**Ministerul Educației Tineretului și Sportului**

**Universitatea Tehnică a Moldovei**

**Catedra Tehnologii Informaționale**

Raport

La lucrarea de laborator nr.1 la

Medii interactive de dezvoltare a produselor soft

A efectuat st.gr 144: Roșca Andrei

A verificat: doctor, conf. univ: Cojanu Irina

**Chișinau 2016**

**Tema:** MEDIUL INTEGRAT C++ BUILDER

**Scopul lucrarii:**

**a)** Însuşirea modului de utilizare a celor mai importante componente ale mediului integrat C++ BUILDER . Realizarea unui program simplu care utilizează componente de tip *TButton, TEdit, Tlabel, RadioButton*  etc.

**b)** Însuşirea modului de utilizare a componentei VCL **TTimer.** Însuşirea modului de utilizare a funcţiilor de lucru cu timpul sistem. Realizarea unor aplicaţii de gestionare a resursei timp.

**c)** Însuşirea modului de utilizare a componentelor VCL **TPaintBox** şi **TPanel.** Însuşirea modului de utilizare a principalelor funcţii grafice ale mediului C++BUILDER . Realizarea unor elemente pentru afişarea grafică a informaţiei (diagramă şi bargraf).

**Notiuni teoretice**:

Componenta **TTimer** se găseşte în **Component Palette** (*pagina System*) .

Obiectul de acest tip permite execuţia in cadrul aplicaţiei a unor funcţii la intervale specificate. În context Windows obiectul TTimer lansează către aplicaţie mesaje la intervale prestabilite.

O particularitate faţă de componentele utilizate în lucrarea precedentă constă in faptul ca acest obiect nu are corespondent grafic pe formă în momentul execuţiei programului.

Componenta **TPaintBox** se găseşte în **Component Palette** (*pagina System*) . Obiectul de acest tip furnizează o componentă *TCanvas* care permite desenarea în interiorul unui dreptunghi, prevenind depăşirea marginilor acestuia.

**Codul sursa (a):**

*Counter.cpp*

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

//---------------------------------------------------------------------------

USEFORM("CounterInteractor.cpp", FirstTask);

//---------------------------------------------------------------------------

WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

try

{

Application->Initialize();

Application->CreateForm(\_\_classid(TFirstTask), &FirstTask);

Application->Run();

}

catch (Exception &exception)

{

Application->ShowException(&exception);

}

catch (...)

{

try

{

throw Exception("");

}

catch (Exception &exception)

{

Application->ShowException(&exception);

}

}

return 0;

}

//---------------------------------------------------------------------------

*CounterInteractor.h*

//---------------------------------------------------------------------------

#ifndef CounterInteractorH

#define CounterInteractorH

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include "sButton.hpp"

#include "sEdit.hpp"

#include "sLabel.hpp"

#include <ComCtrls.hpp>

//---------------------------------------------------------------------------

class TFirstTask : public TForm

{

\_\_published: // IDE-managed Components

TsButton \*upButton;

TsButton \*downButton;

TsEdit \*counterEditor;

TsLabel \*taskLabel;

TsLabel \*editorLabel;

TsButton \*exitButton;

TUpDown \*upDownControl;

void \_\_fastcall onUpButtonClick(TObject \*Sender);

void \_\_fastcall onDownButtonClick(TObject \*Sender);

void \_\_fastcall onExitButtonClick(TObject \*Sender);

void \_\_fastcall onUpDownControlChange(TObject \*Sender,

bool &AllowChange, short NewValue, TUpDownDirection Direction);

private: // User declarations

int counter;

public: // User declarations

\_\_fastcall TFirstTask(TComponent\* Owner);

void printCounter();

};

//---------------------------------------------------------------------------

extern PACKAGE TFirstTask \*FirstTask;

//---------------------------------------------------------------------------

#endif

*CounterInteractor.cpp*

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "CounterInteractor.h"

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sButton"

#pragma link "sEdit"

#pragma link "sLabel"

#pragma resource "\*.dfm"

TFirstTask \*FirstTask;

//---------------------------------------------------------------------------

\_\_fastcall TFirstTask::TFirstTask(TComponent\* Owner)

: TForm(Owner)

{

printCounter();

this->counter = 0;

}

//---------------------------------------------------------------------------

void \_\_fastcall TFirstTask::onUpButtonClick(TObject \*Sender)

{

齃>counter;

printCounter();

}

//---------------------------------------------------------------------------

void \_\_fastcall TFirstTask::onDownButtonClick(TObject \*Sender)

{

--this->counter;

printCounter();

}

//---------------------------------------------------------------------------

void TFirstTask::printCounter() {

counterEditor->Text = this->counter;

}

void \_\_fastcall TFirstTask::onExitButtonClick(TObject \*Sender)

{

Close();

}

//---------------------------------------------------------------------------

void \_\_fastcall TFirstTask::onUpDownControlChange(TObject \*Sender,

bool &AllowChange, short NewValue, TUpDownDirection Direction)

{

this->counter = NewValue;

printCounter();

}

//---------------------------------------------------------------------------

**Codul sursa (b):**

*secondTask.cpp*

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

//---------------------------------------------------------------------------

USEFORM("Unit1.cpp", TimerForm);

//---------------------------------------------------------------------------

WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

try

{

Application->Initialize();

Application->CreateForm(\_\_classid(TTimerForm), &TimerForm);

Application->Run();

}

catch (Exception &exception)

{

Application->ShowException(&exception);

}

catch (...)

{

try

{

throw Exception("");

}

catch (Exception &exception)

{

Application->ShowException(&exception);

}

}

return 0;

}

//---------------------------------------------------------------------------

*Unit1.cpp*

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include "Unit1.h"

#include "dos.h"

#include <stdio.h>

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma link "sButton"

#pragma link "sEdit"

#pragma link "sLabel"

#pragma resource "\*.dfm"

TTimerForm \*TimerForm;

static struct time &operator +=( struct time &lhs, int hundredsOfASecond) {

lhs.ti\_hund += hundredsOfASecond;

if (lhs.ti\_hund >= 100) {

++lhs.ti\_sec;

lhs.ti\_hund = 0;

}

if (lhs.ti\_sec >= 60) {

++lhs.ti\_min;

lhs.ti\_sec = 0;

if (lhs.ti\_min >= 60) {

++lhs.ti\_hour;

lhs.ti\_min = 0;

if (lhs.ti\_hour >= 24)

lhs.ti\_hour = 0;

}

}

return lhs;

}

//---------------------------------------------------------------------------

\_\_fastcall TTimerForm::TTimerForm(TComponent\* Owner)

: TForm(Owner)

{

Timer1->Enabled = false;

Timer2->Enabled = false;

stopButton->Enabled = false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TTimerForm::onExitButtonClick(TObject \*Sender)

{

Close();

}

//---------------------------------------------------------------------------

void \_\_fastcall TTimerForm::onStartButtonClick(TObject \*Sender)

{

Timer1->Enabled = true;

Timer2->Enabled = true;

stopButton->Enabled = true;

startButton->Enabled = !stopButton->Enabled;

zeroButton->Enabled = startButton->Enabled;

}

//---------------------------------------------------------------------------

void \_\_fastcall TTimerForm::onStopButtonClick(TObject \*Sender)

{

Timer1->Enabled = false;

Timer2->Enabled = false;

stopButton->Enabled = false;

startButton->Enabled = !stopButton->Enabled;

zeroButton->Enabled = startButton->Enabled;

}

//---------------------------------------------------------------------------

void \_\_fastcall TTimerForm::onFirstTimerCall(TObject \*Sender)

{

static struct time stopwatchTime = {0};

stopwatchTime += 10;

const int MAX\_BUFFER\_SIZE = 1024;

char buffer[MAX\_BUFFER\_SIZE] = {0};

sprintf(buffer, "%02d:%02d:%02d:%02d", stopwatchTime.ti\_hour,

stopwatchTime.ti\_min,

stopwatchTime.ti\_sec, stopwatchTime.ti\_hund);

stopwatchEditor->Text = buffer;

}

//---------------------------------------------------------------------------

void \_\_fastcall TTimerForm::onTimerTimerCall(TObject \*Sender)

{

const int MAX\_BUFFER\_SIZE = 1024;

struct date currentDate;

struct time currentTime;

getdate(&currentDate);

gettime(&currentTime);

char buffer[MAX\_BUFFER\_SIZE] = {0};

sprintf(buffer, "%02d-%02d-%4d %02d:%02d:%02d", currentDate.da\_mon,

currentDate.da\_day, currentDate.da\_year, currentTime.ti\_hour,

currentTime.ti\_min, currentTime.ti\_sec);

timeEditor->Text = buffer;

}

//---------------------------------------------------------------------------

void \_\_fastcall TTimerForm::onZeroButtonClick(TObject \*Sender)

{

stopwatchEditor->Text = "00:00:00:00";

//timeEditor->Text = "";

}

//---------------------------------------------------------------------------

*Unit1.h*

//---------------------------------------------------------------------------

#ifndef Unit1H

#define Unit1H

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include "sButton.hpp"

#include "sEdit.hpp"

#include "sLabel.hpp"

#include <ExtCtrls.hpp>

//---------------------------------------------------------------------------

class TTimerForm : public TForm

{

\_\_published: // IDE-managed Components

TsEdit \*timeEditor;

TsEdit \*stopwatchEditor;

TsButton \*startButton;

TsButton \*stopButton;

TsButton \*zeroButton;

TsLabel \*taskLabel;

TsLabel \*stopwatchLabel;

TsButton \*exitButton;

TTimer \*Timer1;

TTimer \*Timer2;

void \_\_fastcall onExitButtonClick(TObject \*Sender);

void \_\_fastcall onStartButtonClick(TObject \*Sender);

void \_\_fastcall onStopButtonClick(TObject \*Sender);

void \_\_fastcall onFirstTimerCall(TObject \*Sender);

void \_\_fastcall onTimerTimerCall(TObject \*Sender);

void \_\_fastcall onZeroButtonClick(TObject \*Sender);

private: // User declarations

public: // User declarations

\_\_fastcall TTimerForm(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TTimerForm \*TimerForm;

//---------------------------------------------------------------------------

#endif

*Histogram.cpp*

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

//---------------------------------------------------------------------------

USEFORM("Unit2.cpp", Form2);

//---------------------------------------------------------------------------

WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int)

{

try

{

Application->Initialize();

Application->CreateForm(\_\_classid(TForm2), &Form2);

Application->Run();

}

catch (Exception &exception)

{

Application->ShowException(&exception);

}

catch (...)

{

try

{

throw Exception("");

}

catch (Exception &exception)

{

Application->ShowException(&exception);

}

}

return 0;

}

//---------------------------------------------------------------------------

**Codul sursa (c):**

*Unit2.h*

//---------------------------------------------------------------------------

#ifndef Unit2H

#define Unit2H

//---------------------------------------------------------------------------

#include <Classes.hpp>

#include <Controls.hpp>

#include <StdCtrls.hpp>

#include <Forms.hpp>

#include <ExtCtrls.hpp>

//---------------------------------------------------------------------------

class TForm2 : public TForm

{

\_\_published: // IDE-managed Components

TPaintBox \*PaintBox1;

TPanel \*Panel1;

TPanel \*Panel2;

TButton \*Button1;

TButton \*Button2;

TButton \*Button3;

TTimer \*Timer1;

TTimer \*Timer2;

TEdit \*Edit1;

TLabel \*Label1;

TLabel \*Label2;

void \_\_fastcall Timer1Timer(TObject \*Sender);

void \_\_fastcall PaintBox1Paint(TObject \*Sender);

void \_\_fastcall Timer2Timer(TObject \*Sender);

void \_\_fastcall Button1Click(TObject \*Sender);

void \_\_fastcall Button2Click(TObject \*Sender);

void \_\_fastcall Button3Click(TObject \*Sender);

private: // User declarations

public: // User declarations

\_\_fastcall TForm2(TComponent\* Owner);

};

//---------------------------------------------------------------------------

extern PACKAGE TForm2 \*Form2;

//---------------------------------------------------------------------------

#endif

*Unit2.cpp*

//---------------------------------------------------------------------------

#include <vcl.h>

#pragma hdrstop

#include <dos.h>

#include "Unit2.h"

#include <stdio.h>

#include <time.h>

#include <stdlib.h>

//---------------------------------------------------------------------------

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm2 \*Form2;

//---------------------------------------------------------------------------

\_\_fastcall TForm2::TForm2(TComponent\* Owner)

: TForm(Owner)

{

srand(time(NULL));

Timer2->Enabled = false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Timer1Timer(TObject \*Sender)

{

struct time t;

struct date d;

gettime(&t);

getdate(&d);

char buf[30];

sprintf(buf, "%02d-%02d-%02d %02d:%02d:%02d", d.da\_day, d.da\_mon, d.da\_year,t.ti\_hour,t.ti\_min,t.ti\_sec);

Edit1->Text = buf;

}

//---------------------------------------------------------------------------//---------------------------------------------------------------------------}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::PaintBox1Paint(TObject \*Sender)

{

PaintBox1->Canvas->Pen->Color = clBlack;

PaintBox1->Canvas->Brush->Style = bsHorizontal;

for(int i = 0; i <= 240; i += 10){

PaintBox1->Canvas->MoveTo(0,i);

PaintBox1->Canvas->LineTo(240,i);

PaintBox1->Canvas->MoveTo(i,0);

PaintBox1->Canvas->LineTo(i,240);

}

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Timer2Timer(TObject \*Sender)

{

static int x = 0;

int sign = rand() % 2 == 0 ? 1 : -1;

static int y = ((rand() % 2) \* sign + 120);

PaintBox1->Canvas->Pen->Color = clRed;

PaintBox1->Canvas->Brush->Style = bsHorizontal;

Panel1->Height = y;

PaintBox1->Canvas->MoveTo(x , y);

x =((rand() % 3) + x);

y = ((rand() % 40) \* sign + 120);

PaintBox1->Canvas->LineTo(x,y);

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button1Click(TObject \*Sender)

{

Timer2->Enabled = true;

Button1->Enabled = false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button2Click(TObject \*Sender)

{

Button1->Enabled = true;

Timer2->Enabled = false;

}

//---------------------------------------------------------------------------

void \_\_fastcall TForm2::Button3Click(TObject \*Sender)

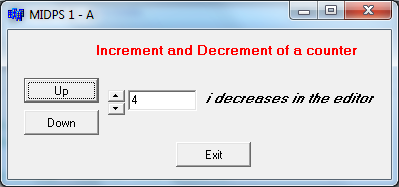
{

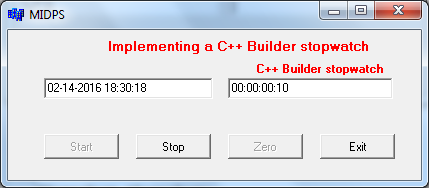
Form2->Close();

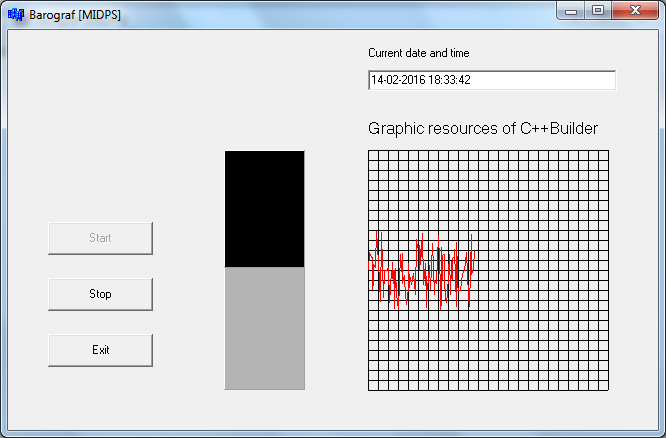
}

//---------------------------------------------------------------------------

Executind aplicatiile, obtinem rezultatul:







Concluzii: In urma efectuarii acestei lucrari de laborator au fost obtinute deprinderi practice de create a aplicatiilor cu interfata grafica utilizind Borland C++ Builder. Au fost studiate asa elemente grafice din VCL ca editoare, buttoane, label-uri, timere, painbox-uri. Am ajuns la concluzia ca astfel de medii interactive usureaza mult procesul de implementare a aplicatiilor.