# 1.Zagadnienia teoretyczne

# 2.Kod programu

using System;

using System.Collections.Generic;

using System.IO;

using System.Media;

using System.Text;

using System.Windows.Forms;

namespace Karta\_dzwiekowa

{

public partial class Form1 : Form

{

private string filePath = "";

// private SoundPlayer player;

private bool wasPlayed, wasRecored = false;

private string fileRecordPath = "";

NAudio.Wave.WaveIn sourceStream = null;

NAudio.Wave.DirectSoundOut soundOut = null;

NAudio.Wave.WaveFileWriter waveFileWriter = null;

public Form1()

{

InitializeComponent();

}

private void buttonFile\_Click(object sender, EventArgs e) // Wybranie pliku do odtworzenia

{

OpenFileDialog file = new OpenFileDialog();

file.Filter = "Audio files (.wav)|\*.wav"; // Akceptowalne formaty plików to WAV

if (file.ShowDialog() == DialogResult.OK)

{

filePath = file.FileName;

labelFilePath.Text = $"Wybrany plik: {filePath}";

FillListBox();

}

}

private void FillListBox() // funkcja zczytująca nagłowek pliku WAV

{

if (!string.IsNullOrEmpty(filePath)) // jesli wybrano plik

{

FileStream fileStream = new FileStream(filePath, FileMode.Open, FileAccess.Read);

BinaryReader reader = new BinaryReader(fileStream);

// odczytanie nagłówka pliku Wave

byte[] wave = reader.ReadBytes(24);

fileStream.Position = 0;

int chunkID = reader.ReadInt32();

int fileSize = reader.ReadInt32();

var fileFormat = Encoding.Default.GetString(wave);

string format = fileFormat.Substring(8, 4);

string subchunk1ID = fileFormat.Substring(12, 8);

int subchunk1Size = reader.ReadInt32();

reader.Close();

// Przypisanie odczytanych wartości do zmiennych tyou string

string chunkIDStr = $"Chunk ID: {chunkID}";

string fileSizeStr = $"Chunk size: {fileSize}";

string fileFormatStr = $"Format: {format}";

string subchunk1IDStr = $"Subchunk ID: {subchunk1ID}";

string subchunk1SizeStr = $"Subchunk Size ID: {subchunk1Size}";

// Wyświetlenie danych w oknie

listBoxFileInfo.Items.Clear();

listBoxFileInfo.Items.AddRange(new string[]

{

"\tNagłówek pliku:",chunkIDStr, fileSizeStr, fileFormatStr,"\tOpis struktury audio:",subchunk1IDStr

,subchunk1SizeStr

});

}

}

// obsługa przycisku play służącego do odtworzenia wcześniej wybranego pliku

private void buttonPlay\_Click(object sender, EventArgs e)

{

if (filePath == String.Empty)

MessageBox.Show("Wybierz plik!"); // jesli sciezka do pliku jest pusta wyswietl komunikat o braku sciezki

else

{

SoundPlayer simpleSound = new SoundPlayer(@filePath);

if (!wasPlayed) // jesli nic nie jest odtwarzane

{

buttonPlay.Text = "STOP"; // zmien napis przycisku

wasPlayed = !wasPlayed;

simpleSound.Play(); // użycie funckji play z biblioteki SoundPlayer

}

else

{

// jeśli coś jest odtwarzane to sytuacja analogiczna tylko z zatrzymaniem odtwarzania

buttonPlay.Text = "PLAY";

wasPlayed = !wasPlayed;

simpleSound.Stop();

}

}

}

// obsługa przycisku wyszukiwania mikrofonu

private void buttonFindDevice\_Click(object sender, EventArgs e)

{

label1.Visible = true;

List<NAudio.Wave.WaveInCapabilities> sources = new List<NAudio.Wave.WaveInCapabilities>();

for (int i = 0; i < NAudio.Wave.WaveIn.DeviceCount; i++)

sources.Add(NAudio.Wave.WaveIn.GetCapabilities(i));

listBoxDevices.Items.Clear(); // jesli już coś było w oknie to je wyczyść

int counter = 0;

foreach (var source in sources)

{

string item = source.ProductName;

listBoxDevices.Items.Add("Mikrofon "+ counter + "->" + item); // dodanie i wyswietlenie do okna znalezionych mikrofonów

counter++;

}

}

private void sourceStream\_DataAvailable(object sender, NAudio.Wave.WaveInEventArgs e)

{

if (waveFileWriter == null)

return;

waveFileWriter.Write(e.Buffer, 0, e.BytesRecorded);

waveFileWriter.Flush();

}

// obsluga przycisku nagraj

private void buttonRecord\_Click(object sender, EventArgs e)

{

// jesli cos nie jest nagrywane

if (wasRecored == false)

{

//jesli nie znaleziono urządzeń (mikrofonu)

if (listBoxDevices.SelectedItems.Count == 0)

return;

// jesli nie została wybrana scieżka do zapisu nagrania wyświetl komunikat

if (fileRecordPath == "")

{

MessageBox.Show("Wybierz miejsce w którym chcesz zapisać plik!");

}

else

{

// nagrywanie do wczesniej wybranego pliku

int deviceNumber = listBoxDevices.SelectedIndex;

sourceStream = new NAudio.Wave.WaveIn();

sourceStream.DeviceNumber = deviceNumber;

sourceStream.WaveFormat = new NAudio.Wave.WaveFormat(44100, NAudio.Wave.WaveIn.GetCapabilities(deviceNumber).Channels); // nadanie czestotliwosci nagrywania, i standardu mono czy stereo wynikającego z urządzenia

sourceStream.DataAvailable += new EventHandler<NAudio.Wave.WaveInEventArgs>(sourceStream\_DataAvailable);

waveFileWriter = new NAudio.Wave.WaveFileWriter(fileRecordPath, sourceStream.WaveFormat);

sourceStream.StartRecording();

buttonRecord.Text = "Nagrywanie...";

wasRecored = true;

}

}

else if (wasRecored == true) // jesli jest już coś nagrywane to zatrzymaj obecne nagrywanie i zmien tekst na przyciskach

{

if (soundOut != null)

{

soundOut.Stop();

soundOut.Dispose();

soundOut = null;

buttonRecord.Text = "Nagraj";

}

if (sourceStream != null)

{

sourceStream.StopRecording();

sourceStream.Dispose();

sourceStream = null;

buttonRecord.Text = "Nagraj";

}

if (waveFileWriter != null)

{

waveFileWriter.Dispose();

waveFileWriter = null;

buttonRecord.Text = "Nagraj";

}

labelRecording.Text = "";

}

}

// funkcja wyboru scieżki do zapisywania nagrania

private void button1\_Click\_1(object sender, EventArgs e)

{

SaveFileDialog save = new SaveFileDialog();

save.Filter = "Wave File (\*.wav)|\*.wav;";

if (save.ShowDialog() != DialogResult.OK) return ;

else

fileRecordPath = save.FileName;

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void groupBox1\_Enter(object sender, EventArgs e)

{

}

}

}