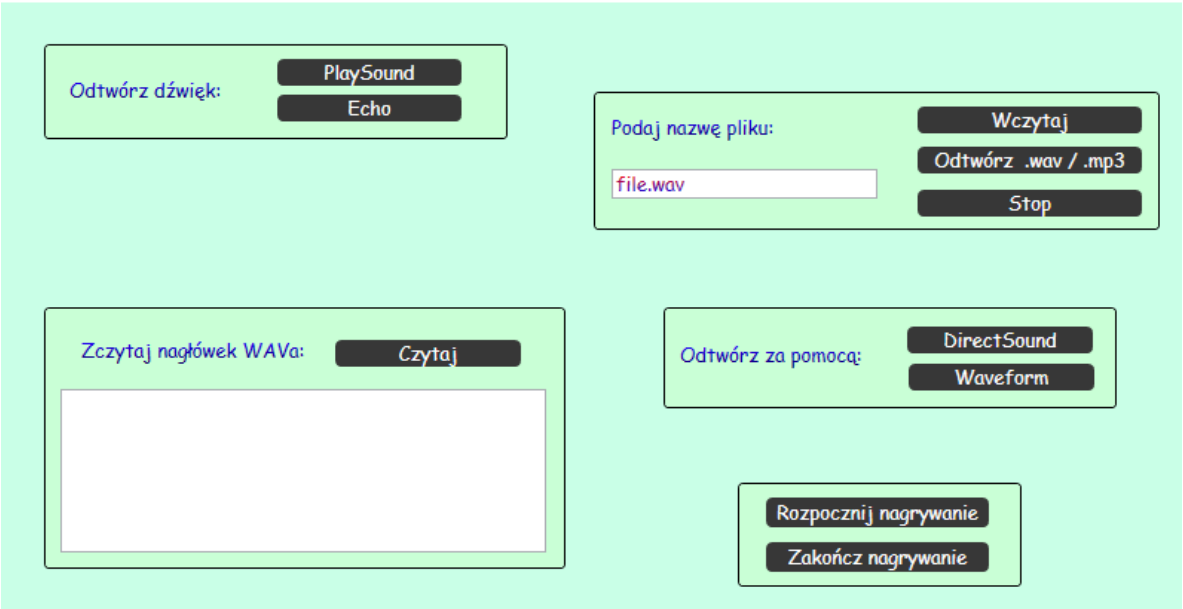
Celem ćwiczenia było zapoznanie się z obsługą karty muzycznej oraz formatem nagłówka plików WAV.

2. Realizacja zadania

Zadaniem przeznaczonym do zrealizowania było napisanie aplikacji służącej do odtwarzania plików WAV oraz umożlwiającej wyświetlenie ich nagłówków.

2.1 Wygląd okna aplikacji

Przy realizacji zadania należało stworzyć aplikacje z prostym interfejsem graficznym użytkownika, umożliwiającym dostęp do funkcji zaimplementowanych w programie. Wygląd okna aplikacji został przedstawiony na rysunku nr 1.



Rysunek 1 Wgląd graficznego interfejsu użytkownika

2.1 Implementacja funkcji aplikacji

Podczas realizacji zadania zaimplementowano funkcje umożliwiające odtwarzanie plików typu WAV oraz mp3. Ponadto aplikacja umożliwi zatrzymanie odtwarzania dźwięku, odtwarzanie za pomocą DirectSound oraz zczytanie nagłówka pliku WAV, a następnie wyświetlenie jego poszczególnych wartości. Dodatkowo aplikacja umożliwia nagranie własnego dźwięku poprzez zczytanie danych z mikrofonu i ich zapis do pliku.

1. Odtwarzanie plików .wav i .mp3

```
public static void Play()
    {

      string Ext = Path.GetExtension(SoundCardHandler.FilePath);

      if (Ext == ".wav")
      {
        Player = new SoundPlayer(FilePath);
        Player.Play();
      }

      if (Ext == ".mp3")
      {
        Wplayer = new WindowsMediaPlayer();
        Wplayer.URL = FilePath;
        Wplayer.controls.play();
      }
    }
```

2.Zatrzymanie odtwarzania

```
public static void PlayDirectSound()
    {
      wave = new NAudio.Wave.WaveFileReader(FilePath);
      output = new NAudio.Wave.DirectSoundOut();
      output.Init(new NAudio.Wave.WaveChannel32(wave));
      output.Play();
    }
```

3. Odtwarzanie pliku za pomocą DirectSound

```
public static void Stop()
    {
      Player?.Stop();
      Wplayer?.controls.stop();
      audioOutput?.Stop();
      output?.Stop();
    }
```

4. Wyświetlanie nagłówka pliku typu WAV

```
 public static string GetHeader()
    {
      var header = new WAVheader();

      using (var fileStream = new FileStream(FilePath, FileMode.Open,
FileAccess.Read))
      using (var binaryReader = new BinaryReader(fileStream))
      {
        try
        {
          //pobiera kolejno dane pliku .wav
          header.riffID = binaryReader.ReadBytes(4);
          header.size = binaryReader.ReadUInt32();
          header.wavID = binaryReader.ReadBytes(4);
          header.fmtID = binaryReader.ReadBytes(4);
          header.fmtSize = binaryReader.ReadUInt32();
          header.format = binaryReader.ReadUInt16();
          header.channels = binaryReader.ReadUInt16();
          header.sampleRate = binaryReader.ReadUInt32();
          header.bytePerSec = binaryReader.ReadUInt32();
          header.blockSize = binaryReader.ReadUInt16();
          header.bit = binaryReader.ReadUInt16();
          header.dataID = binaryReader.ReadBytes(4);
          header.dataSize = binaryReader.ReadUInt32();
        }
        finally
        {
          binaryReader.Close();
          fileStream.Close();
        }
      }
      return header.ToString();
    }
```

5. Nagrywanie dźwięku

```
// pomocnicza funkcja do nagrywania dźwięku
      private static void recordInstance_DataAvailable(object sender,
NAudio.Wave.WaveInEventArgs e)
      {
         if (recordWriter == null) return;
         recordWriter.Write(e.Buffer, 0, e.BytesRecorded);
         recordWriter.Flush();
      }


      // pomocnicza funkcja do nagrywania dźwięku
      private static void recordInstance_RecordingStopped(object sender,
NAudio.Wave.StoppedEventArgs e)
      {
         recordWriter.Dispose();
         recordWriter = null;
         recordInstance.Dispose();
      }


      // początek nagrywania dźwięku
      public static void startRecord()
      {
         recordInstance = new NAudio.Wave.WaveIn();
         recordInstance.WaveFormat = new NAudio.Wave.WaveFormat(48000, 1);

         recordInstance.DataAvailable += new
EventHandler<NAudio.Wave.WaveInEventArgs>(recordInstance_DataAvailable);
         recordInstance.RecordingStopped += new
EventHandler<NAudio.Wave.StoppedEventArgs>(recordInstance_RecordingStopp
ed);

         recordWriter = new NAudio.Wave.WaveFileWriter(recordFile,
recordInstance.WaveFormat);
         recordInstance.StartRecording();
      }


      // koniec nagrywania dźwięku
      public static void stopRecord()
      {
         recordInstance.StopRecording();
      }
```

6. Nałożenie efektu echo

Nie udało się zaimplementować funkcji nakładającej efekt echo.

3. Wnioski

Przy korzystaniu z sytemu Windows możliwe jest korzystanie z różnych sposobów odtwarzania dźwięku. Przy obsługę kary muzycznej pomocne są gotowe funkcje dostępne w bibliotekach.

Aplikacje służącą do obsługi karty muzycznej napisano w języku C#. W aplikacji udało się zaimplementować funkcje służące do odtwarzania i nagrywania dźwięku oraz funkcję umożliwiającą odczytanie i wyświetlenie szczegółów nagłówka plików WAV. Nie udało się zaimplementowanie funkcji służącej do nałożenia efektu echo. Podjęto próbę realizacji tej funkcji przy pomocy biblioteki DirectX, jednak napotkano problem mogący wynikać z nieprawidłowej instalacji tego komponentu.

Bibliografia

[1] Strona internetowa: https://docs.fileformat.com/audio/wav/
[2] Strona internetowa: https://docs.microsoft.com/en-gb/windows/apps/
[3] Strona internetowa: http://slimdx.org
[4] Strona internetowa: https://docs.microsoft.com/en-us/previous-versions/windows/desktop/bb318662(v=vs.85)