Прикладное программирование

Практика №9

Дата 20.02.2023

Практика 09

Библиотека Qt: обработчик событий таймера

Для каждой программы, состоящей из нескольких файлов,

создавайте отдельную папку проекта, например project09-1,

project09-2 и т.д., где "09" - номер практического

занятия, 1, 2 и т.д. - номер задания.

Каждую программу, состоящую из одного файла, сохранять

в отдельном файле для каждого упражнения,

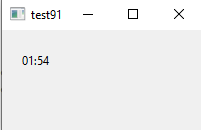
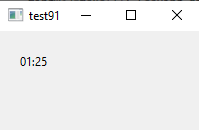
например prog09-1, prog09-2 и т.д., где "09" - номер практического

занятия, 1, 2 и т.д. - номер задания.

1. По примеру 11 из лекции создайте приложение с графическим интерфейсом с применением библиотеки Qt. Приложение моделирует работу секундного таймера. Сначала создайте одномодульную программу с единственным, главным, модулем main.cpp. Затем выполните рефакторинг (реинжениринг) исходного кода программы: создайте отдельный модуль библиотеки с классом MainWindow, которая подключается к основному модулю main.cpp.

Решение.

Скриншот:

Исходный код:

//mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QLabel>

#include <QTime>

#include <QApplication>

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

void Ftimer();

private:

QLabel \*timer;

};

#endif

//mainwindow.cpp

#include "mainwindow.h"

MainWindow::MainWindow(QWidget \*parent) : QMainWindow(parent) {

timer = new QLabel(this);

timer->setGeometry(20,20,30,20);

}

MainWindow::~MainWindow() {

}

void MainWindow::Ftimer() {

int t=120;

while(t>0) {

t--;

QString m = QString::number(t/60);

QString s = QString::number(t-t/60\*60);

if(t/60 < 10 && t-t/60\*60 < 10) timer->setText("0"+m+":"+"0"+s);

if(t/60 < 10 && t-t/60\*60 > 10) timer->setText("0"+m+":"+s);

if(t/60 > 10 && t-t/60\*60 < 10) timer->setText(m+":"+"0"+s);

if(t/60 > 10 && t-t/60\*60 > 10) timer->setText(m+":"+s);

QTime dieTime= QTime::currentTime().addSecs(1);

while (QTime::currentTime() < dieTime)

QCoreApplication::processEvents(QEventLoop::AllEvents, 100);

}

}

//main.cpp

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication app(argc, argv);

MainWindow win;

win.show();

win.Ftimer();

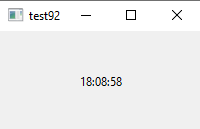
return app.exec();

}

2. В проекте из задания 1 добавьте вывод часов и минут, создайте проект anim021.

Решение:

Скриншот:



Исходный код:

//mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QLabel>

#include <QTime>

#include <QApplication>

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

void Ftimer();

private:

QLabel \*timer;

};

#endif

//mainwindow.cpp

#include "mainwindow.h"

#include <iostream>

#include <QString>

MainWindow::MainWindow(QWidget \*parent) : QMainWindow(parent) {

timer = new QLabel(this);

timer->setGeometry(80,40,80,20);

}

MainWindow::~MainWindow() {

}

void MainWindow::Ftimer() {

int t=65340;

while(t>0) {

t--;

int hour = t/3600;

int minute = (t-hour\*3600)/60;

int second = t-hour\*3600-minute\*60;

QString h = QString::number(hour);

QString m = QString::number(minute);

QString s = QString::number(second);

if(h.length()==1) h="0"+h;

if(m.length()==1) m="0"+m;

if(s.length()==1) s="0"+s;

timer->setText(h+":"+m+":"+s);

QTime dieTime= QTime::currentTime().addSecs(1);

while (QTime::currentTime() < dieTime)

QCoreApplication::processEvents(QEventLoop::AllEvents, 100);

}

}

//main.cpp

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication app(argc, argv);

MainWindow win;

win.show();

win.Ftimer();

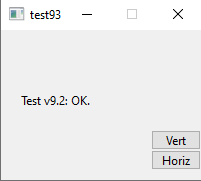
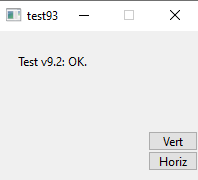
return app.exec();

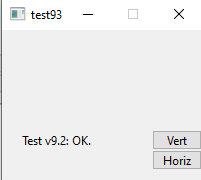
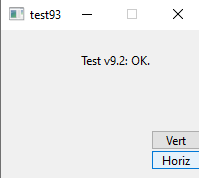
}

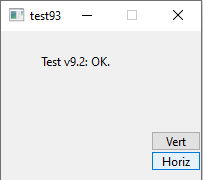
3. Создайте новый проект на основе проекта из задания 1 так, чтобы с помощью QTimer имитировать движение компонента QLabel по горизонтали при нажатии на кнопку Horiz, движение компонента QLabel по вертикали - при нажатии на кнопку Vert.

Решение

Скриншот:







Исходный код:

//mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QLabel>

#include <QTime>

#include <QApplication>

#include <QPushButton>

#include <QTimer>

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private:

QLabel \*Label1;

QTimer \*timer;

QPushButton \*horizbtn;

QPushButton \*vertbtn;

private slots:

void hor();

void ver();

void goH();

void goV();

};

#endif

//mainwindow.cpp

#include "mainwindow.h"

#include <iostream>

int hori = 20;

int vert = 20;

int k = 1;

MainWindow::MainWindow(QWidget \*parent) : QMainWindow(parent) {

this->setFixedWidth(200);

this->setFixedHeight(150);

Label1 = new QLabel(this);

Label1->setGeometry(20,20,100,20);

Label1->setText("Test v9.2: OK.");

horizbtn = new QPushButton(this);

vertbtn = new QPushButton(this);

timer = new QTimer(this);

horizbtn->setText("Vert");

vertbtn->setText("Horiz");

horizbtn->setGeometry(150,100,50,20);

vertbtn->setGeometry(150,120,50,20);

connect(horizbtn, &QPushButton::clicked, this, &MainWindow::hor);

connect(vertbtn, &QPushButton::clicked, this, &MainWindow::ver);

}

MainWindow::~MainWindow() {

}

void MainWindow::goH () {

if(k==1) {

hori=hori+20;

Label1->setGeometry(20,hori,100,20);

}

if(k==0) {

hori=hori-20;

Label1->setGeometry(20,hori,100,20);

}

if(hori>=170) k=0;

if(hori<=10) k=1;

}

void MainWindow::goV () {

if(k==1) {

vert=vert+20;

Label1->setGeometry(vert,20,100,20);

}

if(k==0) {

vert=vert-20;

Label1->setGeometry(vert,20,100,20);

}

if(vert>=140) k=0;

if(vert<=10) k=1;

}

void MainWindow::hor() {

timer->stop();

Label1->setGeometry(20,20,100,20);

connect(timer, SIGNAL(timeout()), SLOT(goH()));

timer->start(1000);

}

void MainWindow::ver() {

timer->stop();

Label1->setGeometry(20,20,100,20);

connect(timer, SIGNAL(timeout()), SLOT(goV()));

timer->start(1000);

}

//main.cpp

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication app(argc, argv);

MainWindow win;

win.show();

return app.exec();

}

4. По примеру 11 из лекции создайте приложение с графическим интерфейсом с применением библиотеки Qt, в котором используется таймер для объектной модели автомобильной дороги, т.е. с подключением библиотеки TRoad с классом TCar. Движение автомобилей имитируется с помощью изменения координаты Left трех компонент QLabel с надписями "A", "B", "C". В программе создается массив объектов класса TCar, свойство координата X каждого автомобиля привязывается к свойству Left

соответствующей надписи.

Скриншот:

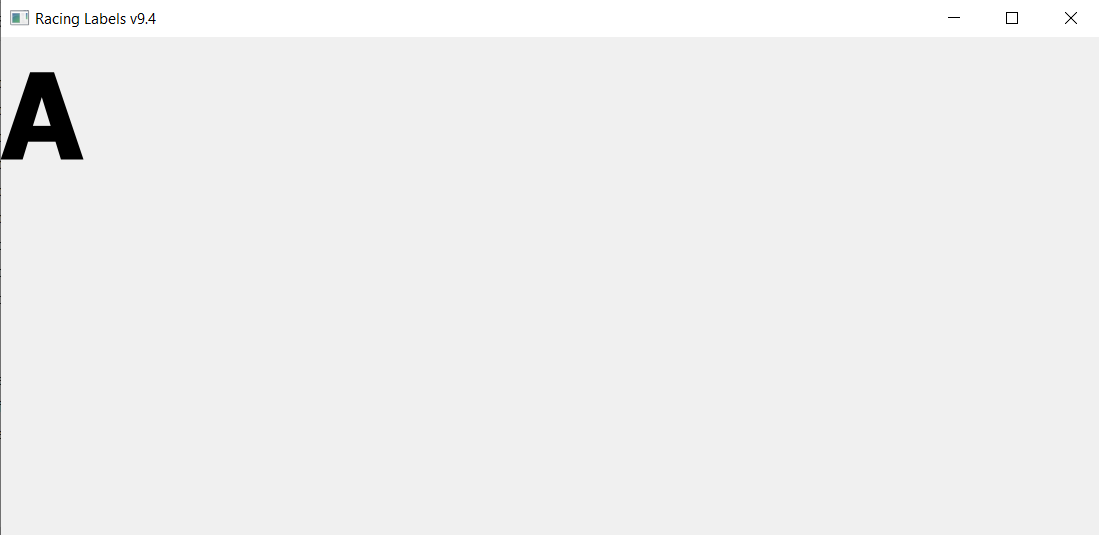


Рис.1 - “Вывод символа А на окно”

Исходный код:

#include <QtWidgets>

class MainWindow : public QMainWindow

{

public:

MainWindow();

private:

QLabel \*carALabel;

};

MainWindow::MainWindow()

{

resize(1100,500);

setWindowTitle("Racing Labels v9.4");

QFont labelFont("TypeWriter",72,QFont::Bold);

carALabel = new QLabel("A",this);

carALabel->setGeometry(0,0,100,150);

carALabel->setFont(labelFont);

}

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

QApplication app(argc, argv);

MainWindow mainWin;

mainWin.show();

return app.exec();

}

Скриншот:

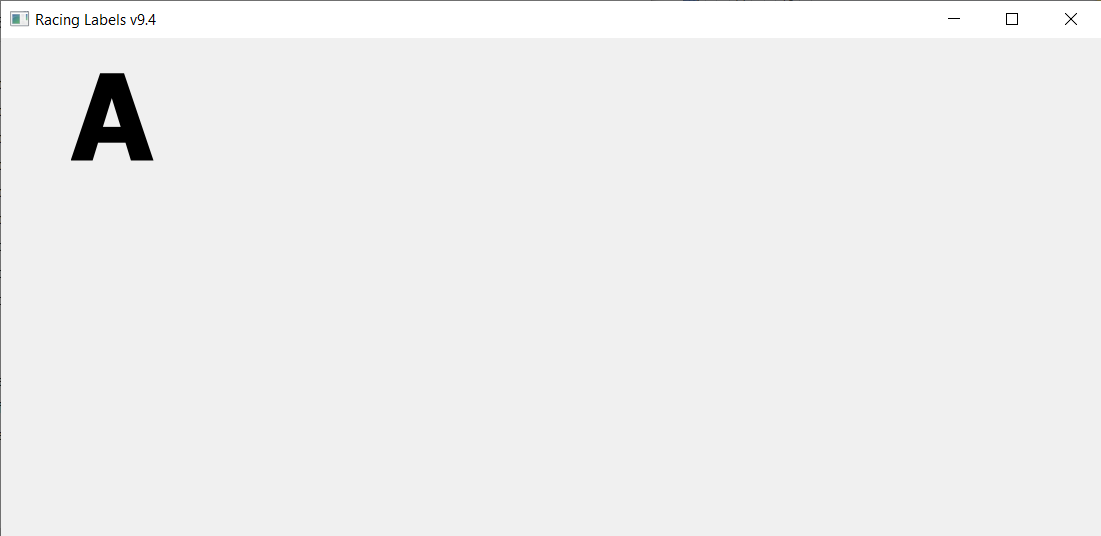


Рис. 2 - “Движение А”

Исходный код:

#include <QtWidgets>

class MainWindow : public QMainWindow

{

public:

MainWindow();

void updateCars();

private:

int xCarA = 0;

QLabel \*carALabel;

QTimer \*timer;

};

MainWindow::MainWindow()

{

resize(1100,500);

setWindowTitle("Racing Labels v9.4");

QFont labelFont("TypeWriter",72,QFont::Bold);

carALabel = new QLabel("A",this);

carALabel->setGeometry(0,0,100,150);

carALabel->setFont(labelFont);

timer = new QTimer(this);

timer->setInterval(200);

timer->start();

connect(timer, &QTimer::timeout,this, &MainWindow::updateCars);

}

void MainWindow::updateCars()

{

xCarA+=5;

carALabel->setGeometry(xCarA,0,100,150);

}

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

QApplication app(argc, argv);

MainWindow mainWin;

mainWin.show();

return app.exec();

}

Скриншот:

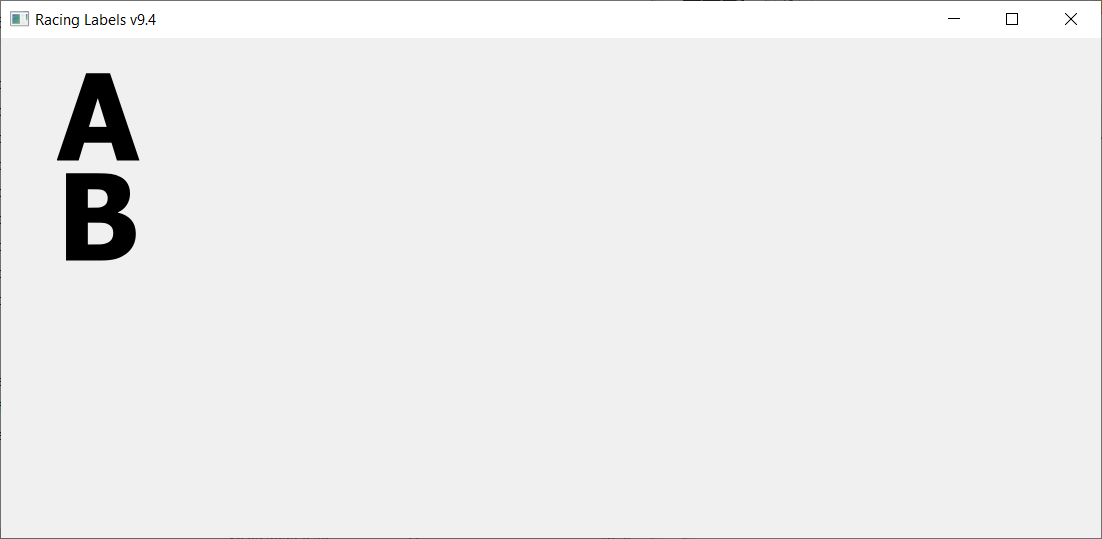


Рис.3 - “Равномерное передвижение А и В”

Исходный код:

#include <QtWidgets>

class Car

{

public:

Car(int newP, double newV);

double X;

int P;

double V;

void move();

};

Car::Car(int newP, double newV)

{

X = 0;

P = newP;

V = newV;

}

class MainWindow : public QMainWindow

{

public:

Car \*\_carA;

Car \*\_carB;

MainWindow(Car \*ptrCarA, Car \*ptrCarB);

void updateCars();

private:

int xCarA = 0;

int xCarB = 0;

int timerInterval;

QLabel \*carALabel;

QTimer \*timer;

QLabel \*carBLabel;

};

MainWindow::MainWindow(Car \*ptrCarA, Car \*ptrCarB)

{

\_carA = ptrCarA;

\_carB = ptrCarB;

resize(1100,500);

setWindowTitle("Racing Labels v9.4");

QFont labelAFont("TypeWriter",72,QFont::Bold);

carALabel = new QLabel("A",this);

carALabel->setGeometry(0,0,100,150);

carALabel->setFont(labelAFont);

QFont labelBFont("Fantasy",72,QFont::Bold);

carBLabel = new QLabel("B",this);

carBLabel->setGeometry(0,100,100,150);

carBLabel->setFont(labelBFont);

timer = new QTimer(this);

timer->setInterval(200);

timer->start();

connect(timer, &QTimer::timeout,this, &MainWindow::updateCars);

}

void MainWindow::updateCars()

{

xCarA+=\_carB->V;

carALabel->setGeometry(xCarA,0,100,150);

xCarB+=\_carB->V;

carBLabel->setGeometry(xCarB,100,100,150);

}

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

QApplication app(argc, argv);

Car carA(1, 2);

Car carB(1, 4);

MainWindow mainWin(&carA,&carB);

mainWin.show();

return app.exec();

}

///

#include <QtWidgets>

class Road

{

public:

Road(double newLength, int newWidth);

double Length;

int Width;

};

Road::Road(double newLength, int newWidth)

{

Length = newLength;

Width = newWidth;

}

class Car

{

public:

Car(int newP, double newV, Road \*ptrRoad);

Road \*\_road;

double X;

int P;

double V;

void move();

};

Car::Car(int newP, double newV, Road \*ptrRoad)

{

\_road = ptrRoad;

X = 0;

P = newP;

V = newV;

}

void Car::move()

{

X+=V;

if (X>\_road->Length)

{

X=0;

}

}

class MainWindow : public QMainWindow

{

public:

Car \*\_carA;

Car \*\_carB;

MainWindow(Car \*ptrCarA, Car \*ptrCarB);

void updateCars();

private:

int xCarA = 0;

int xCarB = 0;

int timerInterval;

QLabel \*carALabel;

QTimer \*timer;

QLabel \*carBLabel;

};

MainWindow::MainWindow(Car \*ptrCarA, Car \*ptrCarB)

{

\_carA = ptrCarA;

\_carB = ptrCarB;

resize(1100,500);

setWindowTitle("Racing Labels v9.4");

QFont labelAFont("TypeWriter",72,QFont::Bold);

carALabel = new QLabel("A",this);

carALabel->setGeometry(0,0,100,150);

carALabel->setFont(labelAFont);

QFont labelBFont("Fantasy",72,QFont::Bold);

carBLabel = new QLabel("B",this);

carBLabel->setGeometry(0,100,100,150);

carBLabel->setFont(labelBFont);

timer = new QTimer(this);

timer->setInterval(200);

timer->start();

connect(timer, &QTimer::timeout,this, &MainWindow::updateCars);

}

void MainWindow::updateCars()

{

//xCarA+=\_carB->V;

\_carA->move();

xCarA = \_carA->X;

carALabel->setGeometry(xCarA,0,100,150);

//xCarB+=\_carB->V;

\_carB->move();

xCarB = \_carB->X;

carBLabel->setGeometry(xCarB,100,100,150);

}

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

QApplication app(argc, argv);

Road road(300,3);

Car carA(1, 6, &road);

Car carB(1, 9, &road);

MainWindow mainWin(&carA,&carB);

mainWin.show();

return app.exec();

}

Скриншот:

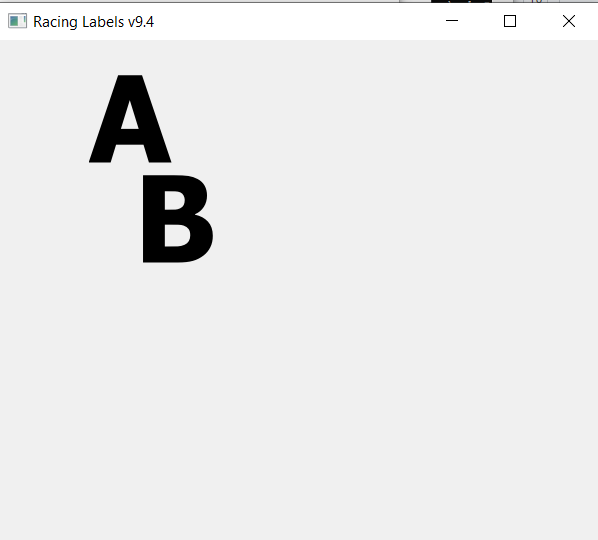


Рис.4 - “Движение А и В”

Исходный код:

#include <QtWidgets>

const int LENGTH\_ROAD = 600;

class Road

{

public:

Road(double newLength, int newWidth);

double Length;

int Width;

};

Road::Road(double newLength, int newWidth)

{

Length = newLength;

Width = newWidth;

}

class Car

{

public:

Car(int newP, double newV, Road \*ptrRoad);

Road \*\_road;

double X;

int P;

double V;

void move();

};

Car::Car(int newP, double newV, Road \*ptrRoad)

{

\_road = ptrRoad;

X = 0;

P = newP;

V = newV;

}

void Car::move()

{

X+=V;

if (X>\_road->Length)

{

X=0;

}

}

class MainWindow : public QMainWindow

{

public:

Car \*\_carA;

Car \*\_carB;

MainWindow(Car \*ptrCarA, Car \*ptrCarB);

void updateCars();

private:

int xCarA = 0;

int xCarB = 0;

int timerInterval;

QLabel \*carALabel;

QTimer \*timer;

QLabel \*carBLabel;

};

MainWindow::MainWindow(Car \*ptrCarA, Car \*ptrCarB)

{

\_carA = ptrCarA;

\_carB = ptrCarB;

resize(LENGTH\_ROAD,500);

setWindowTitle("Racing Labels v9.4");

QFont labelAFont("TypeWriter",72,QFont::Bold);

carALabel = new QLabel("A",this);

carALabel->setGeometry(0,0,100,150);

carALabel->setFont(labelAFont);

QFont labelBFont("Fantasy",72,QFont::Bold);

carBLabel = new QLabel("B",this);

carBLabel->setGeometry(0,100,100,150);

carBLabel->setFont(labelBFont);

timer = new QTimer(this);

timer->setInterval(200);

timer->start();

connect(timer, &QTimer::timeout,this, &MainWindow::updateCars);

}

void MainWindow::updateCars()

{

//xCarA+=\_carB->V;

\_carA->move();

xCarA = \_carA->X;

carALabel->setGeometry(xCarA,0,100,150);

//xCarB+=\_carB->V;

\_carB->move();

xCarB = \_carB->X;

carBLabel->setGeometry(xCarB,100,100,150);

}

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

QApplication app(argc, argv);

Road road(LENGTH\_ROAD,3);

Car carA(1, 6, &road);

Car carB(1, 9, &road);

MainWindow mainWin(&carA,&carB);

mainWin.show();

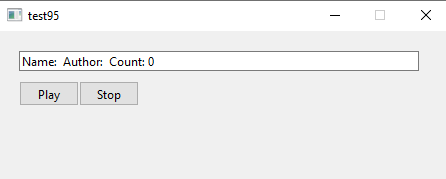
return app.exec();

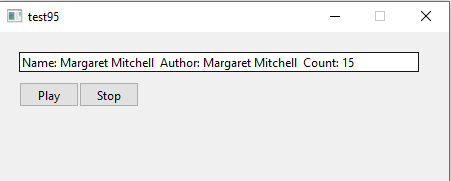
}

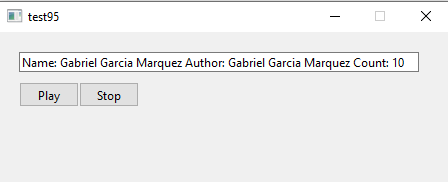
Выводы: По примеру 11 из лекции создайте приложению с графическим интерфейсом с применением библиотеки Qt, в котором используется таймер для объектной модели автомобильной дороги, т.е. с подключением библиотеки TRoad с классом TCar. Создал имитируемое движение автомобилей с помощью изменения координаты Left трех компонент QLabel с надписями "A", "B", "C". В программе создается массив объектов класса TCar, свойство координата X каждого автомобиля привязывается к свойству Left соответствующей надписи.

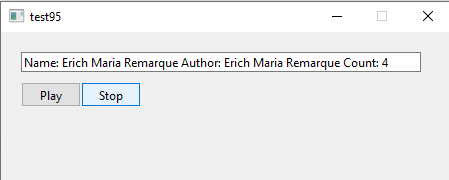
5. Слайд-шоу базы данных TBook. Подключите к проекту библиотеку TBook. Создайте текстовый файл, в котором хранятся сведения о 10 книгах. Из этого файла свойства должны загружаться в массив объектов TBook. Создайте 2 кнопки с надписями Play и Stop. При нажатии на Play запускается вывод сведений о каждой книге в автоматическом режиме с заданным в поле QLineEdit интервалом. При нажатии на кнопку Stop автоматический вывод прекращается.

Скриншот









Исходный код:

//mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include <QMainWindow>

#include <QLabel>

#include <QApplication>

#include <QPushButton>

#include <QTimer>

#include <QLineEdit>

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

MainWindow(QWidget \*parent = nullptr);

~MainWindow();

private:

QLineEdit \*Book;

QTimer \*timer;

QPushButton \*startbutton;

QPushButton \*stopbutton;

private slots:

void showBooks();

void starting();

void stoping();

};

#endif

//mainwindow.cpp

#include "mainwindow.h"

#include <tbook.h>

#include <string>

#include <fstream>

#include <iostream>

TBook books[10];

int i=0;

MainWindow::MainWindow(QWidget \*parent) : QMainWindow(parent) {

this->setFixedWidth(450);

this->setFixedHeight(150);

ifstream fin;

fin.open("C:/Sivkov David/Homework ggpi/9/5/test95/book.txt");

if (fin.is\_open()) {

for (int var = 0; var < 10; var++) {

string temp;

getline(fin, books[var].Name);

getline(fin, books[var].Author);

getline(fin, temp);

books[var].Count = stoi(temp);

}

fin.close();

} else cout << "Can't find a file!" << endl;

Book = new QLineEdit(this);

startbutton = new QPushButton(this);

stopbutton = new QPushButton(this);

Book->setGeometry(20,20,400,20);

startbutton->setGeometry(20,50,60,25);

stopbutton->setGeometry(80,50,60,25);

startbutton->setText("Play");

stopbutton->setText("Stop");

connect(startbutton, &QPushButton::clicked, this, &MainWindow::starting);

connect(stopbutton, &QPushButton::clicked, this, &MainWindow::stoping);

}

MainWindow::~MainWindow() {

}

void MainWindow::showBooks() {

if(i==9) i=0;

Book->setText("Name: " + QString::fromStdString(books[i].Name) +

" Author: " + QString::fromStdString(books[i].Name) +

" Count: " + QString::fromStdString(to\_string(books[i].Count)));

i++;

}

void MainWindow::starting() {

timer = new QTimer(this);

connect(timer, SIGNAL(timeout()), SLOT(showBooks()));

timer->start(1000);

}

void MainWindow::stoping() {

timer->stop();

}

//main.cpp

#include "mainwindow.h"

#include <QApplication>

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

{

QApplication app(argc, argv);

MainWindow win;

win.show();

return app.exec();

}

//tbook.h

#ifndef TBOOK\_H

#define TBOOK\_H

#include <string>

using namespace std;

class TBook {

public:

string Author;

string Name;

int Count;

TBook() {}

};

#endif