Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Вятский государственный университет»

Колледж ВятГУ

**ОТЧЕТ**

**ПО ДОМАШНЕЙ КОНТРОЛЬНОЙ РАБОТЕ №8**

**«ОСНОВЫ СОБЫТИЙНО-ОРИЕНТИРОВАННОГО ПРОГРАММИРОВАНИЯ»**

**«МДК 05.02 РАЗРАБОТКА КОДА ИНФОРМАЦИОННЫХ СИСТЕМ»**

Выполнила: студентка учебной группы

ИСПк-203-52-00

Бочкарёва Виктория Дмитриевна

Преподаватель:

Сергеева Елизавета Григорьевна

Киров

2024

**Цель работы**

Получение базовых навыков реализации приложений с графическим интерфейсом на основе событийно-ориентированной парадигмы.

**Формулировка задания**

Фамилия: Бочкарева

Вариант: 3

Задание:

1. Реализовать приложение «Таймер».
2. Предусмотреть возможность подачи звукового сигнала по истечении заданного промежутка времени.
3. Предусмотреть возможность досрочной остановки таймера.

**Описание алгоритма**

Создается форма TMain с различными компонентами (кнопки, поле для ввода времени, индикатор прогресса и т.д.).

Объявляются переменные Countdown и TotalTime для хранения значения времени обратного отсчета.

Пользователь вводит время в поле ввода в формате "ЧЧ:ММ:СС".

При нажатии кнопки "Start" происходит следующее:

a. Проверяется корректность введенного времени.

b. Введенное время преобразуется в секунды и сохраняется в переменной Countdown.

c. Значение Countdown также сохраняется в переменной TotalTime.

d. Запускается таймер (Timer), который будет вызывать процедуру TimerTimer с заданным интервалом.

При срабатывании таймера вызывается процедура TimerTimer, которая выполняет следующие действия:

a. Уменьшает значение Countdown на 1 секунду.

b. Преобразует оставшееся время в часы, минуты и секунды.

c. Обновляет отображение времени на форме (TimeStop.Caption).

d. Обновляет индикатор прогресса (ProgressBar.Position) в соответствии с оставшимся временем.

e. Если Countdown достиг 0, то:

Останавливает таймер.

Воспроизводит звуковой файл "Ringtone.wav".

Отображает сообщение "Время вышло!" в диалоговом окне.

При нажатии кнопки "Pause" изменяется состояние таймера (Timer.Enabled) на противоположное, что приводит к приостановке или возобновлению обратного отсчета.

При нажатии кнопки "Reset" происходит следующее:

a. Останавливается таймер.

b. Значение Countdown восстанавливается до начального значения (TotalTime).

c. Индикатор прогресса и отображение времени сбрасываются.

**Код программы**

Модуль MainUnit:

unit mainUnit;

{$mode objfpc}{$H+}

interface

uses

Classes, SysUtils, Forms, Controls, Graphics, Dialogs, EditBtn, StdCtrls,

MaskEdit, Buttons, ComCtrls, ExtCtrls, MMSystem;

type

{ TMain }

TMain = class(TForm)

Image1: TImage;

Label1: TLabel;

StartButton: TBitBtn;

PauseButton: TBitBtn;

ResetButton: TBitBtn;

TimeStop: TLabel;

MaskEdit1: TMaskEdit;

ProgressBar: TProgressBar;

Timer: TTimer;

procedure Label1Click(Sender: TObject);

procedure StartButtonClick(Sender: TObject);

procedure PauseButtonClick(Sender: TObject);

procedure ResetButtonClick(Sender: TObject);

procedure TimerTimer(Sender: TObject);

private

public

end;

var

Main: TMain;

Countdown: integer;

TotalTime: Integer;

implementation

{$R \*.lfm}

{ TMain }

procedure TMain.StartButtonClick(Sender: TObject);

var

Hours, Minutes, Seconds: Integer;

begin

if Length(MaskEdit1.Text) = 8 then

begin

if TryStrToInt(Copy(MaskEdit1.Text, 1, 2), Hours) and

TryStrToInt(Copy(MaskEdit1.Text, 4, 2), Minutes) and

TryStrToInt(Copy(MaskEdit1.Text, 7, 2), Seconds) then

begin

Countdown := Hours \* 3600 + Minutes \* 60 + Seconds;

TotalTime := Countdown;

// Запуск таймера

Timer.Enabled := True;

end

else

ShowMessage('Введите время в формате ЧЧ:ММ:СС');

end

else

ShowMessage('Введите время в формате ЧЧ:ММ:СС');

end;

procedure TMain.Label1Click(Sender: TObject);

begin

end;

procedure TMain.PauseButtonClick(Sender: TObject);

begin

// Приостановка таймера

Timer.Enabled := not Timer.Enabled;

end;

procedure TMain.ResetButtonClick(Sender: TObject);

begin

// Сброс таймера

Timer.Enabled := False;

Countdown := TotalTime;

ProgressBar.Position := 0;

TimeStop.Caption := '00:00:00';

end;

procedure TMain.TimerTimer(Sender: TObject);

var

Hours, Minutes, Seconds: Integer;

begin

// Уменьшение времени обратного отсчета

Countdown := Countdown - 1;

// Преобразование обратного отсчета обратно в часы, минуты и секунды

Hours := Countdown div 3600;

Minutes := (Countdown mod 3600) div 60;

Seconds := Countdown mod 60;

// Обновление отображаемого времени

TimeStop.Caption := Format('%.2d:%.2d:%.2d', [Hours, Minutes, Seconds]);

// Обновление полосы прогресса

if TotalTime <> 0 then

ProgressBar.Position := Round((TotalTime - Countdown) / TotalTime \* 100)

else

ProgressBar.Position := 0;

// Проверка, достигло ли время нуля

if Countdown <= 0 then

begin

// Остановка таймера

Timer.Enabled := False;

// Воспроизведение звукового сигнала

sndPlaySound('Ringtone.wav', SND\_ASYNC);

// Отображение сообщения

if MessageDlg('Время вышло!', mtInformation, [mbOK], 0) = mrOK then

begin

// Остановка воспроизведения звука, если пользователь нажал OK

sndPlaySound(nil, SND\_ASYNC);

end;

end;

end;

end.

**Результат выполнения программы**

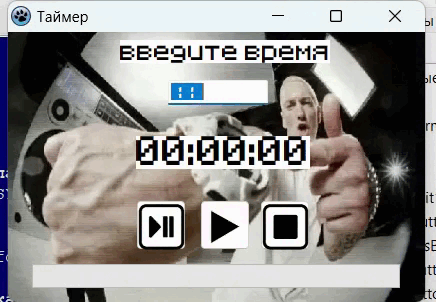
****

Рисунок 1 – Результат выполнения программы

**Вывод**

В результате выполнения домашней контрольной работы поставленная цель была достигнута. Были получены базовые навыки реализации приложений с графическим интерфейсом на основе событийно-ориентированной парадигмы.

В процессе работы была написана программа, которая ведет обратный отсчет от времени, установленным пользователем. Для осуществления графического интерфейса пользователя используются кнопки, метки, поле для ввода, изображение. Для отсчета времени используется компонент timer.

Таким образом, решив все поставленные задачи, а также получив новые знания, были достигнута поставленная цель, получены удовлетворяющие результаты.