Universitatea Tehnică a Moldovei

Faculatatea Calculatoare, Informatica și Microelectronică

Catedra Automatica și Tehnologii Informaționale

**Raport**

**Disciplina:** Medii interactive de dezvoltare a produselor soft

Lucrare de laborator Nr. 1

A efectuat: st.gr. TI-144 Olaru Cristian

A verificat: lect. sup. Cojocaru Svetlana

Chisinau 2016

**Lucrare de laborator Nr. 1**

**Obiectivele lucrării**

**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).

**Facilităţile mediului C++Builder**

Borland C++Builder este un mediu de programare vizual, orientat obiect, pentru dezvoltarea rapidă de aplicaţii (**RAD**) cu scop general şi aplicaţii client/server pentru Windows şi WindowsNT. Folosind C++Builder se pot crea aplicaţii Windows eficiente sciind un minim de cod. Facilităţile semnificative oferite de acestea sunt prezentate succint în cele ce urmează.

# Înalta productivitate a mediului de dezvoltare

Aceasta este favorizată de principalele instrumente furnizate de mediul de dezvoltare integrat (**IDE**) C++Builder şi anume :

*- Visual Form Designer;*

*- Object Inspector;*

*- Component Palette;*

*- Project Manager;*

*- Code Editor;*

*- Debugger.*

Acestea dau posibilitatea utilizatorului să dezvolte rapid aplicaţii având totodată un control complet asupra codului şi resurselor.

**Proiectare drag-and-drop**

Utilizatorul poate crea aplicaţii prin simpla *tragere* (drag and drop) a componentelor din *Component Palette* pe *Form designer* urmată de setarea propietăţilor din *Object Inspector*. *Handler-ele* de evenimente sunt automat create, iar codul lor este complet accesibil. Acest mod de proiectare a unei aplicaţii nu restricţionează în nici un fel accesul programatorului la codul sursă, o aplicaţie putând fi scrisă şi fără a folosi componente vizuale.

## Propietăţi, metode, evenimente

Dezvoltarea rapidă a aplicaţiilor înseamnă suport pentru propietăţile, metodele şi evenimentele obiectelor (*PME*). Propietăţile permit setarea uşoară a caracteristicilor componentelor. Metodele execută acţiuni asupra obiectelor. Evenimentele permit ca aplicaţia să răspundă la mesajele Windows, sau la schimbări de stare a obiectelor. Folosirea modelului PME furnizează un robust şi intuitiv mediu de dezvoltare pentru aplicaţiile Windows.

**Efectuarea Lucrarii :**

**A)**

#include <vcl.h>

#pragma hdrstop

#include "a.h"

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

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

int i=0;

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

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

Form1->Caption="MIDPS 1-A";

Edit1->Text="0";

Label2->Caption="";

}

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

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

Close();

}

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

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{ i=Edit1->Text.ToInt();

Label2->Caption=" 'i' a fost incrementat";

i++ ;

Edit1->Text=(i);

}

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

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{ i=Edit1->Text.ToInt();

Label2->Caption="'i' a fost decrementat";

i--;

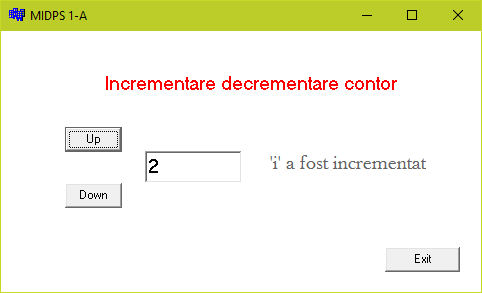
Edit1->Text=(i);

}

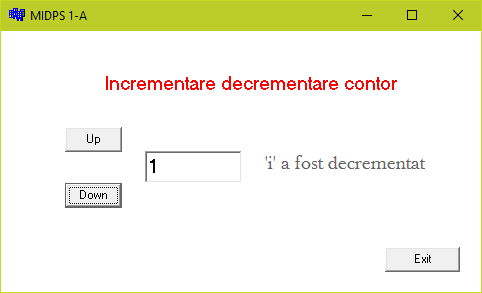
//---------------------------------------------------------------------------

**Rezultat:**

*Incrementare:*



*Decrementare:*



**B)**

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

#include <vcl.h>

#include <dos.h>

#include <stdio.h>

#pragma hdrstop

#include "Unit1.h"

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

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

int zec=0,min,sec;

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

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

Timer1->Enabled=false;

Form1->Caption="MIDPS";

Edit2->Text=" ";

Edit1->Text="00 min 00sec 00zec ";

}

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

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{

Timer1->Enabled=true;

}

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

void \_\_fastcall TForm1::Timer1Timer(TObject \*Sender)

{

zec++;

if(zec==10)

{

zec=0;

sec++;

}

if(sec==60)

{

zec=0;sec=0;

min++;}

Edit1->Text=AnsiString(min)+" min "+ AnsiString(sec)+" sec "+ AnsiString(zec)+" zec ";

}

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

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

Timer1->Enabled=false;

}

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

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

Timer1->Enabled=false;

zec=sec=min=0;

Edit1->Text="00 min 00 sec 00 zec";

}

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

void \_\_fastcall TForm1::Button4Click(TObject \*Sender)

{

Close();

}

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

void \_\_fastcall TForm1::Timer2Timer(TObject \*Sender)

{

char buf[20];

struct date d;

struct time t;

getdate(&d);

gettime(&t);

sprintf(buf,"%02d-%02d-%4d %02d:%02d:%02d",d.da\_day, d.da\_mon,d.da\_year,

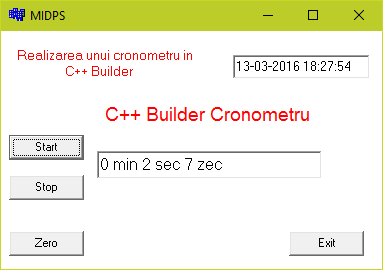
t.ti\_hour,t.ti\_min,t.ti\_sec);

Edit2->Text=(AnsiString)buf;

}

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

**Rezultat:**



**C)**

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

#include <vcl.h>

#include <dos.h>

#include <stdio.h>

#pragma hdrstop

#include "Unit1.h"

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

#pragma package(smart\_init)

#pragma resource "\*.dfm"

TForm1 \*Form1;

int x0=0 ,y0=0,x1=0,y1=0;

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

\_\_fastcall TForm1::TForm1(TComponent\* Owner)

: TForm(Owner)

{

}

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

void \_\_fastcall TForm1::Timer1Timer(TObject \*Sender)

{

char buf[20];

struct date d;

struct time t;

getdate(&d);

gettime(&t);

sprintf(buf,"%02d-%02d-%4d %02d:%02d:%02d",d.da\_day, d.da\_mon,d.da\_year,

t.ti\_hour,t.ti\_min,t.ti\_sec);

Edit1->Text=(AnsiString)buf;

}

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

void \_\_fastcall TForm1::Button1Click(TObject \*Sender)

{ PaintBox1->Repaint();

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

x1 = Form1->PaintBox1->Width;

y1 = Form1->PaintBox1->Height;

x0 = 0;

Form1->PaintBox1->Canvas->MoveTo(0, y1 / 2.0);

Button1->Enabled = false;

Button2->Enabled = true;

Timer2->Enabled = true;

}

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

void \_\_fastcall TForm1::Button3Click(TObject \*Sender)

{

Close();

}

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

void \_\_fastcall TForm1::Button2Click(TObject \*Sender)

{

Button1->Enabled = true;

Button2->Enabled = false;

Timer2->Enabled = false;

}

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

void \_\_fastcall TForm1::PaintBox1Paint(TObject \*Sender)

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

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

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

PaintBox1->Canvas->Rectangle(0, 0, PaintBox1->Width, PaintBox1->Height);

}

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

void \_\_fastcall TForm1::Timer2Timer(TObject \*Sender)

{

y0 = (y1 / 2.0) + (rand() % 91 - 45);

x0 += 4 ;

Form1->PaintBox1->Canvas->LineTo(x0, y0);

Form1->Panel2->Height = y0;

if(x0 > x1)

{

Button2Click(Sender);

}

}

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

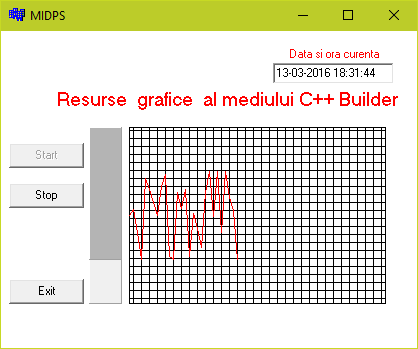
void \_\_fastcall TForm1::FormCreate(TObject \*Sender)

{

}

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

**Rezultat:**



**Concluzie:**

În urma efectuării acestui laborator, am facut cunoștiință mediul de dezvoltare C++ Builder, am înțeles cum funcționează componentele TButton, Ttimer, Label, Edit, PaintBox, Panel etc. C++Builder include unelte care permit dezvoltarea vizuală bazată pe drag-and-drop, făcând programarea mai facilă prin implementarea unui GUI builder.