

## Лабораторна робота № 16

### Знайомство з середовищем програмування C++ Builder

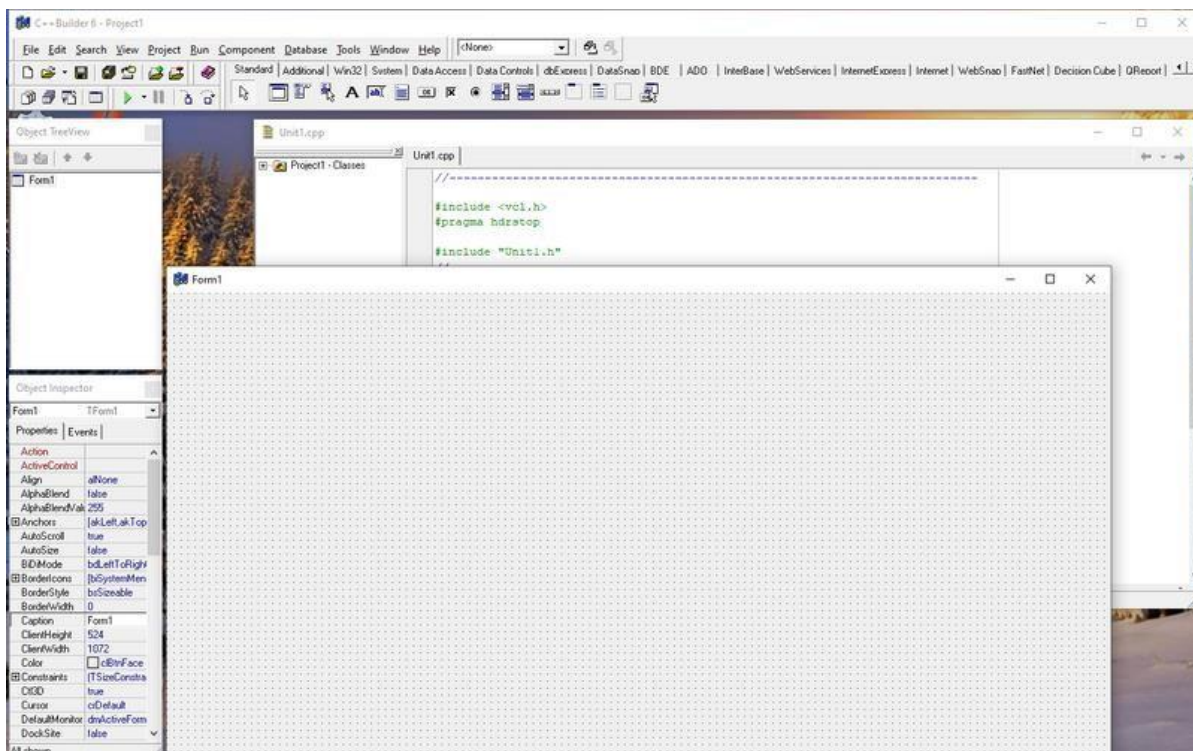
**Мета роботи:** ознайомитися з основними компонентами та інтерфейсом середовища програмування C++ Builder та набути навиків створення та налагодження програмних проєктів.

#### Теоретичні відомості

#### Послідовність організації проєктів в C++ Builder

C++ Builder – це технологія візуального програмування, де автоматизовано її трудомістку частину – створення інтерфейсних програм з діалоговими вікнами. Оболонка C++ Builder надає змогу замість повного самостійного написання програми використовувати великий набір готових візуальних об'єктів, так званих компонентів, піктограми котрих розміщено на відповідних вкладках палітри компонентів.

Вікно середовища розробки програм C++ Builder має вигляд:



У верхній частині вікна міститься головне меню (File, Edit, Search тощо). Нижче розташовано дві інструментальні панелі, які містять низку кнопок та палітру компонентів. Ліворуч головного поля вікна розташовано інспектор об'єктів (Object Inspector). Інспектор об'єктів має дві сторінки: Properties (властивості) та Events (події). Властивості компонентів (назва, колір, розмір тощо) обираються на сторінці Properties, а програми-опрацювачі (натискання клавіш миші чи її рух, натискання клавіш клавіатури, вибір пункту меню тощо) – на сторінці Events у вікні Object Inspector.

Праворуч інспектора об'єктів розташовано конструктор форми “Контейнер”, на який при створенні форми розміщують компоненти. Форма сама є компонентом з назвою **Form**. Без додаткових вказівок заголовок компонента збігається з його назвою, до якої додається порядковий номер, розпочинаючи з 1. Але заголовок можна змінити за допомогою властивості **Caption**.

За конструктором форм розташовано вікно редактора коду Unit1.cpp, в якому прописують тексти програм. Перемикання поміж вікнами форми й редактора можна здійснювати клавішею **F12**.

Для знайомства з середовищем програмування C++ Builder розглянемо кілька прикладів створення програмних проєктів.

## Завдання 1.

### 1 Постановка задачі

Створити програму, в якій можна буде встановлювати колір форми натисканням кнопки. Слід передбачити можливість установлення чотирьох кольорів: жовтого, синього, зеленого, червоного, – а також створення кнопки для виходу з програми.

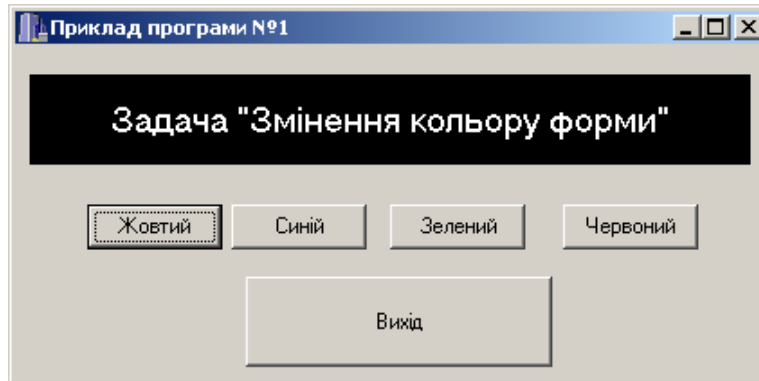


Рисунок 1.1 – Форма програми «Приклад програми № 1»

### 2 Створення форми

На вікно форми слід встановити компоненти: один надпис (Label) і чотири кнопки (Button).

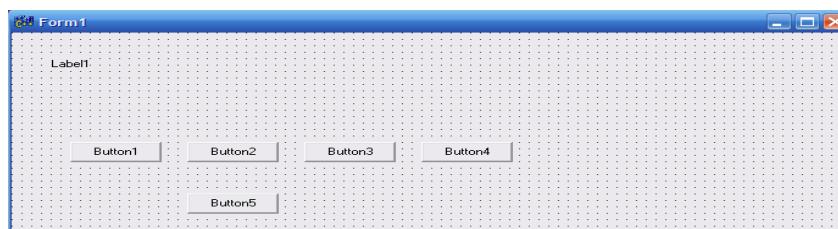


Дані компоненти розташовано на палітрі компонентів на вкладці «Standard»:

Задля розміщення кожного компонента на форму треба:

- 1) виокремити компонент (клацанням лівої кнопки по його піктограмі);
- 2) помістити курсор миші в позицію розміщення компонента на формі;
- 3) клацнути лівою кнопкою миші (опустити компонент).

Після того як ці дії повторити задля розміщування всіх необхідних компонентів, форма набуде вигляду



Надписи, розмір, положення на формі – все це властивості (Properties) компонентів, які можна змінювати при створенні форми засобами інспектора об'єктів (Object Inspector). Щоби змінити властивості компонента, його необхідно виокремити, клацнувши по компонентові лівою кнопкою миші, потім перейти до вікна інспектора об'єктів, на закладці Properties обрати потрібну властивість і встановити її нове значення. Отже, встановимо властивості компонентів згідно з таблицею 1.1.

Таблиця 1.1 – Властивості компонентів

Ім'я компонента	Властивість	Значення властивості	Пояснення
Form1	Caption	Приклад програми № 1	Надпис на формі
Label1	Alignment	taCenter	Розміщення тексту по горизонталі
	AutoSize	false	Автоматичне змінювання розміру
	Caption	Задача "Змінення кольору форми"	Текст надпису
	Color	clBlack	Колір компонента
	Font->Color	clWhite	Колір шрифту надпису
	Font->Size	14	Розмір шрифту надпису
	Height	50	Висота компонента
	Layout	tlCenter	Розміщення тексту по вертикалі
	Width	400	Ширина компонента
Button1	Caption	Жовтий	Надписи на кнопках
Button2	Caption	Синій	
Button3	Caption	Зелений	
Button4	Caption	Червоний	
Button5	Caption	Вихід	Надпис на кнопці
	Height	50	Висота кнопки
	Width	170	Ширина кнопки

### 3 Створювання програми відгуку на подію OnClick

Після створення вигляду форми можна перейти до програмування. Колір форми має змінюватись при клацанні (Click) по відповідних кнопках. Для цього треба створити програми відгуку на подію (Event) OnClick для відповідних кнопок. Програми-відгуки на події створюються (призначаються) у вікні інспектора об'єктів на вкладці подій (Events). Для цього треба:

- 1) виокремити компонент;
- 2) перейти до вікна Object Inspector;
- 3) відкрити вкладку Events;
- 4) двічі клацнути лівою кнопкою миші напроти назви потрібної події.

Задамо відгук на подію OnClick для кнопки Button1. Виконавши всі зазначені чотири кроки, створимо шаблон-заготовку в редакторі коду:

```
void __fastcall TForm1::Button1Click(TObject *Sender)
{
}

```

**Увага!** Наведений вище код створюється при виконуванні зазначених дій *автоматично* середовищем С++ Builder, його не слід записувати.

Всі команди, які мають виконуватись при клацанні по кнопці, треба записати поміж операторними дужками { }. Для кнопки Button1 поміж операторними дужками слід додати команду:

```
Form1->Color = clYellow;
```

Програмний відгук на подію OnClick для кнопки Button1 остаточно матиме вигляд


```
void __fastcall TForm1::Button1Click(TObject *Sender)
{Form1->Color = clYellow;
}
```

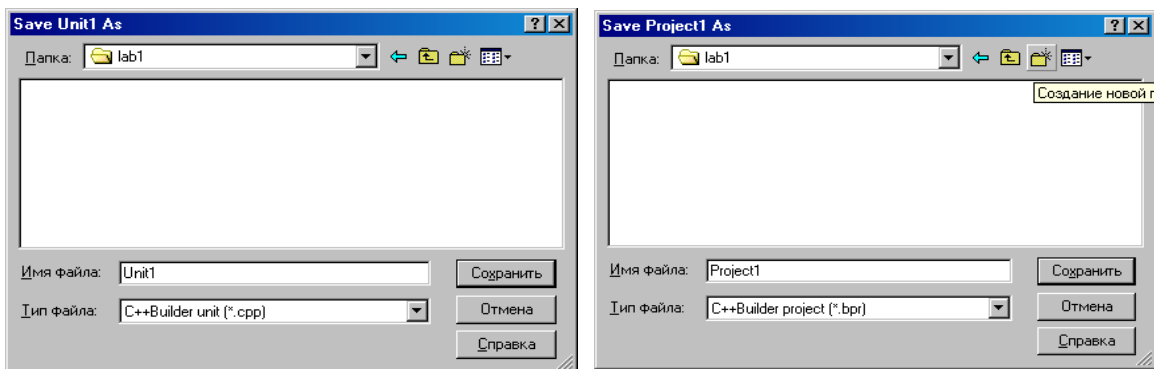
Самостійно запрограмуйте кнопки **Button2, Button3, Button4**.

**П р и м і т к а.** Синій колір – clBlue, зелений – clGreen, червоний – clRed. Для кнопки Button5 (Вихід) програмний код міститиме команду закриття активної форми Close():

```
void __fastcall TForm1::Button5Click(TObject *Sender)
{Close();
}
```

## 4 Запуск програми (проекту)

Перед запуском на виконання програмний проект *обов'язково* слід зберегти в *окремому* каталозі (теці) командою **File->SaveAll** (чи то через ярлик , чи клавішами Shift+Ctrl+S). Внаслідок таких дій почергово з'являться два діалогових вікна, в яких треба у власно створеному новому каталозі зберегти і Unit1 і Project1



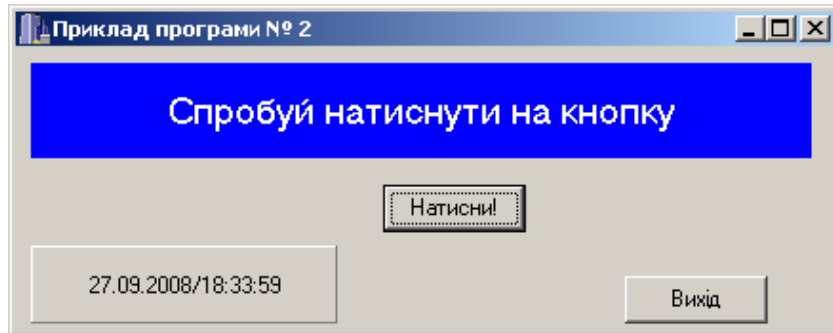
Для запуску проекту слід виконати команду меню **Run->Run** або натиснути на клавіатурі кнопку **F9**. Внаслідок цього має з'явитися форма проекту, зображена на рис. 1.1 (це означає що, що програму написано без помилок).

Якщо в перебігу створювання програми було припущено помилок, то рядок з помилкою висвічуватиметься червоним кольором, а повідомлення компілятора, розташоване під вікном редактора коду, повідомить, якої саме помилки було допущено. Після виправлення усіх помилок слід зберегти зміни і запустити проект на виконання.

## Приклад 2

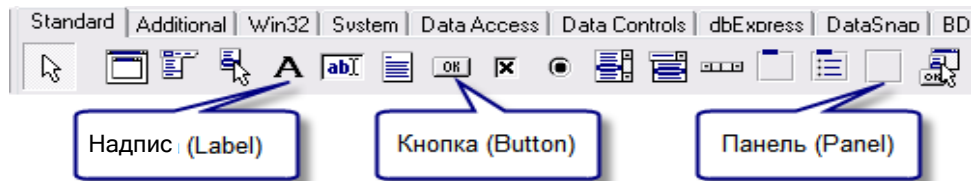
### 1 Постановка задачі

Створити програму, в якій зреалізувати кнопку-“втікача”, тобто кнопку, яка “втікатиме” за намагання натиснути її, окрім того, вивести поточну дату і час на форму. Вікно форми матиме вигляд:

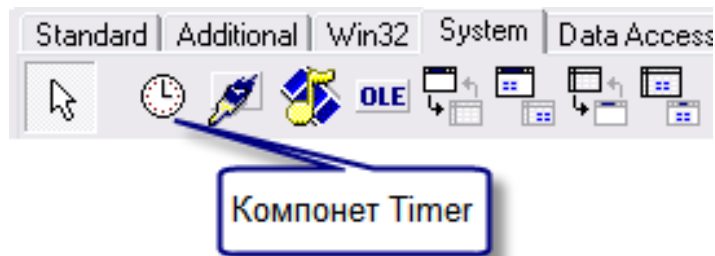


### 2 Створення форми

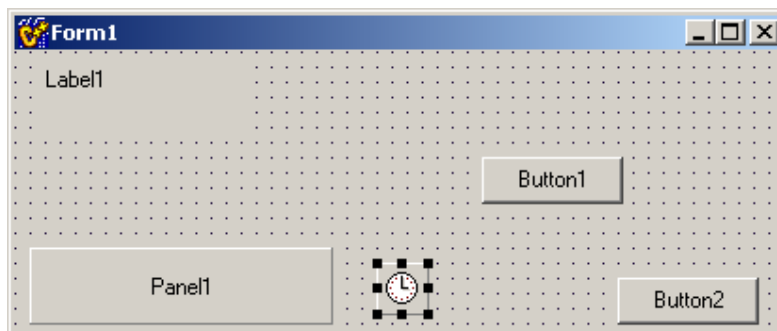
Розташуємо на формі один надпис (Label), одну панель (Panel), дві кнопки (Button) і компонент таймер (Timer). Перші три компоненти містяться на палітрі компонентів на вкладці «Standard»:



Компонент **Timer** міститься на палітрі компонентів на вкладці «System»:



Вигляд форми після розташування всіх компонентів матиме вигляд:



Задамо властивості компонентів, наведені в табл. 1.2.

Таблиця 1.2 – Властивості компонентів

Ім'я компонента	Властивість	Значення властивості	Пояснення
Form1	Caption	Приклад програми № 2	Надпис на формі
	Height	210	Висота форми
	Width	695	Ширина форми
Label1	Alignment	taCenter	Розміщення тексту по горизонталі
	AutoSize	false	Автоматичне змінювання розміру
	Caption	Спробуй натиснути кнопку	Текст надпису
	Color	clBlue	Колір компонента
	Font->Color	clWhite	Колір шрифту надпису
	Font->Size	14	Розмір шрифту надпису
	Height	50	Висота компонента
	Layout	tlCenter	Розміщення тексту по вертикалі
	Width	625	Ширина компонента
Button1	Caption	Натисни!	Надпис на кнопці
Button2	Caption	Вихід	Надпис на кнопці
Timer1	Interval	1000	Інтервал таймера, заданий у мілісекундах
Panel1	Caption		Надпис на панелі

### 3 Створювання програм

Задамо шаблон для відгуку на подію OnMouseMove (подія виникає за наведення на кнопку покажчика миші) для кнопки Button1. Виконавши дії, зазначені в прикладі 1, створимо в редакторі коду такий шаблон:

```
void __fastcall TForm1::Button1MouseMove(TObject *Sender, TShiftState Shift, int X, int Y)
{
}
```

**У в а г а!** Наведений вище код створюється при виконуванні зазначених дій *автоматично* середовищем C++ Builder, його не слід записувати.

Після того як впишемо поміж операторних дужок наступний програмний код, програма відгуку набуде вигляду


```
void __fastcall TForm1::Button1MouseMove(TObject *Sender, TShiftState Shift, int X, int Y)
{ if (Button1->Left <= 0)
Button1->Left = Form1->Width - Button1->Width - 10;
else Button1->Left = Button1->Left - 5;
}
```

Самостійно створіть програму для кнопки Button2 (Вихід) на подію OnClick. Отже, кнопка “втікає”, залишилось вивести дату і час на панель. Задамо для компонента Timer шаблон для події OnTimer (подія відбувається із заданим у властивості Interval інтервалом часу), в якому запишемо такий програмний код: Panel1->Caption = Now().DateToString() + "/" + Now().TimeToString();

**Пояснення.** Функція Now() повертає поточні дату і час. Метод DateTime.ToString() перетворює дату на текст, метод TimeToString() перетворює час на текст.

#### 4 Запуск програми (проекту)

Для запуску проекту на виконання слід виконати команду меню **Run->Run** або натиснути на клавіатурі кнопку **F9**.

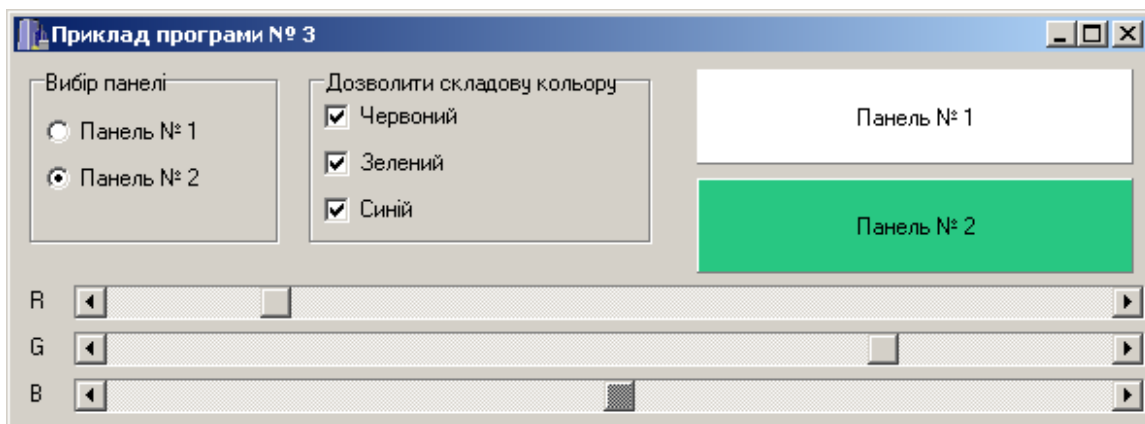
**Увага!** Перед запуском проекту його *обов'язково* слід зберегти в *окремій* теці – команда **File->SaveAll** (  ).

### Приклад 3

#### 1 Постановка задачі

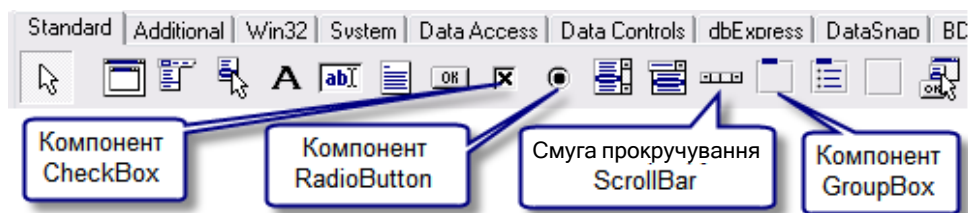
Створити програму, в якій, залежно від вибору користувача, можна встановлювати колір першої чи другої панелі. Здійснити можливість задавати складові трьох основних кольорів: червоного, зеленого, синього. Надати можливість вибору дозволу/заборони змінення певної складової кольорів.

Робочий вигляд форми проекту:

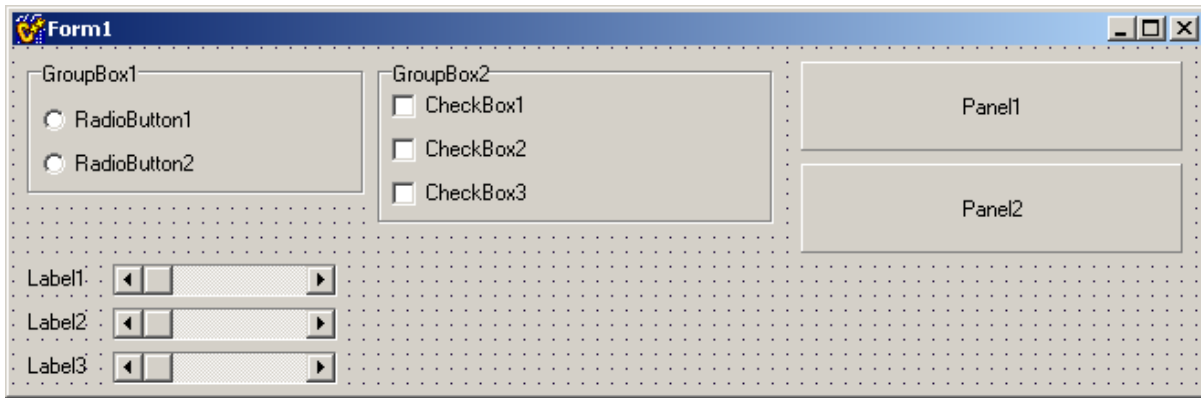


#### 2 Створювання форми

На вікно форми слід встановити такі компоненти: три надписи (Label), два компоненти GroupBox, дві панелі (Panel), три смуги прокручування (ScrollBar). На компонент GroupBox1 розмістимо два компоненти RadioButton, а на GroupBox2 – три компоненти CheckBox. Усі компоненти розташовані на палітрі компонентів на вкладці «Standard»:



Після розташування всіх компонентів форма матиме вигляд



Задамо властивості компонентів, наведені в табл. 1.3.

Таблиця 1.3 – Властивості компонентів

Ім'я компонента	Властивість	Значення властивості	Пояснення
Form1	Caption	Приклад програми № 3	Надпис на формі
	Height	255	Висота форми
	Width	650	Ширина форми
Label1	Caption	R	Текст надпису
Label2	Caption	G	Текст надпису
Label3	Caption	B	Текст надпису
GroupBox1	Caption	Вибір панелі	Надпис компонента
	Height	90	Висота компонента
GroupBox2	Caption	Дозволити складову кольору	Надпис компонента
	Height	90	Висота компонента
RadioButton1	Caption	Панель № 1	Надпис компонента
	Checked	true	Стан вибору
RadioButton2	Caption	Панель № 2	Надпис компонента
ScrollBar1, ScrollBar2, ScrollBar3	Max	255	Максимальне значення прокрутки
	Width	550	Ширина смуги
CheckBox1	Caption	Червоний	Надпис компонента
	Checked	true	Стан вибору
CheckBox2	Caption	Зелений	Надпис компонента
	Checked	true	Стан вибору
CheckBox3	Caption	Синій	Надпис компонента
	Checked	true	Стан вибору
Panel1	Caption	Панель № 1	Надпис панелі
Panel2		Панель № 2	Надпис панелі

### 3 Створювання програм

Задамо шаблон для відгуку на подію OnChange (подія виникає при змінненні позиції повзунка смуги прокрутки) для смуги прокручування ScrollBar1. Коли виконаємо чотири кроки, наведені у прикладі 1, матимемо шаблон в редакторі коду:

```
void __fastcall TForm1::ScrollBar1Change(TObject *Sender)
{
}
```

**Увага!** Наведений вище код створюється при виконуванні зазначених дій автоматично середовищем C++ Builder, його не слід записувати.

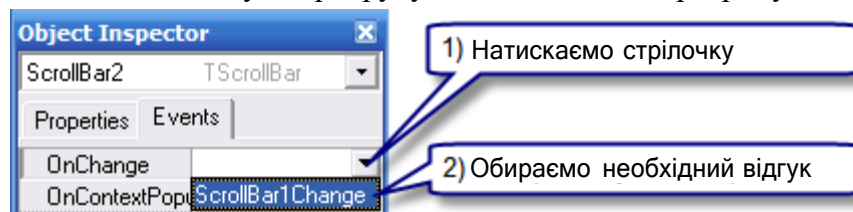
Впишемо поміж операторних дужок такий програмний код:

```
if (RadioButton1->Checked)
Panel1->Color = RGB(ScrollBar1->Position, ScrollBar2->Position,
ScrollBar3->Position);
else
Panel2->Color = RGB(ScrollBar1->Position, ScrollBar2->Position,
ScrollBar3->Position);
```

Програми для відгуку на подію OnChange для компонентів ScrollBar1 та ScrollBar2 є аналогічні, але щоби не писати зайвого коду, зазначимо для ScrollBar2 вже наявну програму-відгук ScrollBar1Change. Для цього:

- 1) виокремимо компонент ScrollBar2;
- 2) у вікні Object Inspector перейдемо до вкладки Events;
- 3) оберемо для події OnChange зі списку ScrollBar1Change:

Аналогічно встановить для смуги прокручування ScrollBar3 програму-відгук для події



OnChange.

Зазначимо, в якій спосіб можна дозволити чи заборонити змінення будь-якої складової кольорів. Для цього у нас є три відповідні “прапорці” (CheckBox). Якщо “прапорець” обрано (властивість Checked), то змінювати складову кольору можна, в іншому разі – ні.

Стан “прапорця” змінюється за його натискання. Отже, опрацьовуватимемо події OnClick компонентів CheckBox1, CheckBox2, CheckBox3. В програмах для події OnClick запишемо:


для CheckBox1: ScrollBar1->Enabled = CheckBox1->Checked;

для CheckBox2: ScrollBar2->Enabled = CheckBox2->Checked;

для CheckBox3: ScrollBar3->Enabled = CheckBox3->Checked;

### 4 Запуск програми (проєкту)

Для запуску проєкту слід виконати команду меню **Run->Run** чи то натиснути на клавіатурі кнопку **F9**.

**Увага!** Перед запуском проєкту його обов'язково слід зберегти в окремій теці – команда **File->SaveAll** (  ).

### **Контрольні запитання**

1. Для чого потрібна форма проекту?
2. Для чого потрібна палітра компонентів?
3. Для чого потрібен інспектор об'єктів?
4. Перелічіть компоненти, які ви вже знаєте.
5. Які властивості компонентів ви змінювали і в який спосіб це зробили?
6. Де записують тексти програм?
7. Для чого потрібні бібліотеки математичних функцій і в який спосіб вони долучаються?
8. В яких бібліотеках містяться математичні функції?
9. В який спосіб можна зберегти проект?
10. В який спосіб можна запустити проект на виконання?