Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Кафедра ЭВМ

Дисциплина: Конструирование программ и языки программирования

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовой работе на тему:

«Аудио-плеер»

Студент: гр.444601 Климович А.А.

Руководитель: Кучук С. А.

Минск, 2016

Содержание

[Введение 1](#_Toc402190757)

[1 Обзор литературы](#_Toc402190758) 1

[2 Структурное проектирование](#_Toc402190759) 1

[3 Функциональное проектирование](#_Toc402190761) 1

[4 Тестирование 1](#_Toc402190761)

[5 Руководство пользователя 1](#_Toc402190762)

[Заключение 1](#_Toc402190763)

[Список используемых источников 1](#_Toc402190764)

**Ввдение**

В данной курсовой работе будет разработан аудиоплеер. Пользователь получит возможность воспроизведения и каталогизации аудиофайлов различных форматов. Поддерживается множество популярных форматов аудиофайлов, такие как WAV, APE, MP3 и другие.

Если рассмотреть воспроизведение аудиофайлов как разовую операцию, то специальные программы для воспроизведения этих файлов совсем необязательны. Так как такое программное средство является стандартным на многих популярных операционных системах, например Windows поставляется вместе с Windows Media Player, Vox – вместе с OS X, а большинство популярных аудиоплееров, таких как Aimp, WinAmp, можно получить совершенно бесплатно.

Однако для того, чтобы воспроизводить аудиофайлы на ПК и ноутбуке, стандартных аудиоплееров зачастую недостаточно, так как они являются стандартными, и не учитывают предпочтения каждого пользователя, потому существует много аналогов. Все дело в настройках внешнего вида и интерфейса для удобной работы пользователя.

Для этого и были созданы множество аналогов, которые можно получить совершенно бесплатно. Аудиоплеер (программа для воспроизведения аудиофайлов) призван максимально облегчить процесс воспроизведения аудиофайлов, также на экономию времени на оптимальную настройку отображения и поиска нужного файла (функция Поиск). Возможно, стоит ради пробы установить на ПК несколько программ для воспроизведения аудиофайлов и облегчить себе процесс воспроизведения нужных для пользователя аудиофайлов.

На данный момент существует большое количество различных аналогов. Winamp Full, JetAudio, AlbumPlayer, KMPlayer, Windows Media Player, Media Player Classic, VLC Media Player, AIMP, aTunes, foobar2000, Songbird, TagScanner.

В своем практическом проекте будет разработано приложение со следующим функционалом: воспроизведение аудиофайлов; сортировка, поиск, добавление аудиофайлов; создание Playlist-а;

1. **Обзор литературы**
   1. **Выбор библиотеки для чтения MP3 файлов**

Наиболее популярным форматом аудиофайлов является MP3. MP3 — это кодек третьего уровня, разработанный командой MPEG, лицензируемый формат файла для хранения аудиоинформации.

MP3 является одним из самых распространённых и популярных форматов цифрового кодирования звуковой информации с потерями. Он широко используется в файлообменных сетях для оценочной передачи музыкальных произведений. Формат может проигрываться практически во всех популярных операционных системах, на большинстве портативных аудиоплееров, а также поддерживается всеми современными моделями музыкальных центров и DVD-плееров. Название файла не всегда может дать понятие о содержимом файла. Для хранения информации о МР3 файла, в него включают специальные метки - ID3 тэги. ID3 (от англ. Identify an MP3) — формат метаданных, наиболее часто используемый в звуковых файлах в формате MP3. ID3 подпись содержит данные о названии трека, альбома, имени исполнителя и т.д., которые используются медиаплеерами и другими программами, а также аппаратными проигрывателями, для отображения информации о файле и автоматического упорядочивания аудиоколлекции.

Файл, содержащий такой тег, начинается последовательностью символов «ID3». Эти символы являются частью заголовка ("Header") тега. Заголовок состоит из 10 байт. Первые три - "ID3", следующие два - версия тега, шестой - флаги, оставшиеся четыре - размер.

Каждый фрейм начинается идентификатором (4 символа, которыми могут быть заглавные латинские буквы и цифры), следующие 4 байта - размер, ещё два - флаги. Можно заметить, что заголовок фрейма также состоит из 10 байт.

В своем курсовом проекте для получение тегов из Mp3 файлов будет использована библиотека taglib-sharp. Данная библиотека предназначена для работы с тэгами медиафайлов(видео, аудио и фото). Библиотека taglib-sharp позволяет получать значение тегов из MP3 файлов без загрузки самого файла.

* 1. **Библиотеки мультимедиа.**

Для выявления наиболее важных характеристик разрабатываемого программного продукта следует провести анализ программы, выполняющих поставленную задачу, а именно Windows Media Player 11 компании Microsoft.

Встроенный медиа проигрыватель, поставляемый вместе с ОС Windows. Позволяет воспроизводить большинство типов аудио и видео файлов. При первом запуске проигрывателя он автоматически выполняет поиск определенных папок по умолчанию, расположенных в библиотеках «Музыка», «Видео», «Изображения» и «ТВ-записи» на компьютере. Если добавить или удалить файлы в этих библиотеках мультимедиа, проигрыватель автоматически обновит доступные мультимедийные файлы. Можно также добавить в библиотеки Windows новые папки, расположенные в другом месте на компьютере или на съемных запоминающих устройствах. При воспроизведении мультимедийного файла, хранящегося на компьютере или на съемном запоминающем устройстве, файл автоматически добавляется в библиотеку проигрывателя, поэтому можно обращаться к этому файлу непосредственно из проигрывателя. Если на компьютере имеется не отслеживаемая папка с мультимедийными файлами, можно включить ее в одну из библиотек Windows, чтобы проигрыватель мог выполнять в ней поиск.

* 1. **Выбор базы данных.**

Для хранения данных существует большое количество реляционных СУБД, среди которых можно выделить системы, описанные в таблице 1.1

Таблица 1.1 – Обзор СУБД

|  |  |
| --- | --- |
| Название СУБД | Описание |
| MsSQL | Microsoft SQL Server — система управления реляционными базами данных (РСУБД), разработанная корпорацией Microsoft. Основной используемый язык запросов – Transact-SQL, создан совместно Microsoft и Sybase. Transact-SQL является реализацией стандарта ANSI/ISO по структурированному языку запросов (SQL) с расширениями. Используется для работы с базами данных размером от персональных до крупных баз данных масштаба предприятия; конкурирует с другими СУБД в этом сегменте рынка.  Написана на языках C, C++, C#.  Текущая версия – MS SQL Server 2014. |
| SQLite | Компактная встраиваемая реляционная база данных. Исходный код библиотеки открыт. Слово «встраиваемая» означает, что SQLite не использует парадигму клиент-сервер, то есть движок SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы.  SQLite поддерживает динамическое типизирование данных. Возможные типы полей: INTEGER, REAL, TEXT, BLOB. |
| MongoDB | Документо-ориентированная СУБД с открытым исходным кодом, не требующая описания схемы таблиц. Написана на языке C++.  База имеет минимизированную семантику для работы с транзакциями. Используемая модель документов хранения данных (JSON/BSON) проще кодируется, проще управляется, а внутренняя группировка релевантных данных обеспечивает дополнительный выигрыш в быстродействии. |

В результате изучения достоинств и недостатков приведенных в таблице 1.1 Обзор СУБД для хранения информации была выбрана СУБД MsSQL.

При работе с различными СУБД раньше для создания, модификации и управления данными писали запросы. Затем программисты перешли на хранимые процедуры. Потом стали использоваться такие технологии, как Database to linq, BLToolkit и NHibernate, однако в настоящее время огромной популярностью пользуется Entity Framework благодаря отсутствию необходимости писать SQL-код и удобному API. Исходя из вышесказанного, было принято решение об использовании Entity Framework при разработке.

В конечном итоге планируется создание кроссплатформенного средства для воспроизведения и каталогизации аудиофайлов. Будут подключены сторонние библиотеки для распознания текста. Средой разработки выбрана Microsoft Visual Studio. В качестве базы данных будет использоваться MS SQL, а в качестве технологии доступа к БД будем использовать Entity Framework. Реализовываться приложение будем на языке С#.

**2** **Функциональное проектирование**

Проектируемое программное средство будет включать в себя следующие модули:

* модуль доступа к БД;
* модуль индексирования;
* модуль управления музыкальной картотекой;
* модуль взаимодействия с пользователем;
* модуль воспроизведения файлов;
  1. **Модуль доступа к базе данных**

Данный модуль предназначен для получения информации из базы данных и записи информации в нее. Модуль состоит из набора классов, представленных на рисунке 3.1.

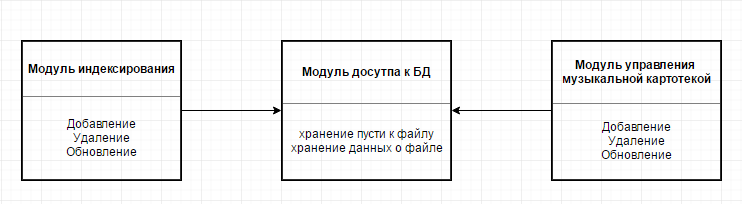


Рисунок 3.1 – Диаграмма классов модуля доступа к базе данных

* 1. **Модуль индексирования**

Индексирование файла отражено на диаграмме взаимодействия, представленной на рисунке 3.2.

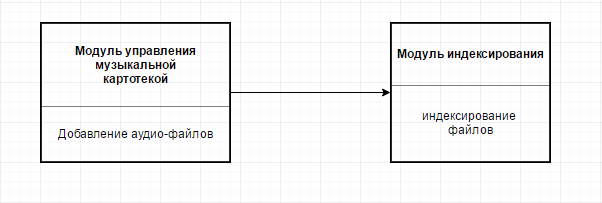


Рисунок 3.2 – Диаграмма классов модуля индексирования

* 1. **Модуль управления музыкальной картотекой**

Данный модуль предназначен для управления музыкальной картотекой. Модуль состоит из набора классов, представленных на рисунке 3.3

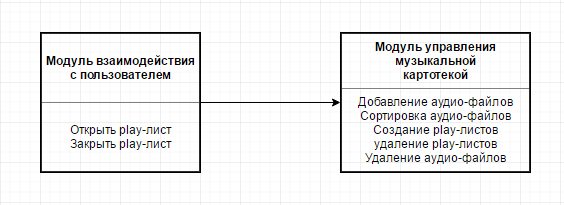


Рисунок 3.3 – Схема управления музыкальной картотекой

.

* 1. **Модуль взаимодействия с пользователем**

Данный модуль предназначен для взаимодействия с пользователем. Модуль состоит из набора классов, представленных на рисунке 3.4

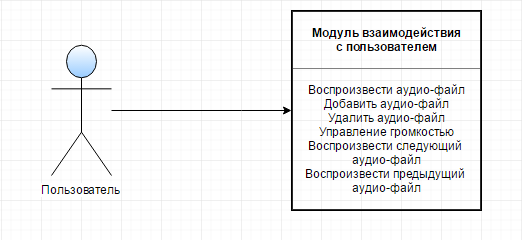


Рисунок 3.4 – Диаграмма взаимодействия с пользователем

* 1. **Модуль воспроизведения аудио-файлов**

Данный модуль предназначен для воспроизведения аудио-файлов. Модуль состоит из набора классов, представленных на рисунке 3.5

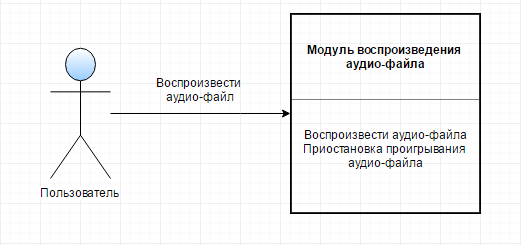


Рисунок 3.5 – схема воспроизведения аудио-файлов

**3 Структурное проектирование**

Проектируемое программное средство будет состоять из компонентов, изображенных на Рисунке 3.1.

Модуль воспроизведения файлов

Модуль индексирования

Модуль доступа к БД

Модуль взаимодействия с пользователем

Модуль управления музыкальной картотекой

Рисунок 3.1 – Структурная схема

Описание компонентов, изображенных на рисунке 3.1:

* модуль доступа к БД – это компонент, предназначенный для получения информации из базы данных и записи информации в нее;
* модуль индексирования – это компонент, предназначенный для получения данных из файла;
* модуль управления музыкальной картотекой – это компонент, предназначенный для добавления, удаления аудио-файлов, создания play-листов, сортировка аудио-файлов в play-листе;
* модуль взаимодействия с пользователем – это компонент, предназначенный для взаимодействия с пользователем;
* модуль воспроизведения аудио-файлов – это компонент, предназначенный для воспроизведения аудио-файлов.

Ниже на рисунке 3.2 представлена схема базы данных.

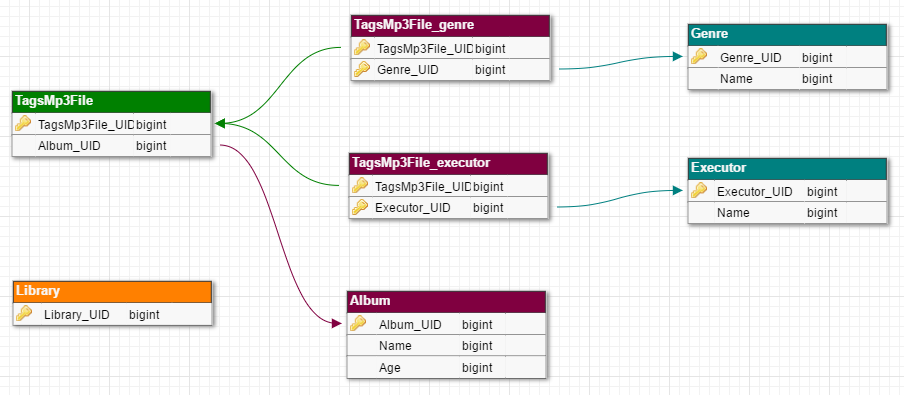


Рисунок 3.2 – Схема базы данных

Ниже в Таблице 3.1 представлено описание структуры базы данных

|  |  |  |
| --- | --- | --- |
| Номер | Таблица | Описание |
|  | TagsMp3File | Хранит в себе ссылки на таблицы TagsMp3File\_genre, Album, TagsMp3File\_executor. |
|  | TagsMp3File\_genre | Хранит в себе ссылки на таблицы Genre и  TagsMp3File. |
|  | TagsMp3File\_executor | Хранит в себе ссылки на таблицы Executor и TagsMp3File. |
|  | Album | Хранит в себе информацию о названии альбома для каждого файла формата Mp3. |
|  | Genre | Хранит в себе информацию о жанре для каждого файла формата Mp3. |
|  | Executor | Хранит в себе информацию о исполнителях для каждого файла формата Mp3. |
|  | Library | Хранит библиотеки |

Таблица 3.1 – Описание структуры базы данных

**4 Тестирование**

Целью тестирования является проверка работоспособности разработанного программного обеспечения. На этой стадии необходимо проверить корректное функционирование разработанного программного обеспечения и соответствие его требованиям, выдвинутым в техническом задании. При выявлении несоответствий работы программы техническому заданию, либо ошибок, требуется доработка программного обеспечения и (или) документации. Разработанное программное обеспечение должно гарантировать устойчивое функционирование независимо от действий конечных пользователей. При возникновении отдельных ошибочных ситуаций разработанное программное обеспечение должно их успешно обрабатывать. Для проведения корректного тестирования сначала необходимо разработать порядок испытаний. Порядок испытаний - это список последовательности действий направленных на проверку корректности работы программы и (или) ее отдельных функциональных частей.

Существует два основных вида тестирования: функциональное и структурное. При функциональном тестировании программа рассматривается как “черный ящик” (то есть ее текст не используется). Происходит проверка соответствия поведения программы ее внешней спецификации. Возможно ли при этом полное тестирование программы? Очевидно, что критерием полноты тестирования в этом случае являлся бы перебор всех возможных значений входных данных, что невыполнимо.

Поскольку исчерпывающее функциональное тестирование невозможно, речь может идти о разработки методов, позволяющих подбирать тесты не “вслепую”, а с большой вероятностью обнаружения ошибок в программе. При структурном тестировании программа рассматривается как “белый ящик” (т.е. ее текст открыт для пользования). Происходит проверка логики программы. Полным тестированием в этом случае будет такое, которое приведет к перебору всех возможных путей на графе передач управления программы (ее управляющем графе). Даже для средних по сложности программ числом таких путей может достигать десятков тысяч.

Таким образом, ни структурное, ни функциональное тестирование не может быть исчерпывающим. Рассмотрим подробнее основные этапы тестирования программных комплексов. В тестирование многомодульных программных комплексов можно выделить четыре этапа:

1. Тестирование отдельных модулей;
2. Совместное тестирование модулей;
3. Тестирование функций программного комплекса (т.е. поиск различий между разработанной программой и ее внешней спецификацией);
4. Тестирование всего комплекса в целом (т.е. поиск несоответствия созданного программного продукта, сформулированным ранее целям проектирования, отраженным обычно в техническом задании).

На первых двух этапах используются, прежде всего, методы структурного тестирования, т.к. на последующих этапах тестирования эти методы использовать сложнее из-за больших размеров проверяемого программного обеспечения; последующие этапы тестирования ориентированы на обнаружение ошибок различного типа, которые не обязательно связаны с логикой программы.

Тестирование модулей проводилось в процессе создания системы, когда данный модуль готов к использованию, при возможных ошибочных ситуациях информация выводилась на консоль, а затем исправлялось место в модуле, где возникала ошибка.

Тестирование всего приложения проводилось методом ручного тестирования путём многократных запусков, разнообразных оправляемых событиях и моделируемых ошибочных ситуаций.

В результате тестирования была получено более защищенное приложения, адекватно реагирующее на возможные ошибочные ситуации.

Таблица 4.1 – Результаты тестирования

|  |  |  |  |
| --- | --- | --- | --- |
| Модуль | Название теста | Сценарий | Пройден |
| Управление  товарами | Добавление и удаление аудио-файлов в play-лист | 1 Запустить приложение   1. Попробовать добавить аудио-файлы в play-list 2. Убедиться, что файлы добавлены 3. Удалить аудио-файлы из play-листа | Да |
| Воспроизведение аудио-файлов из play-листа | 1 Запустить приложение  2 Добавить аудио-файлы в play-лист  2 Воспроизвести аудио-файл  3 Убедиться, что аудио-файл воспроизводится корректно | Да |
| Будет дополнено | | | |

**5 Руководство пользователя**

1. **Заключение**

В ходе курсовой работы был разработан аудиоплеер. Пользователь

получил возможность воспроизведения и каталогизации аудиофайлов

формата Mp3.

Для получение тегов из Mp3 файлов была использована библиотека taglib-sharp. Данная библиотека предназначена для работы с тэгами медиафайлов(видео, аудио и фото). Библиотека taglib-sharp позволяет получать значение тегов из MP3 файлов без загрузки самого файла. Данный полученные с помощью библиотеки taglib-sharp хранятся в базе данных.

В качестве базы данных использовалась MsSQL.

В заключении создания программного средства было проведено тестирование. Тестирование модулей проводилось в процессе создания системы, когда данный модуль готов к использованию, при возможных ошибочных ситуациях информация выводилась на консоль, а затем исправлялось место в модуле, где возникала ошибка.

Тестирование всего приложения проводилось методом ручного тестирования путём многократных запусков, разнообразных оправляемых событиях и моделируемых ошибочных ситуаций.

В результате тестирования была получено более защищенное приложения, адекватно реагирующее на возможные ошибочные ситуации.

В будущем планируется реализовать адаптивный интерфейс и порт на другие устройства.