Петрозаводский государственный университет
Институт математики и информационных технологий
Кафедра информатики и математического обеспечения

Направление подготовки бакалавриата

09.03.04 — Программная инженерия

Отчет о проектной работе по курсу «Основы информатики и программирования»

Приложение «Qt File Manager»

Выполнил:
студент 1 курса группы 22107

Д. А. Куусела _____
подпись

Петрозаводск — 2021

Содержание

Введение

Цель проекта: разработать простой файловый менеджер на C++ и QtQuick.

Задачи проекта:

1. Научиться считывать содержимое директорий при помощи C++
2. Научиться получать информацию о файлах и директориях (размер и дату изменения)
3. Организовать хранение данных о файлах
4. Разработать интерфейс приложения
5. Разработать функции для загрузки информации в QML
6. Реализовать переход по директориям

1 Требования к приложению

1. Возможность просматривать содержимое директорий
2. Возможность переходить в директорию по клику
3. Возможность открыть файл в приложении по умолчанию

2 Проектирование приложения

2.1 main.cpp

Главный C++ файл, запускает создаёт QGuiApplication и передаёт в контекст приложения модуль Utils.

```
// добавляем Utils в QML
Utils utils;
engine.rootContext()->setContextProperty("utils", &utils);
```

2.2 utils.cpp/utils.h

Класс Utils. Содержит функции для получения и хранения информации о файлах. Также реализует функции для изменения текущей рабочей директории (переход в директорию). Конструктор класса Utils:

```
Utils::Utils(QObject *parent) : QObject(parent)
{
    locale = QLocale::system();
    setPath(QDir::currentPath());
    readContentInfo();
}
```

Основные функции и переменные класса:

1. QHash<QString, QFileInfo> currentContent – информация о файлах в текущей директории
2. QList<QString> getDirContent(QString path) – получает список файлов (абсолютные пути) в папке
3. readContentInfo – записывает содержимое текущей директории в currentContent
4. void requestUpdate() – публичный слот для запроса из QML на обновление данных
5. bool isFile(QString path), bool isDir(QString path), bool isAudioFile(QString path) и др. – функции для определения типа файла
6. bool setPath(QString newPath) – переход по пути newPath, если он существует
7. bool openFile(QString path) – открытие файла в приложении по умолчанию (например VLC для .mp4 файла)

2.3 BottomBar.qml

Нижняя панель приложения с информацией.

2.4 DirEntry.qml

Элемент для отображения файла. Содержит иконку, имя и другие данные.

2.5 main.qml

Основной код интерфейса. Отображает список файлов (DirEntry) в виде GridView.

3 Реализация приложения

Для работы с файлами и директориями используются библиотеки Qt:

- QDir
- QDirIterator
- QFile
- QFileInfo
- QMimeType
- QMimeDatabase

Для форматирования дат и приведения размера файла в human readable формат используются библиотеки QDateTime и QLocale.

Для открытия файла в приложении по умолчанию используются QDesktopServices и QUrl.

В итоге приложение содержит:

- 5 модулей (2 C++ и 3 QML)
- 1 C++ класс с 19 функциями
- 150 строк C++ кода и 308 строк QML кода

Заключение

В результате реализован простой файловый менеджер, который позволяет просматривать содержимое пользовательских директорий. Приложение различает аудио, видео, фото, а также файлы с иходным кодом, для разных видов файлов загружаются разные иконки. Реализованы все запланированные функции.

Скриншот приложения: