МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

Учреждение образования

«БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ»

Факультет Информационных технологий

Кафедра Программной инженерии

Специальность 1-40 01 01 Программное обеспечение информационных технологий

Направление специальности 1-40 01 01 10 Программное обеспечение информационных технологий (программирование интернет-изданий)

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

**КУРСОВОГО ПРОЕКТА:**

по дисциплине «Объектно-ориентированные технологии программирования и стандарты проектирования»

Тема Программное средство «SPlayer»

Исполнитель

студент 2 курса группы 5 Сивак Михаил Николаевич

(Ф.И.О.)

Руководитель работы асс.Радиванович Д.А.

(учен. степень, звание, должность, подпись, Ф.И.О.)

Курсовой проект защищен с оценкой

Председатель Пацей Н.В.

(подпись)

Минск 2019

Содержание

[Введение 3](#_Toc9776180)

[1 Обзор аналогов 4](#_Toc9776181)

[2 Анализ требований к программному средству и разработка функциональных требований 9](#_Toc9776182)

[2.1 Требования 9](#_Toc9776183)

[2.2 Описание функциональности SPlayer 10](#_Toc9776184)

[3 Проектирование программного средства 11](#_Toc9776185)

[4 Создание программного средства 13](#_Toc9776186)

[5 Тестирование 14](#_Toc9776187)

[Заключение 17](#_Toc9776188)

[Список используемых источников 18](#_Toc9776189)

[Приложение 19](#_Toc9776190)

# **Введение**

Положительное влияние музыки на человека, как и на всё живое, на самом деле факт, уже неоднократно доказанный многочисленными экспериментами учёных из различных стран.

Многие моменты нашей жизни ассоциируются с определенной песней. И даже когда в памяти меркнут зрительные образы и запахи, достаточно прослушать первые такты знакомой мелодии, чтобы воспоминания ожили. Поэтому музыка благотворно влияет на память человека.

В то же время, многим известно, как музыка помогает человеку справиться с трудными жизненными ситуациями. Музыка способна создать и поддержать нужное настроение. Она может помочь расслабиться, или же наоборот – зарядить необходимой энергией.

После создания первого компьютера, у людей появилась необходимость в компьютерных программах. Для нынешнего поколения, программы тех времён были очень простыми и банальными, но весьма полезными. Это такие программы, как калькулятор, календарь и так далее. Но прогресс не стоит на месте, и со временем программисты научились работать со звуком. Первые аудиоплееры имели весьма ограниченный список функций. Их возможности не выходили за рамки воспроизведения, паузы и остановки музыкальной дорожки. На сегодняшний день у человека есть возможность управлять звуковыми волнами через компьютер, редактировать музыкальные дорожки и многое другое.

В настоящее время существует достаточно много музыкальных проигрывателей, что является большим плюсом для потребителей, так как им предоставлен широкий ассортимент выбора. Программное средство для прослушивания аудио файлов устанавливается по умолчанию почти на любой операционной системе, но не всегда пользователям хватает того функционала и всех возможностей, которые предоставляет стандартные проигрыватели. Но иногда людям очень трудно сделать выбор среди множества программ. Выбор каждого пользователя объясняется его потребностями: кому-то нужно просто воспроизводить аудио, кому-то хочется иметь возможность регулировать и настраивать интерфейс, качество звука (Эквалайзер), кому-то не хватает и этого. В прочем в данном курсовом проекте я планирую создать программное средство, которое будет воспроизводить аудио файлы с возможностью просмотра текста и аккордов проигрываемой мелодии, а также будет реализован функционал, позволяющий искать и сохранять музыку из сети Internet.

# **1 Обзор аналогов**

Для создания нового решения в виде программного средства для проигрывания музыкальных файлов необходимо ознакомиться с уже существующими аналогами. Проанализировав возможности аналогов, все их достоинства и недостатки, можно делать выводы, по которым будет проще создать своё программное обеспечение без уже существующих недостатков в других аудио проигрывателей, а также принять во внимания их преимущества.

В качестве обозреваемых аналогов были взяты программные продукты, связанные с воспроизведением музыки, как наиболее близкие по области применения к разрабатываемому программному средству.

Рассмотрим музыкальный проигрыватель Music Groove (Рисунок 1.1).

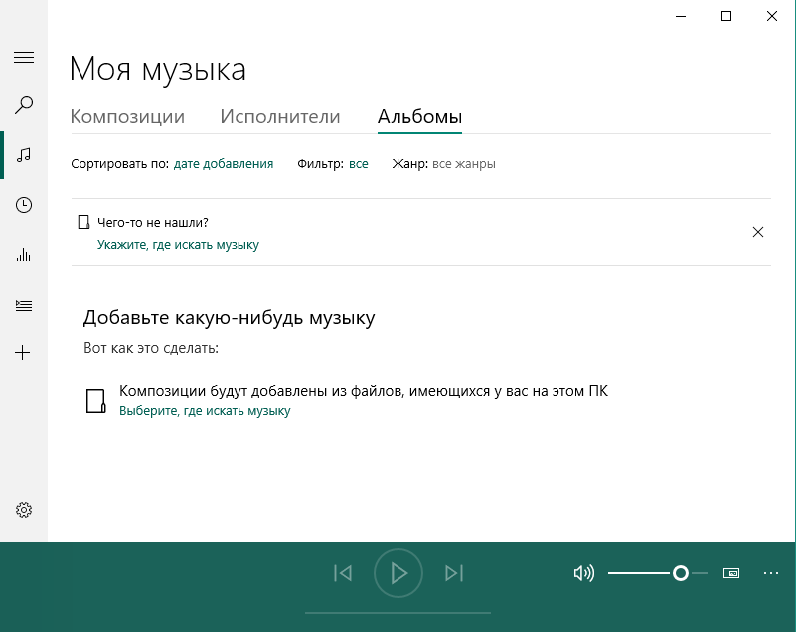


Рисунок 1.1 - Music Groove

Это приложение поддерживает все стандартные функции, связанные и выполняемые с аудиофайлами: воспроизведение, пауза, переход к следующей мелодии, регулирование громкости, изменение баса и прочих характеристик в эквалайзере, а также добавление плейлистов, управление последовательностью воспроизведения, просмотр последних прослушанных мелодий, поиск своих композиций.

Данное ПО вполне устраивает большое количество пользователей, но так как слишком много подобных аналогов с одинаковыми функциями, было бы целесообразно создать что-то новое и необычное, что также может понравиться некоторым пользователям. К примеру, в моём программном средстве будет реализована возможность просмотра текста и аккордов соответствующих им песен.

Рассмотрим сторонние музыкальные плееры для того, чтобы выделить преимущества в их использовании, а также имеющиеся недостатки.

Рассмотрим музыкальный плеер Winamp (Рисунок 1.2).



Рисунок 1.2 – Winamp

Возможности Winamp:

* Воспроизведения файлов популярных форматов mp3, ogg, avi, mpeg и т.д.;
* Качественное объемное звучание Winamp;
* Проигрывание потокового мультимедиа;
* Множество сменных обложек и скинов для внешнего оформления интерфейса Winamp;
* Встроенная функция записи дисков и проигрывания звуковых дисков;
* Обилие настроек, дополнений и прочих параметров;
* Имеется хороший графический эквалайзер;
* Красочные визуальные эффекты воспроизведения;
* Комфортное воспроизведение аудио с опцией плавного перехода;
* Предоставление подробной информации о композициях, исполнителях и альбомах;
* Настройка размера интерактивного окна Winamp;
* Полностью настраиваемые сочетания клавиш плеера Winamp;
* Возможность расширить функциональность при помощи дополнительных подключаемых модулей.

Преимущества Winamp:

* Проигрывание не только популярных аудио, но и видео форматов;
* Десяти-полосный эквалайзер с большим количеством настроек;
* Высокое качество воспроизведения, настраиваемые плейлисты;
* Имеются разнообразные зрительные образы модуля визуализации;
* Удобная медиатека Winamp для ваших записей;
* Множество функциональных возможностей в новых версиях;
* Качественная работа на базе любой актуальной операционной системы Windows и Mac;
* Легкое изменение очереди аудио и видео файлов в списке воспроизведения;
* Winamp поддерживает воспроизведение различных форматов файлов;
* В последней версии, разработчики значительно улучшили настройки эквалайзера;
* Интеграция в панели задач, присутствует русский язык оболочки;
* Низкие системные требования программного обеспечения, которое распространяется бесплатно;
* Импорт в музыкальный проигрыватель из библиотеки iTunes, поддержка Google Music и Apple Music;
* Новая версия Winamp позволяет синхронизировать ПК с приложением Winamp for Android и портативными устройствами Apple, также в ней улучшена оптимизация с Windows 10.

Недостатки Winamp:

* Возможны легкие подтормаживания Winamp для Виндовс при открытии «тяжелых» аудио, видео форматов;
* В ходе демонстрации обложки альбомов, могут возникать периодические сбои;

Рассмотрим медиа и аудио плеер KMPlayer (Рисунок 1.3).

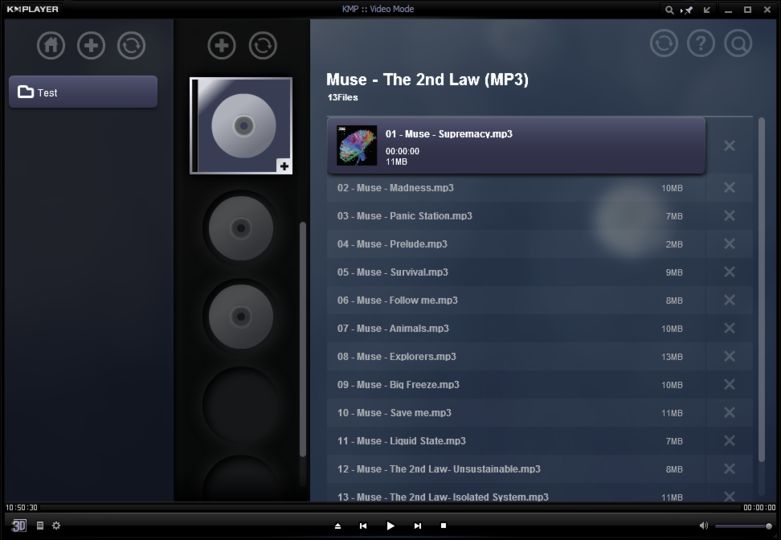


Рисунок 1.3 – KMPlayer

Возможности KMPlayer:

* Поддержка основных форматов DVD, AVI, MKV, FLV, WMV, , Ogg, OGM, WMA, 3GP, MPEG-1/2/4, VCD , RealMedia и QT;
* Настройки эффектов при просмотре видео;
* Поддержка субтитров самых разных форматов;
* Регулировка скорости воспроизведения громкости, яркости, контрастности;
* Создание скриншотов и захват видео в высоком качестве с последующим сохранением в отдельные файлы;
* Командная строка для продвинутых пользователей;
* Умная работа со звуком - приглушение, ночной режим, плавные затухания;

Преимущества KMPlayer:

* Универсальность и всеядность медиапроигрывателя – нужно скачать кмп плеер, и вы получите программу для воспроизведения любых фильмов и музыки;
* Неплохое быстродействие на современных компьютерах – КМ Плеер мгновенно открывает окно просмотра;
* Русифицированный интерфейс с большим количеством эффектов и настроек, чтобы смотреть видео было комфортнее;
* Видео плеер распространяется бесплатно и часто автоматически обновляется с официального сайта;
* Неоднократно награжден как лучший видеоплеер на ПК.

Недостатки KMPlayer:

* Требовательность к ресурсам системы;
* Сложное меню, иногда много времени уходит на поиск простых функций – например, переворот видео на 90" и т.д.;
* Иногда нестабилен в работе, чтобы воспроизвести видео требует перезапуск плеера.

Рассмотрим плеер AIMP (Рисунок 1.4).



Рисунок 1.4 – AIMP

Возможности AIMP:

* Проигрывание музыкальных файлов MP1, MP2, MP3, MPC, FLAC, OGG, APE, MP+, WavPack, S3M, XM, MOD, IT, MO3, MTM, UMX, Speex, WMA, WAV, CDA, AC3, AAC, i32v24.;
* Перекодирует треки в популярные форматы файлов WMA, MP3, WAV, OGG;
* Поддержка плагинов - приложение можно совершенствовать и настраивать под себя;
* Возможность установки глобальных горячих клавиш;
* Записывает звук с микрофона, гарантируя самое лучшее качество;
* Позволяет управлять несколькими трек-листами одновременно;
* Точный и релевантный поиск по плей-листам;
* Продвинутый эквалайзер с высокой чувствительностью;
* Программа AIMP позволяет осуществлять граб-CD и сразу редактировать теги треков;
* Настройка включения и выключения компьютера после определенной записи, по расписанию;
* AIMP поддерживает Интернет-радио.

Преимущества AIMP:

* Простой и доступный интерфейс музыкального проигрывателя;
* Огромное количество скинов в свободном доступе;
* Минимальное потребление ресурсов ОС;
* Качетсвенный 18-полосный эквалайзер в комплекте;
* Отображение обложек альбомов для легкого опознания песни;
* Вывод информации о треке в виде прозрачного тулбара на рабочем столе, когда Аимп свернут;
* Интеграция с «Проводником», добавление вкладки в контекстное меню;
* Возможности редактирования дорожек, сравнимые с базовыми в Adobe Audition;
* Поддержка cue, проигрывания потокового аудио, открытия cd dvd;
* Множества плагинов наилучшего качества в свободном доступе;
* Настраиваемый будильник (планировщик задач);
* Возможн работа с группой файлов;
* Кроссплатформенность, выпускается полный аналог для смартфонов.

Недостатки AIMP:

* Неудобное управление при сворачивании проигрывателя;
* Несмотря на понятный интерфейс аудиоплеера, присутсвуют мелкие ошибки;
* Aimp 4.51 build 2058 - 2070 не всегда применяет ассоциации файлов после инсталяции.

# **2 Анализ требований к программному средству и разработка функциональных требований**

## **2.1 Требования**

**Требования к организации входных данных**

Входные данные для программного средства будут представлены в виде аудио файлов в форматах: MP3, WAV. После поступления данных на вход, они будут воспроизводиться в звуковом формате. Также входными данными будут являться текст песен с аккордами и обложки альбомов исполнителей.

**Требования к организации выходных данных**

Выходные данные будут представлены в виде проигрывания аудиофайла, а также в виде отображения обложек альбомов исполнителей и текстов и аккордов песен, соответствующих текущей проигрываемой мелодии.

**Требование к временным характеристикам**

При загрузке аудио файлов в независимости от количества, время открытия плеера и воспроизведения может быть разным, так как будут использоваться как локальные, так и глобальные (Internet) источники данных.

**Требование к надёжности**

Требования к обеспечению надежного (устойчивого) функционирования программы. Надежное (устойчивое) функционирование программы должно быть обеспечено выполнением совокупности организационно-технических мероприятий, перечень которых приведен ниже:

* Организация бесперебойного питания технических средств;
* Выполнение требований «ГОСТ 31078-2002.
* Защита информации.
* Испытание программных средств на наличие компьютерных вирусов»;
* Необходимый уровень квалификации сотрудников профильных подразделений.
* Время восстановления после отказа.
* Время восстановления после отказа, вызванного сбоем электропитания технических средств (иными внешними факторами), не фатальным сбоем (не крахом) операционной системы, не должно превышать времени, необходимого на перезагрузку операционной системы и запуск программы, при условии соблюдения условий эксплуатации технических и программных средств.
* Время восстановления после отказа, вызванного неисправностью технических средств, фатальным сбоем (крахом) операционной системы, не должно превышать времени, требуемого на устранение неисправностей технических средств и переустановки программных средств.
* Отказы из-за некорректных действий оператора.
* Отказы программы возможны вследствие некорректных действий пользователя при взаимодействии с операционной системой. Во избежание возникновения отказов программы по указанной выше причине следует обеспечить работу конечного пользователя без предоставления ему ограниченного доступа.
* Обеспечить максимально безотказную работу программы при любых входных данных, а также при любых действиях пользователя. Программный продукт должен соответствовать требованиям безопасности, установленным ГОСТ 27451-87, ГОСТ 26104-89.

**Требования к составу и параметрам технических и программных средств**

Минимальные системные требования SPlayer:

* Платформа [OS]: Windows 10, 7, 8, Vista, XP.
* Разрядность: x86 (32-bit) или x64 (64-bit).
* ЦП [CPU]: Dual Core 2 GHz.
* Видеоадаптер [GPU]: аналогичный минимальной версии.
* Оперативная память [RAM]: 1 Gb.
* Аудио-карта [AUDIO]: с поддержкой DX9.
* Контроллер: клавиатура, мышь.

## **2.2 Описание функциональности SPlayer**

Программное средство должно обеспечивать возможность выполнения перечисленных ниже функций:

* Воспроизведение аудиофайлов форматов mp3 и wav.
* Переход к следующему и предыдущему треку.
* Регулирование громкости звука.
* Перемешивание плейлиста.
* Изменение обложки альбома исполнителя в реальном времени в соответствии с играющей мелодией.
* Динамическое получение ссылок из базы данных на папки и файлы, содержащие данные для приложения.
* Считывание текстовых файлов, содержащих текст соответствующей песни, а также аккорды к мелодии и отображение их в скрывающемся окне в реальном времени.
* Добавление музыки в поле плеера с помощью событий Drag и Drop.
* Поиск и скачивание музыки в сети Internet.
* Работа и взаимодействие с Last.fm API, а также API .NET.
* Отправка запросов и получение ответов с помощью протоколоа HTTP.
* Модель отвечающая за обработку и получение данных в формате JSON.
* Возможность устанавливать мелодии на повтор.

# **3 Проектирование программного средства**

В данном приложении используются два модуля: главный модуль (MainWindow), модуль элемента плеера (listMusic). В модуле элемента плеера содержатся действия, связанные с описанием нового компонента, а именно время, номер и название песни. Все остальные возможности аудиоплеера реализованы в главном модуле.

В проекте используются следующие фреймворки, а также библиотеки и компоненты:

* Entity framework v6.2.0 – объектно-ориентированная технология доступа к данным, является object-relational mapping (ORM) решением для .NET Framework от Microsoft. Предоставляет возможность взаимодействия с объектами как посредством LINQ в виде LINQ to Entities, так и с использованием Entity SQL.
* Newtonsoft.Json framework v12.0.2 – это популярная высокопроизводительная среда JSON для .NET для анализирования, запросов и изменений JSON с помощью объектов Json.Net, JObject, JArray и JValue.
* AngleSharp v0.13.0-alpha-739 – Усовершенствованная библиотека парсеров для разметки, анализирующая HTML5, MathML, SVG и CSS для создания DOM на основе официальных спецификаций W3C.
* NAudio v1.9.0 – динамически подключаемая библиотека для работы с аудиофайлами.
* NReco.VideoCinverter v1.1.3 – С# Видео-библиотека, позволяющая кодировать и декодировать практически все известные аудио и видео форматы, такие как mp4, mp3, h264, wav, ogg и многие другие.
* VideoLibrary v2.0.3 – это быстрый и чистый способ загрузки видео с YouTube. Он полностью переносим и не имеет никаких зависимостей.
* YoutubeExplode v4.6.0 – библиотека, которая анализирует общедоступные метаданные на видео YouTube, позволяет загружать их и многое другое. Не требует ключа API и не устанавливает квоты использования.
* YoutubeExtractor v0.10.11 – это библиотека для извлечения ссылки на скачивание из видео YouTube, загрузки их и / или извлечения их звуковой дорожки.
* Material Design – пакет, позволяющий работать с дизайном приложений на высоком уровне, создавать интуитивно понятный интерфейс.

Связи между классами представлены на диаграмме классов (Рисунок 3.1).

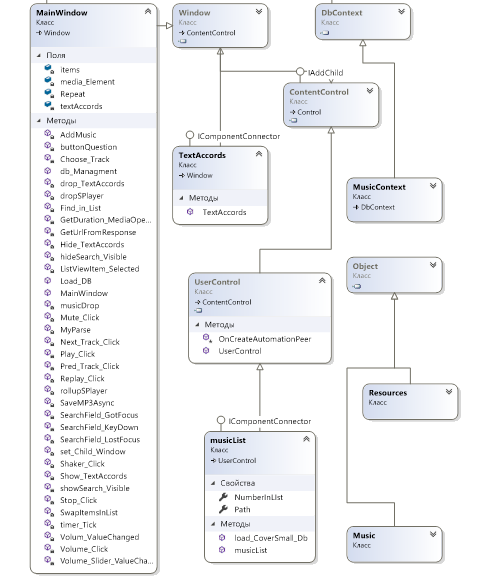
****

Рисунок 3.1 – Диаграмма классов

Список процедур и функций, использование которых необходимо в модуле MainUnit:

* void ListMusic\_Drop(object sender, DragEventArgs e) // функция, осуществляющая загрузку файла в аудиоканал и попытку его воспроизведения.
* void SearchField\_KeyDown(object sender, RoutedEventArgs e) // процедура, отвечающая за скачивание трека из интернета по нажатию клавиши.
* void GetDuration\_MediaOpened(object sender, RoutedEventArgs e) // функция, предназначенная для нахождения длины аудиофайла, загруженного в аудиоканал.
* void timer\_Tick(object sender, EventArgs e)// функция вывода длины трека и времени, прошедшего с начала воспроизведения трека.
* void Choose\_Track(object sender, RoutedEventArgs e) // функция, отвечающая за выбор аудио файла из списка по нажатию.
* void Next\_Track\_Click(object sender, RoutedEventArgs e) // функция, отвечающая за переход к следующему аудио файлу из предложенного списка по нажатию.
* void Pred\_Track\_Click(object sender, RoutedEventArgs e) // функция, отвечающая за переход к предыдущему аудио файлу из предложенного списка по нажатию.
* void Replay\_Click(object sender, RoutedEventArgs e) // функция, отвечающая за повтор текущего аудио файла.
* void AddMusic(string dropfilepath) // функция, отвечающая за добавление музыки в плейлист.
* void Load\_DB() // функция, соединяющая приложение с базой данных и заполняющая её строки.
* void db\_Managment() // функция, работающая с базой данных для получения запрашиваемых данных.
* void GetUrlFromResponse(string response, string substring) // процедура, получающая URL на видео из YouTube.
* void SaveMP3(string VideoURL) // процедура, выполняющаяся в новом потоке, которая сохраняет файл и добавляет его в плейлист с помощью функции void AddMusic(string dropfilepath).
* void Show\_TextAccords(object sender, RoutedEventArgs e) // событие, которое показывает дочернее окно для вывода данных из базы данных.
* void Hide\_TextAccords(object sender, RoutedEventArgs e) // событие, которое скрывает дочернее окно.
* void drop\_TextAccords(object sender, EventArgs e) // событие, позволяющее перемещать дочернее окно при перетаскивании главного.
* void set\_Child\_Window(object sender, RoutedEventArgs e) // метод, устанавливающий главное окно владельцем дочернего.
* void Shaker\_Click(object sender, RoutedEventArgs e) // функция, позволяющая перемешать треки в плейлисте.
* void load\_CoverSmall\_Db() // метод, отвечающий за отображение обложки каждой мелодии, добавленной в плейлист.

Диаграмма базы данных, спроектированной в ходе разработки приложения представлена на рисунке 3.2.

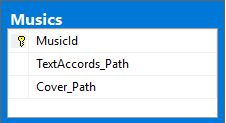


Рисунок 3.2 – Диаграмма базы данных

# **4 Создание программного средства**

В качестве интерфейса прикладного программирования был выбран обширный API-интерфейс – Windows Presentation Foundation (WPF), предназначенный для создания настольных программ с графически насыщенным пользовательским интерфейсом. В основе графической технологии WPF лежит мощная инфраструктура, основанная на DirectX. Это является одним из основных отличий WPF от более ранней технологии создания пользовательских интерфейсов – Windows Forms.

Для работы с WPF использовался объектно-ориентированный язык программирования с С-подобным синтаксисом – С#, разработанный для создания приложений на платформе Microsoft .NET Framework.

Чтобы осуществлять связь между базой данных и приложением на C# необходим посредник. И именно таким посредником является технология ADO.NET Entity Framework. Эта предоставляет собой технологию работы с данными, которая основана на платформе .NET Framework. Эта технология представляет нам набор классов, через которые мы можем отправлять запросы к базам данных, устанавливать подключения, получать ответ от базы данных и производить ряд других операций.

Для дизайна приложения используется пакет Material Design от компании Google, который позволяет создавать насыщенный и интуитивно понятный дизайн.

# **5 Тестирование**

При тестировании программного средства были замечены некоторые недостатки, такие как: зависание проигрывания аудио при подгрузке данных из SQL Server, а также при скачивании музыки из сети Internet. Но приведённые выше проблемы были решены с помощью управления потоками приложения, что позволило обеспечить асинхронную подгрузку данных, в то время, как основной поток позволяет работать с приложением дальше при выполнении второстепенных.

На рисунке 5.1 представлена демонстрация воспроизведения аудио с отображением всей информации о ней, в том числе и обложки альбома исполнителя.

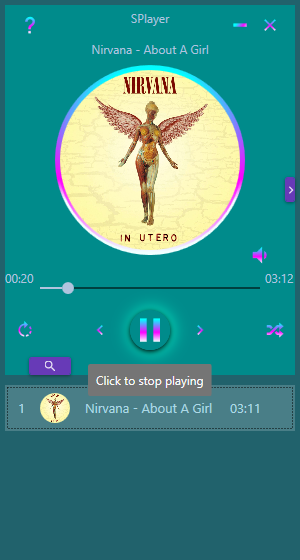


Рисунок 5.1 – Тест воспроизведения

На рисунке 5.2 представлена демонстрация отображения текста и аккордов текущей композиции в дочернем окне.

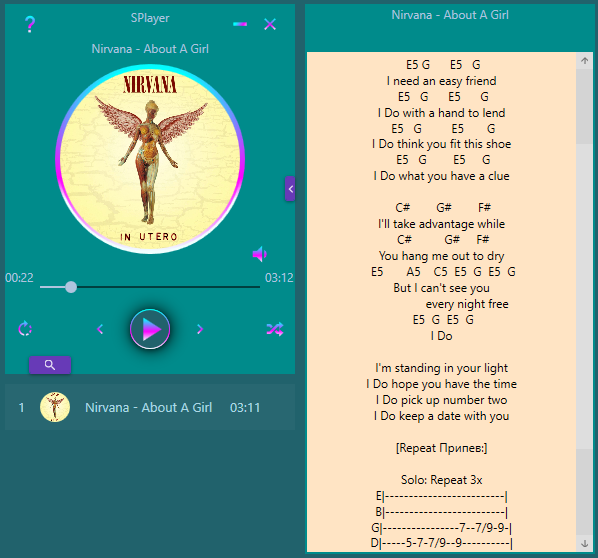


Рисунок 5.2 – Тестирование отображения данных

На рисунке 5.3 демонстрируются изменения в программе при отсутствии текста текущей песни, а также обложки.

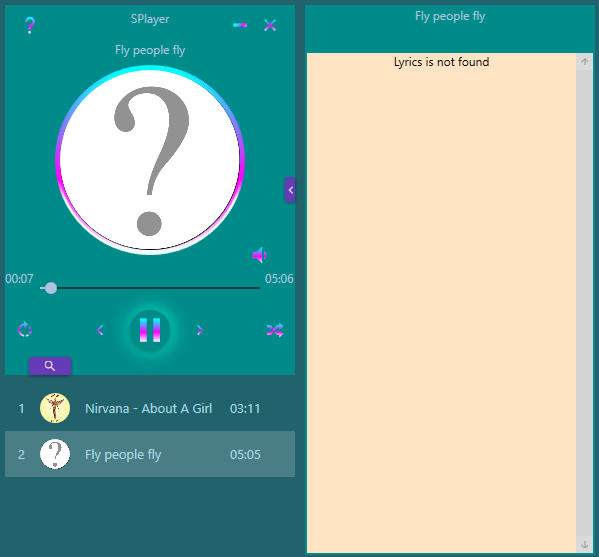


Рисунок 5.3 – Тест на отсутствие данных

На рисунке 5.4 демонстрируется открытие поля для поиска в интернете.

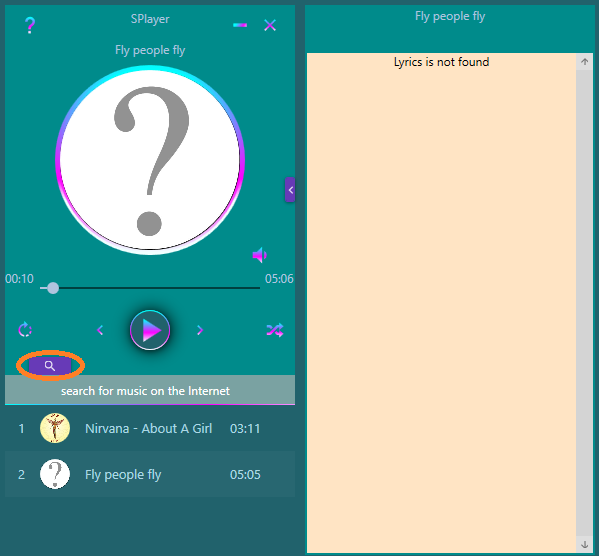


Рисунок 5.4 – Тест кнопки поиска

На рисунке 5.5 представлено тестирование поиска и скачивания музыки из интернета.

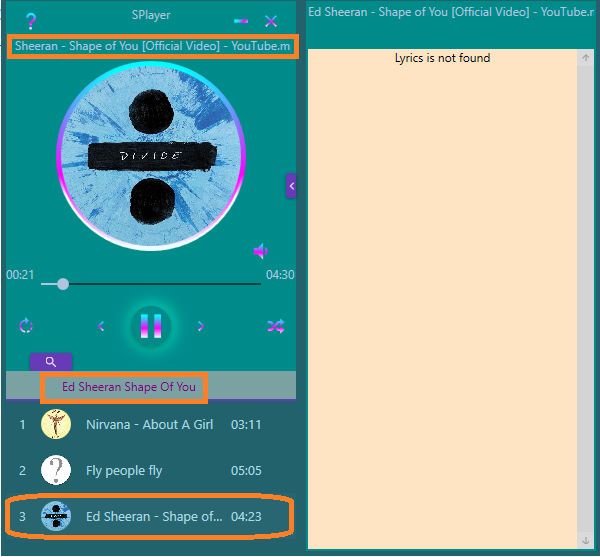


Рисунок 5.5 – Тест поиска и скачивания музыки

После того, как откроется окно ввода для поиска музыки в интеренете, следует ввести название композиции, которую пользователь желает сохранить и далее воспроизвести. После осуществления ввода названия песни, следует нажать клавишу Enter. Далее через небольшой промежуток времени в нашем плейлисте отобразится сохранённая композиция, после чего пользователь может воспроизвести её по нажатии на неё, как и остальные аудио, располагающиеся на локальных источниках.

# **Заключение**

На сегодняшний день неотъемлемой частью любого приложения является соединение с интернетом. В рамках данного курсового проекта был выполнен аудиоплеер с поиском музыки в интернете и её последующим скачиванием и отображением, прослушиванием аудиофайлов форматов MP3 и WAV. Также был добавлен функционал для отображения текста и аккордов текущей проигрываемой композиции, что даёт возможность пользователям петь вместе с музыкой. В процессе разработки были выполнены поставленные цели и задачи.

Разработка алгоритмов воспроизведения аудио и обработки звука потребовала изучения библиотеки NAudio library и некоторых физических явлений, в частности явления звука. В ходе проектирования программного средства были получены навыки относительно применения стандартных и подключения нестандартных компонентов среды Visual Studio Community 2017. Также потребовалось освоение функций Windows API и ряда возможностей протокола UDP, RTCP, RTP, для скачивания данных из интернета.

Архитектура программы является модульной и гибкой, что обеспечивает потенциал для развития. На примере других приложений, рассмотренных в разделе “Обзор аналогов”, можно определить, что программа может быть доработана. Могут быть реализованы такие функции как онлайн-вещания радиостанций, функции работы с записями и прочее.

Созданное программное средство обладает относительно небольшим спектром возможностей, но в свою очередь удобным и понятным интерфейс, наличие всех необходимых функций позволяют пользователю с комфортом прослушивать аудиозаписи.

Также приложение подойдёт тем, кто давно хотел научиться играть на гитаре. Пользователь может слушать мотив мелодий, чтобы понять, какой ритм и бой нужно брать для игры на гитаре, после чего, открыв панель с аккордами и текстом, он без особого труда сможет изучить всю мелодию. Также приложение удобно для тех, кто просто любит подпевать при проигрывании песен.

Таким образом программное средство нацелено на пользователей, которые неравнодушны к музыке и нуждаются в приложении, способном удовлетворить их творческие, а также просто любительские потребности.

# **Список используемых источников**

[1] Metanit. Сайт о программировании [Электронный ресурс]. – Электронные данные. – Режим доступа: https://metanit.com/sharp/wpf.php.

[2] Microsoft Docs [Электронный ресурс]. – Электронные данные. – Режим доступа: https://docs.microsoft.com/en-us/previous-versions/dotnet/netframework-4.0/aa970268(v=vs.100).

[3] WPF Tutorial [Электронный ресурс]. – Электронные данные. – Режим доступа: https://wpf-tutorial.com.

[4] Шилдт, Г. Полный справочник по С#: учеб. Пособие. – СПб, 2007.

[5] Microsoft Docs [Электронный ресурс]. – Электронные данные. – Режим доступа: https://docs.microsoft.com/en-us/dotnet/csharp/.

[6] Professor Web .NET and Web programming [Электронный ресурс]. – Электронные данные. – Режим доступа: https://professorweb.ru/my/WPF/base\_WPF/level1/info\_WPF.php.

[7] Мак-Дональд, М. WPF: Windows Presentation Foundation в .NET 4.0 с примерами на C# 2010 для профессионалов. Создание Windows приложений нового поколения. – Apress, 2011. – 1016 с.

# **Приложение**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace SPlayer.DataModel

{

public class Music

{

public string MusicId { get; set; }

public string TextAccords\_Path { get; set; }

public string Cover\_Path { get; set; }

}

}

Листинг 1 – Создание модели данных

namespace SPlayer.DataModel

{

using System;

using System.Data.Entity;

using System.Linq;

public class MusicContext : DbContext

{

public MusicContext()

: base("name=MusicContext")

{

}

public DbSet<Music> Musics { get; set; }

}

}

Листинг 2 – Создание контекста базы данных

public void Load\_DB()

{

try

{

using (DataModel.MusicContext db = new DataModel.MusicContext())

{

DataModel.Music music\_Nirvana = new DataModel.Music { MusicId = "Nirvana",

TextAccords\_Path = "TextAccords/Nirvana", Cover\_Path = "Covers/Nirvana" };

DataModel.Music music\_Rainbow = new DataModel.Music { MusicId = "Rainbow", TextAccords\_Path = "TextAccords/Rainbow", Cover\_Path = "Covers/Rainbow" };

DataModel.Music music\_Scorpions = new DataModel.Music { MusicId =

"Scorpions", TextAccords\_Path = "TextAccords/Scorpions", Cover\_Path =

"Covers/Scorpions" };

DataModel.Music music\_EdSheeran = new DataModel.Music { MusicId =

"EdSheeran", TextAccords\_Path = "TextAccords/EdSheeran", Cover\_Path =

"Covers/EdSheeran" };

db.Musics.Add(music\_Nirvana);

db.Musics.Add(music\_Rainbow);

db.Musics.Add(music\_Scorpions);

db.Musics.Add(music\_EdSheeran);

db.SaveChanges();

}

}

catch

{}

}

Листинг 3 – Добавление данных в базу

public void db\_Managment()

{

this.Dispatcher.BeginInvoke(new Action(() =>

{

string str\_MusicName = Track\_name.Text.ToString(); //Full name of current music

string save\_MusicName = str\_MusicName; //Saving full name of music

str\_MusicName = new string(str\_MusicName.TakeWhile(x => x != '-').ToArray());

//Cut full name for a search in db

string search = str\_MusicName.Replace(" ", "");

string save\_TextAccords = "";

string save\_Cover = "";

string new\_save = save\_MusicName.Replace(" ", "");

string path\_Parent = Path.GetFullPath(@"..\..\");

try

{

using (DataModel.MusicContext db = new DataModel.MusicContext())

{

var music = db.Musics;

foreach (DataModel.Music m in music)

{

if (m.MusicId == search)

{

save\_TextAccords = m.TextAccords\_Path;

save\_Cover = m.Cover\_Path;

}

}

}

}

catch

{

textAccords.txt.Text = "The Lyrics is not in the database :(";

}

try

{

using (StreamReader sr = new StreamReader(path\_Parent +

save\_TextAccords + "/" + new\_save + ".txt", System.Text.Encoding.UTF8))

{

textAccords.txt.Text = sr.ReadToEnd();

}

}

catch

{

textAccords.txt.Text = "Lyrics is not found";

}

try

{

cover\_Circle.ImageSource = BitmapFrame.Create(new Uri(path\_Parent +

save\_Cover + "/" + search + ".jpg"));

}

catch

{

cover\_Circle.ImageSource = BitmapFrame.Create(new Uri(path\_Parent +

"Covers/NotFound.jpg"));

}

}

));

}

Листинг 4 – Обращение к базе данных

private void SearchField\_GotFocus(object sender, RoutedEventArgs e)

{

if (SearchField.Text == "search for music on the Internet")

{

SearchField.Text = "";

SearchField.Foreground = new SolidColorBrush((Color)ColorConverter.ConvertFromString("Purple"));

}

}

private void SearchField\_LostFocus(object sender, RoutedEventArgs e)

{

SearchField.Text = "search for music on the Internet";

SearchField.Foreground = Brushes.White;

}

private string MyParse(string songname)

{

string result = "";

for (int i = 0; i < songname.Length; i++)

{

if (songname[i] == ' ')

{

result += "+";

}

else

result += songname[i];

}

return result;

}

private void SearchField\_KeyDown(object sender, KeyEventArgs e)

{

string answer = "";

string SongName = SearchField.Text;

if ((e.Key == Key.Enter) && (SongName != "search for music on the Internet") &&

(SongName != ""))

{

var MyRequest = MyParse(SongName);

string url = $"http://ws.audioscrobbler.com/2.0/?method=track.search&track={MyRequest}&api\_key=57ee3318536b23ee81d6b27e36997cde&format=json&limit=1";

try

{

HttpWebRequest httpWebRequest = (HttpWebRequest)WebRequest.Create(url);

HttpWebResponse httpWebResponse =

(HttpWebResponse)httpWebRequest.GetResponse();

string response;

try

{

var streamReader = new StreamReader(httpWebResponse.GetResponseStream());

response = streamReader.ReadToEnd();

string subString = "https://www.last.fm/music/";

answer = GetUrlFromResponse(response, subString);

httpWebRequest = (HttpWebRequest)WebRequest.Create(answer);

httpWebResponse = (HttpWebResponse)httpWebRequest.GetResponse();

streamReader = new StreamReader(httpWebResponse.GetResponseStream());

response = streamReader.ReadToEnd();

subString = "https://www.youtube.com/watch";

answer = GetUrlFromResponse(response, subString);

SaveMP3Async(answer);

}

catch

{

MessageBox.Show($"\"{SongName}\" not found");

}

}

catch

{

MessageBox.Show("No connection to the Internet");

}

}

}

private string GetUrlFromResponse(string response, string subString)

{

string answer = "";

int indexOfSubstring = response.IndexOf(subString);

if (indexOfSubstring > 0)

{

int i = indexOfSubstring;

char findSymbol = '"';

while ((response[i] != findSymbol) && (i < response.Length))

{

answer += response[i];

i++;

}

if (i >= response.Length)

throw new ArgumentException();

return answer;

}

else

throw new ArgumentException();

}

private async void SaveMP3Async(string VideoURL)

{

try

{

var youtube = YouTube.Default;

var vid = await youtube.GetVideoAsync(VideoURL);

File.WriteAllBytes(vid.FullName, await vid.GetBytesAsync());

var inputFile = new MediaFile { Filename = vid.FullName };

var outputFile = new MediaFile { Filename = $"{vid.FullName}.mp3" };

using (var engine = new Engine())

{

engine.GetMetadata(inputFile);

engine.Convert(inputFile, outputFile);

}

AddMusic(outputFile.Filename);

}

catch

{

MessageBox.Show($"Your song not found");

}

}

}

Листинг 5 – Обзор методов для поиска и загрузки музыки