**СОДЕРЖАНИЕ**

ВВЕДЕНИЕ………………………………………………………………………..5

1 ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РЕШЕНИЯ

ПОСТАВЛЕННОЙ ЗАДАЧИ………..………………………………………...…6

2 ОБОСНОВАНИЕ ВЫБРАННЫХ МЕТОДОВ И АЛГОРИТМОВ…………..7

3 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПРОГРАММИСТА……………………...8

4 ОПИСАНИЕ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ…………………………9

5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ…………………………………………11

ЗАКЛЮЧЕНИЕ………………………………………………………………….12

СПИСОК ЛИТЕРАТУРЫ….

ПРИЛОЖЕНИЕ А….

ПРИЛОЖЕНИЕ Б….

ПРИЛОЖЕНИЕ В….

ПРИЛОЖЕНИЕ Г….

ПРИЛОЖЕНИЕ Д….

**ВВЕДЕНИЕ**

Чтение книг всегда являлось важной части жизни каждое человека независимо от рода его деятельности. В 21 веке все большую и большую популярность приобретает чтение книг в онлайн формате на телефоне или компьютере. Так как мне нравится читать, то было решено попробовать создать свое приложение для чтения книг.

Объектно-ориентированное программирование представляет собой концепцию программирования, в которой программа представляет из себя совокупность взаимодействующих объектов, каждый из которых является экземпляром какого-либо класса, а сами классы образуют иерархию наследования. Один из наиболее популярных языков программирования, поддерживающим данную концепцию, является C++.

C++ — компилируемый статически типизированный язык программирования. Он поддерживает такие парадигмы программирования, как процедурное, объектно-ориентированное и обобщенное. C++ включает в себя как возможности низкоуровневых, так и высокоуровневых языков программирования.

Основными принципами объектно-ориентированного программирования являются инкапсуляция, наследование и полиморфизм. C++ предоставляет возможности для реализации данных концепций. Инкапсуляция позволяет ограничить доступ к свойствам объектов другим частям этой же программы. Наследование предоставляет возможность повторного использования кода, что помогает уменьшить количество кода и увеличить скорость его написания. Полиморфизм позволяет быть программе более гибкой.

C++ достаточно универсальный язык программирования, что позволяет его использовать для написания операционных систем, драйверов устройств, приложений, игр. Исходя из этого можно сделать вывод, что данный язык программирования достаточно удобен для данной курсовой работы.

**1 ОБЗОР МЕТОДОВ И АЛГОРИТМОВ РЕШЕНИЯ**

**ПОСТАВЛЕННОЙ ЗАДАЧИ**

В настоящее время fb2 (fb – fiction book, «художественная книга») является популярным форматом для хранения книг. Fb2-файл – это частный случай формата xml. Текст в документе хранится в специальных тегах – элементах типов параграфов. Ключевым компонентом такого файла является тег <body>. Он содержит непосредственно текст книги. Во всем документе таких тегов может быть несколько – дополнительные блоки используются для хранения сносок, комментариев и примечаний.

Стандарт fb2 включает в себя лишь минимально необходимый набор тегов для “оформления” художественной литературы, что упрощает его обработку. Причем в самом файле не находится никакой информации о стилях текста, что позволяет задать в программе все атрибуты текста или сделать возможным изменять их пользователем. Недостатком формата является отсутствие функционала для сложной верстки текста, что делает его неподходящим для учебников, справочников и технической литературы, о чем говорит само название формата.

Для решения поставленной задачи в программе были созданы следующие классы:

* MainWindow. Этот класс предназначен для создания главного окна приложения;
* Book. Этот класс содержит в себе текст книги, номер текущей страницы, общее число страниц, максимальную длину строки и максимальное число строк на странице. Также данный класс содержит в себе методы для считывания текста из fb2 файла и методы доступа к приватным полям;
* RecentOpenedFilesWindow. Этот класс предназначен для создания диалогового окна, в котором пользователь может выбрать одну из ранее открытых книг на последней открытой странице либо удалить любую такую книгу;
* WidgetStyle. Этот класс содержит в себе статические методы, каждый из которых возвращает стили для виджетов на главном и диалоговом окнах.
* List. Этот шаблонный класс является реализацией односвязного списка.
* Exception. Этот класс является базовым классом исключений, от которого наследуются классы для различных типов исключений.
* ArgumentNullException. Этот класс обрабатывает исключение, когда какой-либо аргумент отсутсвует.
* OutOfRangeException. Этот класс обрабатывает исключение, когда пользователь пытается перейти на несуществующую страницы.

**2 ОБОСНОВАНИЕ ВЫБРАННЫХ МЕТОЛОВ И АЛГОРИТМОВ**

Для написание данного курсового проекта был выбран фреймворк Qt, основанный на языке программирования C++. Qt обладает большим количеством классов, которые могут пригодиться в процессе разработки программного обеспечения, начиная от элементов графического интерфейса и заканчивая классами, предназначенными для работы с сетью, базами данных и XML документами, удобной системой общения между виджетами приложения с помощью сигналов и слотов и хорошей документацией, позволяющей за короткий промежуток времени разобраться в устройстве фреймворка.

Для реализации программной логики используется язык C++ по причине высокой производительности и поддержки объектно-ориентированного программирования.

**3 ОПИСАНИЕ ПРОГРАММЫ ДЛЯ ПРОГРАММИСТА**

Диаграмма классов позволяет на увидеть зависимости, которые есть между классами в программе.

**4 ОПИСАНИЕ АЛГОРИТМОВ РЕШЕНИЯ ЗАДАЧИ**

**4.1 Алгоритм по шагам метода void Book::ParseBookFile()**

Шаг 1. Создание объекта класса QFile bookFile(pathToBookFile).

Шаг 2. Если !bookFile.open(QIODevice::Text | QIODevice::ReadOnly) перейти к шагу 3, иначе перейти к шагу 5.

Шаг 3. Создание окна с ошибкой QMessageBox::critical(nullptr, “”, “Что-то пошло не так! Файл невозможно открыть!”).

Шаг 4. Конец.

Шаг 5. Создание объекта класса QXmlStreamReader xmlFile(&bookFile).

Шаг 6. Если !xmlFile.atEnd() перейти к шагу 7, иначе перейти к шагу 14.

Шаг 7. xmlFile.readNextStartElement().

Шаг 8. Если xmlFile.isEndElement() перейте к шагу 6, иначе перейти к шагу 9.

Шаг 9. Создание объекта класса QString

Шаг 10. Если tag == “body” перейти к шагу 11, иначе перейте к шагу 13.

Шаг 11. SetBookText(xmlFile).

Шаг 12. Перейти к шагу 14.

Шаг 13. Перейти к шагу 6.

Шаг 14. bookFile.close().

Шаг 15. Конец.

**4.2 Алгоритм по шагам метода void Book::SetBookText**

**(QXmlStreamReader& xmlFile)**

Шаг 1. Созданием объекта класса Qstring tag.

Шаг 2. Если tag != body перейти к шагу 3, иначе пеерйти к шагу 12.

Шаг 3. xmlFile.readNextStartElement().

Шаг 4. tag = xmlFile.name().toString().

Шаг 5. Если xmlFile.isEndElement() перейте к шагу 2, иначе перейти к шагу 6.

Шаг 6. Пока tag != "p" && tag != "title" перейти к шагу 7, иначе перейти к шагу 9.

Шаг 7. xmlFile.readNextStartElement().

Шаг 8. tag = xmlFile.name().toString().

Шаг 9. Если tag == "p" перейти к шагу 10, иначе перейти к шагу 11.

Шаг 10. SetParagraphStyle(xmlFile.readElementText

(QXmlStreamReader::ReadElementTextBehaviour::IncludeChildElements)).

Шаг 11. SetTitleStyle(xmlFile).

Шаг 12. SetTotalPagesNumber().

Шаг 13. Конец.

**4.3 Алгоритм по шагам метода void MainWindow::SetBookLabelText**

**(List<QString> bookText, int currentPageNumber)**

Шаг 1. Создание объекта класса QString pageText.

Шаг 2. Переменная int maxStringAmount = book.GetMaxStringAmount().

Шаг 3. Цикл по i = 0 пока i < maxStringAmount с шагом 1.

Шаг 4. Переменная int index = (currentPageNumber - 1) \* maxStringAmount + i.

Шаг 5. Если index > bookText.size() – 1 перейти к шагу 7.

Шаг 6. pageText.append(bookText[index]).

Шаг 7. ui->bookLabel->setText(pageText).

Шаг 8. Конец.

**4.4 Блок-схема метода void Book::SetTitleStyle(QXmlStreamReader& xmlFile)**

Данный метод предназначен для того, чтобы установить различные заголовки книги по центру жирным шрифтом.

Блок-схема данного метода приведена на чертеже

**4.5 Блок-схема метода void Book::SetParagraphStyle(QString paragraphText)**

Данный метод предназначен для того, чтобы установить абзац текста книги с соответствующими ему стилями.

Блок-схема данного метода приведена на чертеже

**5 РУКОВОДСТВО ПОЛЬЗОВАТЕЛЯ**

Для удобства написания программы было решены создать две папки: include и src. Папка include содержит в себе заголовочные файлы каждого класса, а папка include – файлы с исходным кодом. Так же была создана папка img, в который хранятся все изображения, используемые в программе.

Для того, чтобы запустить программу, нужно с официального сайта скачать библиотеку Qt. Далее необходимо создать проект Qt Widget Application, добавить в него папку с include, src и img и запустить проект.

Для того, чтобы начать читать, необходимо:

1. Нажать кнопку “Выбрать файл” в левой части окна;
2. В открывшемся диалоговом окне выбрать файл, имеющий расширения .fb2;
3. Нажать кнопку “Открыть”.

После открытия файла можно перейти на следующую страницы, нажав

кнопку ->; перейти на предыдущую страницу, нажав кнопку <-; ввести в поле внизу экрана номер страницы, на которую необходимо перейти, и нажать на кнопку с изображенной на ней лупой.

Также в программе реализована функция открытия ранее открытых файлов в случае, если таковые имеются. Для этого необходимо нажать кнопку “Выбрать ранее открытый файл” и в появившемся диалоговом окне выбрать любой из ранее открытых файлов. После этого выбранный файл будет открыть на последней открытой странице. При необходимости ранее выбранные файлы можно удалять. Для этого необходимо нажать на кнопку с изображенной на ней мусоркой.

**ЗАКЛЮЧЕНИЕ**

В ходе данного курсового проекта было изучено объектно-ориентированное программирование на примере языка C++, а так же фреймворк Qt. В конечном итоге была разработана программа для чтения книг, имеющая понятный пользовательский интерфейс с возможностью открывать fb2-файлы. Однако данный проект может быть усовершенствован в одном из следующих направлений:

* добавление возможности открыть электронные книги в различных форматах (например epub или pdf);
* улучшение пользовательского интерфейса;
* добавление возможности масштабировать текст;
* добавление возможности меня свойства текста (шрифт, размер).