МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РЕСПУБЛИКИ БЕЛАРУСЬ

БЕЛОРУССКИЙ НАЦИОНАЛЬНЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

## Факультет информационных технологий и робототехники

Кафедра программного обеспечения информационных систем и технологий

**Отчет по лабораторной работе № 14**

по дисциплине: ”Разработка простейшего приложения в визуальной среде WPF”

на тему: ***”Разработка комплексного приложения в WPF”***

Вариант 14

Выполнил**:** студент группы 107012019 Кузьмич И.В.

Принял**:** Гурский Н.Н.

Минск 2020

***Цель лабораторной работы:*** Показать умение создания современного приложения в визуальной среде.

## Постановка задачи: Разработать современное приложение, содержащее расширенный список компонентов управления программой и представления данных в различных видах.

***Задание:***

Разработать приложение, поддерживающее различные способы управления вычислительным процессом с помощьюTPageControl, TTabSheet, TСhart***,*** стандартных диалоговых компонентов, ТPopupMenu***,*** TMainMenu, TToolBar и др. на примере лабораторной работы №8.

***Код программы:***

*Файл App.xaml.cs*

*﻿using System;*

*using System.Collections.Generic;*

*using System.Configuration;*

*using System.Data;*

*using System.Linq;*

*using System.Threading.Tasks;*

*using System.Windows;*

*namespace Audio\_Player*

*{*

*/// <summary>*

*/// Interaction logic for App.xaml*

*/// </summary>*

*public partial class App : Application*

*{*

*}*

*}*

*Файл Audio.cs*

*﻿using System;*

*using System.Collections.Generic;*

*using System.Linq;*

*using System.Text;*

*using System.Threading.Tasks;*

*namespace Audio\_Player*

*{*

*[Serializable]*

*public class Audio*

*{*

*public string Name { get; set; }*

*public string Title { get; set; }*

*public string DirectoryName { get; set; }*

*public string Singer { get; set; }*

*public string Album { get; set; }*

*public TimeSpan Duration { get; set; }*

*public Audio(string Name, string Title, string DirectoryName, string Singer, string Album, TimeSpan Duration)*

*{*

*this.Name = Name;*

*this.Title = Title;*

*this.DirectoryName = DirectoryName;*

*this.Singer = Singer;*

*this.Album = Album;*

*this.Duration = Duration;*

*if(String.IsNullOrEmpty(this.Album))*

*{*

*this.Album = "Неизвестный альбом";*

*}*

*if(String.IsNullOrEmpty(this.Singer))*

*{*

*this.Singer = "Неизвестный исполнитель";*

*}*

*}*

*}*

*}*

*Файл ChangePage.cs*

*﻿using System;*

*using System.Windows;*

*using System.Windows.Controls;*

*using System.Windows.Input;*

*using System.Windows.Media.Animation;*

*namespace Audio\_Player*

*{*

*public partial class MainWindow : Window*

*{*

*private void SetVisibleMain()*

*{*

*PListControl.Visibility = Visibility.Collapsed;*

*AddFolder.Visibility = Visibility.Collapsed;*

*PListInfo.Visibility = Visibility.Collapsed;*

*PlayGrid.Visibility = Visibility.Visible;*

*}*

*private void SetVisibleFolderAdder()*

*{*

*PListControl.Visibility = Visibility.Collapsed;*

*AddFolder.Visibility = Visibility.Visible;*

*PListInfo.Visibility = Visibility.Collapsed;*

*PlayGrid.Visibility = Visibility.Collapsed;*

*}*

*private void SetVisiblePlayListInfo()*

*{*

*PListControl.Visibility = Visibility.Collapsed;*

*AddFolder.Visibility = Visibility.Collapsed;*

*PListInfo.Visibility = Visibility.Visible;*

*PlayGrid.Visibility = Visibility.Collapsed;*

*}*

*}*

*}*

*Файл LoadAudio.cs*

*﻿using System;*

*using System.Collections.Generic;*

*using System.Linq;*

*using System.IO;*

*using System.Windows;*

*using TagLib;*

*using System.Text;*

*using System.Threading.Tasks;*

*namespace Audio\_Player*

*{*

*class LoadAudio*

*{*

*string[] Formats = new string[] { ".aif", ".m3u", ".m4a", ".mid", ".mp3", ".mpa", ".wav", ".wma" };*

*[Obsolete]*

*public List<Audio> GetMusic(string Path)*

*{*

*List<Audio> AudioPathList = new List<Audio>();*

*try*

*{*

*DirectoryInfo dir = new DirectoryInfo(Path);*

*FileInfo[] files = dir.GetFiles();*

*foreach (FileInfo f in files)*

*{*

*if (Formats.Contains(f.Extension))*

*{*

*TagLib.File FD = TagLib.File.Create(f.FullName);*

*AudioPathList.Add(new Audio(f.Name, FD.Tag.Title, f.DirectoryName, FD.Tag.FirstArtist, FD.Tag.Album, FD.Properties.Duration));*

*}*

*}*

*/\**

*foreach (DirectoryInfo d in dir.GetDirectories())*

*{*

*GetMusic(Path + @"\" + d.Name);*

*} \*/*

*return AudioPathList;*

*}*

*catch (Exception e)*

*{*

*MessageBox.Show(e.Message);*

*return AudioPathList;*

*}*

*}*

*}*

*}*

*Файл MainWindow.xaml.cs*

*﻿using System;*

*using System.Collections.Generic;*

*using System.IO;*

*using System.Linq;*

*using System.Windows;*

*using System.Windows.Controls;*

*using System.Windows.Input;*

*using System.Windows.Media.Animation;*

*using System.Windows.Media.Imaging;*

*namespace Audio\_Player*

*{*

*public partial class MainWindow : Window*

*{*

*public List<string> Paths = new List<string>();*

*public int CurrentIndex = 0;*

*public bool Playing = false;*

*private LoadAudio load\_audios = new LoadAudio();*

*public PlayList mainPL = new PlayList() { Name = "Main Playlist" };*

*public List<PlayList> playLists { get; set; } = new List<PlayList>();*

*public List<Audio> CurrentList;*

*private TimeSpan TotalTime;*

*public MainWindow()*

*{*

*InitializeComponent();*

*DeserializeData();*

*CurrentList = mainPL.AudioList;*

*}*

*private void ms\_MediaEnded(object sender, RoutedEventArgs e)*

*{*

*if (RepeatButt.IsChecked == true)*

*{*

*ms.Position = TimeSpan.Zero;*

*ms.Play();*

*}*

*else*

*{*

*if (CurrentIndex < CurrentList.Count - 1)*

*ChangeAudio(CurrentList[++CurrentIndex]);*

*}*

*}*

*private void Audio\_MouseLeftButtonDown(object sender, MouseButtonEventArgs e)*

*{*

*var audioContext = ((sender as Grid).DataContext as Audio); // получаем данные нажатого grid-а*

*ChangeAudio(audioContext);// воспроизводим*

*}*

*private void Window\_SizeChanged(object sender, SizeChangedEventArgs e)*

*{*

*if(this.ActualWidth < 850)*

*{*

*TitleSizer.Visibility = Visibility.Hidden;*

*AudioSlider.Visibility = Visibility.Hidden;*

*}*

*else*

*{*

*TitleSizer.Visibility = Visibility.Visible;*

*AudioSlider.Visibility = Visibility.Visible;*

*BotButtons.Children[3].Visibility = Visibility.Visible;*

*}*

*if(this.ActualWidth < 600)*

*{*

*BotButtons.Children[5].Visibility = Visibility.Collapsed;*

*BotButtons.Children[4].Visibility = Visibility.Collapsed;*

*}*

*else*

*{*

*BotButtons.Children[5].Visibility = Visibility.Visible;*

*BotButtons.Children[4].Visibility = Visibility.Visible;*

*}*

*}*

*private void Window\_Closed(object sender, EventArgs e)*

*{*

*MyDialogWindow dialog = new MyDialogWindow();*

*dialog.TX.Text = "Сохранить данные?";*

*dialog.Owner = this;*

*dialog.ShowDialog();*

*if (dialog.DialogResult.HasValue && dialog.DialogResult.Value)*

*{*

*SerializeData();*

*}*

*}*

*public BitmapImage LoadImage(string text, bool decode)*

*{*

*BitmapImage bitmap = new BitmapImage();*

*bitmap.BeginInit();*

*bitmap.UriSource = new Uri(text, UriKind.Absolute);*

*bitmap.CacheOption = BitmapCacheOption.OnLoad;*

*if (decode)*

*{*

*bitmap.DecodePixelWidth = 200;*

*bitmap.DecodePixelHeight = 200;*

*}*

*bitmap.EndInit();*

*return bitmap;*

*}*

*private void ChangeAudio(Audio audioContext)*

*{*

*ms.Source = new Uri(audioContext.DirectoryName + "\\" + audioContext.Name);*

*BottomPanel.Visibility = Visibility.Visible;*

*TopPlayGrid.DataContext = audioContext; // обновляем верхнюю панель*

*Playing = true;*

*ms.Play();*

*CurrentIndex = CurrentList.IndexOf(audioContext);*

*(PlayButton.Child as Image).Source = LoadImage(@"pack://application:,,,/Resources/Pause.png", false);*

*if(!System.IO.File.Exists(audioContext.DirectoryName + "\\" + audioContext.Name)) // проверка наличия файла перед воспроизведением*

*{*

*// если такой файл не существует - создается диалоговое окно*

*MyDialogWindow dialog = new MyDialogWindow();*

*dialog.TX.Text = "Этот файл был удален или перемещен... Удалить данные о нем?";*

*dialog.Owner = this;*

*dialog.ShowDialog();*

*if (dialog.DialogResult.HasValue && dialog.DialogResult.Value)*

*{*

*playLists = playLists.Select(x => { x.AudioList = x.AudioList.Where*

*(s => s.Name != audioContext.Name || s.DirectoryName != audioContext.DirectoryName).ToList(); x.GetTime(); return x; }).ToList(); // с списка плейлистов, создаем новый список, в котором не будет такого файла*

*mainPL.AudioList = mainPL.AudioList.Where(x => x.Name != audioContext.Name || x.DirectoryName != audioContext.DirectoryName).ToList(); // то же делаем и для главного плейлиста*

*CurrentList = CurrentList.Where(x => x.Name != audioContext.Name || x.DirectoryName != audioContext.DirectoryName).ToList(); // то же и для текущего списка воспроизведения*

*Play.ItemsSource = CurrentList; // обновляем главний список воспроизведения*

*if (CurrentList.Count > 0)*

*{*

*ChangeAudio(CurrentList[--CurrentIndex]); // воспроизводим следующую песню*

*}*

*return;*

*}*

*else*

*{*

*ChangeAudio(CurrentList[++CurrentIndex]); // воспроизводим следующую песню*

*return;*

*}*

*}*

*TagLib.File Ds = TagLib.File.Create(audioContext.DirectoryName + "\\" + audioContext.Name);*

*if (Ds.Tag.Pictures.Length > 0)*

*{*

*TagLib.IPicture pic = Ds.Tag.Pictures[0];*

*MemoryStream s = new MemoryStream(pic.Data.Data);*

*s.Seek(0, SeekOrigin.Begin);*

*BitmapImage bitmap = new BitmapImage();*

*bitmap.BeginInit();*

*bitmap.StreamSource = s;*

*bitmap.CacheOption = BitmapCacheOption.OnLoad;*

*bitmap.DecodePixelWidth = 200;*

*bitmap.DecodePixelHeight = 200;*

*bitmap.EndInit();*

*s.Close();*

*Img\_Audio.Source = bitmap;*

*GC.Collect();*

*}*

*else*

*{*

*Img\_Audio.Source = LoadImage(@"pack://application:,,,/Resources/NoImg.jpg", true);*

*}*

*BottomInfo.DataContext = audioContext;*

*}*

*private void PL\_Click(object sender, MouseButtonEventArgs e)*

*{*

*SetVisiblePlayListInfo();*

*Bounce(PListInfo, this.ActualWidth, this.ActualHeight);*

*PListInfo.DataContext = null;*

*PListInfo.DataContext = (sender as Border).DataContext;*

*}*

*public void Bounce(Grid elem, double mainWin\_ActW, double mainWin\_ActH)*

*{*

*ThicknessAnimation Anim = new ThicknessAnimation();*

*Anim.From = new Thickness(0, -elem.ActualHeight, 0, 0);*

*Anim.To = new Thickness(0);*

*elem.BeginAnimation(MarginProperty, Anim);*

*}*

*private void Grid\_MouseDown(object sender, MouseButtonEventArgs e)*

*{*

*DragMove();*

*}*

*}*

*}*

*Файл MyDialogWindow.xaml.cs*

*﻿using System;*

*using System.Collections.Generic;*

*using System.Linq;*

*using System.Text;*

*using System.Threading.Tasks;*

*using System.Windows;*

*using System.Windows.Controls;*

*using System.Windows.Data;*

*using System.Windows.Documents;*

*using System.Windows.Input;*

*using System.Windows.Media;*

*using System.Windows.Media.Imaging;*

*using System.Windows.Shapes;*

*namespace Audio\_Player*

*{*

*public partial class MyDialogWindow : Window*

*{*

*public MyDialogWindow()*

*{*

*InitializeComponent();*

*}*

*private void OkB\_Click(object sender, RoutedEventArgs e)*

*{*

*DialogResult = true;*

*this.Close();*

*}*

*private void CancB\_Click(object sender, RoutedEventArgs e)*

*{*

*DialogResult = false;*

*this.Close();*

*}*

*}*

*}*

*Файл PlayList.cs*

*﻿using System;*

*using System.Collections.Generic;*

*using System.Linq;*

*using System.Text;*

*using System.Threading.Tasks;*

*namespace Audio\_Player*

*{*

*[Serializable]*

*public class PlayList*

*{*

*public List<Audio> AudioList { get; set; } = new List<Audio>();*

*public string Name { get; set; }*

*public int AudioTime { get; set; }*

*public void GetTime()*

*{*

*AudioTime = 0;*

*foreach(Audio val in AudioList)*

*{*

*AudioTime += val.Duration.Minutes;*

*}*

*}*

*}*

*}*

*Файл SerializationData.cs*

*﻿using System.Collections.Generic;*

*using System.IO;*

*using System.Runtime.Serialization.Formatters.Binary;*

*using System.Windows;*

*namespace Audio\_Player*

*{*

*public partial class MainWindow : Window*

*{*

*private void DeserializeData()*

*{*

*//считываем данные*

*BinaryFormatter formatter = new BinaryFormatter();*

*using (FileStream fs = new FileStream("P\_S.dat", FileMode.OpenOrCreate))*

*{*

*if (fs.Length != 0)*

*Paths = (List<string>)formatter.Deserialize(fs);*

*}*

*ListPaths.ItemsSource = new List<string>(Paths);*

*using (FileStream fs = new FileStream("P\_List.dat", FileMode.OpenOrCreate))*

*{*

*if (fs.Length != 0)*

*mainPL.AudioList = (List<Audio>)formatter.Deserialize(fs);*

*}*

*//обновляем окна*

*ListPaths.ItemsSource = new List<string>(Paths);*

*Play.ItemsSource = new List<Audio>(mainPL.AudioList);*

*}*

*private void SerializeData()*

*{*

*//сохраняем пути, главный плейлист и плейлисты*

*BinaryFormatter formatter = new BinaryFormatter();*

*using (FileStream fs = new FileStream("P\_S.dat", FileMode.OpenOrCreate))*

*{*

*formatter.Serialize(fs, Paths);*

*}*

*using (FileStream fs = new FileStream("P\_List.dat", FileMode.OpenOrCreate))*

*{*

*formatter.Serialize(fs, mainPL.AudioList);*

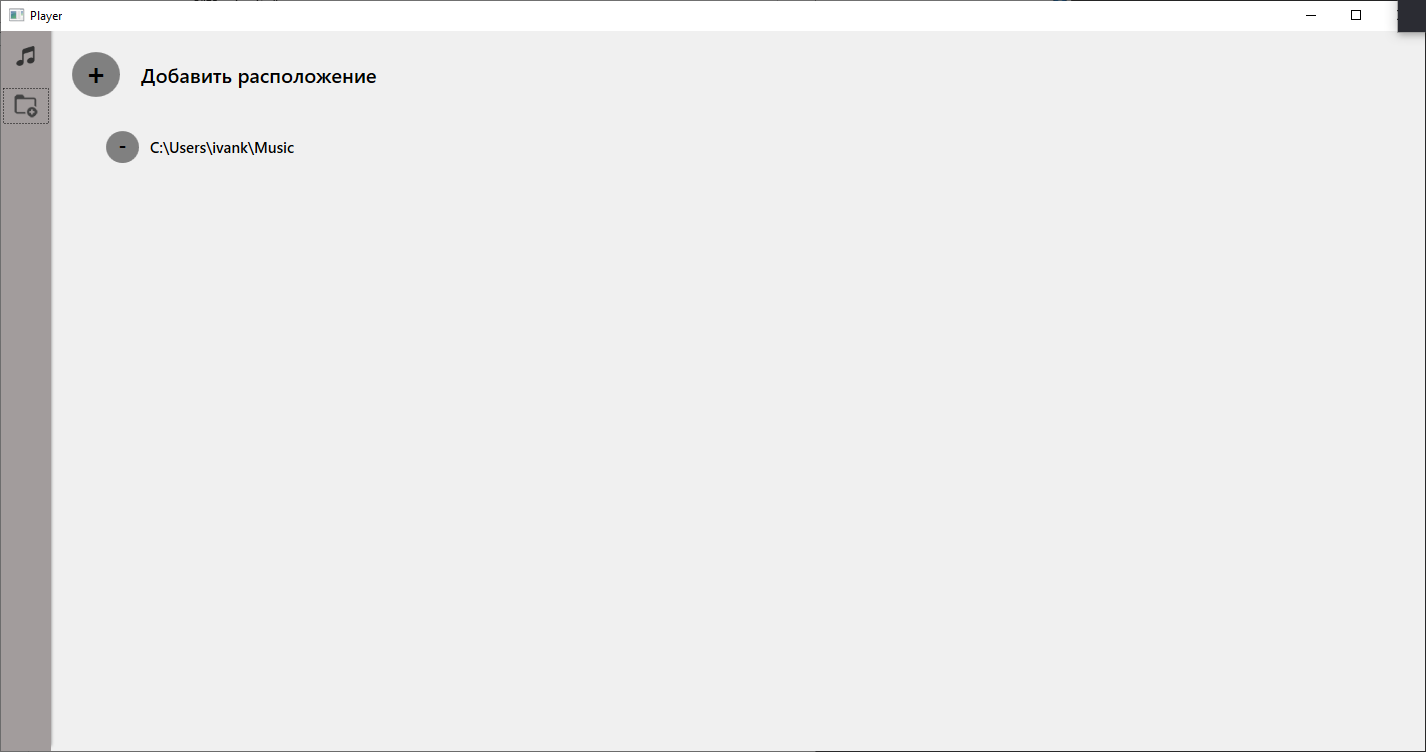
*}*

*}*

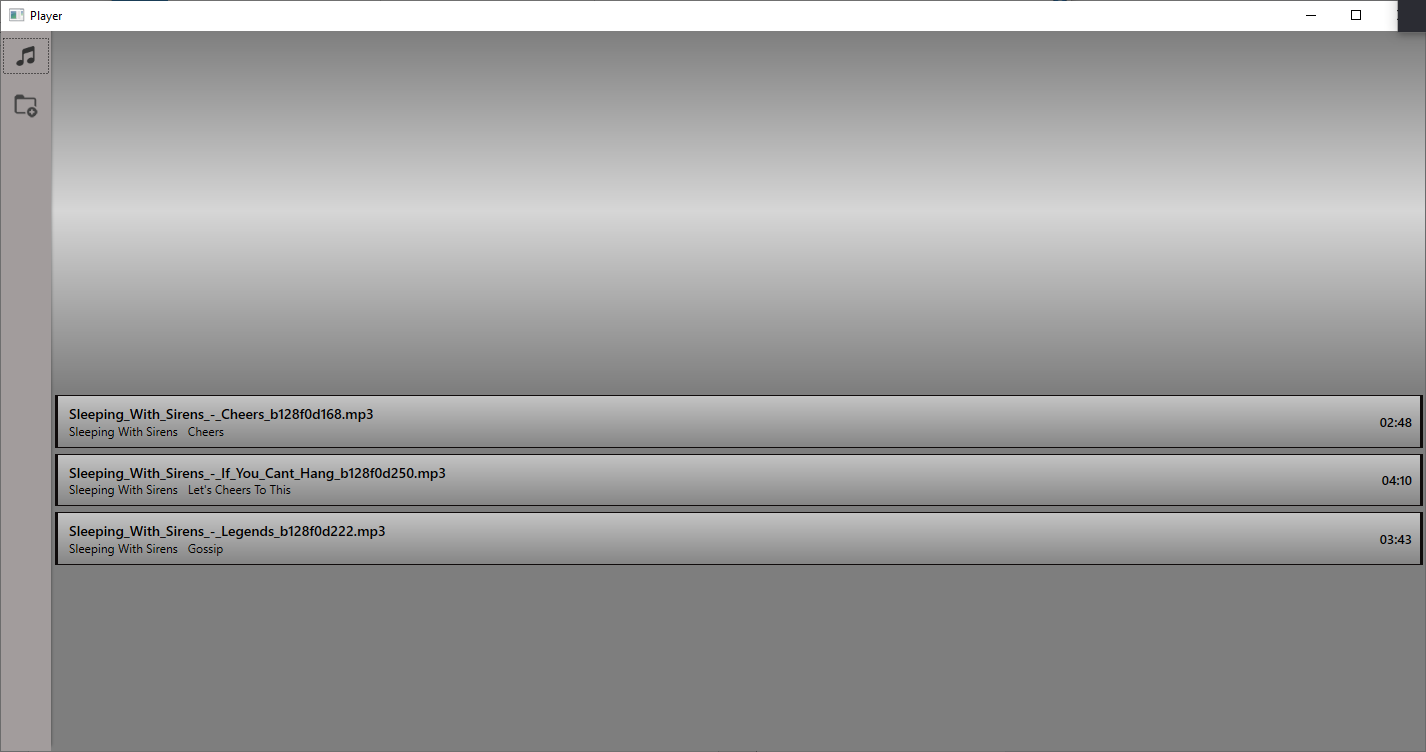
*}*

*}*

***Скриншоты результатов:***



**Рисунок 1. Результат работы программы**



**Рисунок 2. Результат работы программы**