МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ

«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

КАФЕДРА «ИИТ»

Лабораторная работа №1,2

По дисциплине «ОСиСП»

Выполнила:

студент ФЭИС

группы ПО-5

Нерода А. А.

Проверил:

Дряпко А. В.

Брест 2021

**Цель: приобрести практические навыки проектирования и разработки приложений с графическим пользовательским интерфейсом в ОС Windows средствами Qt**

**Задание:**

1) Выбрать тему из перечисленных ниже или предложить свою (тематика – игры, системные

программы и утилиты для ОС Windows);

2) Вписать свою фамилию напротив выбранной темы в файле;

3) Разработать программу с графическим пользовательским интерфейсом, реализующую указан-

ный функционал, с использованием фреймворка Qt.

**Код программы:**

**main.cpp**  
  
#include "player.h"

#include <QApplication>

int **main**(int argc, char \*argv[]) {

QApplication a(argc, argv);

Player w;

w.show();

*return* a.exec();

}

**player.cpp**

#include "player.h"

#include "ui\_player.h"

Player::Player(QWidget \*parent) : QMainWindow(parent), ui(*new* Ui::Player) {

ui->setupUi(*this*);

ui->just\_plauingTrack->setVisible(*false*);

playListModel = *new* QStandardItemModel(*this*);

ui->playlistView->setModel(playListModel);

playListModel->setHorizontalHeaderLabels(QStringList() << tr("Audio Track")<< tr("File Path"));

ui->playlistView->hideColumn(1);

ui->playlistView->verticalHeader()->setVisible(*false*);

ui->playlistView->setSelectionMode(QAbstractItemView::SingleSelection); *//* *Разрешаем* *выделять* *только* *одну* *строку*

ui->playlistView->setEditTriggers(QAbstractItemView::NoEditTriggers); *//* *Отключаем* *редактирование*

ui->playlistView->horizontalHeader()->setStretchLastSection(*true*); *//* *по* *ширине* *строки*

player = *new* QMediaPlayer(*this*);

playlist = *new* QMediaPlaylist(player);

player->setPlaylist(playlist);

playlist->setPlaybackMode(QMediaPlaylist::Loop);

connect(playlist, &QMediaPlaylist::currentIndexChanged, [*this*](int index) {

ui->just\_plauingTrack->setText(playListModel->data(playListModel->index(index, 0)).toString());

});

}

Player::~**Player**() {

*delete* ui;

*delete* playListModel;

*delete* playlist;

*delete* player;

}

void Player::**on\_btn\_add\_clicked**() {

QStringList new\_files = QFileDialog::getOpenFileNames(*this*, tr("Open files"), QString(), tr("Audio Files (\*.mp3)"));

*if*(!new\_files.isEmpty()) {

QStringList write\_to\_globallist;

*if*(count\_tracks\_num != 0) {

*for*(int i = 0; i < new\_files.size(); i++) {

bool flag = *false*;

*for*(int j = 0; j < files\_list.size(); j++) {

*if*(new\_files[i] == files\_list[j]) {

flag = *true*;

}

}

*if*(!flag) {

write\_to\_globallist.append(new\_files[i]);

}

}

add\_playlist(write\_to\_globallist);

}

*else* add\_playlist(new\_files);

}

}

void Player::add\_playlist(QStringList list) {

*//* *устанавливаем* *данные* *по* *именам* *и* *пути* *к* *файлам*

*//* *в* *плейлист* *и* *таблицу* *отображающую* *плейлист*

*foreach* (QString filePath, list) {

files\_list.append(list);

count\_tracks\_num++;

QList<QStandardItem \*> items;

items.append(*new* QStandardItem(QDir(filePath).dirName()));

items.append(*new* QStandardItem(filePath));

playListModel->appendRow(items);

playlist->addMedia(QUrl(filePath));

}

ui->count\_tracks->setText(QString::number(count\_tracks\_num));

}

void Player::**on\_btn\_play\_clicked**() {

player->play();

ui->just\_plauingTrack->setVisible(*true*);

}

void Player::**on\_btn\_previous\_clicked**() {

playlist->previous();

}

void Player::**on\_btn\_pause\_clicked**() {

player->pause();

}

void Player::**on\_btn\_stop\_clicked**() {

player->stop();

ui->just\_plauingTrack->setVisible(*false*);

}

void Player::**on\_btn\_next\_clicked**() {

playlist->next();

}

void Player::on\_playlistView\_doubleClicked(*const* QModelIndex &index) {

playlist->setCurrentIndex(index.row());

on\_btn\_play\_clicked();

}

**player.h**

#ifndef PLAYER\_H

#define PLAYER\_H

#include <QMainWindow>

#include <QWidget>

#include <QFileDialog>

#include <QStandardItemModel>

#include <QMediaPlayer>

#include <QMediaPlaylist>

#include <QDebug>

#include <math.h>

QT\_BEGIN\_NAMESPACE

*namespace* **Ui** { *class* **Player**; }

QT\_END\_NAMESPACE

*class* Player : *public* QMainWindow {

Q\_OBJECT

*public*:

Player(QWidget \*parent = *nullptr*);

~Player();

*private* *slots*:

void add\_playlist(QStringList list);

void on\_btn\_add\_clicked();

void on\_btn\_play\_clicked();

void on\_btn\_previous\_clicked();

void on\_btn\_pause\_clicked();

void on\_btn\_stop\_clicked();

void on\_btn\_next\_clicked();

void on\_playlistView\_doubleClicked(*const* QModelIndex &index);

*private*:

Ui::Player \*ui;

QStringList files\_list;

int count\_tracks\_num = 0;

QStandardItemModel \*playListModel;

QMediaPlayer \*player;

QMediaPlaylist \*playlist;

};

#endif *//* *PLAYER\_H*

**Результаты работы программы:**

