Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»

Факультет компьютерных систем и сетей
Кафедра электронных вычислительных машин
Дисциплина: Программирование на языках высокого уровня

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА
к курсовой работе
на тему
«Видеоплеер»

БГУИР КР 1-40 02 01 310 ПЗ

Студент К. А. Каражан
Руководитель Е. В.Богдан

МИНСК 2023

Учреждение образования
«Белорусский государственный университет
информатики и радиоэлектроники»

Факультет компьютерных систем и сетей

УТВЕРЖДАЮ
Заведующий кафедрой ЭВМ

(подпись)
2023 г.

ЗАДАНИЕ
по курсовому проектированию

Студенту *Каражан Ксении Александровне*

1. Тема проекта *«Видеоплеер»*
2. Срок сдачи студентом законченного проекта *11 декабря 2023 г.*
3. Исходные данные к проекту: картинки jpg для иконок приложения в папке icon
4. Содержание расчётно-пояснительной записки (перечень вопросов, которые подлежат разработке)
   1. Введение.
   2. Задание.
   3. Обзор литературы.
   3.1. Обзор методов и алгоритмов решения поставленной задачи.
   4.Функциональное проектирование.
   4.1. Структура входных и выходных данных.
   4.2. Разработка диаграммы классов.
   4.3. Описание классов.
   5.Разработка программных модулей.
   5.1. Разработка схем алгоритмов (два наиболее важных метода).
   5.2. Разработка алгоритмов (описание алгоритмов по шагам для двух методов).
   6.Результаты работы.
   7.Заключение
   8.Литература
   9.Приложения
5. Перечень графического материала (с точным обозначением обязательных чертежей и графиков)
   1. *Диаграмма классов.*
   2. *Схема алгоритма loadPlaylist() .*
   3. *Схема алгоритма openFolderAndAddToPlaylist().*
6. Консультант по проекту (с обозначением разделов проекта) Е.В.Богдан

7. Дата выдачи задания 15 *сентября 2023 г.*

8. Календарный график работы над проектом на весь период проектирования (с обозначением сроков выполнения и трудоёмкости отдельных этапов):

1. Выбор задания. Разработка содержания пояснительной записки. Перечень графического материала – 15 %;
разделы 2, 3 – 10 %;
разделы 4 к – 20 %;
разделы 5 к – 35 %;
раздел 6,7,8 – 5 %;
раздел 9 к – 5%;
оформление пояснительной записки и графического материала к 11.12.23 – 10 %
Защита курсового проекта с 21.12 по 28.12.23г.

РУКОВОДИТЕЛЬ Е.В.Богдан

Задание принял к исполнению

К.А Каражан

(дата и подпись студента)

СОДЕРЖАНИЕ

ВВЕДЕНИЕ.....5
1 ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ.....6
2. ОБЗОР ЛИТЕРАТУРЫ.....8
2.1 Обзор методов и алгоритвом решения поставленной задачи.....15
3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВНИЕ.....17
3.1 Структура входных и выходных данных.....17
3.2 Разработка диаграммы классов.....17
3.3 Описание классов.....17
4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ.....21
4.1 Разработка схем алгоритмов.....21
4.2 Разработка алгоритмов.....21
5 РЕЗУЛЬТАТ РАБОТЫ.....23
ЗАКЛЮЧЕНИЕ.....31
СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ.....32
ПРИЛОЖЕНИЕ А.....33
ПРИЛОЖЕНИЕ Б.....34
ПРИЛОЖЕНИЕ В.....35
ПРИЛОЖЕНИЕ Г.....36
ПРИЛОЖЕНИЕ Д.....37

ВВЕДЕНИЕ

Изучение принципов объектно-ориентированного программирования (ООП) предоставляет разработчикам эффективные инструменты для организации кода и разработки масштабируемых приложений. Ознакомление с принципами позволяет разработчикам расширить свои навыки, обеспечивая более интуитивный и гибкий подход к созданию программного обеспечения.

Научное внимание текущего курсового проекта направлено на разработку видеоплеера на языке программирования C++ с использованием фреймворка Qt, с акцентом на основах объектно-ориентированного программирования. Современные технологии разработки на C++ включают использование стандартных библиотек, инструментов разработки и передовых методов программирования, с учётом применения объектно-ориентированного подхода для создания систем, обладающих модульностью и легкочитаемостью.

В современное время, где требования к программному обеспечению непрерывно растут, объектно-ориентированное программирование (ООП) становится неотъемлемой частью подготовки профессиональных разработчиков.

Объектно-ориентированное программирование (ООП), позволяющее создавать программы, основанные на объектах, а не на процедурах. Это позволяет создавать более гибкие и масштабируемые Template Library), которые предоставляют широкий набор готовых компонентов для решения различных задач.

Использование современных инструментов разработки, таких как среды разработки (IDE), отладчики и компиляторы, которые позволяют ускорить процесс разработки и улучшить качество кода.

Видеоплееры играют важную роль в современном мире, предоставляя пользователям возможность просматривать и воспроизводить контент. Они имеют множество преимуществ и важности, которые делают их неотъемлемой частью нашего повседневного использования. Несколько ключевых важных особенностей:

Поддержка различных форматов видео: Видеоплееры могут поддерживать множество форматов видео, таких как TS, MTS, M2TS, MXF, MP4, M4V, MOV, MPG, MPEG, AVI, FLV, RMVB, WMV, ASF, MKV, SWF, VOB, OGM, WTV, WebM и другие.

Высокое качество видео: Видеоплееры поддерживают воспроизведение видео высокого разрешения, такого как Full HD, 4K, 5K и даже 8K.

Управление воспроизведением: Видеоплееры предоставляют пользователям возможность управлять воспроизведением видео, включая перемотку вперёд и назад, переключение на полноэкранный режим, выбор звуковой дорожки и другие функции.

Совместимость с различными операционными системами:

Видеоплееры обычно совместимы с различными операционными системами, такими как Windows, Mac и другие. Это позволяет пользователям использовать видеоплееры на любом устройстве, независимо от его операционной системы.

Бесплатность или доступная цена: Многие видеоплееры доступны бесплатно или имеют доступную цену, что делает их доступными для широкого круга пользователей.

Расширенные функции: Некоторые видеоплееры, такие как PotPlayer и VLC Player, предлагают расширенные функции, такие как синхронизация субтитров, фильтры видео/аудио, изменение скорости воспроизведения и другие.

Экономичность: Большинство видеоплееров бесплатны или имеют доступную цену, что делает их доступными для широкого круга пользователей. Некоторые из них предлагают дополнительные функции, такие как блокировка рекламы, за которую они взимают плату 5.

Цель настоящего курсового проекта заключается в исследовании особенностей объектно-ориентированного программирования, в том числе анализ современных технологий и их применение при создании видеоплеера с высоким уровнем абстракции и гибкости.

Результатом проекта будет создание функционального видеоплеера, предоставляющего пользователю возможность взаимодействия с мультимедийным контентом и обладающего интуитивным интерфейсом для удобного управления функциональностью.

1. ЗАДАНИЕ НА КУРСОВУЮ РАБОТУ

Овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня C++, овладеть практическими навыками проектирования и разработки законченного, отлаженного и протестированного программного продукта с использованием языка высокого уровня C++[1]. Разработать видеоплеер с использование среды разработки Qt. Понимание основных принципов ООП:

При разработке используется Qt, кроссплатформенный фреймворк для разработки приложений на C++. Он предоставляет разработчикам множество инструментов и библиотек для создания графического интерфейса пользователя, работы с сетью, базами данных, мультимедиа и многого другого.

Создание удобного интерфейса:

- Разработать привлекательный и интуитивно понятный интерфейс для пользователя.
- Обеспечить простоту использования основных функций (воспроизведение, пауза, перемотка и т. д.).

Поддержка различных форматов видео:

- Обеспечить воспроизведение видео в различных форматах (MP4, AVI, MKV и т. д.).

Масштабируемость и адаптивность:

- Обеспечить масштабируемость для поддержки различных разрешений видео.

Контроль за воспроизведением:

- Добавить функциональность управления скоростью воспроизведения.
- Реализовать возможность выбора конкретного времени в видео.

Плейлисты:

- Возможность создать/сохранить плейлист;
- Возможность включить видео непосредственно из плейлиста.

2. ОБЗОР ЛИТЕРАТУРЫ

Перед началом разработки следовало ознакомиться с основными концепциями и инструментами Qt C++, такими как сигналы и слоты, система событий, элементы управления интерфейса пользователя (QWidget, QLabel, QPushButton и т. д.), а также основами работы с мультимедийными данными.

Ключевым этапом является изучение классов, ответственных за мультимедийное воспроизведение, таких как QMediaPlayer. Необходимо углублённое понимание основных функций этих классов, возможностей настройки воспроизведения, обработки событий и управления медиаресурсами.

Важным аспектом является анализ документации по работе с видео- и аудиокодеками с целью обеспечения поддержки требуемых форматов. Кроме того, рекомендуется ознакомиться с образцами кода, демонстрирующими создание пользовательского интерфейса видеоплеера с использованием Qt, для усвоения эффективных методов интеграции компонентов и обеспечения удобства использования приложения.

Документация Qt:

Getting Started with Qt: Раздел "Getting Started" в документации предоставляется информация о том, как установить Qt, настроить среду разработки и создать простое приложение.

Qt Multimedia Example: Примеры видео- и аудиоплееров.

Overview: Введение в фреймворк, его основные концепции и принципы. Создание графического интерфейса: Qt Widgets[9]: Информация о виджетах Qt, базовых элементах управления, таких как кнопки, поля ввода и другие.

Примеры кода и Учебные проекты[7]: Qt Examples: Обширный набор примеров кода для различных компонентов Qt.Qt Tutorials: Учебные проекты и туториалы, позволяющие освоить различные аспекты фреймворка.

Форумы и Сообщества: Qt Forum: Онлайн-форумы, где разработчики могут задавать вопросы, делиться опытом и получать поддержку от сообщества Qt.

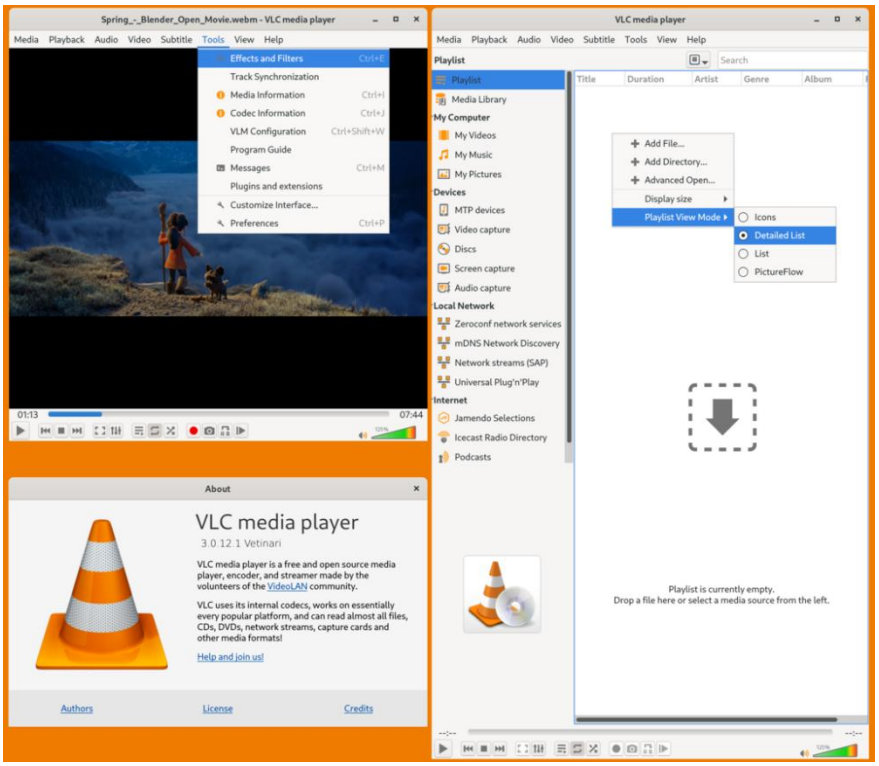
Для разработки и оценки инструментов приложения необходимо изучить другие видеоплееры. Популярными считаются VLC Media Player, Windows Media Player, Media Player Classic HC, KMPlayer.

Стоит рассмотреть каждый с плюсами и минусами, дабы определить необходимые аспекты.

Плееры будут рассмотрены в соответствии с перечислением.

VLC Media Player был разработан проектом VideoLAN и имеет дружественное сообщество разработчиков по всему миру.

Интерфейс приложения:



Ниже приведены плюсы и минусы

Плюсы:

- Поддержка множества форматов: VLC способен воспроизводить большинство распространенных аудио- и видеоформатов, а также потоковое воспроизведение мультимедийных файлов;
- Открытый исходный код: Проект VLC является свободным программным обеспечением с открытым исходным кодом. Это позволяет пользователям не только бесплатно использовать программу, но и изменять ее по своему усмотрению;
- Кроссплатформенность: VLC поддерживает различные операционные системы, что делает его удобным для использования на компьютерах с разными ОС. Пользователи могут легко переносить свои настройки и списки воспроизведения между разными устройствами;
- Простой пользовательский интерфейс: VLC предоставляет простой и интуитивно понятный интерфейс, что делает его пригодным для использования как опытными пользователями, так и новичками;
- Широкие возможности настройки: Пользователи могут настраивать VLC под свои потребности, регулировать параметры видео и аудио, изменять внешний вид интерфейса и многое другое;
- Поддержка субтитров и многоязычность: VLC обеспечивает поддержку субтитров, а также может воспроизводить мультимедийные файлы с различными аудио- и субтитральными дорожками;

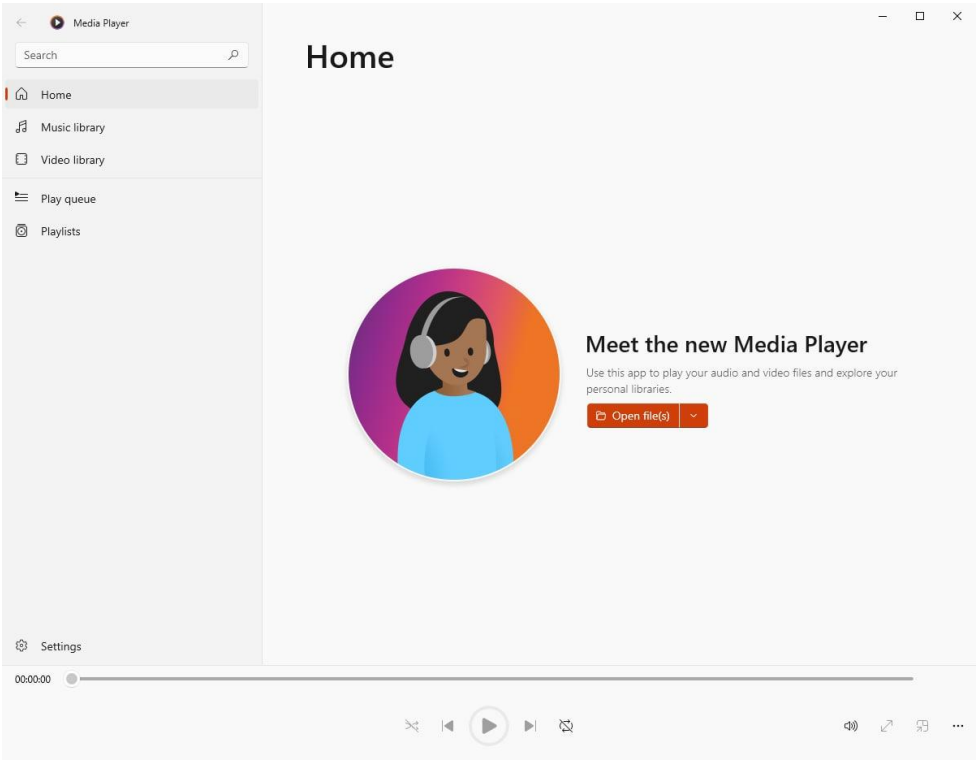
- Расширенные функции потокового воспроизведения: VLC способен воспроизводить потоковое видео и аудио из сети, поддерживая протоколы, такие как HTTP, RTP, RTSP, MMS.

Минусы:

- Интерфейс: Для некоторых пользователей интерфейс VLC может показаться менее привлекательным и современным по сравнению с некоторыми другими медиаплеерами. Это вопрос вкуса, но для тех, кто ценит стильные и эстетичные интерфейсы, VLC может показаться менее привлекательным.
- Сложность настроек: Возможности настройки VLC могут показаться сложными для некоторых пользователей, особенно для тех, кто не знаком с продвинутыми параметрами воспроизведения и потоковой передачи. Для простого использования может потребоваться время для изучения различных функций.
- Потребление ресурсов: В некоторых случаях VLC может потреблять больше системных ресурсов, особенно при воспроизведении высококачественных видеофайлов. Это может сказаться на производительности компьютера, особенно если у вас старое или менее мощное устройство.
- Не всегда стабильный плеер: Некоторые пользователи сообщают о случайных сбоях и неполадках при использовании VLC, таких как зависания или вылеты программы. Вопреки широкой поддержке и обновлениям, неисправности могут встречаться.
- Отсутствие облачной интеграции: По сравнению с некоторыми современными медиаплеерами, VLC может иметь ограниченные возможности интеграции с облачными сервисами для потокового воспроизведения и синхронизации мультимедийных библиотек.
- Расположение файлов. Программа при, например, записи плейлиста не учитывает то, что пользователь может переместить файл в другое место, что приведёт к ошибке.

Следующим плеерером будет Windows Media Player, который является продуктом Microsoft, и часто обновляется вместе с операционной системой Windows.

Интерфейс приложения:



Ниже приведены плюсы и минусы.

Плюсы:

- Интеграция с Windows: Windows Media Player предустановлен в операционной системе Windows, что делает его частью стандартного пакета программ. Это удобно для пользователей, так как они могут начать использовать медиаплеер сразу после установки ОС;
- Простой интерфейс: Интерфейс Windows Media Player прост и интуитивно понятен. Это делает программу привлекательной для тех, кто предпочитает простоту в использовании;
- Легкость использования: Воспроизведение мультимедийных файлов в Windows Media Player — простая задача. Программа легко обнаруживает и воспроизводит музыку и видео без лишних сложностей;
- Интеграция с библиотекой: Windows Media Player умеет автоматически сканировать и добавлять мультимедийные файлы в библиотеку, облегчая управление и поиск контента;
- Поддержка множества форматов: Встроенная поддержка многих популярных аудио- и видеоформатов, что позволяет пользователям воспроизводить разнообразные мультимедийные файлы.

Минусы:

- Ограниченные возможности: По сравнению с некоторыми другими медиаплеерами, Windows Media Player имеет ограниченные возможности настройки и функциональности. Он может показаться недостаточно мощным для продвинутых пользователей;
- Отсутствие поддержки для некоторых форматов: Некоторые более редкие или современные форматы файлов могут не поддерживаться

Windows Media Player без установки дополнительных кодеков;

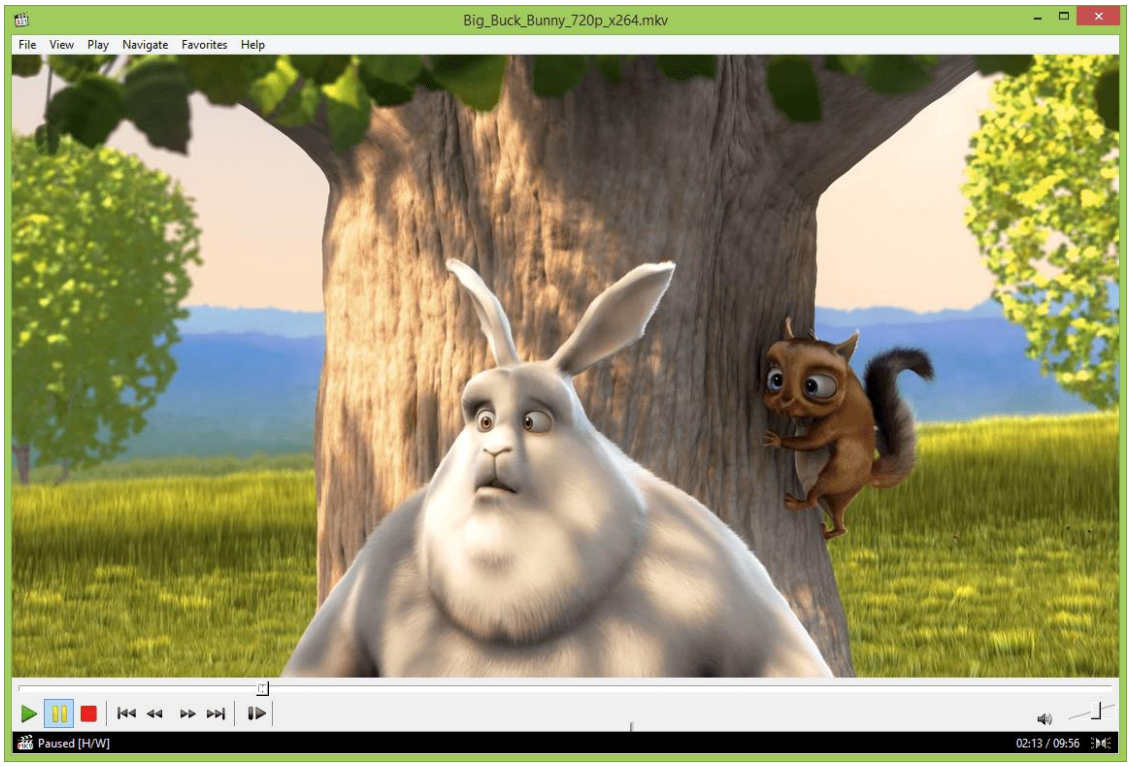
● Отсутствие кроссплатформенности: Windows Media Player доступен только для операционных систем Windows, что может быть недостатком для пользователей, работающих на других платформах;

● Неактивное обновление: В последние годы Microsoft придает меньше внимания разработке и обновлению Windows Media Player в сравнении с другими своими продуктами, такими как Microsoft Edge;

● Ограниченные возможности потокового воспроизведения: Пользователям, которым важна возможность стриминга и потокового воспроизведения, могут потребоваться более современные решения, так как Windows Media Player имеет ограниченные функции в этой области.

Далее, Media Player Classic Home Cinema, или MPC-HC, предоставляет эффективный и гибкий медиаплеер с широкими возможностями настройки.

Интерфейс приложения:



Ниже приведены плюсы и минусы.

Плюсы:

● Лёгкость и быстрота: MPC-HC является лёгким и быстрым медиаплеером, что делает его отличным выбором для пользователей, которым важна эффективная работа программы на компьютере с ограниченными ресурсами;

● Поддержка множества форматов: MPC-HC поддерживает множество аудио- и видеоформатов, а также форматы субтитров. Это

позволяет воспроизводить разнообразный контент без необходимости установки дополнительных кодеков;

- Гибкость настройки: MPC-HC предоставляет продвинутым пользователям широкие возможности настройки. Это включает в себя параметры воспроизведения, настройки видео и аудио, а также возможность использования различных внешних фильтров;
- Активная разработка (на момент последнего обновления): В определённый период времени MPC-HC активно обновлялся, что обеспечивало пользователей новыми функциями и исправлениями ошибок;
- Наличие портативной версии: MPC-HC предоставляет портативную версию, которую можно использовать без установки. Это удобно для пользователей, которым необходимо быстро запустить медиаплеер без изменения системы.

И последний видеоплеер - KMPlayer. Корейский проигрыватель звуковых и видео файлов для Microsoft Windows и macOS, также есть мобильная версия для Android.

Интерфейс приложения:

Ниже приведены плюсы и минусы.

- Интуитивный интерфейс: KMPlayer предоставляет стильный и интуитивно понятный пользовательский интерфейс, что делает его привлекательным для широкого круга пользователей;
- Поддержка широкого спектра форматов: KMPlayer включает в себя встроенные кодеки и обеспечивает поддержку множества аудио- и видеоформатов без необходимости установки дополнительных кодековж;
- Продвинутые возможности настройки: Пользователи могут настраивать параметры воспроизведения, включая яркость, контрастность, насыщенность, а также применять различные фильтры и эффекты;

- Обработка высококачественного контента: KMPlayer хорошо справляется с воспроизведением высококачественных видеофайлов и поддерживает воспроизведение видео с разрешением до 4K;
- Поддержка 3D-видео: KMPlayer предоставляет возможность воспроизведения 3D-видео с использованием подходящих очков, что может быть интересным для любителей такого контента.

Минусы:

- Встроенные объявления: В некоторых версиях KMPlayer присутствуют встроенные рекламные элементы, которые могут раздражать пользователей. Однако можно найти версии без рекламы или отключить их в настройках;
- Отсутствие кроссплатформенности: KMPlayer доступен только для операционной системы Windows, что ограничивает его использование для пользователей, работающих на других платформах;
- Отсутствие облачной интеграции: В отличие от некоторых современных медиаплееров, KMPlayer не предоставляет широких возможностей интеграции с облачными сервисами;
- Ограниченная поддержка потокового воспроизведения: В сравнении с некоторыми другими медиаплеерами, KMPlayer может иметь ограниченные возможности в области потокового воспроизведения мультимедийного контента из сети;
- Отсутствие официальных обновлений (на момент последнего обновления): Некоторые пользователи отмечают, что KMPlayer может не обновляться регулярно, что может вызывать опасения относительно безопасности и совместимости с новым контентом.

Нет идеальных вещей, и технологий это касается особенно. Создать идеальный видеоплеер нельзя, но достичь наилучшее возможное качество вполне возможно.

Проанализировав все вышеперечисленные видеоплееры, можно подчеркнуть детали, необходимые при разработке собственного приложения.

Стоит начать с общих плюсов и минусов.

Общие плюсы:

- Интуитивный интерфейс: Все рассмотренные медиаплееры (VLC, Windows Media Player, Media Player Classic HC, KMPlayer) предоставляют интуитивно понятные пользовательские интерфейсы, что облегчает их использование для широкой аудитории;
- Поддержка форматов: Все медиаплееры обеспечивают поддержку различных аудио- и видеоформатов, что является ключевым аспектом для универсального воспроизведения мультимедийного контента;
- Возможности настройки: Важными плюсами являются возможности настройки параметров воспроизведения, что предоставляет пользователям больше контроля над просмотром и прослушиванием;
- Эффективность ресурсов: Некоторые медиаплееры, такие как

MPC-HC, KMPlayer, предоставляют легкие и быстрые решения, что особенно важно для пользователей с ограниченными ресурсами компьютера.

Общие минусы:

- Отсутствие обновлений и поддержки: Критическим недостатком является отсутствие регулярных обновлений и поддержки, что может влиять на безопасность и совместимость с новым контентом (например, MPC-HC).
- Реклама: Наличие встроенной рекламы может раздражать пользователей и ухудшать общий опыт использования.
- Ограниченные функции потокового воспроизведения: Некоторые медиаплееры могут иметь ограниченные возможности в области потокового воспроизведения;

Выводы:

При разработке видеоплеера следует обратить внимание на следующие аспекты:

- Поддержка форматов: Видеоплеер должен обеспечивать широкую поддержку аудио- и видеоформатов для универсального воспроизведения контента;
- Интуитивный интерфейс: Пользовательский интерфейс должен быть интуитивно понятным и привлекательным для широкой аудитории;
- Возможности настройки: Предоставление пользовательского контроля над параметрами воспроизведения повышает привлекательность плеера;
- Эффективность ресурсов: Приложение должно быть оптимизировано для эффективного использования ресурсов, чтобы обеспечить плавное воспроизведение даже на устройствах с ограниченными возможностями;
- Регулярные обновления и поддержка: Обеспечение регулярных обновлений и поддержки является ключевым для обеспечения безопасности, исправления ошибок и совместимости с новыми стандартами;
- Отсутствие навязчивой рекламы: В случае использования рекламы, она не должна быть навязчивой и должна учитывать удовлетворение пользовательского опыта.

2.1 Обзор методов и алгоритмов решения поставленной задачи

В ходе разработки программы были задействованы разнообразные возможности языка программирования C++, которые будут подробно рассмотрены далее. Для решения поставленной задачи был выбран язык программирования C++ и применена методология объектно-ориентированного программирования [4].

Программа включает в себя механизм обработки исключительных ситуаций. Этот механизм предназначен для описания реакции программы на возможные ошибки, которые могут возникнуть в процессе её выполнения и

привести к невозможности или бессмысленности дальнейшей обработки основного алгоритма программы [5].

Касательно интерфейса пользователя, в коде создаются элементы управления, такие как кнопки воспроизведения, паузы, остановки, перемотки и отображения плейлиста. Дополнительно проведена настройка стилей и расположения элементов интерфейса для улучшения пользовательского опыта.

В области мультимедийной обработки использован класс QMediaPlayer для эффективного управления воспроизведением видео. Реализована обработка видео- и аудиокодеков, обеспечивающая поддержку различных форматов мультимедийных файлов. Также предусмотрен механизм обработки исключительных ситуаций для управления ошибками в процессе выполнения программы.

Относительно настройки воспроизведения, реализован слайдер для регулировки скорости воспроизведения, а также механизмы управления громкостью и перемоткой видео вперёд и назад.

В сфере работы с плейлистом создан и настроен плейлист для хранения и управления списком видеофайлов. Реализованы функции загрузки и сохранения плейлиста с использованием диалоговых окон для выбора файла.

Система сохранения и восстановления состояния позволяет сохранять текущее состояние видеоплеера, включая последний открытый файл и его позицию, с последующим восстановлением при повторном запуске программы.

3 ФУНКЦИОНАЛЬНОЕ ПРОЕКТИРОВАНИЕ

В данном разделе описываются входные и выходные данные программы, диаграмма классов, а также приводится описание используемых классов и их методов.

3.1 Входные данные

Для работы программы и отображения иконок были сделанные специальные jpg картинки, помещённые в папку icon.

3.2 Разработка диаграммы классов

Диаграмма классов данной работы показана в приложении А.

3.3 Описание классов

3.3.1 Класс `MainWindow`

`MainWindow` – Класс предназначен для управления основным окном видеоплеера. Он инкапсулирует в себе логику и элементы управления, наследует функции и характеристики QMainWindow.

Описание полей класса:

`Ui::MainWindow *ui` - указатель на объект Ui. Предназначен для работы с виджетами.

`videoPlayer (QMediaPlayer)` – Указатель на объект класса QMediaPlayer, отвечающий за воспроизведение аудио и видео.

`videoWidget (QVideoWidget)` - Указатель на объект класса QVideoWidget, представляющий виджет для отображения видео.

`videoContainer (QWidget)` - Указатель на объект класса QWidget, используемый в качестве контейнера для видео.

`controlWidget (QWidget)` - Указатель на объект класса QWidget, представляющий виджет для управления воспроизведением.

`slider (QSlider)` - Указатель на объект класса QSlider, представляющий слайдер для управления временем воспроизведения видео.

`volume (QSlider)` - Указатель на объект класса QSlider, представляющий слайдер для управления уровнем громкости.

`speedSlider (QSlider)` - Указатель на объект класса QSlider, представляющий слайдер для управления скоростью воспроизведения.

`Mduration (qint64)` - Переменная типа qint64, содержащая общую длительность текущего видео.

`startTimeLabel (TimeLabel)` - Указатель на объект класса TimeLabel, представляющий виджет для отображения текущего времени

воспроизведения.
endTimeLabel (TimeLabel) – Указатель на объект класса TimeLabel, представляющий виджет для отображения общей длительности видео.
playlistWidget (PlaylistWidget) – Указатель на объект класса PlaylistWidget, представляющий виджет для отображения и управления плейлистом.
isPlaylistVisible (bool) – Флаг типа bool, указывающий, видим ли в данный момент плейлист.
videoPositions (QMap<QString, qint64>) – Контейнер QMap, содержащий информацию о позициях воспроизведения для каждого видео в плейлисте.
lastVideoFilePath (QString) – Строковая переменная, содержащая путь к последнему выбранному видеофайлу.
appSettings (QSettings) – Указатель на объект класса QSettings, представляющий настройки приложения.

Описание методов:

openFile() – Открывает диалоговое окно для выбора видеофайла.
loadVideo(const QString &videoFilePath) – Загружает выбранный видеофайл для воспроизведения.
onPlaylistItemClicked(const QString &videoFilePath) – Обрабатывает событие выбора элемента в плейлисте, начинает воспроизведение выбранного видеофайла.
togglePlayPause() – Переключает состояние воспроизведения/паузы видео.
on_btnStop_clicked() – Останавливает воспроизведение видео.
on_btnBackward_clicked() – Возвращает видео на 5 секунд назад.
on_btnForward_clicked() – Перематывает видео на 5 секунд вперёд.
on_btnShowPlaylist_clicked() – Скрывает/показывает плейлист.
loadPlaylist() – Загружает плейлист с сохранёнными видеофайлами.
savePlaylist(const QString &playlistPath, const QList<QString> &filePaths) – Сохраняет текущий плейлист в указанный файл.
saveCurrentPlaylist() – Сохраняет текущий плейлист.
openFolderAndAddToPlaylist() – Открывает диалоговое окно для выбора папки и добавляет видеофайлы из этой папки в плейлист.
onSpeedSliderChanged(int value) – Обрабатывает изменение положения слайдера скорости воспроизведения.
setPlaybackSpeed025()...setPlaybackSpeed2() – Устанавливает соответствующую скорость воспроизведения.
onVolumeSliderChanged(int value) – Обрабатывает изменение положения слайдера громкости.
increaseVolume()...offVolume() – Управляют уровнем громкости.
updateCurrentTime(qint64 position) – Обновляет текущее время

воспроизведения видео.
`updateTotalDuration(qint64 duration)` - Обновляет общую длительность видео.
`saveLastVideoPosition()` - Сохраняет последнюю позицию воспроизведения видео.
`loadLastVideoPosition()` - Загружает последнюю позицию воспроизведения видео.
`closeEvent(QCloseEvent *event)` - Обрабатывает событие закрытия главного окна.
`closeApp()` - Завершает выполнение приложения.
`showHelpMessage()` - Отображает справочное сообщение о приложении.
`MainWindow(QWidget parent = nullptr)` - конструктор класса.

3.3.2 Класс `PlaylistWidget`

`PlaylistWidget` - Класс представляет виджет плейлиста, который наследует от QListWidget и предоставляет дополнительную функциональность для управления плейлистом и обработки событий кликов на элементах.

Описание полей класса:

`playlist (QMediaPlaylist)` - Указатель на объект класса QMediaPlaylist, представляющий плейлист для хранения и управления списком аудио- и видеофайлов.

`playlistItemClicked(const QString &videoFilePath)` – Сигнал, отправляемый при клике на элемент плейлиста. Передаёт полный путь к выбранному видеофайлу.

`handleItemClick(QListWidgetItem item)` – Слот, обрабатывающий событие клика на элемент плейлиста. Извлекает полный путь к выбранному видеофайлу из элемента и отправляет соответствующий сигнал.

Описание методов:
`PlaylistWidget(QWidget parent = nullptr)` – Конструктор класса.
`addItem(const QString &folderPath, const QString &videoFileName)` - Метод для добавления элемента в плейлист. Принимает путь к папке и имя видеофайла. Создаёт полный путь к видеофайлу и добавляет его в плейлист.
`getPlaylist()` - Метод для получения текущего плейлиста в виде QStringList.

3.3.3 Класс `timelabel`

`timelabel` – Класс наследует функционал и элементы QLabel, который предназначен для отображения информации о времени в виде текста.

Описание полей класса:

`mediaPlayer (QMediaPlayer)` - Указатель на объект класса QMediaPlayer, представляющий плеер, с которым связан виджет.

`videoDuration (qint64)` - Переменная для хранения общей длительности текущего видеофайла в миллисекундах.

Описание методов:

`TimeLabel(QWidget parent = nullptr)` - Конструктор класса.

`setMediaPlayer(QMediaPlayer player)` - Метод для установки плеера, с которым будет связан TimeLabel.

`updateCurrentTime()` - Метод для обновления отображаемого времени текущего воспроизведения. Извлекает текущую позицию видеоплеера и обновляет текст виджета с учётом формата времени.

`updateTotalDuration()` - Метод для обновления отображаемой общей длительности видео. Извлекает общую длительность текущего видеофайла из медиаплеера и обновляет текст виджета с учётом формата времени.

4 РАЗРАБОТКА ПРОГРАММНЫХ МОДУЛЕЙ

4.1 Разработка схем алгоритмов

Метод `loadPlaylist()` загружает позицию последнего просмотренного видео. Схема метода `loadPlaylist()` показана в приложении Б.

Метод `openFolderAndAddToPlaylist()` сохраняет текущий плейлист.

Схема метода `openFolderAndAddToPlaylist()` показана в приложении В.

4.2 Разработка алгоритмов

Метод `saveCurrentPlaylist()` класса `MainWindow`.

Шаг 1. Запрос имени и расположения файла у пользователя;

Шаг 2. Вывод диалогового окна для выбора имени и места сохранения файла плейлиста;

Шаг 3. Применяется метод QFileDialog::getSaveFileName;

Шаг 4. Проверка, был ли выбран файл;

Шаг 5. Проверяется, был ли пользователем выбран файл или нажал "Отмена";

Шаг 6. Объявляется список строк filePaths, который будет использоваться для хранения путей к файлам;

Шаг 7. Начинается цикл, который будет итерироваться по всем элементам виджета плейлиста (playlistWidget);

Шаг 8. Для каждой итерации цикла получаем указатель на объект типа QListWidgetItem по текущему индексу i;

Шаг 9. Из данных элемента виджета (Qt::UserRole) извлекается закодированный путь к папке и сохраняется в строку encodedFolderPath;

Шаг 10. Закодированный путь декодируется из процент-кодировки и сохраняется в строке folderPath;

Шаг 11. Строится полный путь к файлу, объединяя декодированный путь к папке и имя видеофайла. Результат сохраняется в строке filePath;

Шаг 12. Полный путь к файлу добавляется в список filePaths. Этот список накапливает все пути к файлам из виджета плейлиста на каждой итерации цикла;

Шаг 13. Метод saveCurrentPlaylist вызывает метод savePlaylist, передавая ей выбранный путь и список файлов;

Шаг 14. Создание объекта QFile, который представляет файл плейлиста;

Шаг 15. Файл открывается в режиме записи и текстового доступа. Если открытие прошло успешно, код внутри условия выполняется;

Шаг 16. Создание объекта QTextStream, который используется для

записи текста в файл;

Шаг 17. Запись строки #EXTM3U, которая указывает на формат M3U;

Шаг 18. Итерация по списку путей к файлам, переданным в метод;

Шаг 19. Создание объекта QUrl из пути к файлу;

Шаг 20. Кодировка URL видеофайла. Так как M3U может использовать разные кодировки, то для удобства и широкого применения используется UTF-8;

Шаг 21. Запись в файл кодированный URL видеофайла;

Шаг 22. Завершение цикла, закрытие файла;

Шаг 23. Если открытие файла не удалось, выводится предупреждение;

Шаг 24. Выводится информационное сообщение о том, что плейлист был успешно сохранен;

Метод `loadLastVideoPosition()` класса `MainWindow`.

Шаг 1: Создание объекта QSettings для доступа к настройкам приложения;

Шаг 2: Получение пути к последнему воспроизведённому видеофайлу из настроек;

Шаг 3: Проверка, что путь не пуст и файл существует;

Шаг 5: Запрос пользователя о загрузке последней позиции;

Шаг 6: Обработка ответа пользователя;

Шаг 7: Обновление переменной lastVideoFilePath перед загрузкой видео. Это необходимо для того, чтобы пользователь при открытии приложения в следующий раз, не открыв другое видео, не потерял сохранение;

Шаг 8: Добавление видео в плейлист;

Шаг 9: Установка медиа и позиции;

Шаг 10: Воспроизведение видео;

Шаг 11: Вывод отладочной информации;

Шаг 12: При выборе “No” выводится информация об отказе пользователя от загрузки последней позиции;

Шаг 13: В случае ошибки выводится информация о том, что сохранённой позиции воспроизведения не найдено.

5. РЕЗУЛЬТАТ РАБОТЫ

На рисунке 5.1 изображена начало работы программы. При старте программы пользователь может открыть файл/видео, сохранить и открыть плейлист.

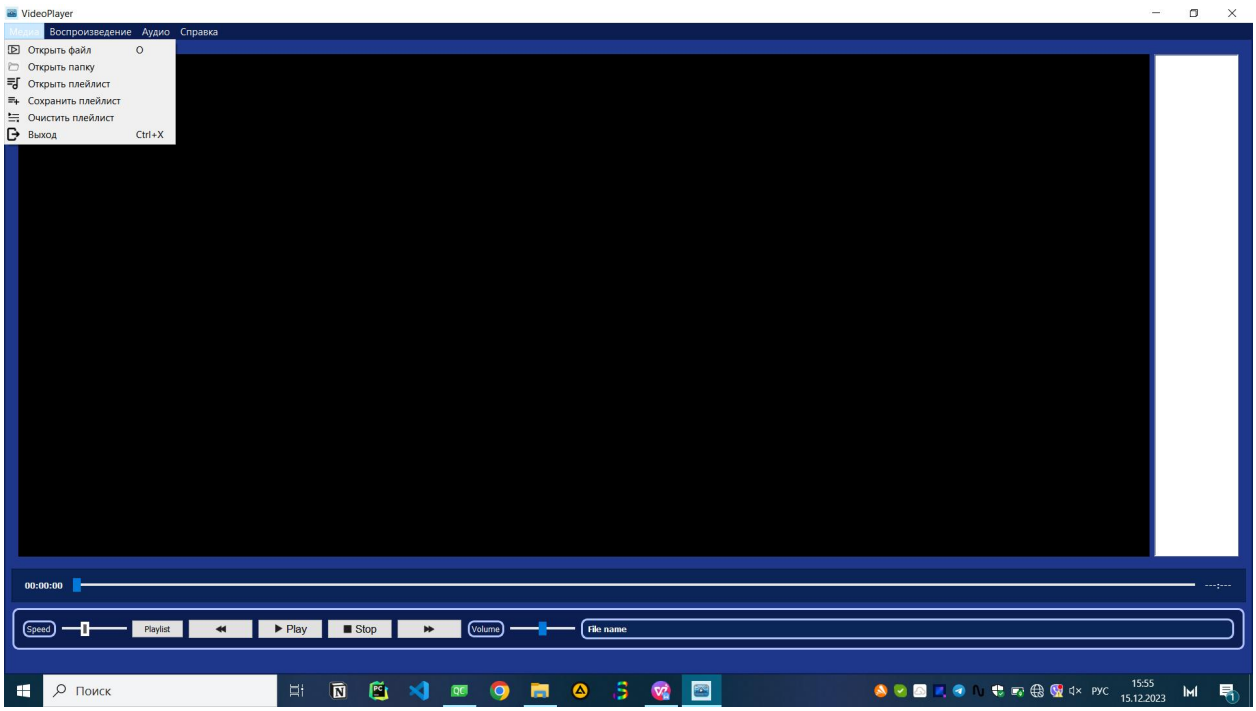


Рисунок 5.1 – Начало работы программы

На рисунке 5.2 показана работа плейлиста и видеоплеера. Пользователь может переключаться между видео с помощью нажатия на название.

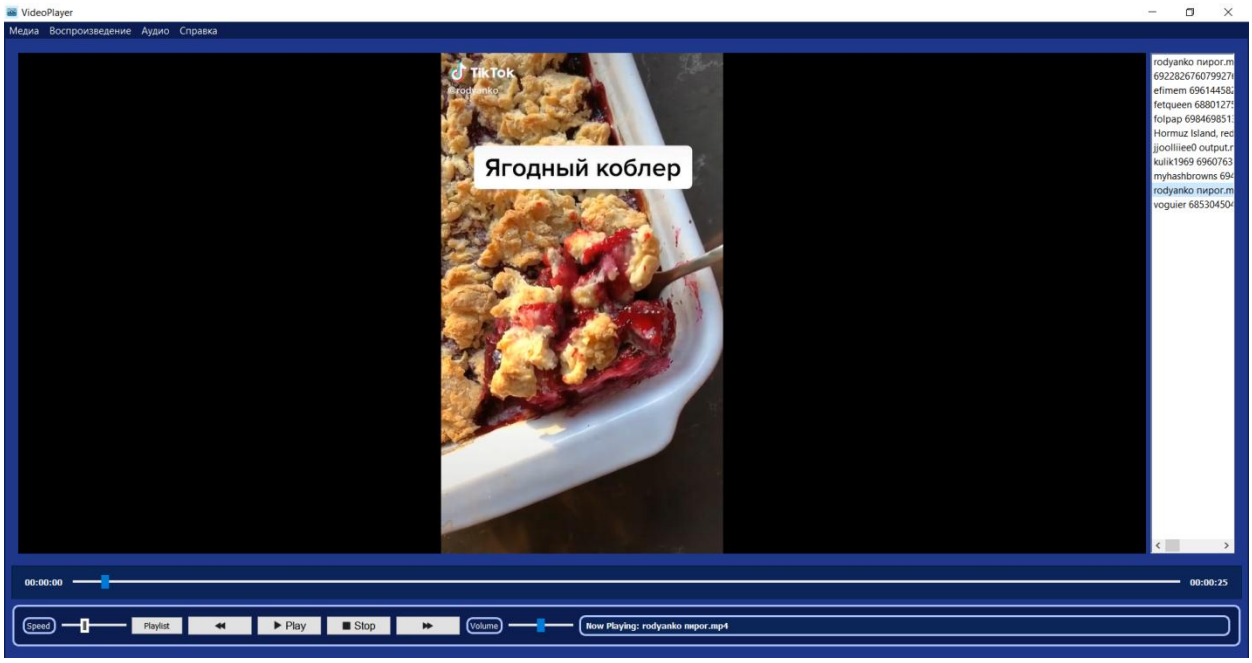


Рисунок 5.2 – Работа видеоплеера и плейлиста

На рисунке 5.3 отражена работа загрузки контрольной точки последнего просмотренного видеоролика. При нажатии на кнопку “Yes” функция срабатывает. При нажатии на кнопку “No” программа запускается как обычно.

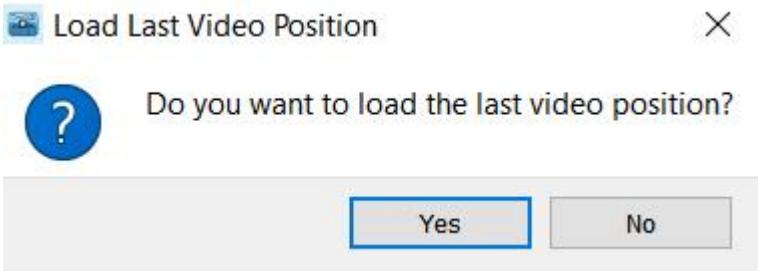


Рисунок 5.3 – Окно загрузки сохранённой точки

На рисунке 5.4 показана демонстрация функции загрузки плейлиста. При удачном выполнении выводится сообщение.

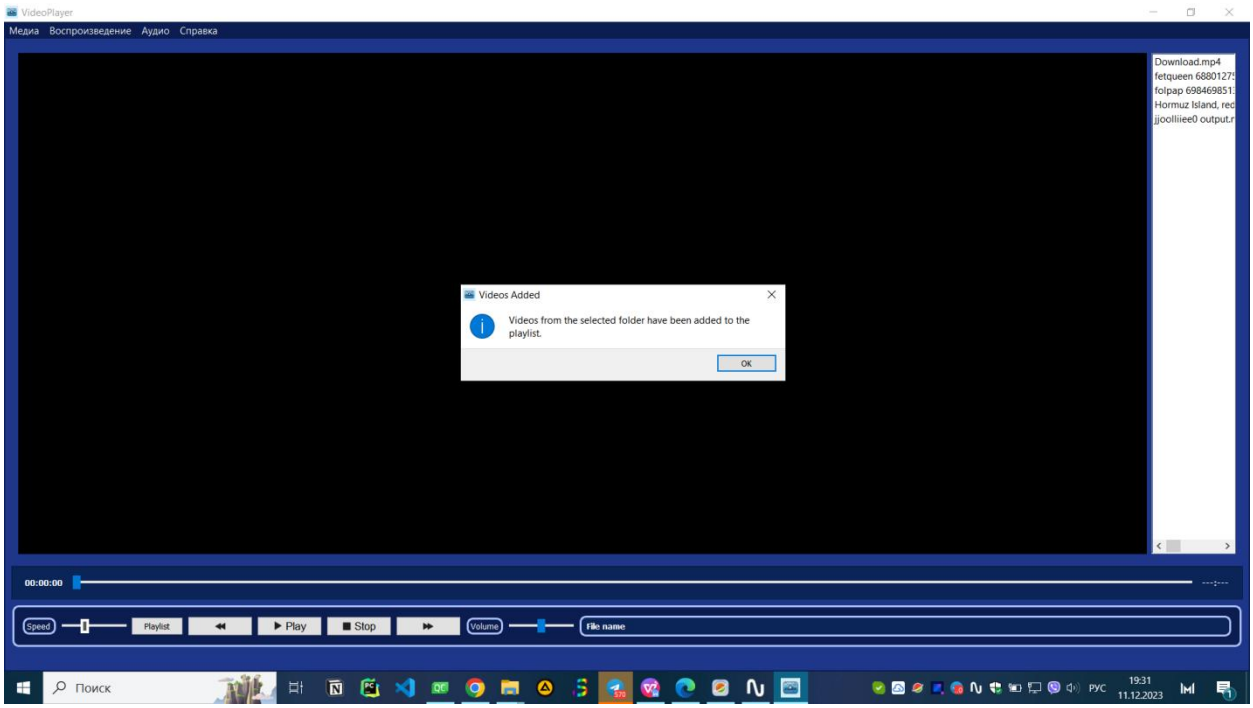


Рисунок 5.4 – Загрузка плейлиста

Рисунок 5.5 показывает возможности плеера: изменение скорости, скачок на 5 секунд вперёд и назад, пауза/плей и стоп.

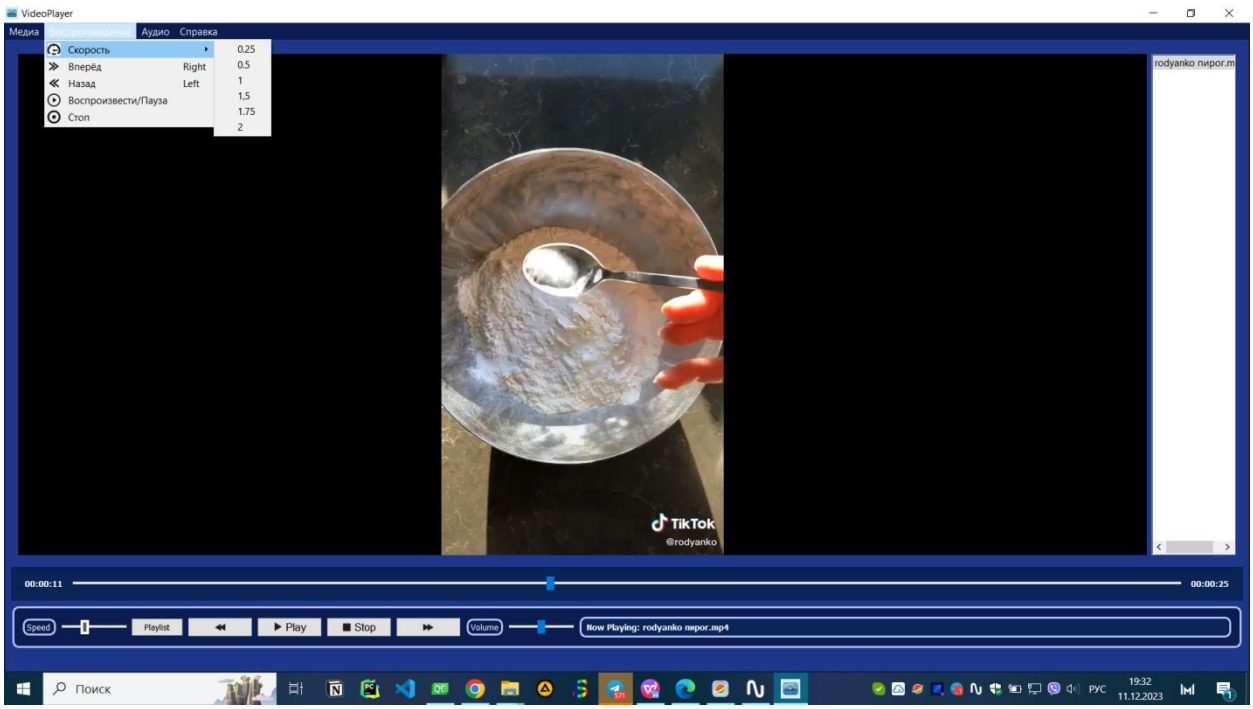


Рисунок 5.5 – Демонстрация функций

Рисунок 5.6 показывает функционал меню «Аудио». Пользователь может увеличивать, уменьшать, отключать звук.

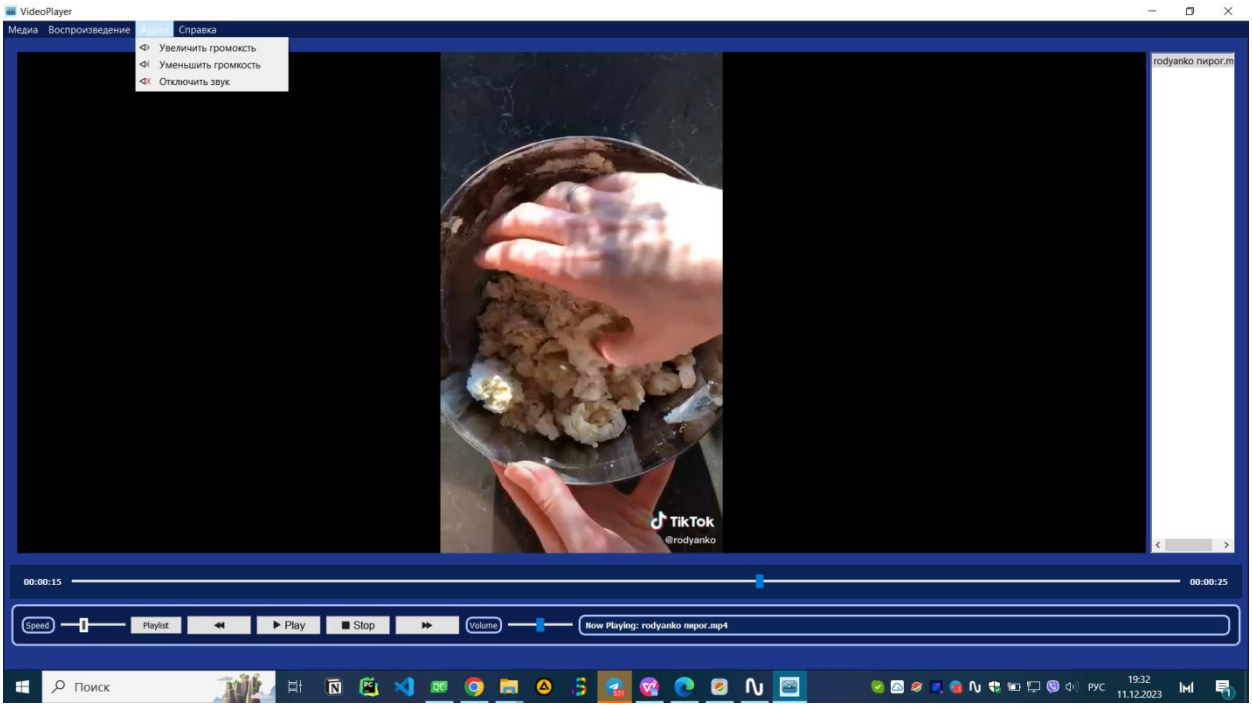


Рисунок 5.6 – Регулирование звука

Для информации о приложении можно обратиться в меню "Справка", пункт "Информация".

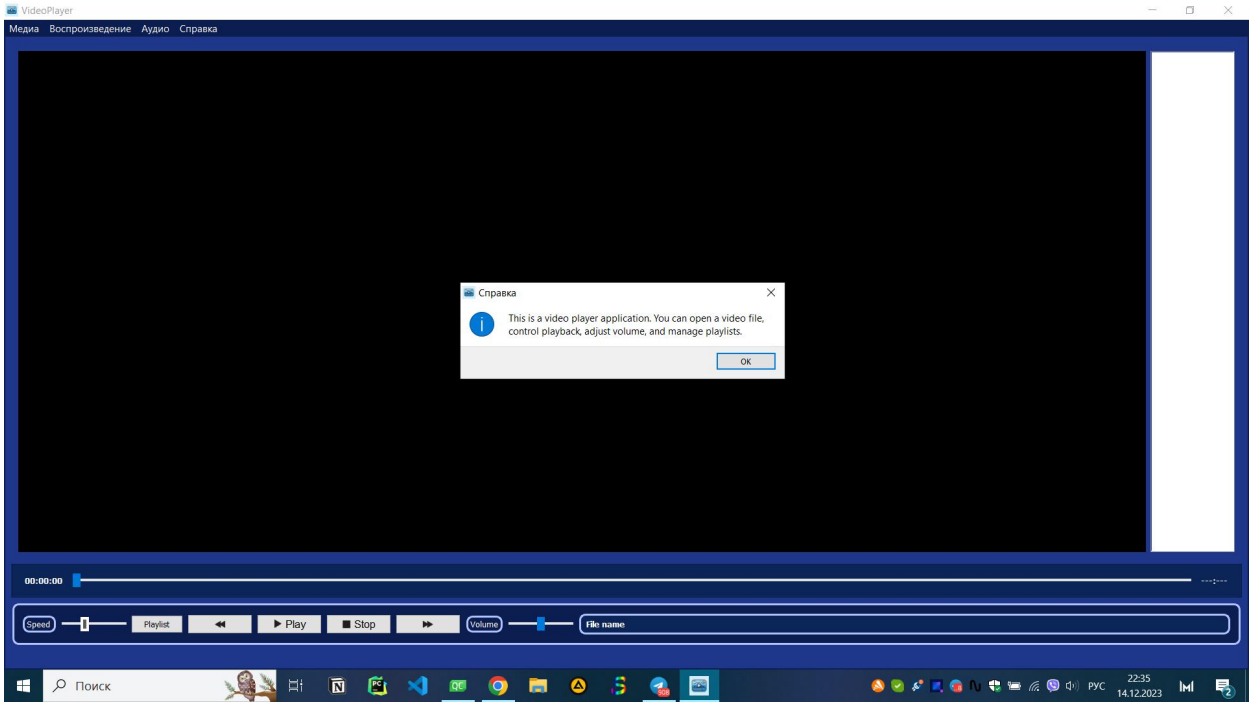


Рисунок 5.7 - Справка

При желании пользователь может взаимодействовать с плейлистом,

сохранив, открыв или очистив его. Рассмотрим процесс поподробнее.

Во-первых, необходимо загрузить хотя бы одно видео в плейлист. Пользователю нужно открыть меню "Медиа" и выбрать пункт по желанию. В данном примере загружаются видео из папки.

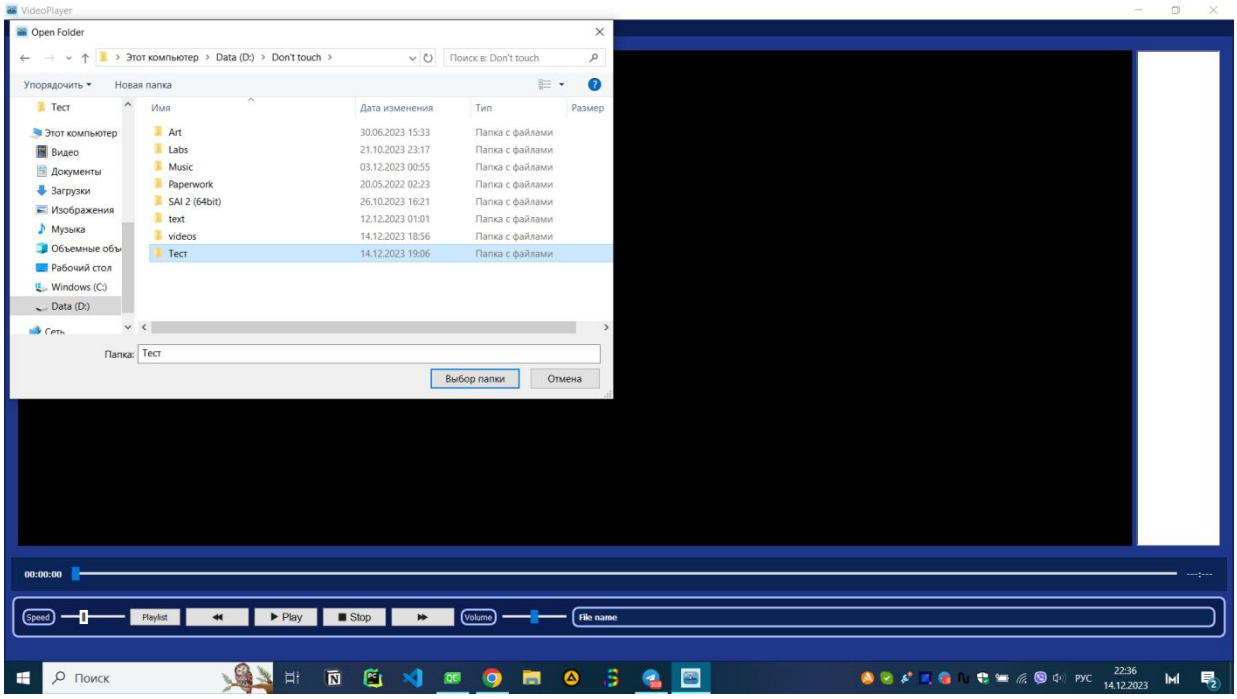


Рисунок 5.8 - Загрузка видео

Далее, на рисунке 5.9 видно, что из русскоязычной папки загружаются видео, в дальнейшем это будет важно.

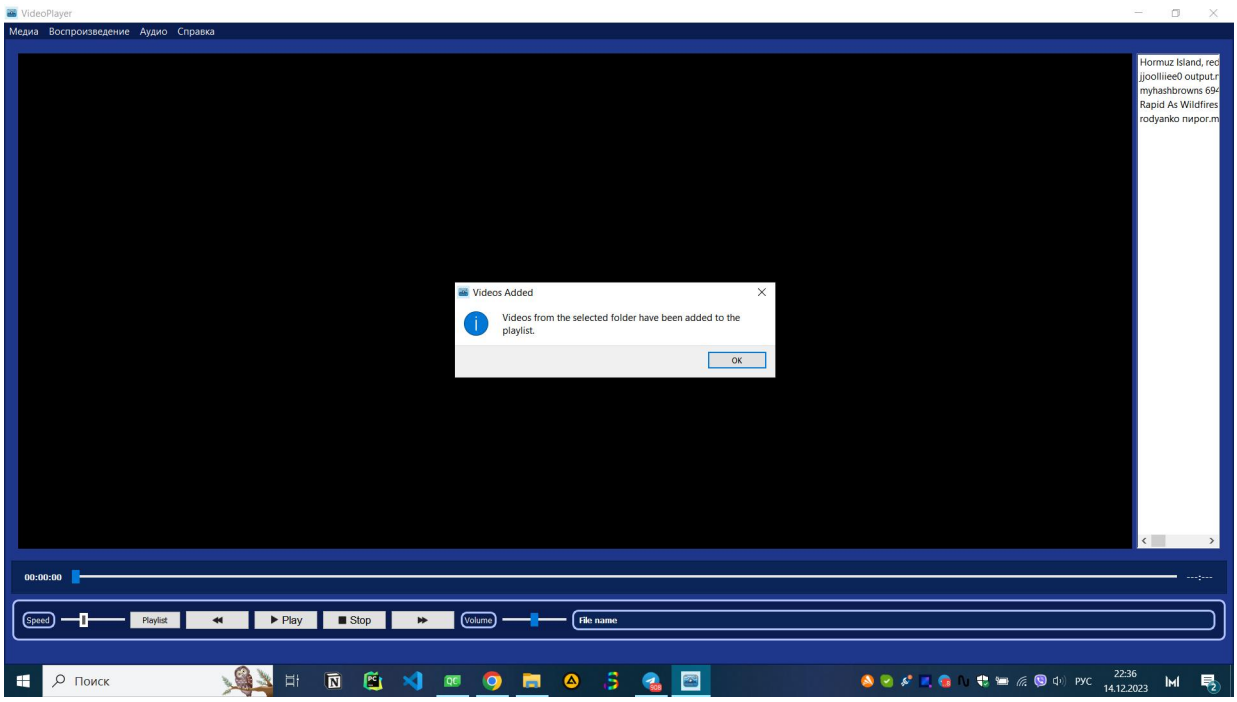


Рисунок 5.9 - Загрузка видео в плейлист

Можно свободно взаимодействовать с плейлистом. На рисунке 5.10 пользователь выбрал пункт "Сохранить плейлист".

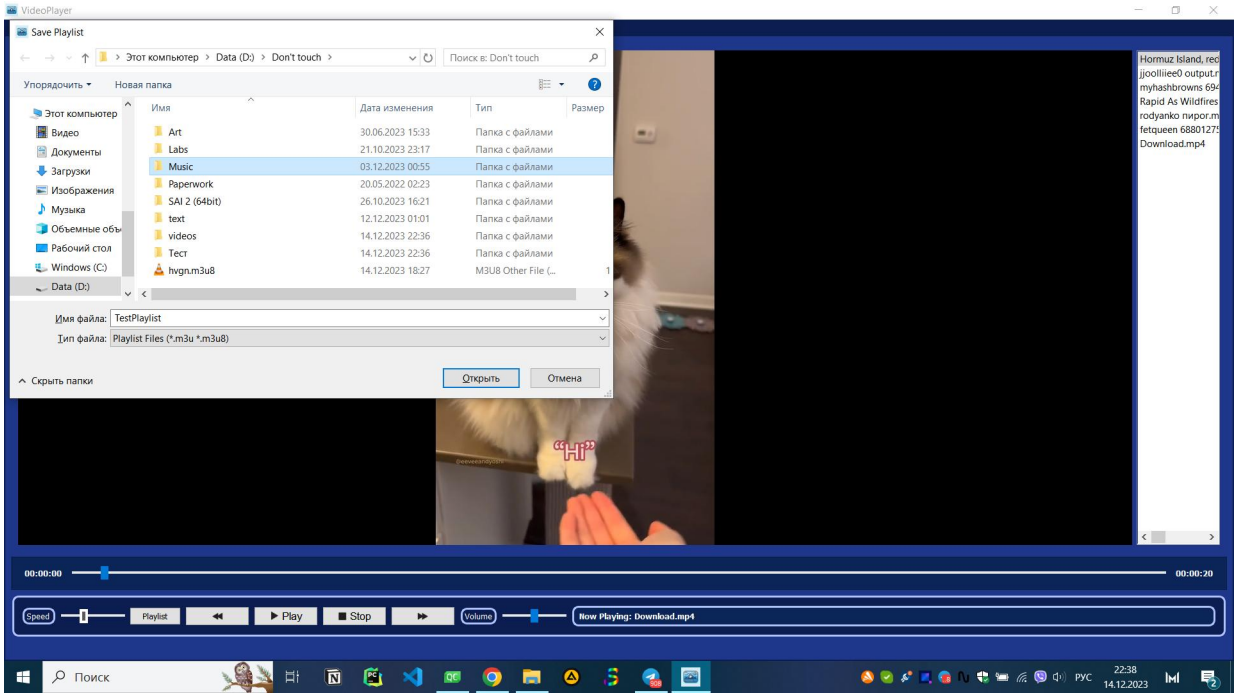


Рисунок 5.10 - Сохранение плейлиста

Откроем файл m3u в блокноте. При изучении VLC Media Player был замечен недочёт процесса сохранения плейлиста. Он заключается в "относительной" записи файлов.

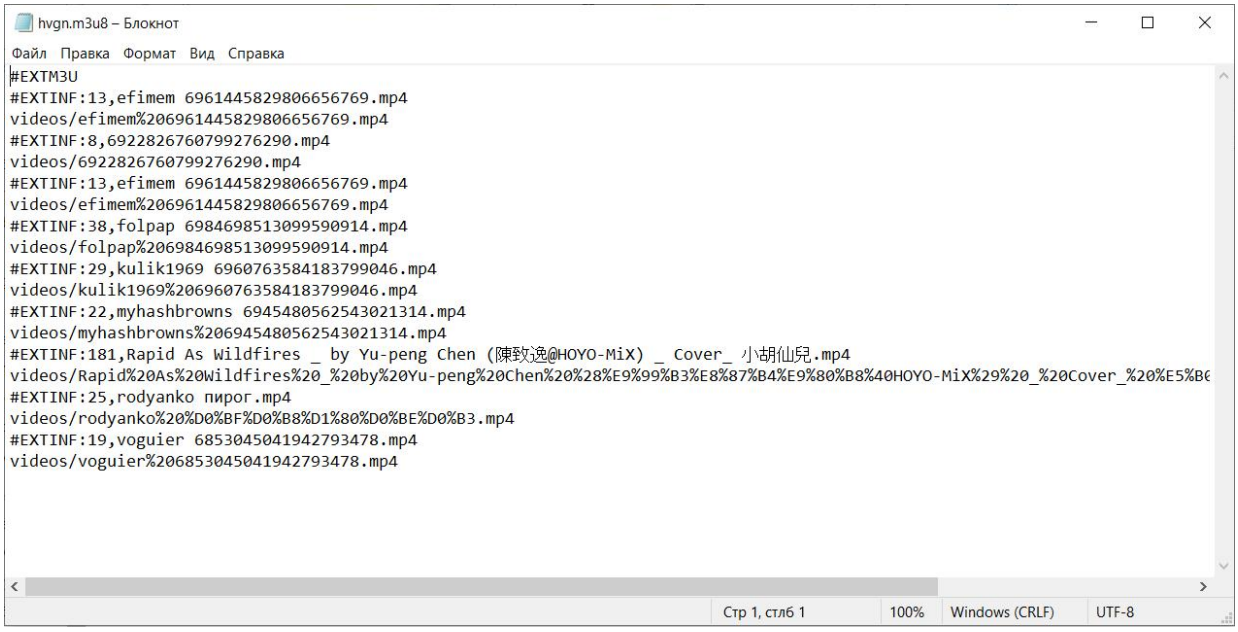


Рисунок 5.11 - Файл плейлиста, созданного VLC Media Player

Как можно заметить, в файл записываются длительность видео, название и адрес, который зависит от положения видео относительно файла. Это достаточно серьёзная проблема, потому что при перемещении файла в другую папку плейлист перестанет работать. Поэтому в ходе разработки вместо "относительного" пути записывается полный путь. Так как не все медиаплееры и кодеки способны обрабатывать кириллицу, иероглифы и не классические ASCII символы, имена и адреса файлов декодируются в UTF-8.

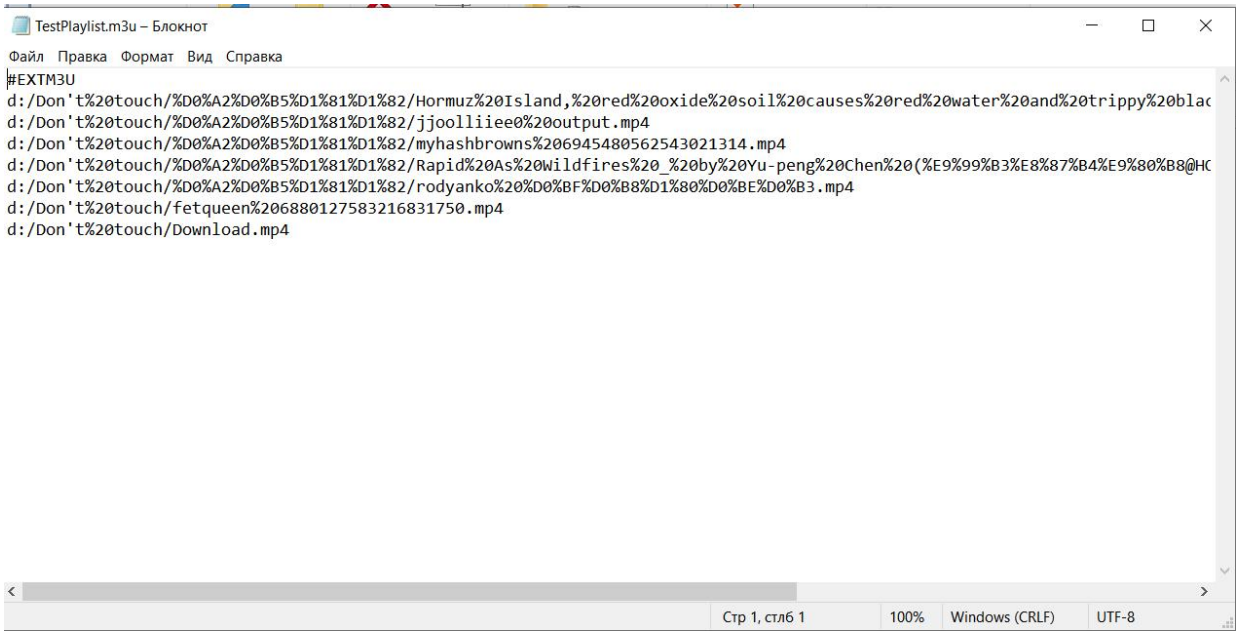


Рисунок 5.12 - Файл плейлиста, созданного разработанным плеером

Также, Qt обрабатывает кириллицу, а с иероглифами есть некоторые затруднения, которые необходимо учитывать при разработке программ, связанные с языками.

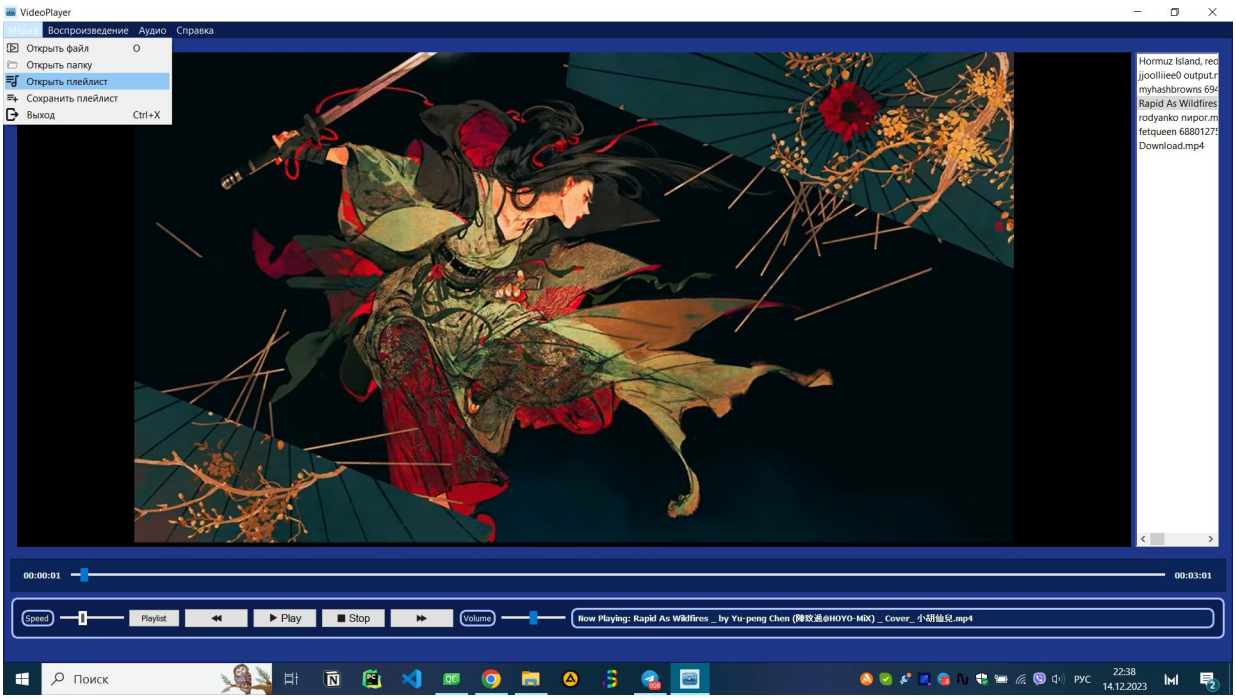


Рисунок 5.13 - Результат открытия созданного программой плейлиста

```
========================
========================
"Rapid%20As%20Wildfires%20_%20by%20Yu-peng%20Chen%20(%E9%99%B3%E8%87%B4%E9%80%B8@HOYO-MiX)%20_%20Cover_%20%E5%B0%8F%E8%83%A1%E4%BB%99%E5%85%92.mp4"
"Rapid As Wildfires _ by Yu-peng Chen (???@HOYO-MiX) _ Cover_ ????.mp4"
========================
========================
"rodyanko%20%D0%BF%D0%B8%D1%80%D0%BE%D0%B3.mp4"
"rodyanko пирог.mp4"
========================
========================
"fetqueen%206880127583216831750.mp4"
"fetqueen 6880127583216831750.mp4"
```

Рисунок 5.14 - Вывод имён файлов с кодировкой и в привычной форме в отладчике Qt

ЗАКЛЮЧЕНИЕ

В процессе выполнения курсовой работы, посвящённой разработке видеоплеера, были успешно достигнуты первоначально поставленные цели. Реализован видеоплеер с базовыми функциями, позволяющие просматривать видео и плейлист.

В дальнейшем можно реализовать более продвинутые функции, например, просмотр субтитров, изменение цвета/контраста изображения.

Итог: Полученные знания в области работы с видеоданными, а также опыт в создании графического интерфейса на основе фреймворка Qt, предоставляют отличные возможности для будущих проектов. Этот опыт может быть успешно применен в разработке различных мультимедийных приложений, таких как видеоредакторы, фильтры видеоизображений и другие приложения, связанные с обработкой видеоданных.

Основанный на данной работе опыт и знания позволяют в дальнейшем изучить область мультимедийных технологий и программирования, открывают перспективы для участия в проектах, направленных на улучшение пользовательского опыта в области мультимедийных приложений.

В ходе выполнения курсовой работы произошло значительное расширение навыков программирования на языке C++. Также были освоены аспекты взаимодействия с графикой и разработки графических интерфейсов с применением фреймворка Qt. Полученные компетенции предоставляют устойчивую базу для последующих исследований и разработок в области компьютерного зрения и графики.

СПИСОК ЛИТЕРАТУРЫ

[1] "Объектно-ориентированное программирование на C++" Бьярн Страуструп
[2] "Язык программирования C++" Герберт Шилдт
[3] "C++ Primer" Липман, Лажойе, Му, Хопкинс
[4] "Алгоритмы. Построение и анализ" Кормен, Лейзерсон, Ривест, Штайн
[5] "Введение в алгоритмы" Кормен, Лейзерсон, Ривест, Штайн
[6] "Алгоритмы на C++" Роберт Седжвик, Кевин Уэйн
[7] " Qt Examples And Tutorials" [Электронный ресурс]. - Режим доступа: https://doc.qt.io/qt-5/qtexamplesandtutorials.html Дата доступа: 22.10.2023
[8] "Структуры данных и алгоритмы в C++" Robert Lafore
[9] "Qt.5.10 Профессиональное программирование на C++" Шлее
[10] "Beautiful C++: 30 Core Guidelines for Writing Clean, Safe and Fast Code" J. Guy Davidson, Kate Gregory

ПРИЛОЖЕНИЕ А
(обязательное)
Диаграмма классов

ПРИЛОЖЕНИЕ Б
(обязательное)
Схема метода `loadPlaylist()`

ПРИЛОЖЕНИЕ В
(обязательное)
Схема метода `openFolderAndAddToPlaylist()`

ПРИЛОЖЕНИЕ Г
(обязательное)
Код программы

ПРИЛОЖЕНИЕ Д
(обязательное)
Ведомость документов