Ministerul Educaţiei al Republicii Moldova

Universitatea Tehnică a Moldovei

Catedra: Automatica și Tehnologii Informaționale

RAPORT

Lucrare de laborator Nr.1
la disciplina:
Medii Interactive de Dezvoltare a Produselor Soft

A efectuat: st. gr. TI-144 Plotnicu R.
A verificat: lect.univ. Cojanu I.

Chişinău 2015

Lucrarea de Laborator Nr.1

Mediu intergrat C++ Builder

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 pe obiecte, 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.

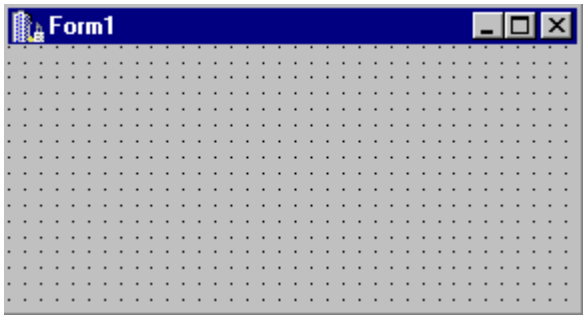
1 IDE (Mediul de Dezvoltare Integrat)

Elementele mediului integrat de dezvoltare sunt:

— Meniu principal (Main Menu);
— Forma (Form);
— Editorul de cod (Code Editor);
— Bara cu instrumente (Toolbar);
— Paleta cu componente (Component Palette);
— Tabelul cu proprietăţi ale obiectelor (Object Inspector);
— Administratorul de program (Program Manager).

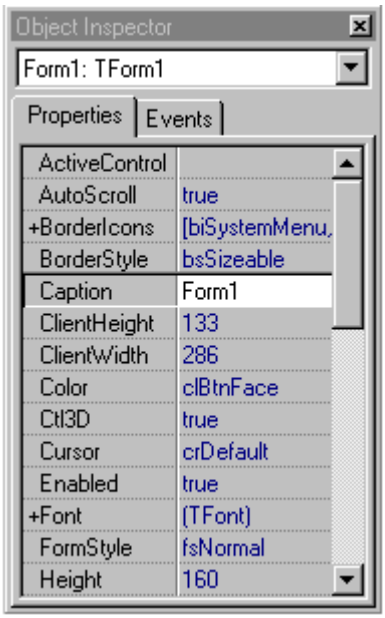
Proiectare drag-and-drop

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.



C++Builder Help

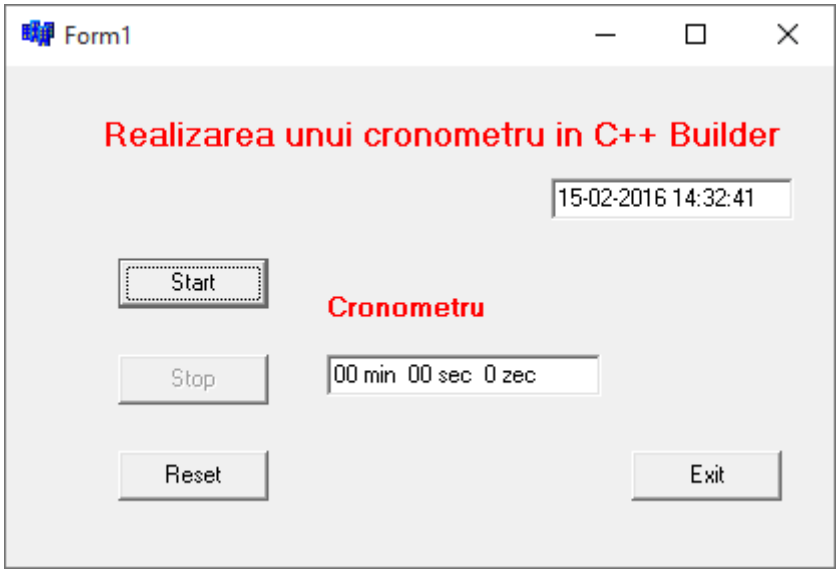
Mediul C++Builder oferă un ghid practic, care conţine peste 3000 de pagini de documentaţie despre IDE, VCL, baze de date şi tehnici de programare.

Efectuarea programelor din sarcina lucrării în mediul C++ Builder

a) cronometru

```
#include <vcl.h>
#include <stdio.h>
#include <iostream>
#include <sstream>
#pragma hdrstop

#include "Unit1.h"
#include "Dos.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"

using namespace std;

TForm1 *Form1;

struct date d;
struct time t;

void PrintTime();

int seconds;
int minutes;
int zecimals;

//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
      seconds = 0;
      minutes = 0;
      zecimals = 0;
      PrintTime();
      Timer2->Enabled = false;
      stop->Enabled = false;
}
//---------------------------------------------------------------------------


void __fastcall TForm1::exitClick(TObject *Sender)
{
      Close();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Timer1Timer(TObject *Sender)
{
      char buf[20];
      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);
      timeBox->Text=(AnsiString)buf;
}
//---------------------------------------------------------------------------
void PrintTime()
{
```

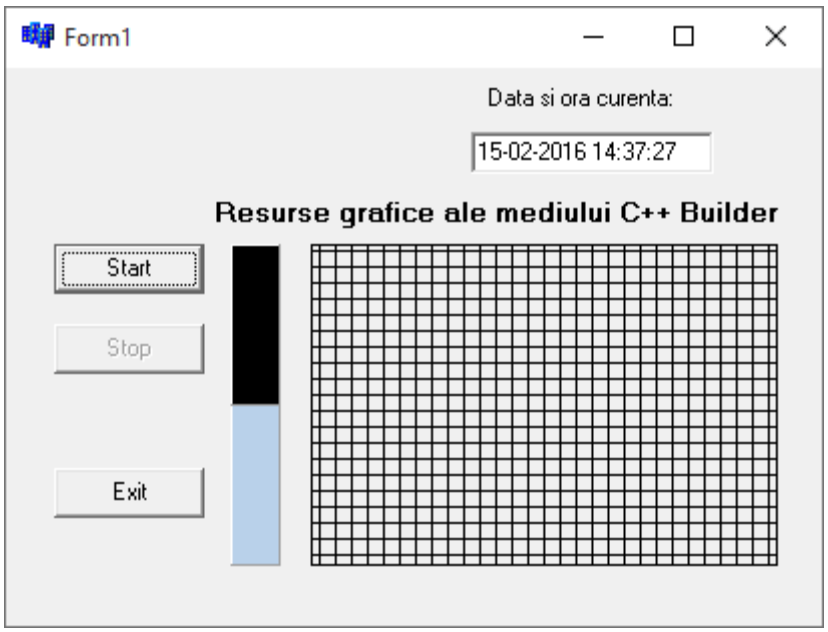
```
	stringstream timeFormat;
	timeFormat << ((minutes < 10) ? "0" : "") << minutes << " min  ";
	timeFormat << ((seconds < 10) ? "0" : "") << seconds << " sec  ";
	timeFormat << zecimals << " zec";
	string temp = timeFormat.str();
	Form1->cronBox->Text = (AnsiString)temp.c_str();
}
void __fastcall TForm1::startClick(TObject *Sender)
{
	Timer2->Enabled = true;
	start->Enabled = false;
	reset->Enabled = false;
	stop->Enabled = true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::stopClick(TObject *Sender)
{
	Timer2->Enabled = false;
	start->Enabled = true;
	reset->Enabled = true;
	stop->Enabled = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::resetClick(TObject *Sender)
{
	seconds = 0;
	minutes = 0;
	zecimals = 0;
	PrintTime();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Timer2Timer(TObject *Sender)
{
	if(zecimals == 9)
	{
		zecimals = 0;
		if(seconds == 59)
		{
			seconds = 0;
			minutes++;
		}
		else seconds++;
	}
	else zecimals++;
	PrintTime();
}
```



b) grafic

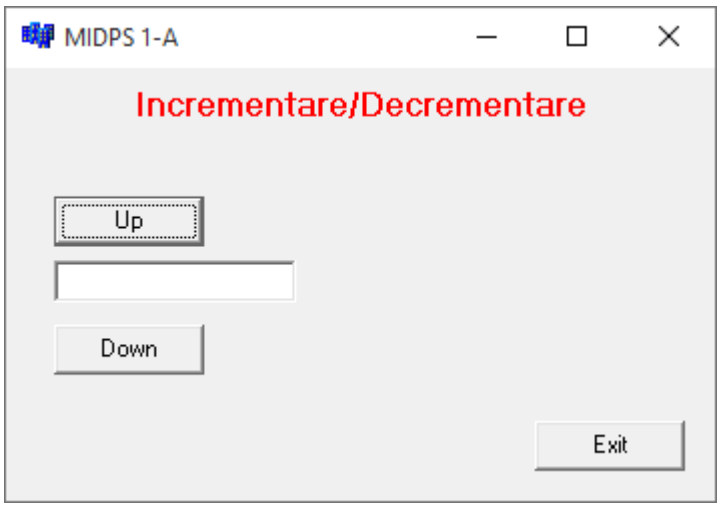
```
#include <vcl.h>
#pragma hdrstop
#include <stdio.h>

#include "Unit1.h"
#include "Dos.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
struct time t;
struct date d;
int width;
int height;
int x;
int y;


TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
	: TForm(Owner)
{
	stop->Enabled = false;
	srand(time(NULL));
}
void __fastcall TForm1::exitClick(TObject *Sender)
{
	Close();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Timer1Timer(TObject *Sender)
{
	char buf[20];
	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);
	timeBox->Text=(AnsiString)buf;
```

```
}
//---------------------------------------------------------------------------


void __fastcall TForm1::paintBoxPaint(TObject *Sender)
{
      paintBox->Canvas->Pen->Color = clBlack;
      paintBox->Canvas->Brush->Color = clBlack;
      paintBox->Canvas->Brush->Style = bsCross;
      paintBox->Canvas->Rectangle(0, 0, paintBox->Width, paintBox->Height);
}
//---------------------------------------------------------------------------

void __fastcall TForm1::startClick(TObject *Sender)
{
      paintBox->Repaint();
      paintBox->Canvas->Pen->Color = clRed;
      width = Form1->paintBox->Width;
      height = Form1->paintBox->Height;
      x = 0;
      paintBox->Canvas->MoveTo(0, height / 2.0);
      stop->Enabled = true;
      start->Enabled = false;
      Timer2->Enabled = true;
}
//---------------------------------------------------------------------------

void DrawLine()
{
      y = (height / 2.0) + (rand() % 41 - 20);
      Form1->paintBox->Canvas->LineTo(++x, y);
      Form1->Panel2->Height = y;
}
void __fastcall TForm1::Timer2Timer(TObject *Sender)
{
  if(x == width)
            stopClick(Sender);
      else DrawLine();
}
//---------------------------------------------------------------------------

void __fastcall TForm1::stopClick(TObject *Sender)
{
      Timer2->Enabled = false;
      start->Enabled = true;
      stop->Enabled = false;
}
```



c) incrementare

```
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
	: TForm(Owner)
{
}
//---------------------------------------------------------------------------

void __fastcall TForm1::exitClick(TObject *Sender)
{
	Close();
}

void __fastcall TForm1::upClick(TObject *Sender)
{
	textBox->Text = textBox->Text.ToInt() + 1;
	status->Caption = "Incrementare i";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::downClick(TObject *Sender)
{
	textBox->Text = textBox->Text.ToInt() - 1;
	status->Caption = "Decrementare i";
}
```



Concluzie

În urma realizării acestei lucrări de laborator, am facut cunoștiință cu mediul de dezvoltare C++ Builder, am aflat cum funcționează componentele TButton, Ttimer, Label, Edit, PaintBox, Panel etc. Am însuşit modul de utilizare a principalelor funcţii grafice ale mediului C++BUILDER . Aceste funcții facilitează lucrul programatorului și crește productivitatea. Am aflat cum funcționează funcțiile de lucru cu timpul sistem. C++ Builder este destul de performant și conține tot de ce e nevoie pentru a crea o fereastra ce stă la baza unui program.