Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерных систем и сетей

Кафедра программного обеспечения информационных технологий

Дисциплина: Операционные системы и системное программирование (ОСиСП)

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

к курсовому проекту на тему

«Программное средство аудиозаписи “BahRecorder»

БГУИР КП 1–40 01 01 15 ПЗ

Выполнил

студент, гр. 851004 Борисюк Ф.А.

Проверил: Шульга Е.С.

Минск 2020

Содержание

[Введение 5](#_Toc498266598)

[1 Анализ литературных источников 6](#_Toc498266599)

[1.1 Анализ существующих приложений 6](#_Toc498266600)

[2 Постановка задачи 8](#_Toc498266601)

[3 Разработка программного средства 9](#_Toc498266602)

[3.1 Описание используемых технологий 9](#_Toc498266603)

[3.2 Структура программного средства 10](#_Toc498266604)

[4 Тестирование и проверка работоспособности программного средства 19](#_Toc498266605)

[5 Руководство по установке и использованию программного средства 26](#_Toc498266606)

[Заключение 29](#_Toc498266607)

[Список использованных источников 30](#_Toc498266608)

[Приложение А 31](#_Toc498266609)

Введение

В современном мире наиболее развивающаяся отрасль – это отрасль компьютеров. Еще несколько десятилетий назад все персональные компьютеры можно было пересчитать по пальцам. Однако, сейчас их количество возросло настолько, что в каждой семье есть один, а может быть и более компьютеров. В наше время информационные технологии занимают центральное место в интеллектуализации общества, развитии его системы образования и культуры.

Операционная система Windows заняла свое место среди постребителей благодаря блестящим возможностям создания удобных графических интерфесов, что в свою очередь позволило программистам плодотворно создавать понятные и приятные в использовании прикладные программы.

WindowsAPI (англ. applicationprogramminginterfaces) — общеенаименование целого набора базовых функций интерфейсов программирования приложений операционных систем семейств Microsoft Windows корпорации «Майкрософт». Является самым прямым способом взаимодействия приложенийсWindows.

Windows API спроектирован для использования в языке Си для написания прикладных программ, предназначенных для работы под управлением операционной системы MS Windows. Работа через Windows API — это наиболее близкий к операционной системе способ взаимодействия с ней из прикладных программ. Windows API представляет собой множество функций, структур данных и числовых констант, следующих соглашениям языка Си. Все языки программирования, способные вызывать такие функции и оперировать такими типами данных в программах, исполняемых в среде Windows, могут пользоваться этим API. В частности, это языки C++, Pascal, VisualBasic и многие другие.

# Анализ литературных источников

## Анализ существующих приложений

Компания Microsft поставляет в стандартных версиях WindowsXP и младше, Windows 7 программу для записи звука с микрофона компьютера. В обоих версиях операционной системы данная программа называется «Звукозапись». Остановимся на каждой из версий подробнее.

В WindowXP«Звукозапись» (рисунок 1) имела очень неоднозначный функционал.

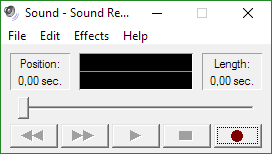


Рисунок 1 – «Звукозапись» WindowsXP

Имея очень простной интерфейс программа удобна для выполнения своей прямой задачи – создание простых (без высокой дискретизации) аудиофайлов со средней длиной воспроизведения до 5 секунд. Время записи было ограничено 1 минутой. Пользователям приходилось идти на различные ухищрения для увеличения длины записи. На сайте Microsoft есть отдельная статья о том как это сделать [3]. Однако программа умеет применять различные звуковые эффекты наример «Эхо» или «Реверсия». Вместе с тем не понятно, как применять данные эффекты, если аудиозапись очень короткая.

В версии Windows 7 (рисунок 2) программу значительно доработали.

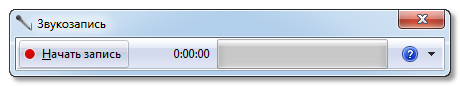


Рисунок 2 – «Звукозапись» Windows 7

С помощью программы «Звукозапись» в Windows 7 можно записать звук в звуковой файл и сохранить его на компьютере. Звук можно записывать с различных звуковых устройств (например, с микрофона), подключенных к звуковой плате компьютера. Виды источников ввода звука, с которых можно записать звук, зависят от имеющихся звуковых устройств и источников входа на звуковой плате.

В новой версии «Звукозаписи» разработчики убрали функционально малополезные эффекты аудио, также программа начала записывать звук на диск, поэтому длина записи ограничена только количеством свободного дискового пространства. Такжеразработчики реализовали запись в формат WMA (WindowsMediaAudio). Данный формат предназначен для сжатия аудиофайла однако требует дополнительных кодеков, также формат WMAне получил такого широкого распространения как например MP3, следовательно использовать аудиозаписи на других ОС затруднительно.

Формат звука WAV- это цифровой формат звука, по своему качеству примерно одинаковый с форматом AUDIO, но объем измеряется не в минутах, а в байтах, мегабайтах и т.д. Данный формат может храниться как на жёстком диске ПК, что делает его весьма доступным для работы со звуковым оформлением сценария, так и на компакт-диске.

WAVфайл состоит из двух чётких областей. Одна из них – заголовок файла, другая – область данных.

Заголовки WAV-файла используют стандартный формат RIFF. Первые 8 байт файла - стандартный заголовок секции RIFF, который имеет ID секции "RIFF" и размер секции, равный размеру файла минус 8 байт, используемых для RIFF-заголовка. Первые 4 байта данных в секции "RIFF" определяют тип ресурса, который можно найти в секции. WAV-файлы всегда используют тип ресурса "WAVE". После типа ресурса (ID "WAVE") идут все секции звукового файла, которые определяют аудиосигнал.

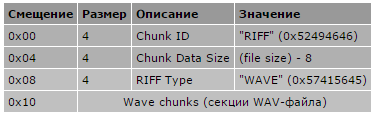


Рисунок 3 – Значение полей в секции RIFF

Хотя существует довольно много типов секций, заданных для файлов WAV, но большинство WAV-файлов содержат только две из них - секцию формата ("fmt ") и секцию данных ("data"). Это именно те секции, которые необходимы для описания формата выборок аудиоданных, и для хранения самих аудиоданных. Хотя официальная спецификация не задает жесткий порядок следования секций, наилучшей практикой будет размещение секции формата перед секцией данных. Многие программы ожидают именно такой порядок секций, и он наиболее разумен для передачи аудиоданных через медленные, последовательные источники наподобие Интернет. Иначе если формат придет после данных, то перед стартом воспроизведения необходимо считать и запомнить все аудиоданные, только после получения формата запускать воспроизведение.

Все секции формата RIFF и соответственно секции Wave сохраняются в следующем формате (см. Рисунок 4). Заметьте, что даже вышеупомянутая секция RIFF соответствует этому формату.

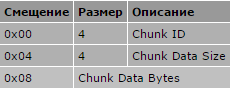


Рисунок 4 - Формат секций RIFF и Wave

Также формат WAV может иметь своё индивидуальное название на любом носителе (каждый файл вы можете назвать, как вам захочется, и это название будет сохраняться при копировании или переносе). Не нуждается в специальных программах для копирования, например, с лазерного диска на жёсткий диск, что делает удобным перемещение звуковых файлов с компьютера на компьютер.

Но звук в формате WAV не может быть воспроизведён на простом лазерном проигрывателе, этот формат может быть обработан только программами компьютера или CD-плеерами.

# Постановка задачи

После анализа приложений было принятно решение об создании приложения, способную записывать и воспроизводить аудиофайлы формата WAV.

Для выполнения поставленной задачи было решено разработать следующий функционал:

* Возможность записывать аудиофайлы в формате WAV
* Возможность прослушивать аудиофайлы в формате WAV
* Возможность сохранить записанный аудиофайл
* Возможность открыть аудиофайлы в формате WAV
* Отображение времени записи
* Отображение времени воспроизведения
* Отображение состояния воспроизведения
* Смена позиции воспроизведения (перемотка)
* Пауза, возобновление записи/воспроизведения аудиофайла

# Разработка программного средства

## Описание используемых технологий

Разработка программного средства велась на языке программирования C++ с использованием интерфейса прикладного программирования WindowsAPI. Для работы с музыкальными файлами использована звуковая библиотека BASSaudiolibrary. Также для создания программного средства была использована интегрированная среда разработки MicrosoftVisualStudio.

* **Язык программирования C++**

[Компилируемый](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D0%B8%D1%80%D1%83%D0%B5%D0%BC%D1%8B%D0%B9_%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F),[статически типизированный](https://ru.wikipedia.org/wiki/%D0%A1%D1%82%D0%B0%D1%82%D0%B8%D1%87%D0%B5%D1%81%D0%BA%D0%B0%D1%8F_%D1%82%D0%B8%D0%BF%D0%B8%D0%B7%D0%B0%D1%86%D0%B8%D1%8F)[язык программирования](https://ru.wikipedia.org/wiki/%D0%AF%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F). Имеет богатую стандартную библиотеку, которая включает в себя распространённые контейнеры и алгоритмы, ввод-вывод, регулярные выражения, поддержку многопоточности. C++ широко используется для разработки программного обеспечения. Область его применения включает создание[операционных систем](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0), разнообразных прикладных программ,[драйверов](https://ru.wikipedia.org/wiki/%D0%94%D1%80%D0%B0%D0%B9%D0%B2%D0%B5%D1%80)устройств, приложений для встраиваемых систем, высокопроизводительных серверов, а также развлекательных приложений. Существует множество реализаций языка C++для различных платформ как бесплатных, так и коммерческих.

Синтаксис C++ унаследован от языка[C](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F)). Одним из принципов разработки этого языка было сохранение совместимости с C. Тем не менее, C++ не является в строгом смысле надмножеством C. Множество программ, которые могут одинаково успешно транслироваться как[компиляторами](https://ru.wikipedia.org/wiki/%D0%9A%D0%BE%D0%BC%D0%BF%D0%B8%D0%BB%D1%8F%D1%82%D0%BE%D1%80)C, так и компиляторами C++, довольно велико, но не включает все возможные программы на C [4].

* **Windows API**

Windows API – общее наименование целого набора базовых функций [интерфейсов программирования приложений](https://ru.wikipedia.org/wiki/%D0%98%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F_%D0%BF%D1%80%D0%B8%D0%BB%D0%BE%D0%B6%D0%B5%D0%BD%D0%B8%D0%B9)операционных систем семейств[Microsoft Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows). Является самым прямым способом взаимодействия приложений с ОС Windows.

Windows APIспроектирован для использования в языке [C](https://ru.wikipedia.org/wiki/%D0%A1%D0%B8_(%D1%8F%D0%B7%D1%8B%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D1%8F))для написания[прикладных программ](https://ru.wikipedia.org/wiki/%D0%9F%D1%80%D0%B8%D0%BA%D0%BB%D0%B0%D0%B4%D0%BD%D0%BE%D0%B5_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B5_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D0%B5), предназначенных для работы под управлением операционной системы MS Windows. Работа черезWindows API– это наиболее близкий к операционной системе способ взаимодействия с ней из прикладных программ.

Windows API представляет собой множество функций, структур данных и числовых констант, следующих соглашениям языка C. Все языки программирования, способные вызывать такие функции и оперировать такими типами данных в программах, исполняемых в среде Windows, могут пользоваться этим API [5].

* **BASS audio lirary**

Звуковая[библиотека](https://ru.wikipedia.org/wiki/%D0%91%D0%B8%D0%B1%D0%BB%D0%B8%D0%BE%D1%82%D0%B5%D0%BA%D0%B0_(%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%B8%D1%80%D0%BE%D0%B2%D0%B0%D0%BD%D0%B8%D0%B5)), предназначена для использования в[операционных системах](https://ru.wikipedia.org/wiki/%D0%9E%D0%BF%D0%B5%D1%80%D0%B0%D1%86%D0%B8%D0%BE%D0%BD%D0%BD%D0%B0%D1%8F_%D1%81%D0%B8%D1%81%D1%82%D0%B5%D0%BC%D0%B0)[Windows](https://ru.wikipedia.org/wiki/Microsoft_Windows)и[macOS](https://ru.wikipedia.org/wiki/Mac_OS_X), также доступна для[Linux](https://ru.wikipedia.org/wiki/Linux), [Android](https://ru.wikipedia.org/wiki/Android)и[iOS](https://ru.wikipedia.org/wiki/IOS)-платформ. Цель библиотеки заключается в том, чтобы предоставить[разработчикам](https://ru.wikipedia.org/wiki/%D0%A0%D0%B0%D0%B7%D1%80%D0%B0%D0%B1%D0%BE%D1%82%D1%87%D0%B8%D0%BA_%D0%BF%D1%80%D0%BE%D0%B3%D1%80%D0%B0%D0%BC%D0%BC%D0%BD%D0%BE%D0%B3%D0%BE_%D0%BE%D0%B1%D0%B5%D1%81%D0%BF%D0%B5%D1%87%D0%B5%D0%BD%D0%B8%D1%8F)образец аудио-центра, в частности, обеспечить работу с потоками ([MP3](https://ru.wikipedia.org/wiki/MP3),[MP2](https://ru.wikipedia.org/wiki/MP2), [MP1](https://ru.wikipedia.org/wiki/MP1), [OGG](https://ru.wikipedia.org/wiki/OGG), [WAV](https://ru.wikipedia.org/wiki/WAV), [AIFF](https://ru.wikipedia.org/wiki/AIFF)), [MOD](https://ru.wikipedia.org/wiki/MOD)файлами ([XM](https://ru.wikipedia.org/wiki/XM),[IT](https://ru.wikipedia.org/wiki/IT), [S3M](https://ru.wikipedia.org/wiki/S3M), [MOD](https://ru.wikipedia.org/wiki/MOD), [MTM](https://ru.wikipedia.org/w/index.php?title=MTM&action=edit&redlink=1), [UMX](https://ru.wikipedia.org/w/index.php?title=UMX&action=edit&redlink=1)), [MO3](https://ru.wikipedia.org/wiki/MO3)файлами ([mp3](https://ru.wikipedia.org/wiki/Mp3),[ogg](https://ru.wikipedia.org/wiki/Ogg)и сжатых[MOD](https://ru.wikipedia.org/wiki/MOD)файлов), функциями записи, редактирования тегов аудиофайлов, конвертирования популярных форматов. Весь набор возможностей предоставляется в виде[dll-библиотеки](https://ru.wikipedia.org/wiki/DLL)размером около 110 килобайт [6].

* **Microsoft Visual Studio**

Интегрированнаясредаразработки (англ. IDE–IntegratedDevelopmentInvironment) –комплекспрограммныхсредств, используемый для разработки программного обеспечения (ПО).

MicrosoftVisualStudio–набор инструментов для создания ПО: от планирования до разработки пользовательского интерфейса, написания кода, тестирования, отладки, анализа качества кода и производительности. VisualStudioможно использовать для создания различных типов приложений: от простых приложений для магазина и игр для мобильных устройств до больших и сложных систем, обслуживающих предприятия и центры обработки данных. Основные виды программных средств, которые можно создавать с использованием VisualStudio:

* приложения и игры, которые выполняются не только на платформе Windows, но и на Androidи iOS;
* веб-сайты и веб-службы на основе ASP.NET, JQuery, AngularJSи других популярных платформ;
* приложения для самых разных устройств и платформ, включая

Office, Sharepoint, Hololens, Kinect, InternetofThings;

* игры и графические приложения для разных устройств Windows, включая Xbox, с поддержкой DirectX.

По умолчанию VisualStudioобеспечиваетподдержкуC#, C, C++, JavaScript, F#, VisualBasic. Хорошо интегрируется и работает со сторонними приложениями, например, Unityи ApacheCordova. Имеется также возможность самостоятельно расширять VisualStudioпутём создания плагинов для выполнения специализированных задач[7].

## Структура программного средства

Программное средство состоит из однооконногоприложения (см. Рисунок 5), описанного в файле BahRecorder.cpp.

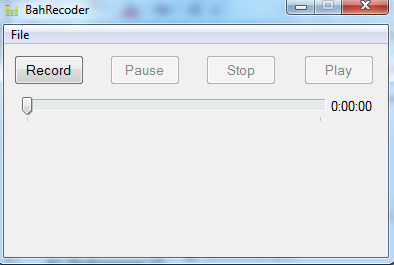
.

Рисунок 5 – Главное окно

### Запись аудиофайла

Для аудиозаписи создается буфер.

char \*recbuf = nullptr;

DWORD reclen;

При инициализации устройства данных буфер связывает с ним. Для начала непосредственно записи вызывается функция:

BASS\_RecordStart(FREQ, CHANS, 0, RecordingCallback, nullptr)

Для обработки собщений записывающей функции служит колбэкRecordingCallback, основная задача которого довыделение памяти при ее оканчании.

Осуществление записи начинается по нажатии кнопки “Record” (см. Рисунок 7).

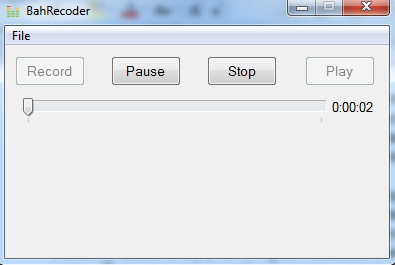
.

Рисунок 7 – Запись аудио

### Приостановка/возобновление проигрованиязаписи

Функция, реализованная в bass.dllуниверсальна как для канала воспроизведения, так и для канала записи, следовательно, при ее вызове достаточно определить идет запись или воспроизведени,для этого служит флаг:

charbass\_using;

следовательно, при 0 указывается, что идет запись, при 1 – воспроизведение. Зная какое уменно устройство используется, функции BASS\_ChannelPause() и BASS\_ChannelPlay() могут приостановить и возобновить поток соответственно. Приостановка воспроизведения записи происходит после нажатия кнопки “Pause” в главном окне приложения(см. Рисунок 8).

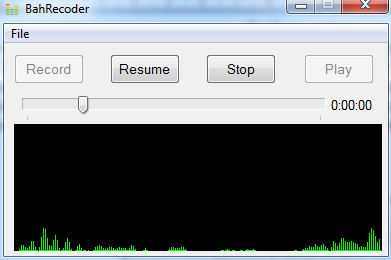


Рисунок 8 – Приостановка воспроизведения

Возобновление воспроизведения записи осуществляется по нажатию кнопки “Resume” (см. Рисунок 9).

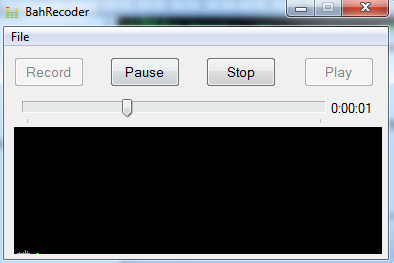


Рисунок 9 – Возобновление воспроизведения

### Сохраниние/открытие файла

Функции сохранения (см.Рисунок 10) и открытия (см. Рисунок 11) файла используют стандартное Windowsдиалоговое окно для открытия и сохранения файлов. Отметим, что с помощью данного окна программа получает имя целевого файла, сохранение и чтение реализуются отдельно.

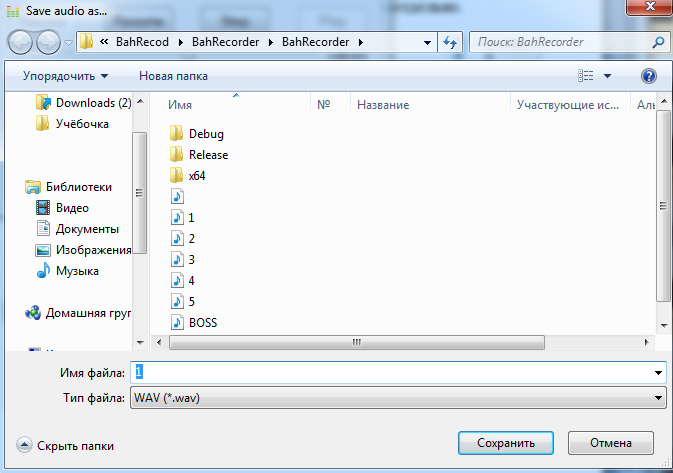


Рисунок 10 – Сохранение записи

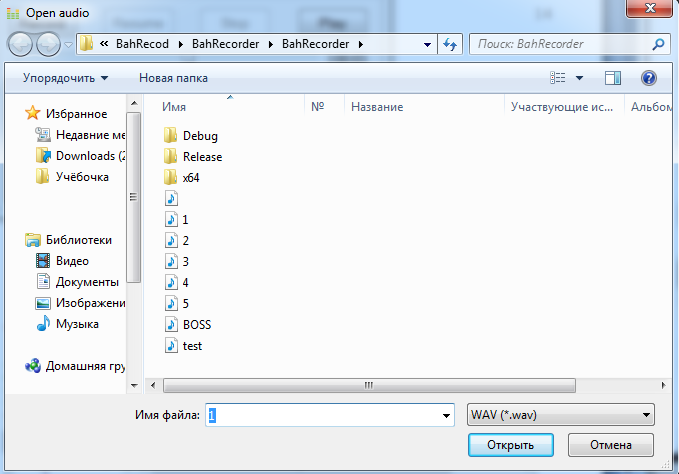


Рисунок 11 – Открытие записи

### Перемотка аудиозаписи

Вовремя воспроизведения аудиозаписи, при перетаскивании бегунка состояния воспроизведения, происходит перемещение в потоке вопроизведения с помощью функции BASS\_ChannelSetPosition().

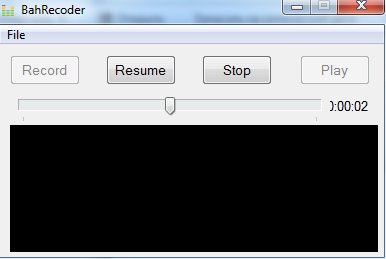


Рисунок 12 – Перемотка аудиозаписи



Рисунок 13 – Схема алгоритма записи аудио



Рисунок 14 – Схема алгоритма проигрывание макросов

Реализация всех возможностей приложения расположена в ряде классов, объекты которых используются функциями из файла WinMain.cpp. Каждый класс состоит из двух файлов: файл \*.hсодержит описание класса, файл \*.cppсодержит реализацию класса. Далее более подробно описаны все классы приложения.

* **Класс BahRecorder**

Реализует построение и отрисовку спектра текущего канала воспроизведения.

Таблица 1 – Методы класса BahRecorder

|  |  |  |  |
| --- | --- | --- | --- |
| №  п.п. | Наименование  метода | Назначение  метода | Параметры |
| 1 | Update Spectrum | Выводит различных спектров при воспроизведении | uTimerID (время проигрываемой записи);  uMsg(преслыаемые сообщения) |
| 2 | RecordingCallback | Удаление записываемой аудиозаписи, если в оперативной памяти недостаточно место | length(размер записи);  user(пользователь) |
| 3 | InitDevice | Проверка на присутствие записывающего устройства на компьютере | Отсутствуют |
| 4 | StartRecording | Выполнить начала записи, без записи на диск | Отсутствуют |
| 5 | StopRecording | Остановка записи | Отсутствуют |
| 6 | Play | Воспроизведение уже открытой записи или новой записи | Отсутствуют |
| 7 | GetFileName | Отрисовывает в заданном окне по заданным координатам пустой спектр | save(проверка на неопходимость сохранения файла) |
| 8 | WriteToDisk | Запись аудио на диск | Отсутствуют |
| 9 | OpenFile | Открытие файла | Отсутствуют |

# Тестирование и проверка работоспособности программного средства

Таблица 6 – Тест 1

|  |  |
| --- | --- |
| Тестовая  ситуация | Запуск записи при отсутствии микрофона |
| Исходный набор данных | Запуск приложения |
| Ожидаемый результат | Ошибка обращения по адресу |
| Фактический результат | 10 |

Таблица 7 – Тест 2

|  |  |
| --- | --- |
| Тестовая  ситуация | Попытка поспроизвести файл формата MP3 |
| Исходный набор данных | Открытие MP3 файл |
| Ожидаемый результат | Аварийное завершение |
| Фактический результат | Будут видны только WAV файлы |

Таблица 8 – Тест 3

|  |  |
| --- | --- |
| Тестовая  ситуация | Многократное прослушивание |
| Исходный набор данных | Прослушивание одной записи многократное количество раз |
| Ожидаемый результат | Переполнение памяти |
| Фактический результат | Память будет очищаться каждый раз, когда файл будет воспроизведён |

Таблица 9 – Тест 4

|  |  |
| --- | --- |
| Тестовая  ситуация | Выходи из программы при проигрывании |
| Исходный набор данных | Нажатие кнопки выход при воспроизведении |
| Ожидаемый результат | Некорректное завершение, ошибка звукового устройства |
| Фактический результат | Закрытие успешно |

Таблица 10 – Тест 5

|  |  |
| --- | --- |
| Тестовая  ситуация | Изменение положения воспроизведения вперед |
| Исходный набор данных | Передвинуть палзунок |
| Ожидаемый результат | Ползунок передвинут |
| Фактический результат |  |

Таблица 11 – Тест 6

|  |  |
| --- | --- |
| Тестовая  ситуация | Запуск приложения при отсутствующей библиотеке bass.dll с программой в одной папке |
| Исходный набор данных | Удаление библиотеки bass.dll |
| Ожидаемый результат | Приложение не запустится |
| Фактический результат | Приложение не запустится |

# Руководство по установке и использованию программного средства

Для работы с приложением нужно запуститьисполняемый файл программы`BahRecorder.exe`.Появитсяглавное окно программы (рисунок 5).

При незагруженном треке пользователю доступна функция записи, для ее выполнения нужно нажать кнопку **Record**(см. Рисунок 15).После ее нажатия начнется запись, об этом свидетельствует тикающий таймер.Также становятся доступны кнопки **Stop**и**Pause**(см. Рисунок 16).

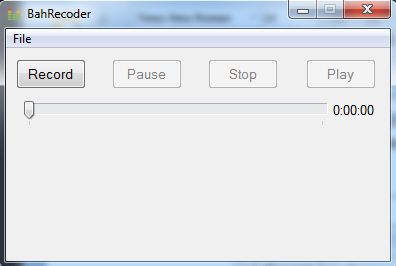


Рисунок 15 – Доступна только кнопка “Record”

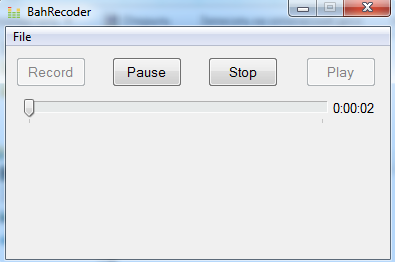


Рисунок 16 – Нажатие кнопки “Record”

Кнопка **Stop**остановить запись аудио и кнопка **Play**станут доступны. Кнопка **Pause**принажатии приостанавливает запись, делая ее недоступной для проигрывания, сменяя свою подпись на **Resume**,которая при нажатии возобновит запись аудиофайла (см. Рисунок 17).

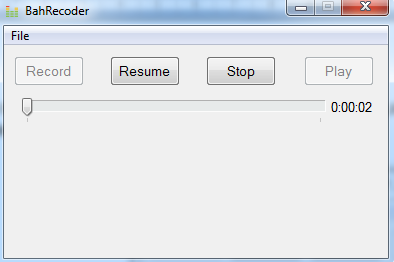


Рисунок 17 – Нажатиеи обновление кнопки “Pause”

При нажатии кнопки **Stop**(см. Рисунок 18)происходит остановка записи трекаи для его воспроизведения нужно нажать кнопку **Play**.

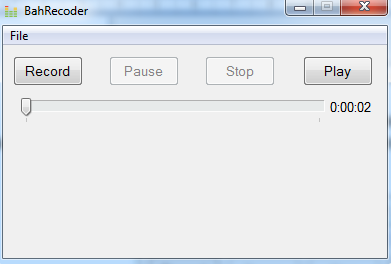


Рисунок 18 – Нажатие кнопки “Stop”

Во время воспроизведения аудиофайла его можно приостаноить/возобновить (**Pause**/**Resume**), остановить воспроизведение (**Stop**), перемещая бегунок перемотать трек на интересующую позицию. После остановки аудиофайла становиться доступна запись нового.

Также после окончания записи файла становиться доступно его сохранение и открытие другого файла. Для этого нужно нажать вкладку меню **File** (см. Рисунок 19) и выбрать необходимую функцию **OpenFile** или **SaveFile**соответственно. В данной вкладке меню доступна функция завершения программы **Exit**.

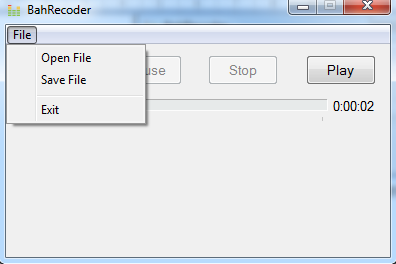


Рисунок 19 – Вкладка **File** главного меню

Заключение

Было разработано приложение, целью которого являлось создание легкого в использовании средства, которое работает с легковесной портируемой и широкоподдериваемой библиотекой bass.dll. Данная цель была достигнута.

Данное приложении всячески тестировалось, что в итоге привело к стабильной работе приложения во многих ситуациях.

Был более подробно изучен сам язык С++, а также работа с WindowsAPI.

Приложение можно в дальнейшем расширить, а именно, реализовать механизмы записи треков в различных сжимаемых форматах, добавить функцию записи, эквалайзер.

Списокиспользованных источников

[1] MSDN - обущающий портал Microsoft [Электронный ресурс]. Win32 application. Режимдоступа: https://msdn.microsoft/.

[2] Макс Шлее, Профессиональное программирование на С++ – М.: БХВ-Петербург, 2010. – 883 с.

[3] MicrosoftSupport[Электронныйресурс]. How to Increase the Maximum Recording Time in the Sound Recorder Utility – Режимдоступа: <http://support.microsoft.com/kb/82215>

[4] Федоренко Ю.П., Алгоритмы и программы на C++ Builder –М.: ДМК Пресс,2010. –544 с.

[5] Мартин Р. Чистый код: создание, анализ, рефакторинг. Библиотека программиста. – СПб.: Питер, 2010. – 464 с.: ил.

[6] Un4seen Developments [Электронныйресурс]. BASS.dll Documentation. Режимдоступа: http://www.un4seen.com/doc/#bass/bass.html.

ПриложениеА

(обязательное)

Текст программных модулей приложения

Содержимое файлаBahRecorder.cpp

// BahRecorder.cpp : Defines the entry point for the application.

//

#include "stdafx.h"

#include "BahRecorder.h"

#define MAX\_LOADSTRING 100

// Global Variables:

HINSTANCE hInst; // current instance

WCHAR szTitle[MAX\_LOADSTRING]; // The title bar text

WCHAR szWindowClass[MAX\_LOADSTRING]; // the main window class name

// Forward declarations of functions included in this code module:

ATOM MyRegisterClass(HINSTANCE hInstance);

BOOL InitInstance(HINSTANCE, int);

LRESULT CALLBACK WndProc(HWND, UINT, WPARAM, LPARAM);

INT\_PTR CALLBACK About(HWND, UINT, WPARAM, LPARAM);

BOOL InitDevice();

#define FREQ 44100

#define CHANS 2

#define BUFSTEP 200000

char \*recbuf = nullptr;

DWORD reclen;

HRECORD rchan = 0;

HSTREAM chan = 0;

voidStartRecording();

voidStopRecording();

voidWriteToDisk();

voidOpenFile();

void Play();

HWND hWnd;

HWND bRecord;

HWND bStop;

HWND bPlay;

HWND bPause;

HWND hwndTrackBar;

HWND lTime;

HWND lFileName;

int Time;

intrecodingTime;

charbass\_using = 0;

QWORD len;

QWORD time;

int APIENTRY wWinMain(\_In\_ HINSTANCE hInstance,

\_In\_opt\_ HINSTANCE hPrevInstance,

\_In\_ LPWSTR lpCmdLine,

\_In\_ intnCmdShow)

{

UNREFERENCED\_PARAMETER(hPrevInstance);

UNREFERENCED\_PARAMETER(lpCmdLine);

// Initialize global strings

LoadString(hInstance, IDS\_APP\_TITLE, (LPSTR)szTitle, MAX\_LOADSTRING);

LoadStringW(hInstance, IDC\_BAHRECORDER, szWindowClass, MAX\_LOADSTRING);

MyRegisterClass(hInstance);

// Perform application initialization:

if (!InitInstance (hInstance, nCmdShow))

{

return FALSE;

}

HACCEL hAccelTable = LoadAccelerators(hInstance, MAKEINTRESOURCE(IDC\_BAHRECORDER));

MSG msg;

// Main message loop:

while (GetMessage(&msg, nullptr, 0, 0))

{

if (!TranslateAccelerator(msg.hwnd, hAccelTable, &msg))

{

TranslateMessage(&msg);

DispatchMessage(&msg);

}

}

return (int) msg.wParam;

}

//

// FUNCTION: MyRegisterClass()

//

// PURPOSE: Registers the window class.

//

ATOM MyRegisterClass(HINSTANCE hInstance)

{

WNDCLASSEXW wcex;

wcex.cbSize = sizeof(WNDCLASSEX);

HBRUSH hb= ::CreateSolidBrush(RGB(240, 240, 240));

wcex.style = CS\_HREDRAW | CS\_VREDRAW;

wcex.lpfnWndProc = WndProc;

wcex.cbClsExtra = 0;

wcex.cbWndExtra = 0;

wcex.hInstance = hInstance;

wcex.hIcon = LoadIcon(hInstance, MAKEINTRESOURCE(MAIN\_ICO));

wcex.hCursor = LoadCursor(nullptr, IDC\_ARROW);

wcex.hbrBackground =hb;

wcex.lpszMenuName = MAKEINTRESOURCEW(IDC\_BAHRECORDER);

wcex.lpszClassName =szWindowClass;

wcex.hIconSm = LoadIcon(wcex.hInstance, MAKEINTRESOURCE(SMALL\_ICO));

returnRegisterClassExW(&wcex);

}

//

// FUNCTION: InitInstance(HINSTANCE, int)

//

// PURPOSE: Saves instance handle and creates main window

//

// COMMENTS:

//

// In this function, we save the instance handle in a global variable and

// create and display the main program window.

//

BOOL InitInstance(HINSTANCE hInstance, intnCmdShow)

{

hInst = hInstance; // Store instance handle in our global variable

INITCOMMONCONTROLSEX init = { sizeof(INITCOMMONCONTROLSEX), ICC\_WIN95\_CLASSES };

InitCommonControlsEx(&init);

hWnd = CreateWindowW(szWindowClass, szTitle, WS\_OVERLAPPED | WS\_CAPTION | WS\_SYSMENU | WS\_MINIMIZEBOX,

CW\_USEDEFAULT, 0, 345, 170, nullptr, nullptr, hInstance, nullptr);

bRecord = CreateWindow("BUTTON", "Record", WS\_CHILD | WS\_VISIBLE | BS\_PUSHBUTTON, 10, 10, 70, 30, hWnd, (HMENU)IDB\_RECORD, hInstance, 0);

bPause = CreateWindow("BUTTON", "Pause", WS\_CHILD | WS\_VISIBLE | BS\_PUSHBUTTON, 90, 10, 70, 30, hWnd, (HMENU)IDB\_PAUSE, hInstance, 0);

bStop = CreateWindow("BUTTON", "Stop", WS\_CHILD | WS\_VISIBLE | BS\_PUSHBUTTON, 170, 10, 70, 30, hWnd, (HMENU)IDB\_STOP, hInstance, 0);

bPlay = CreateWindow("BUTTON", "Play", WS\_CHILD | WS\_VISIBLE | BS\_PUSHBUTTON, 250, 10, 70, 30, hWnd, (HMENU)IDB\_PLAY, hInstance, 0);

hwndTrackBar = CreateWindowEx(0, TRACKBAR\_CLASS, "Trackbar Control", WS\_CHILD | WS\_VISIBLE | TBS\_ENABLESELRANGE, 10, 50, 265, 30, hWnd, (HMENU)IDT\_TRACKBAR, hInst, NULL);

lTime = CreateWindow("static", "0:00:00",

WS\_CHILD | WS\_VISIBLE | WS\_TABSTOP,

276, 53, 43, 16,

hWnd, (HMENU)IDL\_TIME,

hInstance, NULL);

lFileName = CreateWindow("static", "File name",

WS\_CHILD | WS\_VISIBLE | WS\_TABSTOP,

10, 80, 325, 16,

hWnd, (HMENU)IDL\_FILENAME,

hInstance, NULL);

HFONT font = CreateFont(16, 0, 0, 0, FW\_REGULAR,

false, false, false, ANSI\_CHARSET,

OUT\_DEFAULT\_PRECIS, CLIP\_DEFAULT\_PRECIS,

CLEARTYPE\_QUALITY, FF\_SWISS,

"MS Shell Dlg");

SendMessage(bRecord, WM\_SETFONT, WPARAM(font), 0);

SendMessage(bStop, WM\_SETFONT, WPARAM(font), 0);

SendMessage(bPlay, WM\_SETFONT, WPARAM(font), 0);

SendMessage(bPause, WM\_SETFONT, WPARAM(font), 0);

SendMessage(lTime, WM\_SETFONT, WPARAM(font), 0);

SendMessage(lFileName, WM\_SETFONT, WPARAM(font), 0);

EnableWindow(bStop, false);

EnableWindow(bPlay, false);

EnableWindow(bPause, false);

if ((!BASS\_RecordInit(-1)))

{

BASS\_RecordFree();

BASS\_Free();

MessageBox(NULL, "Error", "Recoding devise don't initialized!", 0);

}

if (!hWnd)

{

return FALSE;

}

ShowWindow(hWnd, nCmdShow);

UpdateWindow(hWnd);

return TRUE;

}

//

// FUNCTION: WndProc(HWND, UINT, WPARAM, LPARAM)

//

// PURPOSE: Processes messages for the main window.

//

// WM\_COMMAND - process the application menu

// WM\_PAINT - Paint the main window

// WM\_DESTROY - post a quit message and return

//

//

LRESULT CALLBACK WndProc(HWND hWnd, UINT message, WPARAM wParam, LPARAM lParam)

{

switch (message)

{

case WM\_COMMAND:

{

intwmId = LOWORD(wParam);

// Parse the menu selections:

switch (wmId)

{

case IDM\_OPEN:

EnableWindow(bStop, false);

EnableWindow(bPlay, true);

EnableWindow(bPause, false);

EnableWindow(bRecord, true);

OpenFile();

break;

case IDM\_SAVE:

WriteToDisk();

break;

case IDM\_ABOUT:

DialogBox(hInst, MAKEINTRESOURCE(IDD\_ABOUTBOX), hWnd, About);

break;

case IDM\_EXIT:

DestroyWindow(hWnd);

break;

case IDB\_PAUSE:

if (bass\_using == 0)

{

if (GetWindowTextLength((HWND)bPause) == 5) // when Pause pressed

{

SetWindowText((HWND)bPause, "Resume");

KillTimer(hWnd, 300);

BASS\_ChannelPause(rchan);

}

else //when Resume pressed

{

BASS\_ChannelPlay(rchan, false);

SetTimer(hWnd, 300, 1000, (TIMERPROC)NULL);

SetWindowText((HWND)bPause, "Pause");

}

}

else

{

if (GetWindowTextLength((HWND)bPause) == 5) // when Pause pressed

{

SetWindowText((HWND)bPause, "Resume");

KillTimer(hWnd, 300);

BASS\_ChannelPause(chan);

}

else //when Resume pressed

{

BASS\_ChannelPlay(chan, false);

SetTimer(hWnd, 300, 1000, (TIMERPROC)NULL);

SetWindowText((HWND)bPause, "Pause");

}

}

break;

case IDB\_RECORD:

if (InitDevice())

{

bass\_using = 0;

EnableWindow(bStop, true);

EnableWindow(bPlay, false);

EnableWindow(bPause, true);

EnableWindow(bRecord, false);

StartRecording();

recodingTime = 0;

SetTimer(hWnd, 300, 1000, (TIMERPROC)NULL);

SetWindowText(lFileName, "File name");

}

break;

case IDB\_STOP:

EnableWindow(bStop, false);

EnableWindow(bPlay, true);

EnableWindow(bPause, false);

EnableWindow(bRecord, true);

SendMessage(hwndTrackBar, TBM\_SETPOS, (WPARAM)TRUE, 0);

if (bass\_using == 0)

{

StopRecording();

KillTimer(hWnd, 300);

}

else

{

BASS\_ChannelStop(chan);

KillTimer(hWnd, 300);

}

StopRecording();

break;

case IDB\_PLAY:

bass\_using = 1;

EnableWindow(bStop, true);

EnableWindow(bPlay, false);

EnableWindow(bPause, true);

EnableWindow(bRecord, false);

Play();

Time = 0;

SetTimer(hWnd, 300, 1000, (TIMERPROC)NULL);

if (bass\_using == 0)

{

if (!BASS\_ChannelIsActive(rchan))

KillTimer(hWnd, 300);

}

else

{

if (!BASS\_ChannelIsActive(chan))

KillTimer(hWnd, 300);

}

break;

default:

returnDefWindowProc(hWnd, message, wParam, lParam);

}

}

break;

case WM\_TIMER:

switch (wParam)

{

case 300:

recodingTime++;

if (bass\_using == 1)

{

time = BASS\_ChannelGetPosition(chan, BASS\_POS\_BYTE);

Time = (int)BASS\_ChannelBytes2Seconds(chan, time);

SendMessage(hwndTrackBar, TBM\_SETPOS, (WPARAM)TRUE, (int)((double)time / len \* 100));

if (!BASS\_ChannelIsActive(chan))

{

SendMessage(hwndTrackBar, TBM\_SETPOS, (WPARAM)TRUE, 0);

KillTimer(hWnd, 300);

EnableWindow(bRecord, true);

EnableWindow(bPlay, true);

EnableWindow(bStop, false);

EnableWindow(bPause, false);

}

}

else

{

Time = recodingTime;

}

charformTime[8];

formTime[0] = char(48 + Time / 3600);

formTime[1] = ':';

formTime[2] = char(48 + Time % 3600 / 60 / 10);

formTime[3] = char(48 + Time % 3600 / 60 % 10);

formTime[4] = ':';

formTime[5] = char(48 + Time % 3600 % 60 / 10);

formTime[6] = char(48 + Time % 3600 % 60 % 10);

formTime[7] = '\0';

SetWindowText(lTime, formTime);

break;

}

case WM\_HSCROLL:

{

if (LOWORD(wParam) != TB\_THUMBPOSITION)

{

break;

}

inttrackBarPosition = SendDlgItemMessage(hWnd, IDT\_TRACKBAR, TBM\_GETPOS, 0,0);

QWORD position = (double)trackBarPosition / 100 \* len;

BASS\_ChannelSetPosition(chan, position, BASS\_POS\_BYTE);

}

break;

case WM\_PAINT:

{

PAINTSTRUCT ps;

HDC hdc = BeginPaint(hWnd, &ps);

EndPaint(hWnd, &ps);

}

break;

case WM\_DESTROY:

BASS\_RecordFree();

BASS\_Free();

PostQuitMessage(0);

break;

default:

returnDefWindowProc(hWnd, message, wParam, lParam);

}

return 0;

}

// Message handler for about box.

INT\_PTR CALLBACK About(HWND hDlg, UINT message, WPARAM wParam, LPARAM lParam)

{

UNREFERENCED\_PARAMETER(lParam);

switch (message)

{

case WM\_INITDIALOG:

return (INT\_PTR)TRUE;

case WM\_COMMAND:

if (LOWORD(wParam) == IDOK || LOWORD(wParam) == IDCANCEL)

{

EndDialog(hDlg, LOWORD(wParam));

return (INT\_PTR)TRUE;

}

break;

}

return (INT\_PTR)FALSE;

}

BOOL CALLBACK RecordingCallback(HRECORD handle, const void \*buffer, DWORD length, void \*user)

{

if ((reclen%BUFSTEP) + length >= BUFSTEP) {

recbuf = (char\*)realloc(recbuf, ((reclen + length) / BUFSTEP + 1)\*BUFSTEP);

if (!recbuf) {

rchan = 0;

MessageBox(NULL, "Error", "Out of memmory!",0);

return false;

}

}

memcpy(recbuf + reclen, buffer, length);

reclen += length;

return true;

}

BOOL InitDevice()

{

BASS\_RecordFree();

if (!BASS\_RecordInit(-1)) {

MessageBox(NULL, "Error", "Recording devise don't initialised!", 0);

return false;

}

else

return true;

}

voidStartRecording()

{

WAVEFORMATEX \*wf;

if (recbuf)

{

BASS\_StreamFree(chan);

chan = 0;

free(recbuf);

recbuf = nullptr;

BASS\_Free();

}

recbuf = (char\*)malloc(BUFSTEP);

reclen = 44;

memcpy(recbuf, "RIFF\0\0\0\0WAVEfmt \20\0\0\0", 20);

memcpy(recbuf + 36, "data\0\0\0\0", 8);

wf = (WAVEFORMATEX\*)(recbuf + 20);

wf->wFormatTag = 1;

wf->nChannels = CHANS;

wf->wBitsPerSample = 16;

wf->nSamplesPerSec = FREQ;

wf->nBlockAlign = wf->nChannels\*wf->wBitsPerSample / 8;

wf->nAvgBytesPerSec = wf->nSamplesPerSec\*wf->nBlockAlign;

rchan = BASS\_RecordStart(FREQ, CHANS, 0, RecordingCallback, nullptr);

if (!rchan) {

free(recbuf);

MessageBox(NULL, "Error", "Recoding hash't done!", 0);

recbuf = 0;

}

}

voidStopRecording()

{

BASS\_ChannelStop(rchan);

rchan = 0;

\*(DWORD\*)(recbuf + 4) = reclen - 8;

\*(DWORD\*)(recbuf + 40) = reclen - 44;

}

void Play()

{

FILE \*fp;

charszTempName[MAX\_PATH];

tmpnam\_s(szTempName, MAX\_PATH);

fopen\_s(&fp,szTempName, "wb");

fwrite(recbuf, reclen, 1, fp);

fclose(fp);

BASS\_Init(-1, 44100, BASS\_DEVICE\_3D, 0, NULL);

chan = BASS\_StreamCreateFile(FALSE, szTempName, 0, 0, 0);

len = BASS\_ChannelGetLength(chan, BASS\_POS\_BYTE);

BASS\_ChannelPlay(chan, TRUE);

remove(szTempName);

}

char\* GetFileName(bool save) {

static OPENFILENAME ofn;

static char fullpath[255], filename[256], dir[256];

ofn.lStructSize = sizeof(OPENFILENAME);

ofn.hwndOwner = hWnd;

ofn.hInstance = hInst;

ofn.lpstrFilter = "WAV (\*.wav)\0\*.wav\0";

ofn.nFilterIndex = 1;

ofn.lpstrFile = fullpath;

ofn.nMaxFile = sizeof(fullpath);

ofn.lpstrFileTitle = filename;

ofn.nMaxFileTitle = sizeof(filename);

ofn.lpstrInitialDir = dir;

ofn.Flags = OFN\_PATHMUSTEXIST | OFN\_OVERWRITEPROMPT | OFN\_HIDEREADONLY | OFN\_EXPLORER;

if (save) {

ofn.lpstrTitle = "Save audio as...";

GetSaveFileName(&ofn);

}

else {

ofn.lpstrTitle = "Open audio";

GetOpenFileName(&ofn);

}

returnfullpath;

}

voidWriteToDisk()

{

FILE \*fp;

char\* file = GetFileName(true);

if (!strchr(file,'.'))

{

strcat(file, ".wav");

}

fp = fopen(file, "wb");

fwrite(recbuf, reclen, 1, fp);

fclose(fp);

PathStripPath(file);

SetWindowText(lFileName, file);

}

voidOpenFile()

{

FILE \* fp;

free(recbuf);

char\* file = GetFileName(false);

fp = fopen(file, "rb");

fseek(fp, 0, SEEK\_END);

reclen = ftell(fp);

rewind(fp);

recbuf = (char\*)malloc(sizeof(char)\*reclen);

fread(recbuf, 1, reclen, fp);

fclose(fp);

PathStripPath(file);

SetWindowText(lFileName, file);

SetWindowText(lTime, "0:00:00");

}

1. Содержимое файлаResource.h

//{{NO\_DEPENDENCIES}}

// Включаемый файл, созданный в MicrosoftVisualC++.

// ИспользуетсяBahRecorder.rc

//

#define IDC\_MYICON 2

#define IDD\_BAHRECORDER\_DIALOG 102

#define IDS\_APP\_TITLE 107

#define IDD\_ABOUTBOX 103

#define IDM\_ABOUT 104

#define IDM\_EXIT 105

#define IDC\_BAHRECORDER 109

#define IDB\_RECORD 110

#define IDB\_PAUSE 111

#define IDB\_STOP 112

#define IDB\_PLAY 113

#define IDT\_TRACKBAR 114

#define IDL\_TIME 115

#define IDL\_FILENAME 116

#define IDR\_MAINFRAME 128

#define SMALL\_ICO 135

#define MAIN\_ICO 136

#define ID\_FILE\_EXIT 32771

#define IDM\_OPEN 32773

#define IDM\_SAVE 32774

#define IDC\_STATIC -1

// Next default values for new objects

//

#ifdef APSTUDIO\_INVOKED

#ifndef APSTUDIO\_READONLY\_SYMBOLS

#define \_APS\_NO\_MFC 1

#define \_APS\_NEXT\_RESOURCE\_VALUE 138

#define \_APS\_NEXT\_COMMAND\_VALUE 32775

#define \_APS\_NEXT\_CONTROL\_VALUE 1000

#define \_APS\_NEXT\_SYMED\_VALUE 110

#endif

#endif

1. Содержимое файла BahRecorder.h

#pragma once

#include "resource.h"

#include "bass.h"

#include <cstdio>

#include <commctrl.h>

#include <Commdlg.h>

#include <Shlobj.h>

#include <minwinbase.h>

#include <tchar.h>

#include <Shlwapi.h>

#pragma comment(lib,"Comctl32.lib")

#pragma comment(lib,"Shlwapi.lib")

#pragma comment(linker,"\"/manifestdependency:type='win32' \

name='Microsoft.Windows.Common-Controls' version='6.0.0.0' \

processorArchitecture='\*' publicKeyToken='6595b64144ccf1df' language='\*'\"")

#pragma comment(lib,"bass.lib")

ВЕДОМОСТЬДОКУМЕНТОВ

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Обозначение | | | | | Наименование | | | | Дополнитель-ные сведения | | |
|  | | | | | Текстовые документы | | | |  | | |
|  | | | | |  | | | |  | | |
| БГУИР КП 1–40 01 01 15 ПЗ | | | | | Пояснительная записка | | | | 34с. | | |
|  | | | | |  | | | |  | | |
|  | | | | | Графические документы | | | |  | | |
|  | | | | |  | | | |  | | |
| ГУИР.851004-01 СА | | | | | Схема алгоритма проигрования макросов | | | | Формат А1 | | |
|  | | | | | Схема алгоритма | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  | | | | |  | | | |  | | |
|  |  |  |  |  | БГУИР КП 1–40 01 01 15 ПЗ | | | | | | |
|  |  |  |  |  |
|  |  |  |  |  |
| Измммм | Лист | № докум. | Подп | Дата | Программное средство аудио записи  “BahRecorder” | Литера | | | | Лист | Листов |
| Разраб. | | Борисюк Ф.А. |  | 21.12 | Т |  |  | | 35 | 35 |
| Пров. | | Шульга Е.С. |  | 21.12 | Кафедра ПОИТ  гр. 851004 | | | | | |
|  | |  |  |  |
|  | |  |  |  |
|  | |  |  |  |