Министерство Образования Республики Молдова

Технический Университет Молдовы

Кафедра Автоматики и Информационных Технологий

Лабораторная работа №1

По дисциплине: «MIDPS»

Тема: «Использование стандартных компонентов в C++ Builder»

Выполнила: студент группы TI-145:

Кысса Виктория

Проверил: старший преподаватель:

Кожокару Светлана

Кишинёв 2016

Цель работы:

Изучить принципы создания простейших Windows-приложений с графическим интерфейсом; изучить основные свойства компонентов Label, Edit, Memo, Button, CheckBox, RadioButton.

Теоретическая часть:

Компоненты. Основным строительным элементом визуального программирования является *компонент.* В свою очередь, для компонентов характерно наличие *свойств* и *событий.* Компоненты позволяют пользователю программы выполнять различные действия, например, щелкать на кнопках или вводить данные. Все компоненты, размещаемые в формах, имеют *уникальные* имена. Имя задается с помощью свойства `Name`. Можно изменить имя компонента во время работы над приложением.

Стандартные компоненты. На первой странице (Standard) Палитры Компонент размещены наиболее часто используемые компоненты.

`TButton` – прямоугольная кнопка с надписью. Поместив `TButton` на форму, Вы по двойному щелчку можете создать заготовку обработчика события нажатия кнопки. Далее нужно заполнить заготовку кодом.

Вот основные свойства элемента управления CommandButton: *`Left`* (позиция элемента управления относительно левого края его контейнера), *`Top`* (позиция элемента относительно верхнего края его контейнера), *`Height`* (высота), *`Width`* (ширина), *`Enabled`* (определяет, можно ли пользователю работать с этим элементом управления), *`Visible`* (видимость во время выполнения программы), *`Caption`* (подпись).

Все указанные свойства можно менять как во время разработки программы, так и во время ее работы. Чтобы изменить свойства кнопки во время работы программы, необходимо в процедуре использовать команду вида:

```
НазваниеЭлемента.НазваниеСвойства := НовоеЗначение
```

Например, если в форме имеется кнопка `cmd1`, то можно изменить ее свойства следующим образом:

```
cmd1->Caption = "Нажми на меня";
cmd1->Left = 200;
```

При этом новое значение для свойства `Caption` указывается в кавычках, так как оно имеет тип `String`, а для свойства `Left` – без кавычек, так как оно имеет числовой тип.

`TLabel` служит для отображения текста на экране. Вы можете изменить шрифт и цвет метки, если дважды щелкнете на свойство `Font` в Инспекторе Объектов. Текст метки является значением свойства `Caption`. Свойство `Alignment` определяет способ выравнивания текста. Чтобы размер шрифта автоматически соответствовал максимальному заполнению области, установите значение `true` свойства `AutoSize`. Чтобы весь текст можно было увидеть внутри короткой области, задайте значение true свойства `WordWrap` (перенос слов). Установкой значения true свойства `Transparent` вы можете оставить видимой часть другой компоненты сквозь название, расположенное прямо на ней.

TEdit - стандартный управляющий элемент Windows для ввода. Он может быть использован для отображения короткого фрагмента текста и позволяет пользователю вводить текст во время выполнения программы. Начальное содержимое области редактирования определяет строка, являющаяся значением свойства Text. Свойство Font определяет параметры шрифта текстового поля. Свойство AutoSize разрешает или запрещает текстовому полю динамически изменять размер. Установив свойство ReadOnly в значение True, мы запрещаем пользователю программы вводить данные в текстовое поле. Свойство MaxLength определяет число символов, которые можно ввести в текстовое поле. Свойство SelText содержит текущий выделенный фрагмент строки в текстовом поле. Свойства SelStart, SelLength возвращают начальную позицию и длину выделенного фрагмента строки в текстовом поле.

TMemo – отображает прямоугольную область редактируемого ввода множественных строк информации на форме. Начальное содержимое области редактирования определяет массив строк, являющийся значением свойства Lines. Окно редактора элементов списка открывается кнопкой в графе значении этого свойства.

Timer - таймер, позволяет выполнять действия через определенные промежутки времени.

PaintBox (рамка рисования) создает на форме элемент, на котором можно рисовать

Цель работы:

Разработать три приложения используя стандартные компоненты.

Приложение №1

- Две кнопки (Button 1 şи 2) для инкрементации (UP) и соответственно декрементации (DOWN) переменной **i** ;
- Кнопка (Button 3) для выхода (Exit);
- Окошко для отображения (Edit1) значения переменной **i**;
- Две надписи (Label1 и 2) для отображения текста: «**Incrementare decrementare conto**» и «Статус i».

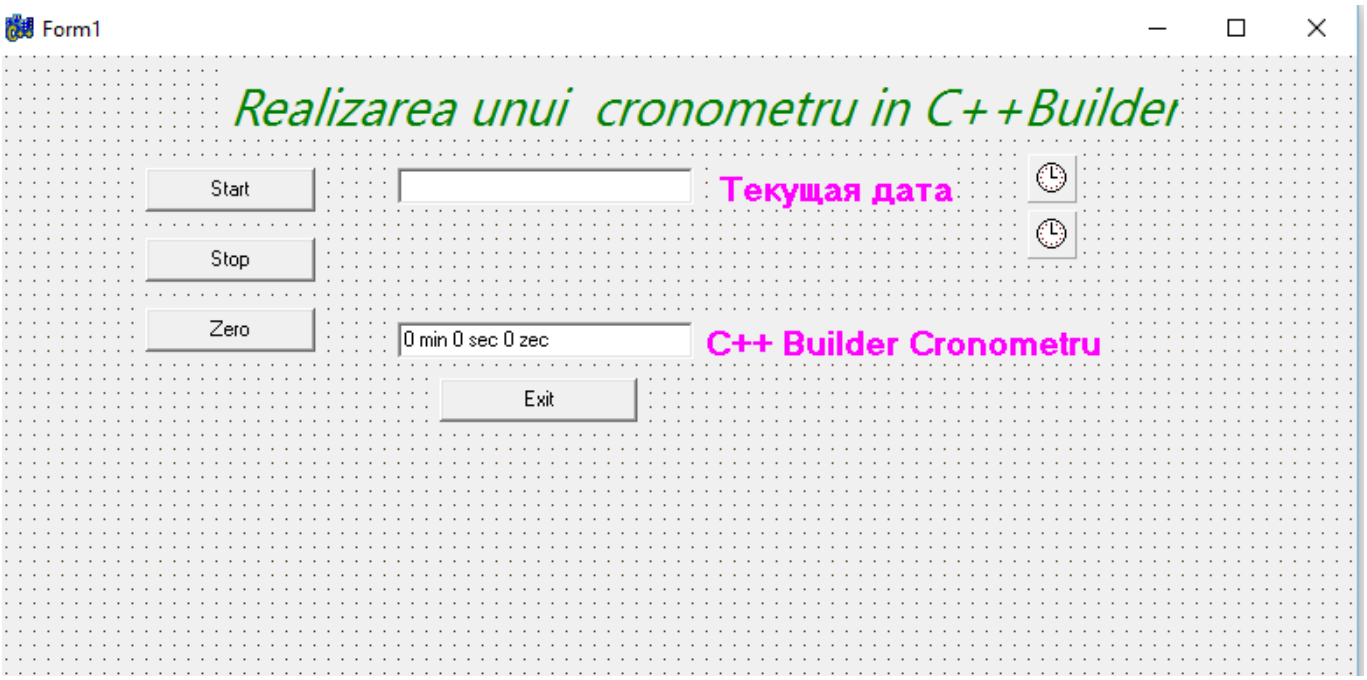
Листинг программы:

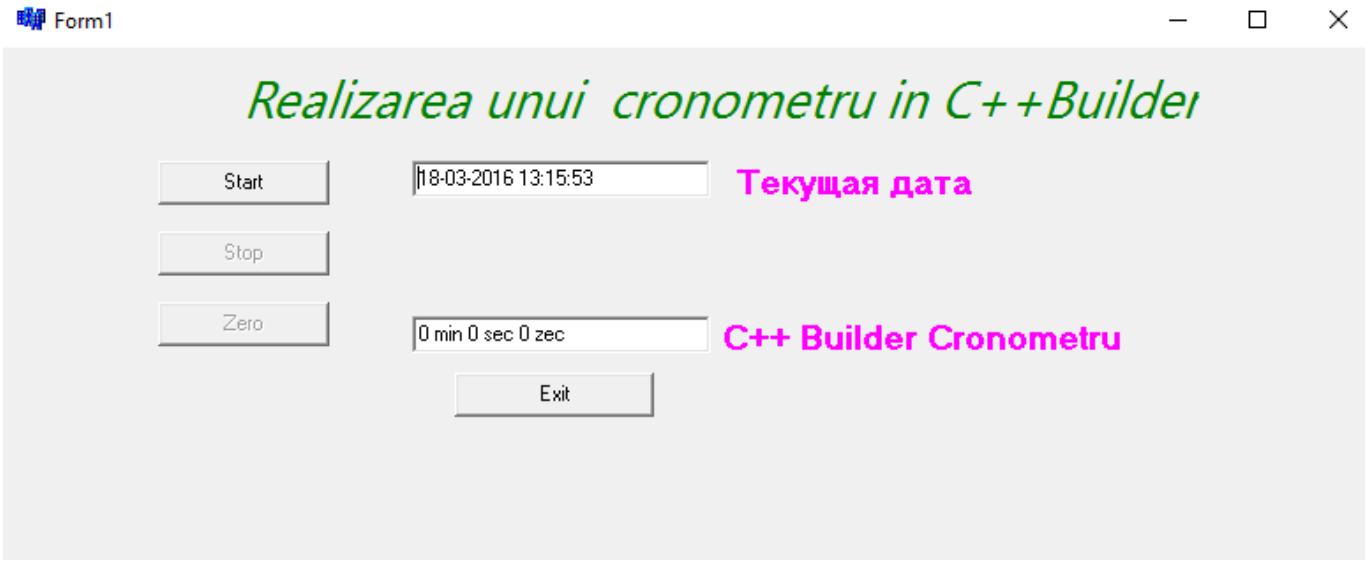
```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include "Lab1a.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
int i=0;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
      : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
i++;
Edit1->Text=i;
Label1->Caption="i была инкрементированна  ";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button2Click(TObject *Sender)
{
i--;
Edit1-> Text=i;
Label1->Caption="i была декрементированна ";
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button3Click(TObject *Sender)
{
```

```
Close();
}
//---------------------------------------------------------------------------
```

Приложение №2 «Хронометр»

- Форма (*Form1*)**;**
- Четыре кнопки (*Button 1, 2, 3 , 4*), выполняющие следующие функции:
  - Button1 – запуск хронометра (**Start**);
  - Button2 – остановка хронометра (Caption **Stop**);
  - Button3 – обнуление (**Zero**);
  - Button4 – выход (**Exit**).
- Два таймера (*Timer1* şi *Timer2*), выполняющие следующие функции:
  - Timer1 (*Интервал=1000 мс*) - текущее время;
  - Timer2 (*Интервал=100 мс*) – время хронометра;
- Два окошка для отображения (*Edit1* si *Edit2*), использованные для:
  - Edit1 – отображение реального времени;
  - Edit2 – время хронометра;
- Две надписи (*Label1* si *Label2*);





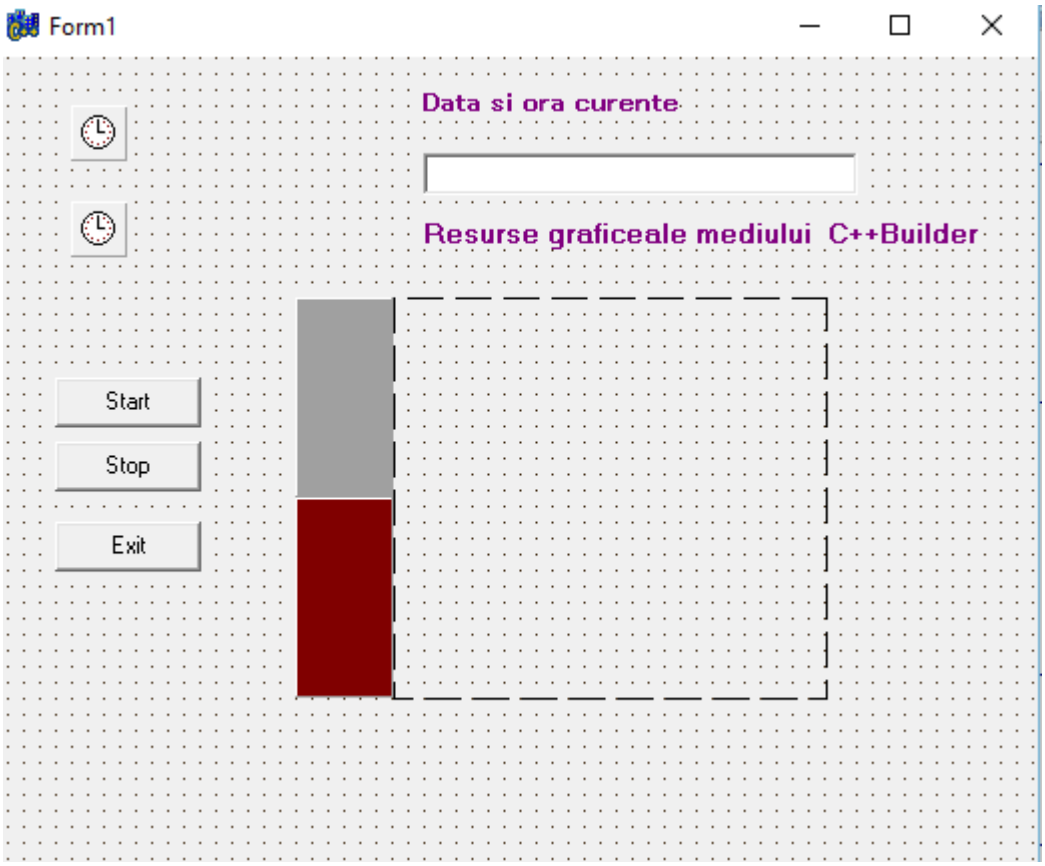
Листинг программы:

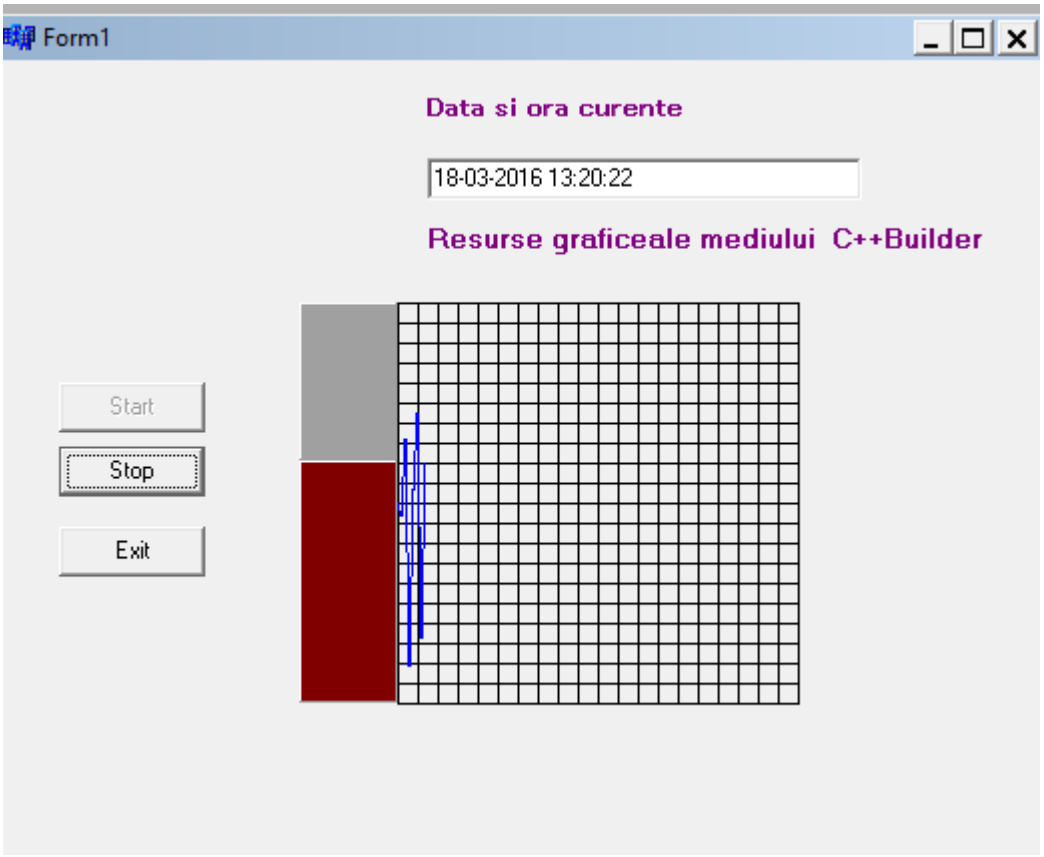
```
//---------------------------------------------------------------------------

#include <vcl.h>
#pragma hdrstop

#include <stdio.h>
#include "Lab1b.h"
#include "dos.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
int i=0;
struct date d;
struct time t;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
Timer2->Enabled = true;
Button1->Enabled = false;
Button2->Enabled = true;
Button3->Enabled = false;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button4Click(TObject *Sender)
{
Close();
}
//---------------------------------------------------------------------------
```

```
void __fastcall TForm1::Button2Click(TObject *Sender)
{
Timer2->Enabled = false;
Button1->Enabled = false;
Button2->Enabled = false;
Button3->Enabled = true;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button3Click(TObject *Sender)
{
i=0;
Edit2->Text="0 min 0 sec 0 zec";
Button1->Enabled = true;
Button2->Enabled = false;
Button3->Enabled = false;
}
//---------------------------------------------------------------------------

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);
Edit1->Text=(AnsiString)buf;

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

void __fastcall TForm1::Timer2Timer(TObject *Sender)
{
i++;
Edit2->Text = IntToStr(i/600) + " min " + (i%600)/10 + " sec " + IntToStr(i%10) + " zec";
}
//---------------------------------------------------------------------------
```

Приложение №3

- Форма (*Form1*)**;**
- Три кнопки (*Button 1*, *2*, *3*):
  - Buton1 – активация барографа (**Start**);
  - Buton2 – остановка барографа(Caption **Stop**);
  - Buton3 – выход (Caption **Exit**).
- Два таймера (*Timer1* şi *Timer2*) :
  - Timer1 (*Interval=1000 ms*)-текущее время;
  - Timer2 (*Interval=500 ms*) – время барографа;
- Окошко для отображения (*Edit1*);
- Две надписи (*Label1* si *Label2*);
- Две панели (Panel1, Panal2);
- PaintBox1;





Листинг программы:

```
//---------------------------------------------------------------------------

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

```
#include <stdio.h>
#include "Lab1c.h"
#include "dos.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
struct date d;
struct time t;
int x=0;
int y, yprev;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------


void __fastcall TForm1::StartClick(TObject *Sender)
{
x = 0;
Timer2->Enabled = true;
Start->Enabled = false;
Stop->Enabled = true;
Form1->Repaint();
for (int i = 0; i < 201; i+=10){
PaintBox1->Canvas->MoveTo(i,0);
PaintBox1->Canvas->LineTo(i,201);
}
for (int i = 0; i < 201; i+=10){
PaintBox1->Canvas->MoveTo(0,i);
PaintBox1->Canvas->LineTo(201,i);
}
PaintBox1->Canvas->MoveTo(0,100);
}
//---------------------------------------------------------------------------
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);
Edit1->Text=(AnsiString)buf;

}
//---------------------------------------------------------------------------
void __fastcall TForm1::StopClick(TObject *Sender)
{
Timer2->Enabled=false;
Start->Enabled=true;
```

```
Stop->Enabled=false;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Timer2Timer(TObject *Sender)
{
  randomize();
yprev = y;
y = random(201)+1;
x+=1;
if(x >= 201){
Form1->Repaint();
for (int i = 0; i < 201; i+=10){
PaintBox1->Canvas->MoveTo(i,0);
PaintBox1->Canvas->LineTo(i,201);
}
for (int i = 0; i < 201; i+=10){
PaintBox1->Canvas->MoveTo(0,i);
PaintBox1->Canvas->LineTo(201,i);
}
PaintBox1->Canvas->MoveTo(0,yprev);
x=0;
}
PaintBox1->Canvas->Pen->Color = clBlue;
PaintBox1->Canvas->LineTo(x,y);
PaintBox1->Canvas->Pen->Color = clBlack;
Panel1->Height = y;
Panel2->Top = Panel1->Top + Panel1->Height;
Panel2->Height = 200 - Panel1->Height;


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

Вывод:

В ходе лабораторной работы я изучила основы работы в интегрированной среде разработки: «C++ Builder». Необходимо было разработать 3 простых приложения: инкрементирование и декрементирование, хронограф и барограф. В ходе выполнения получились три небольшие программы, включающее в себя принципы создания простейших Windows-приложении с графическим интерфейсом. Были применены основные свойства таких компонентов, как: Label, Edit, PaintBox, Button, Timer, Panel.