

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ФАХОВИЙ КОЛЕДЖ РАКЕТНО-КОСМІЧНОГО МАШИНОБУДУВАННЯ
ДНІПРОВСЬКОГО НАЦІОНАЛЬНОГО УНІВЕРСИТЕТУ ім. О.Гончара

МЕТОДИЧНА РОЗРОБКА
для забезпечення самостійної роботи студентів

З НАВЧАЛЬНОЇ ДИСЦИПЛІНИ

«Інструментальні засоби візуального програмування»

для студентів денної форми навчання за спеціальністю

121 «Інженерія програмного забезпечення»

(код і назва спеціальності)

Розроблено викладачем

_____/ Гапоненко Н.В./

Розглянуто та схвалено

на засіданні ЦК Пі

Протокол № від .20

Голова ЦК Пі

План самостійного вивчення теми № 1

з навчальної дисципліни "Інструментальні засоби візуального програмування"

Тема	Палітра компонентів середовища візуального програмування
Мета	Ознайомитись з палітрою компонентів середовища візуального програмування
	<i>знати: інструментарій середовища візуального програмування</i>
	<i>вміти: використовувати інструментарій середовища візуального програмування</i>

Час – 2 години

Навчальні питання:

- 1 Інтегроване середовище розробки
- 2 Палітра компонентів середовища
- 3 Використання палітри компонентів середовища

Методичні рекомендації

Самостійно вивчаючи запропоновану тему, студент повинен засвоїти такі основні положення:

<i>По-перше</i>	Інтегроване середовище розробки — це програмний продукт, який об'єднує текстовий редактор, компілятор, відладчик і довідкову систему
<i>По-друге</i>	Палітра компонентів — це вікно зі згрупованими за сенсом піктограмами компонентів, призначеними для використання в програмі
<i>По-третє</i>	Пакет включає інструментарій, необхідний для роботи в середовищі програмування

Література

Архангельский, А. Я.	Программирование	в	C++	Builder	[Текст]	/
А. Я. Архангельский.						
- М. : Бином-Пресс, 2010. - 1230 с.						

Навчальні матеріали до самостійного вивчення теми

Палітра компонентів

Палітра компонентів — це багатосторінковий блокнот, на сторінках якого розміщуються кнопки виклику компонентів, що знаходяться в бібліотеці компонентів або які є шаблонами. Останні відрізняються тим, що не розміщуються ні в якому пакеті компонентів.

Компоненти, що відображаються, при роботі програми мають зовнішній вигляд, схожий з тим, який вони мають в конструкторі форм. Компоненти, що не відображаються, при роботі програми або повністю не відображаються (як, наприклад, таймер), або їх зображення істотно відрізняється від того, як вони зображалися в конструкторі форм (наприклад, меню).

Після установки Embarcadero C++ Builder сторінка стандартних компонентів Standard містить наведені нижче стандартні компоненти.

TMainMenu дозволяє вам помістити головне меню в програму. При розміщенні TMainMenu на форму це виглядає, як просто іконка. Створення меню включає три кроки: розміщення TMainMenu на форму, виклик дизайнера меню через властивість Items в інспектор об'єктів, визначення пунктів меню в редакторі меню. Цей компонент дозволяє розмістити у програмі лінійку головного меню з випадаючими підпунктами.

PopupMenu — випадаюче локальне меню (контекстне). Це меню виникає при натисканні правої клавіші миші на відповідному компоненті. Контекстне меню може бути створене практично для всіх компонентів Embarcadero C++ Builder і формується за допомогою редактора меню.

Button — кнопка. Цей компонент призначається для керування виконанням програми. Помістивши TButton на форму, за подвійним клацанням можна створити обробник події натискання кнопки і написати свій програмний код.

Checkbox — незалежний перемикач. Цей компонент призначається для вибору, причому можна вибрати один, всі або ні одного з варіантів, що

пропонуються. TCheckBox відображає рядок тексту з маленьким віконцем поруч, де можна поставити позначку.

RadioButton — залежний перемикач (опція). Цей компонент призначається для вибору тільки одного з варіантів (опцій), що пропонуються. Вибір іншого варіанту виключає варіант, обраний раніше.

ListBox — список рядків з вертикальною лінійкою скролінгу. Призначається для вибору рядка. Головна властивість – Items. Номер обраного рядка відомий у програмі як властивість ItemIndex. Неопублікована властивість Count (цілого типу) визначає кількість рядків списку. Номер першого елемента списку = 0. За допомогою властивості Sorted = True можна сортувати список на етапі розробки або виконання додатку.

ComboBox — комбінований рядок введення. Цей компонент схожий на компонент ListBox і також призначається для вибору рядка. Основною відзнакою цього компоненту від компонента ListBox є наявність поля вводу. Це поле вводу нагадує компонент Edit — рядок введення. Нумерація стрічок починається з нуля. Є кілька типів ComboBox, але найбільш популярний випадає вниз (drop-down combo box), який можна бачити внизу вікна діалогу вибору файлу

Головна властивість — Items, яка містить список рядків. Для компоненту ComboBox номер обраної стрічки відомий у програмі як властивість ItemIndex. Нумерація стрічок починається з нуля. Текст, розміщений у полі вводу, розміщується у властивості Text.

ScrollBar — лінійка скролінгу. Цей компонент призначається для розміщення на компоненті, який потребує прокрутки для перегляду розміщених на ньому елементів. TScrollbar — смуга прокрутки, з'являється автоматично в об'єктах редагування, ListBox'ах при необхідності прокрутки тексту для перегляду.

GroupBox — панель із заголовком для розміщення на ній компонентів. TGroupBox використовується для візуальних цілей і для вказівки Windows, який порядок переміщення по компонентах на формі (при натисканні клавіші TAB).

RadioGroup — панель із заголовком для розміщення на ній перемикачів.

Panel — панель для розміщення на ній компонентів. Для компонентів типа TPanel різні варіанти видів рамки панелей встановлені за допомогою властивостей BevelInner (внутрішня рамка), BevelOuter (зовнішня рамка) і BorderStyle (стиль бодюра). Вирівнювання тексту усередині компоненту визначається властивістю Alignment, яка дозволяє вирівнювати текст по лівому, правому краю або по центру клієнтської області і панелі.

ActionList — список операцій для компонентів (меню, кнопок і т. д.).

Є також компоненти для роботи з текстом Label, Edit, Memo.

План самостійного вивчення теми № 2

з навчальної дисципліни "Інструментальні засоби візуального програмування"

Тема	Інструментарій середовища візуального програмування.
	Створення візуального додатку. Робота з проєктом
Мета	Ознайомитись з інструментарієм середовища візуального програмування, навчитись створювати візуальний додаток, працювати з проєктом
	<i>знати:інструментарій середовища візуального програмування</i>
	<i>вміти:використовувати інструментарій середовища візуального програмування</i>

Час – 2 години

Навчальні питання:

- 1 Інтегроване середовище розробки
- 2 Створення проєкту візуального додатку
- 3 Робота з проєктом візуального додатку

Методичні рекомендації

Самостійно вивчаючи запропоновану тему, студент повинен засвоїти такі основні положення:

<i>По-перше</i>	Інтегроване середовище розробки — це програмний продукт, який об'єднує текстовий редактор, компілятор, відладчик і довідкову систему
<i>По-друге</i>	Проєкт — це набір взаємопов'язаних вихідних файлів, компіляція і компонування яких дозволяє створити виконувану програму
<i>По-третє</i>	Пакет включає інструментарій, необхідний для роботи в середовищі програмування

Література

Архангельский, А. Я. Программирование в C++ Builder [Текст] / А. Я. Архангельский.	/
- М. : Бином-Пресс, 2010. - 1230 с.	

Навчальні матеріали до самостійного вивчення теми

ВСТУП

Integrated Development Environment (інтегроване середовище розробки), або, скорочено, IDE — це програмний продукт, об'єднуючий текстовий редактор, компілятор, відладчик і довідкову систему. Embarcadero C++ Builder — середовище візуального програмування, призначене для швидкого проектування та розробки додатків.

Основні інструменти інтегрованого середовища розробки (IDE) Embarcadero C++ Builder

На рисунку 1.2 показаний Embarcadero C++ Builder відразу після запуску. Це інтегроване середовище розробки (IDE), що включає в себе чотири основних елементи. Нагорі знаходиться головне вікно. Воно містить звичайну лінійку меню , інструментальну панель (ліворуч) і палітру компонентів (багатосторінкова панель праворуч).

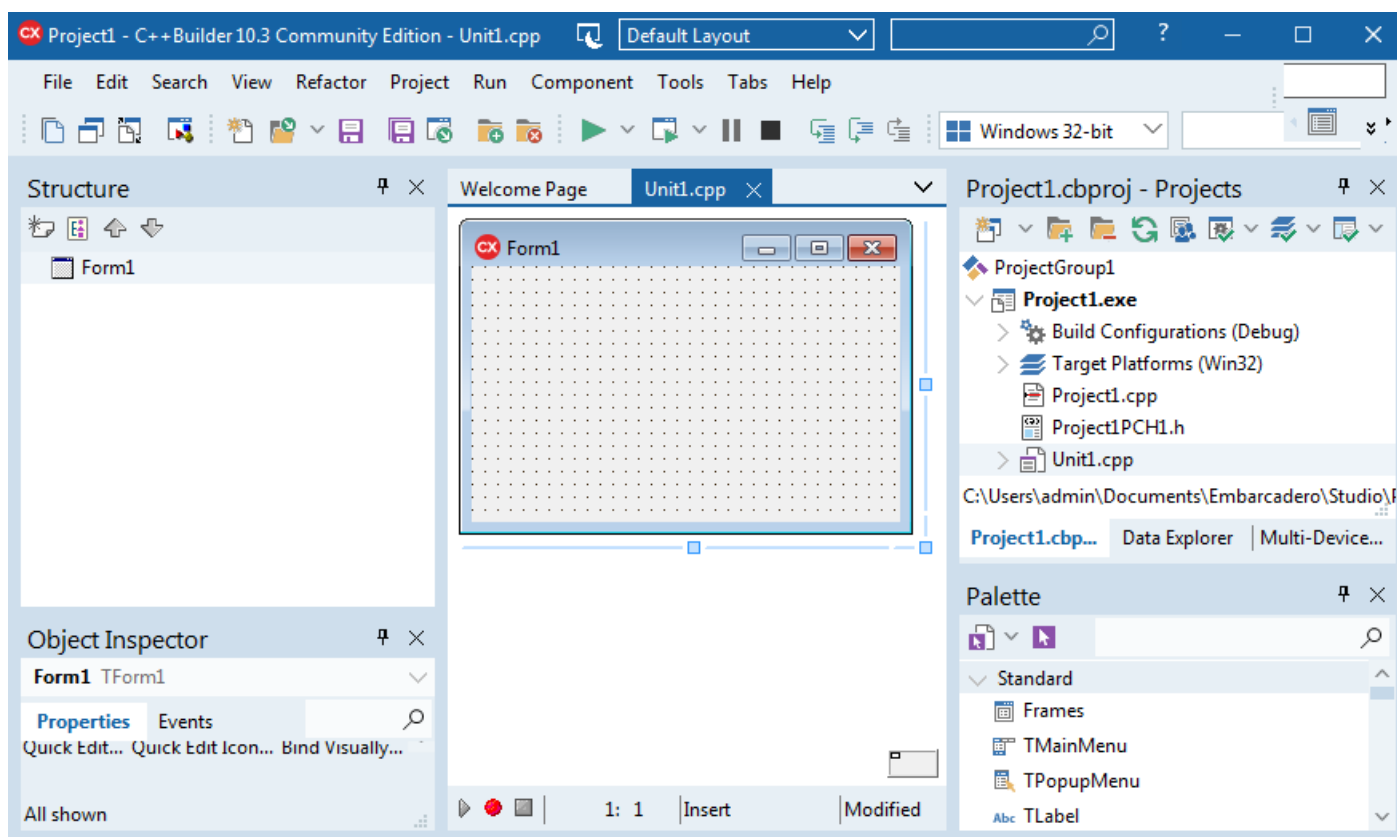


Рис. 1.2 – Вигляд інтегрованого середовища розробки

Правіше інспектора об'єктів розташовується конструктор форм. При створенні проекту конструктор відображає порожню форму. Форма — це центральний елемент візуального програмування. Вона може представляти головне вікно програми, дочірнє вікно, діалогову панель. На ній розміщуються різні елементи керування (типові і найпоширеніші — командна кнопка), звані візуальними компонентами. Існують також і невізуальні компоненти, наприклад, таймери і компоненти зв'язку з базами даних. У інспекторі об'єктів зіставляються подіям компонентів написані програмістом процедури обробки. Це, по суті, і є візуальне програмування, що базується на компонентній моделі.

Нарешті, під конструктором форм знаходиться вікно редактора коду. Його функції не обмежуються редагуванням вихідного тексту програми.

При першому запуску середовища до лівої сторони редактора коду пристиковане вікно оглядача класів, що наведено на рисунку 1.3.

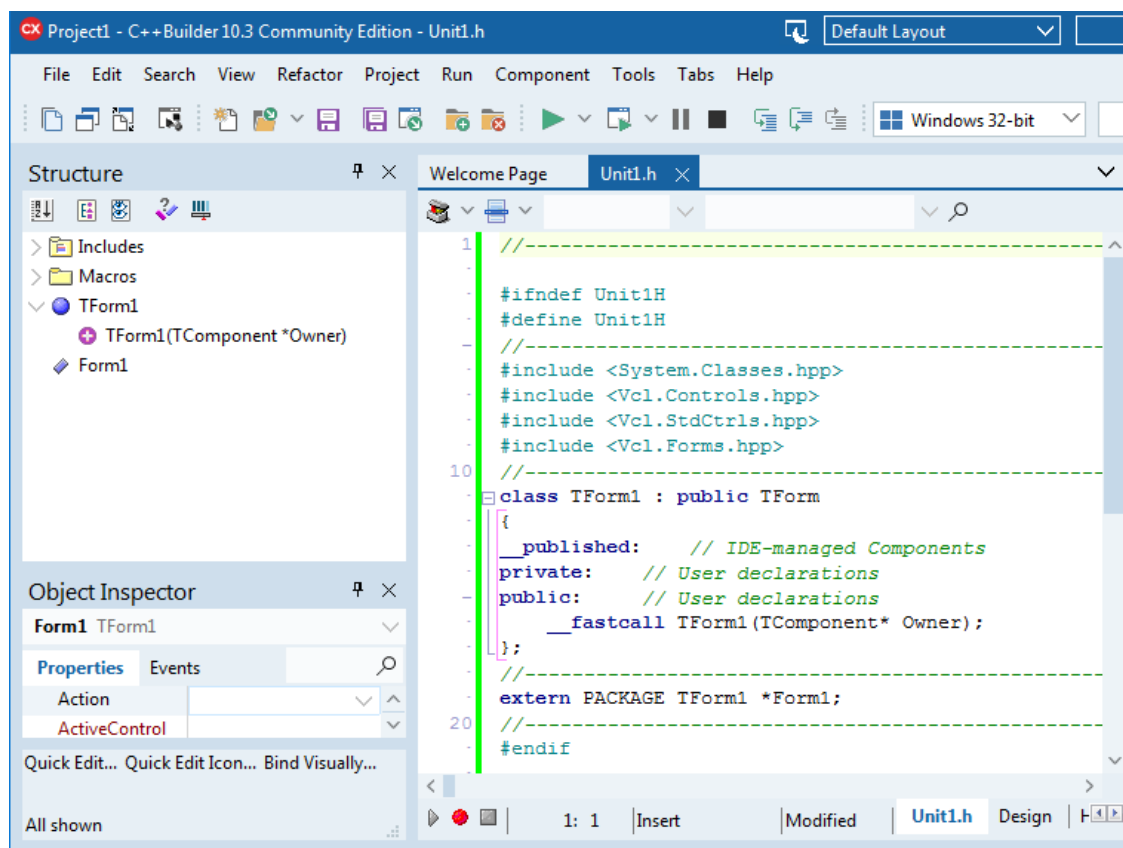


Рис. 1.3 – Вікно редактора коду з оглядачем класів

Інструментальне вікно Embarcadero C++ Builder може бути зістиковано з іншим вікном. У разі стикування по центру вікно стає багатосторінковим, з закладками, що дозволяють перемикатися між сторінками.

В Embarcadero C++ Builder, як і в багатьох інших сучасних програмах, багато операцій реалізуються через контекстні меню, які викликаються натисканням правої кнопки в тому чи іншому вікні.

В редакторі можна відкривати відразу декілька файлів вихідного коду. При цьому він також стає багатосторінковим вікном із закладками. Написи на закладках відображають імена файлів.

Основними елементами середовища програмування Embarcadero C++ Builder є:

- головне вікно , що включає головне меню, панелі інструментів, палітру компонентів;
- текстовий редактор (Code Editor) , що відображається одним або декількома вікнами;
- конструктор форм (Form) , що відображається одним або декількома вікнами - по одному на кожну форму;
- інспектор об'єктів (Object Inspector).

Головне меню Embarcadero C++ Builder складається з наступних підменю:

- File (робота з файлами) ;
- Edit (редагування) ;
- Search (пошук інформації) ;
- View (перегляд інформації) ;
- Project (параметри проекту) ;
- Run (виконання програми) ;
- Component (бібліотека компонентів);
- Database (база даних) ;
- Tools (інструментальні засоби) ;
- Help (довідкова система).

Панелі інструментів, що формуються при установці пакету Embarcadero C++ Builder, містять ряд кнопок з графічним зображенням дій, які вони виконують. Як правило, вони дублюють деякі найбільш часто використовувані команди головного меню, тому панелі інструментів можуть самі розглядатися як своєрідне однорівневе меню.

3 Проект у середовищі візуального програмування Embarcadero C++ Builder

Проект у середовищі візуального програмування Embarcadero C++ Builder створюється за натисканням пункту меню File => Windows VCL Application. В середовищі відразу відкривається візуальна форма.

Для кожного нового проекту в середовищі автоматично відображається вікно проектувальника форм (Form Designer), яке являє собою головне вікно майбутнього програми за замовчуванням має ім'я Form1.

Вікно редактора коду (Code Editor) має заголовок Unit1.cpp і знаходиться (після запуску середовища) позаду вікна проектувальника форм. Редагування програм здійснюється в редакторі. Embarcadero C++ Builder володіє найпотужнішим, вбудованим в редактор графічним відладчиком, що дозволяє знаходити і усувати помилки в коді.

Інспектор об'єктів служить для перегляду опублікованих властивостей компонентів форми (або самої форми), завдання їм значень, завдання оброблювачів подій. У ньому містяться характеристики або активна форма, або її активного компонента. Звичайно інспектор об'єктів виводиться на екран автоматично при запуску середовища, або за допомогою команди головного меню View|Object Inspector (F11).

Основну частину інспектора об'єктів займає двохсторінковий блокнот. На його першій сторінці (Properties) (Властивість) розміщені опубліковані властивості об'єкта із вказівкою імені властивості і його значень. Перед деякими іменами може стояти знак +. Це означає, що властивість є комбінованою і складається з декількох компонентів.

Значення властивостей відображаються або рядками уведення, або комбінованими рядками уведення. Комбінований рядок уведення відрізняється наявністю в правій частині кнопки із зображенням трикутника, спрямованого вниз, нажавши яку можна відкрити список, що випадає, припустимих значень властивості. У деяких рядках уведення перебуває кнопка із зображенням

трьох крапок. Це означає, що при натисканні кнопки виводиться вікно, за допомогою якого задається ряд параметрів комбінованої властивості.

Друга сторінка (Events) (Подія) блокнота містить імена подій, які може обробляти даний об'єкт, із вказівкою імен оброблювачів подій або відсутністю таких. Якщо імені оброблювача подій нічого не відповідає, то дана подія об'єктом не обробляється. При активізації оброблювача події відбувається перехід до сторінки текстового редактора, у якій перебуває текст цього оброблювача.

У верхній частині інспектора об'єктів перебуває комбінований рядок введення, список якої містить імена активної форми й всіх її компонентів.

Інспектор об'єктів — це один із трьох найцінніших інструментів розробки інтерфейсу додатка. Два інших — це проектувальник форм (Form Designer) і редактор коду (Code Editor).

Компонент таблиця комірок. Компоненти кнопки. Панелі інструментів

Актуальність і основна ціль теми:

- ознайомити з основними властивостями та методами компоненту таблиця комірок;
- ознайомити з основними властивостями та методами кнопок.

Студент повинен знати:

- основні властивості та методи компоненту – таблиця комірок;
- основні властивості та методи компоненту – кнопка.

Студент повинен уміти:

- використати основні властивості та методи таблиці комірок для розробки програм;
- використати основні властивості та методи кнопок для розробки програм.

Основні питання:

1. Основні властивості компоненту таблиця комірок (StringGrid).
2. Властивість Options.
3. Основні події компоненту.
4. Подія OnSelectCell.
5. Типи кнопок.
6. Загальні властивості кнопок.
7. Кнопки BitBtn.
8. Радіокнопки.

Питання для самоперевірки і контролю:

1. Для чого призначений компонент-таблиця?
2. Які властивості компоненту-таблиці вам відомі?
3. Які методи компоненту-таблиці вам відомі?

4. Як використати подію OnSelectCell?
5. Для чого призначені різні типи кнопок?
6. Як реагують на події кнопки?
7. Які загальні властивості кнопок?
8. Які властивості кнопки BitBtn?
9. Яке призначення радіокнопки?

Рекомендована література:

1. Архангельский, А. Я. Программирование в С++ Builder [Текст] / А. Я. Архангельский. - М. : Бином-Пресс, 2010. - 1230 с.
2. Архангельский, А. Я. Delphi 7. Справочное пособие [Текст] / А. Я. Архангельский. - М. : Бином-Пресс, 2004. - 1024 с.

Таблиця комірок - компонент StringGrid

Компонент **StringGrid** є таблицею для відображення даних у вигляді рядків і стовпців. Цей компонент також називають таблицею, сіткою рядків або просто сіткою. Таблиця може мати смуги прокрутки.

Дані таблиці можуть бути тільки для читання або редагованими. Таблиця може мати задану кількість фіксованих (верхніх) рядків і стовпців (зліва). У них можна задати заголовки стовпців (у фіксованих рядках) і пояснення до вмісту кожного рядка (у фіксованих стовпцях).

Список основних опублікованих властивостей StringGrid даний в табл.1. Всі властивості доступні під час виконання додатку. Виводити тексти в таблицю можна програмно в кожен осередок або по стовпцях або по рядках за допомогою методів класу TStrings.

Таблиця 1 - Основні опубліковані властивості компоненту StringGrid

Властивість	Призначення
ColCount	Кількість рядків таблиці
RowCount	Кількість стовпців таблиці
FixedCols	Кількість фіксованих (крайніх лівих) стовпців таблиці
FixedRows	Кількість фіксованих (верхніх) рядків таблиці
DefaultColWidth	Ширина стовпця в пікселях
DefaultRowHeight	Висота рядка в пікселях
Color	Колір фону частини таблиці, що заповнюється
FixedColor	Колір фону фіксованої частини таблиці
GridLineWidth	Товщина ліній сітки таблиці в пікселях
ScrollBars	Можливість відображення полос прокрутки
Options	Параметри таблиці

Властивість `ScrollBars` визначає можливість відображення смуг прокрутки за допомогою одного з наступних значень:

- `ssNone` — смуги прокрутки не допускаються;
- `ssHorizontal` - допускається горизонтальна смуга;
- `ssVertical` - допускається вертикальна смуга;
- `ssBoth` - допускаються обидві смуги прокрутки (за умовчанням).

Якщо смуги прокрутки встановлені, вони з'являються і зникають в процесі виконання додатку автоматично, в міру необхідності.

Властивість `Options`

Властивість `Options` є безліччю параметрів, кожний з яких типу `Boolean`. Воно може приймати набір з наступних значень:

- `goFixedVertLine` і `goFixedHorzLine` - відображати в сітці для фіксованих елементів вертикальні і горизонтальні лінії відповідно;
- `goVertLine` і `goHorzLine` - відображати в сітці для області даних вертикальні і горизонтальні лінії відповідно;
- `goRangeSelect` - дозволити користувачу вибір розміру осередків; ігнорується при `goEditing = True`;
- `goDrawFocusSelected` - вибраний осередок виділяється прямокутною рамкою і кольором;

- goRowSizing і goColSizing - можна змінювати висоту рядків і ширину стовпців з даними;
- goRowMoving і goColMoving - можна переміщати рядки і стовпці з даними за допомогою миші;
- goEditing - можна редагувати дані;
- goTabs - допускається переміщення між осередками за допомогою клавіші Tab;
- goRowSelect - можна вибрати весь рядок;
- goAlwaysShowJiditor - сітка не блокує режим редагування;
- goThumbTracking - дані обновляються в процесі прокручування таблиці (а не тільки після відпуску бігунка прокрутки).

В процесі виконання додатку можна використовувати ряд неопублікованих властивостей компоненту. Зокрема:

- LeftCol і TopCol - індекси першого видимого на екрані прокручуваного (з даними) стовпця і першого видимого прокручуваного рядка;
- ColWidths[int Index] і RowHeights[int Index] - дозволяють задати в пікселях ширину стовпця і висоту рядка з номером Index.

Кількість рядків і стовпців можна змінити з програми за допомогою властивості ColCount. Наприклад, за допомогою операторів:

```
// - додати стовпець
StringGrid1.ColCount := StringGrid1.ColCount + 1;
// - додати рядок
StringGrid1.RowCount := StringGrid1.RowCount + 1;
```

Можна очистити доданий рядок. Наприклад, оператором:

```
StringGrid1.Rows[StringGrid1.RowCount - 1].Clear;
```

Видалення останнього рядка можна виконати оператором:

```
StringGrid1.RowCount := StringGrid1.RowCount - 1;
```

Основні властивості компонентів, що визначають дані, що відображаються у табл.2.

Таблиця 2 - Основні властивості компонентів, що визначають дані

Cells [col] [row]	Рядок із комірки з індексами col, row
Cols [Index]	Список рядків із стовпця з номером Index
Rows [Index]	Список стовпців із рядка з номером Index

При зверненні до вмісту осередку в списку її номерів першим указується номер стовпця (Col), другим - номер рядка (Row).

Рядки і стовпці нумеруються, починаючи з нуля до RowCount - 1 і до ColCount - 1 відповідно.

Всі ці властивості доступні під час виконання додатку. Виводити тексти можна програмно по окремих осередках, використовуючи властивість Cells, або відразу по стовпцях (властивістю Cols) або по рядках (властивістю Rows).

За допомогою властивостей Cols і Rows можна виконувати операції над рядками і стовпцями. Наприклад, копіювати їх значення в простий список за допомогою операторів:

```
// -копіювання стовпця
ListBox1.Items.Assign(StringGrid1.Cols[3]);
// - копіювання рядка
ListBox2.Items.Assign(StringGrid1.Rows[4]);
```

В основному компонент StringGrid використовується для вибору або коректування користувачем значень, відображених в елементах таблиці.

Подія OnSelectCell

Серед безлічі подій компоненту StringGrid треба відзначити подію **OnSelectCell**. Вона виникає у момент вибору користувачем осередку (клацанням в осередку). Приклад заголовка методу для реакції додатку на вибір елементу таблиці:

```
void __fastcall TForm1::StringGrid1SelectCell(TObject
*Sender, int ACol, int ARow, bool &CanSelect)
{
    if ((ARow == 1) && (ACol == 1)) CanSelect = false;
}
```

де ACol, ARow - параметри, які містять номери стовпця і рядка відповідно;

CanSelect - параметр, який визначає допустимість вибору даного осередку; з його допомогою можна дозволити (для CanSelect = True) або заборонити (для CanSelect = False) вибір будь-якого елементу таблиці.

Наприклад оператором можна заборонити вибір осередків 1-й рядка і 1-го стовпця:

```
if ((ARow = 1) or (ACol = 1)) CanSelect := false;
або
if ((ARow = 1) and (ACol = 1)) CanSelect := false;
StringGrid1.Cells[2][1] = "66";
```

При виборі осередку можна використовувати операторів для обробки даних вибраного рядка. Наприклад, можна вивести на мітки номера рядка і стовпця вибраного осередку і її значення операторами:

```
Label1.Caption := "Вибраний осередок " + IntToStr(ARow) + ":"
+ IntToStr(ACol);
Label2.Caption := StringGrid1.Cells[ACol][ARow]; //вивід
вмісту.
```

Компонент StringGrid може реагувати на різні події. Наприклад, на наступні:

OnClick - при клацанні на компоненті;

OnDblClick - при подвійному клацанні на компоненті;

OnSelectCell - при виборі осередку компоненту.

Компоненти доступу до бази даних

Створення підключення до бази даних у візуальному додатку

Актуальність і основна ціль теми:

- ознайомити з основними властивостями компонентів доступу до бази даних;
- ознайомити з основними методами компонентів доступу до бази даних.

Основні питання:

9. Компонент ADOConnection.

10. Компонент ADOTable.

11. Компонент DataSource.

Питання для самоперевірки і контролю:

10. Як створити підключення до бази даних?

11. Як вибрати дані з таблиці БД?

12. Як вивести дані на форму?

Рекомендована література:

Архангельский, А. Я. Программирование в C++ Builder [Текст] / А. Я. Архангельский. - М. : Бином-Пресс, 2010. - 1230 с.

Компонент ADOConnection.

Треба задати властивість ConnectionString, де вказати драйвер та параметри підключення до бази даних.

Після виконаних налагоджень властивість Connected=true виконує підключення. Доцільно використовувати одне підключення для всього програмного додатку.

Компонент TDataSource.

Компонент DataSource діє як посередник між компонентами TDataSet (TTable , TQuery , TStoredProc) і компонентами Data Controls — елементами управління , що забезпечують представлення даних на формі. Компоненти TDataSet управляють зв'язками з бібліотекою Borland Database Engine (BDE) , а компонент DataSource управляє зв'язками з даними в компонентах Data Controls.

У типових додатках БД компонент DataSource , як правило , пов'язаний з одним компонентом TDataSet (TTable або TQuery) і з одним або більше компонентами Data Controls (такими , як DBGrid , DBEdit та ін.) Зв'язок цього компоненту з компонентами TDataSet і DataControls здійснюється з використанням наступних властивостей і подій.

Властивість DataSet компонента DataSource ідентифікує ім'я компонента TDataSet. Можна привласнити значення властивості DataSet на етапі виконання або за допомогою інспектора об'єктів на етапі проектування .

Властивість Enabled компонента DataSource активізує або зупиняє взаємозв'язок між компонентами TDataSource і Data Controls. Якщо значення властивості Enabled одно true, то компоненти Data Controls, пов'язані з TDataSource, сприймають зміни набору даних.

Використання властивості Enabled дозволяє тимчасово роз'єднувати візуальні компоненти Data Controls і TDataSource, наприклад, для того, щоб у разі пошуку в таблиці з великою кількістю записів не відображати на екрані перегортування всієї таблиці.

Властивість AutoEdit компонента DataSource контролює, як ініціюється редагування в компонентах Data Controls. Якщо значення властивості AutoEdit одно true, то режим редагування починається безпосередньо при отриманні фокусу компонентом Data Controls, зв'язаним з даним компонентом TDataSet. В іншому випадку режим редагування починається, коли викликається метод Edit компонента TDataSet, наприклад, після натиснення користувачем кнопки Edit на компоненті DBNavigator.

Подія onDataChange компонента DataSource настає, коли відбувається зміна значення поля, запису, таблиці, запиту.

Подія OnUpdateData компонента DataSource настає, коли користувач намагається змінити поточний запис в TDataSet. Оброблювач цієї події слід створювати, коли потрібно дотримати умови посилальної цілісності або обмеження, що накладаються на значення полів змінною бази даних.

Компонент TTable.

Найбільш простим способом звернення до таблиць баз даних є використання компонента TTable, що надає доступ до однієї таблиці.

Для цієї мети найбільше часто використовуються наведені нижче властивості.

Active — вказує, відкрита (true) чи ні (false) дана таблиця.

DatabaseName — ім'я каталогу, що містить шукану таблицю, або псевдонім (alias) видаленої БД (псевдоніми встановлюються за допомогою відповідних утиліт, наприклад, MeSQLConnector). Ця властивість може бути змінена тільки в разі, якщо таблиця закрита.

Exclusive — якщо ця властивість приймає значення true, то ніякий інший користувач не може відкрити таблицю, якщо вона відкрита даним додатком. Якщо ця властивість одно false (значення за замовчуванням), то інші користувачі можуть відкривати цю таблицю.

IndexName — ідентифікує вторинний індекс для таблиці. Цю властивість не можна змінити, поки таблиця відкрита.

MasterFields — визначає ім'я поля для створення зв'язку з іншою таблицею.

MasterSource — ім'я компонента TDataSource, за допомогою якого TTable буде отримувати дані з пов'язаної таблиці.

ReadOnly — якщо ця властивість true, таблиця відкрита в режимі " тільки для читання ". Не можна змінити властивість ReadOnly, поки таблиця відкрита.

Eof, Bof — ці властивості приймають значення true, коли покажчик поточного запису розташований на останній або відповідно першого запису таблиці.

Вивести інформацію з логічного абору можна у компоненти класу TDataControls такі як DBGrid, DBEdit, DBComboBox та інші. Для цього треба задати властивість DataSource, а для компонентів, що потребують виведення значення одного поля, ще FieldName.

Розробка інтерфейсу додатку

Наведемо характеристики Windows-додатків. Внаслідок того, що програмна архітектура Windows-додатків заснована на принципі обміну повідомленнями, всі вони мають загальні програмні модулі, які зазвичай явно включаються до початкового коду. Завдяки використанню бібліотек компонентів цей процес здійснюється в автоматичному режимі.

Будь-який Windows-додаток включає спеціальну функцію, яка не використовується безпосередньо програмою, а викликається самою операційною системою. Подібна функція отримала найменування функції вікна. Вона викликається на виконання в Windows в тому випадку, коли система передає повідомлення програмі. Завдяки цьому здійснюється взаємодія між програмою і системою. Функція вікна передає повідомлення, використовуючи власні параметри.

Додатки (прикладні програми) Embarcadero C++ Builder є інтерактивними системами, в яких для організації взаємодії між користувачем і програмою використовуються методи (підпрограми), керовані подіями.

Обмін інформацією між об'єктами здійснюється за допомогою повідомлень. Повідомлення є результатом появи подій.

Програмування, кероване подіями, не виключає використання процедурного програмування.

Подія — це відгук на зовнішній вплив. Суть програмування, керованого подіями, складається у відстеженні таких подій, які вимагають реакції програми. У процесі функціонування операційної системи (ОС) Windows виникає багато різних подій, але тільки деякі з них вимагають відгуку конкретного додатка. Середовище Embarcadero C++ Builder пов'язує методи (реакцію) програми з подіями, що відбуваються в ОС. Мова програмування, що використовується в Embarcadero C++ Builder, на самому нижньому рівні тісно пов'язана з внутрішніми функціями Windows. Цей зв'язок прихований в компонентах, об'єктах і методах. З їх допомогою система візуального програмування спрощує створення Windows-додатків.

Програма для Windows — це набір об'єктів, що посилають і приймають повідомлення. Кожен з об'єктів, що відповідають елементам інтерфейсу Windows, може містити обробники різних повідомлень.

Основним вікном, яке розробляється, є форма. У процесі розробки програми при розміщенні об'єкта на формі (наприклад, кнопки) у візуальному середовищі основні параметри об'єкта (розмір, положення на екрані, колір тощо) відразу відображаються у вигляді реального компонента на формі, а відповідний йому код мовою C++ автоматично записується в вихідний файл форми, який відображає об'єкт в процесі виконання програми. Потім цей вихідний код компілюється у виконуваний машинний код.