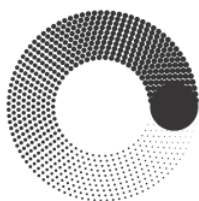


**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ**



МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

**Факультет информационных технологий
Кафедра Информатики и информационных технологий**

**направление подготовки 09.03.02 «Информационные системы и
технологии»,**

ЛАБОРАТОРНАЯ РАБОТА №6

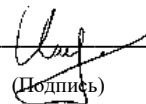
Дисциплина: Технология прикладного программирования

Выполнил: студент группы 231-338

Шаура И. М.

Дата, подпись 11.02.2024

(Дата)


(Подпись)

Проверил: ст. преп. Калмыков Е. А.

(Оценка)

Дата, подпись _____

(Дата)

(Подпись)

Замечания:

Москва

2024

Задание:

Разработать программу "аудиоплеер" или программу "видеоплеер" на выбор.

Функционал:

1. Воспроизведение аудио/видео.
2. Загрузка нескольких файлов.
3. Сохранение и открытие плейлиста

Код XAML

```
<Window x:Class="_6.AudioPlayer.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
        xmlns:mc="http://schemas.openxmlformats.org/markup-compatibility/2006"
        xmlns:local="clr-namespace:_6.AudioPlayer"
        mc:Ignorable="d"
        Title="AudioPlayer" Height="300" Width="400" ResizeMode="NoResize">
    <StackPanel Margin="10">
        <Label Name="songTitle" Content="Ничего не играет..." HorizontalContentAlignment="Center"
            Margin="2"/>
        <Label Name="textStatus" Content="" HorizontalContentAlignment="Center" Margin="2" />
        <Slider Name="Slider" Width="300" ValueChanged="Slider_ValueChanged" Value="{Binding
            CurrentPosition}" Minimum="0" Maximum="{Binding CurrentMaximumPosition}"/>
        <WrapPanel HorizontalAlignment="Center">
            <Button Content="⊕" Name="addToListButton" Background="Transparent" BorderThickness="0"
                FontSize="10" Click="addToListButton_Click"/>
            <Button Content="⏮" Name="playPauseButton" Background="Transparent" BorderThickness="0"
                FontSize="20" Click="playPauseButton_Click"/>
            <Button Content="⏭" Name="skipButton" Background="Transparent" BorderThickness="0"
                FontSize="20" Click="skipButton_Click"/>
        </WrapPanel>
        <ListBox Name="PlaylistLB" Margin="10" ItemsSource="{Binding Playlist}" Height="80">
            <ListBox.ItemTemplate>
                <DataTemplate DataType="Track">
                    <StackPanel Orientation="Horizontal" MouseEnter="PlaylistItem_MouseEnter">
                        <TextBlock Text="{Binding TrackName}" />
                        <Label/>
                        <Button Content="▶" Click="LBPlayButton_Click" Background="Transparent"
                            BorderThickness="0"/>
                        <Button Content="✕" Click="LBRemoveButton_Click" Background="Transparent"
                            BorderThickness="0"/>
                    </StackPanel>
                </DataTemplate>
            </ListBox.ItemTemplate>
        </ListBox>
        <WrapPanel HorizontalAlignment="Center">
            <Button Content="Сохранить плейлист" Click="SavePlaylistButton_Click"/>
            <Label/>
            <Button Content="Открыть плейлист" Click="OpenPlaylistButton_Click"/>
        </WrapPanel>
    </StackPanel>
</Window>
```

Код C#

```
using Microsoft.Win32;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.IO;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Media;
using System.Windows.Threading;

namespace _6.AudioPlayer
{
    public class Track
    {
        public required string TrackName { get; set; }
        public required string TrackPath { get; set; }
    }

    public partial class MainWindow : Window, INotifyPropertyChanged
    {
        public event PropertyChangedEventHandler? PropertyChanged;

        private void OnPropertyChanged(string propertyName)
        {
            PropertyChanged?.Invoke(this, new PropertyChangedEventArgs(propertyName));
        }

        private readonly MediaPlayer mediaPlayer = new();
        private bool _isPlaying = false;
        private int _playingIndex = 0;

        public DispatcherTimer Timer { get; set; }
        public ObservableCollection<Track> Playlist { get; set; } = [];
        public Track MouseOnTrack { get; set; }

        private double _currentPosition;

        public double CurrentPosition
        {
            get => _currentPosition;
            set
            {
                _currentPosition = value;
                OnPropertyChanged(nameof(CurrentPosition));
            }
        }

        private double _currentMaximumPosition;

        public double CurrentMaximumPosition
        {
            get => _currentMaximumPosition;
            set
            {
                _currentMaximumPosition = value;
                OnPropertyChanged(nameof(CurrentMaximumPosition));
            }
        }
    }
}
```

```

public MainWindow()
{
    InitializeComponent();
    DataContext = this;
    Timer = new()
    {
        Interval = TimeSpan.FromSeconds(1)
    };
    Timer.Tick += TimerTick;
    Timer.Start();
}

private void addToListButton_Click(object sender, RoutedEventArgs e)
{
    OpenFileDialog openFileDialog = new()
    {
        Title = "Выберите аудиофайл",
        Filter = "MP3 files (*.mp3)|*.mp3|WAV files (*.wav)|*.wav"
    };

    if (openFileDialog.ShowDialog() == true)
    {
        Playlist.Add(new() { TrackName = openFileDialog.SafeFileName, TrackPath =
openFileDialog.FileName });
        if (Playlist.Count > 1) return;
        mediaPlayer.Open(new Uri(openFileDialog.FileName));
    }
}

private void playPauseButton_Click(object sender, RoutedEventArgs e)
{
    if (_playingIndex == Playlist.Count - 1)
    {
        mediaPlayer.Close();
        _playingIndex = 0;
        mediaPlayer.Open(new Uri(Playlist[0].TrackPath));
        mediaPlayer.Play();
        _isPlaying = true;
        return;
    }
    if (_isPlaying)
    {
        mediaPlayer.Pause();
        _isPlaying = false;
    }
    else
    {
        mediaPlayer.Play();
        _isPlaying = true;
    }
}

private void skipButton_Click(object sender, RoutedEventArgs e)
{
    NextTrack();
}

private void LBPlayButton_Click(object sender, RoutedEventArgs e)
{
    _playingIndex = Playlist.IndexOf(Playlist.First(x => x == MouseOnTrack));
}

```

```

        mediaPlayer.Open(new Uri(MouseOnTrack.TrackPath));
        mediaPlayer.Play();
        _isPlaying = true;
    }

    private void LBRemoveButton_Click(object sender, RoutedEventArgs e)
    {
        mediaPlayer.Close();
        Playlist.Remove(MouseOnTrack);
        _isPlaying = false;
    }

    private void PlaylistItem_MouseEnter(object sender, System.Windows.Input.MouseEventArgs e)
    {
        MouseOnTrack = (Track)((StackPanel)sender).DataContext;
    }

    void TimerTick(object? sender, EventArgs e)
    {
        if (!mediaPlayer.NaturalDuration.HasTimeSpan) return;
        if (mediaPlayer.Source != null)
        {
            textStatus.Content = string.Format("{0} / {1}", mediaPlayer.Position.ToString(@"mm\:ss"),
mediaPlayer.NaturalDuration.TimeSpan.ToString(@"mm\:ss"));
            if (_playingIndex < Playlist.Count)
                songTitle.Content = Playlist[_playingIndex].TrackName;
            CurrentPosition = mediaPlayer.Position.TotalMilliseconds;
            CurrentMaximumPosition = mediaPlayer.NaturalDuration.TimeSpan.TotalMilliseconds;
        }
        if (CurrentPosition == mediaPlayer.NaturalDuration.TimeSpan.TotalMilliseconds)
        {
            NextTrack();
        }
    }

    private void Slider_ValueChanged(object sender, RoutedPropertyChangedEventArgs<double> e)
    {
        mediaPlayer.Position = TimeSpan.FromMilliseconds(Slider.Value);
    }

    private void NextTrack()
    {
        if (_playingIndex < Playlist.Count - 1)
        {
            _playingIndex++;
            mediaPlayer.Close();
            mediaPlayer.Open(new Uri(Playlist[_playingIndex].TrackPath));
            mediaPlayer.Play();
        }
    }

    private void SavePlaylistButton_Click(object sender, RoutedEventArgs e)
    {
        string playlist = "";
        foreach (var item in Playlist)
            playlist += item.TrackPath + "#" + item.TrackName + "\n";
        SaveFileDialog saveFileDialog = new()
        {
            Title = "Сохранить",
            Filter = "Text file (*.txt)|*.txt"
        };
        if (saveFileDialog.ShowDialog() == true)

```

```

        {
            File.WriteAllText(saveFileDialog.FileName, playlist);
            MessageBox.Show($"Плейлист сохранен по пути {saveFileDialog.FileName}");
        }
    }

    private void OpenPlaylistButton_Click(object sender, RoutedEventArgs e)
    {
        Playlist.Clear();
        OpenFileDialog openFileDialog = new()
        {
            Title = "Открыть",
            Filter = "Text file (*.txt)*.txt"
        };
        if (openFileDialog.ShowDialog() == true)
        {
            List<string> playlist =[.. File.ReadAllLines(openFileDialog.FileName)];
            foreach (var item in playlist)
            {
                var indexOfSeparator = item.IndexOf(item.First(x => x == '#'));
                var track = new Track
                {
                    TrackPath = item[..indexOfSeparator],
                    TrackName = item.Substring(indexOfSeparator + 1, item.Length - indexOfSeparator - 1)
                };
                Playlist.Add(track);
            }
            mediaPlayer.Open(new Uri(Playlist[0].TrackPath));
            MessageBox.Show($"Плейлист успешно импортирован!");
        }
    }
}

```

Скриншоты приложения:

