5. Розв'язуємо задачу (5) і здійснюємо перехід до нового розподілу. Для цього, для кожного елемента $y_k = 0$ розв'язку задачі (5) приймаємо $x_{i_k r}$=0 і $x_{i_k p_k}$=1. У такий спосіб отримуємо новий розподіл X, при якому умова (3) виконується для індексу $j=r$.

6. Приймаємо $\varepsilon_r = 1$ і переходимо до пункту 2.

7. X — знайдений розподіл.

8. Кінець.

1. Баранов В. И., Стечкин Б. С. Экстремальные комбинаторные задачии их приложения. М.,1989. 2. Марданов С. С. Сокращение размерности целочисельной задачи о ранце и ее решение паралельным алгоритом динамического програмрования.//Дискрет. математика. 1992. Т4. № 2. С.32-38. 3. Мину М. Мамематическое програмирование. Теория и алгоритмы. М., 1990. 4. Цегелик Г. Г. Системы распределенных баз данных. Львов, 1990. 5. Nemhauser G. L., Wosley L. A. Combinatorial and Integer Programming. John Wiley, New York, 1988. 6. Legedre J. P., Minoux M. Une application de dualite m programmation en nobers entiers-selection et affections optimalis d'une flotte d'avions.// R.A.I.R.O. 1977. Vol.11, №2, P 201-222.

Стаття надійшла до редколегії 04.03.96

УДК 681.3.06

М.Ю. Щербина

Реалізація Outline з використанням Microsoft Visual C++ 1.51 і бібліотеки класів Microsoft Foundation Class*

Компілятор Microsoft Visual C++ 1.51 є потужним засобом для розробки програм під Microsoft Windows. Бібліотека класів Microsoft Foundation Class (MFC) і наявні засоби візуального програмування дають змогу спростити і прискорити процес створення програм. Проте MFC орієнтована на використання стандартних Windows-компонент, таких, як Button, Edit, ListBox і т.п. Фактично MFC можна вважати заміною Windows SDK.

---

* Ця робота була частково підтримана Міжнародною Соросівською програмою підтримки освіти в галузі точних наук (ISSEP), грант № U051215.

Іноді виникає потреба використовувати нестандартні елементи, як от елемент, відомий під назвою Outline. Прикладом Outline може бути вікно програми File Manager, яке містить дерево піддиректорій. Таким чином, під Outline розумітимемо вікно, яке відображає певним чином деревовидну структуру і дає змогу виконувати над нею задані дії: розгортати підрівень або піддерево поточної вершини, згортати відкрите піддерево, різним чином рухатись по вузлах дерева.

Вікна типу Outline використовуються у Windows-програмах настільки часто, що цей тип вікон можна вважати "стандартним". На жаль, реалізація Outline-класа відсутня і в MFC, і в MFC Samples. З огляду на це розроблена власна реалізація Outline. Сформулюємо задачі, які має виконувати Outline-об'єкт.

1. Надавати набір методів для формування деревовидної структури.

2. Відображати частину дерева, яка відкрита у даний момент.

3. Дозволяти навігацію по дереву з використанням стандартних клавіш:

й — перейти до попереднього пункту;

к — перейти до наступного пункту;

Ctrl + й — перейти до попереднього пункту на тому самому рівні;

Ctrl + к — перейти до наступного пункту на тому самому рівні;

з — перейти до пункту на рівень вище;

и — перейти до пункту на рівень нижче.

4. Дозволяти дії над деревом з використанням стандартних клавіш:

Enter — розгорнути/згорнути підрівень/піддерево поточної вершини (аналогічно подвійному натисканню на кнопку миші);

+ — розгорнути підрівень поточної вершини;

* — розгорнути піддерево поточної вершини;

Ctrl + * — розгорнути все дерево;

- — згорнути піддерево поточної вершини.

5. Дозволяти навігацію і згортання/розгортання за допомогою миші.

Одне з найважливіших завдань при реалізації Outline — забезпечити відображення дерева у вікні. В існуючих реалізаціях, які відомі автору, для цього використовуються різні методи. Реалізація Outline у Borland Delphi — це повне створення свого класу вікна з обробкою будь-яких подій і цілковито самостійно реалізованою процедурою малювання вікна. На думку автора, цей метод невиправдано ускладнює

задачу малювання та обробки клавіатури і миші. Було прийняте рішення реалізувати Outline на базі стандартного для Windows класу ListBox. Для нормальної роботи ListBox, що створюється, повинен мати стилі LBS_OWNERDRAWFIXED, LBS_HASSTRINGS та LBS_NOREDRAW. Решта стилів можна вказувати опціонально. Зрозуміло, що стилі LBS_EXTENDEDSEL та LBS_MULTIPLESEL не можуть бути вказані.

Реалізація Outline містить два класи: COutline, породжений від CListBox, який і використовується у програмах, та клас COutlineNode, який репрезентує вузол дерева і має внутрішнє використання.

Кожен вузол дерева має такі поля:

void* Data — зберігає вказівник на деякі дані, що асоціюються з конкретним вузлом; це реалізовано для заміни функцій типу GetItemDataPtr, які мають внутрішнє використання (з кожним пунктом ListBox якраз і асоціюється об'єкт класу COutlineNode);

BOOL Last — вказує, чи заданий вузол останній серед синів свого батьківського елемента; використовується насамперед для малювання;

int Level — вказує рівень заданого вузла (0 для першого рівня); використовується насамперед для малювання;

COutlineNode* Parent — вказівник на батьківський вузол; використовується для малювання.

Розглянемо методи, які реалізовані у класі COutline. Насамперед необхідно перевизначити віртуальний метод void DrawItem (LPDRAWITEMSTRUCT lpDIS), який відповідає за малювання заданого пункту з ListBox. Структура DRAWITEMSTRUCT містить поле UINT itemID, яке і вказує індекс пункту, який малюватиметься. Алгоритм малювання виглядає так:

Якщо акція малювання ODA_DRAWENTIRE, то:

1. Розрахувати відступ заданого пункта (використовуючи поле Level). Намалювати горизонтальну лінію перед піктограмою та текстом.

2. Якщо пункт останній на своєму рівні (поле Last), то намалювати вертикальну лінію половинної висоти, інакше — вертикальну лінію повної висоти.

3. Провести цикл, рухаючись по батьківських елементах до кореня дерева. Для кожного батька, якщо він не останній на своєму рівні (поле Last), намалювати на відповідному рівні (поле Level) вертикальну лінію.

4. Намалювати відповідну піктограму залежно від того, чи має синів пункт, що малюється.

5. Вивести текст.

Якщо акція ODA_SELECT або елемент обраний (ODS_SELECTED) та акція ODA_DRAWENTIRE, то проінвертувати текст.

Запропонований автором алгоритм створювався поступово, і метою вдосконалень було максимально пришвидшити малювання. У результаті перемалювання (наприклад, при прокручуванні) дерева, що містить понад 750 вузлів, відбувається непомітно для користувача.

Інший цікавий момент — реалізація додавання елементів у Outline. У Borland Delphi дерево цілком міститься у пам'яті. У даній реалізації метод Expand, який відкриває синів обраного елемента, реалізований як абстрактний віртуальний метод. Він автоматично викликається при обробці подій від клавіатури та миші і має використовувати метод AddChild для додавання елементів у дерево.

Під час розробки Outline виникло чимало проблем, які не достатньо широко висвітлені у літературі з програмування під Windows і в документації фірми Microsoft. По-перше, метод SetCurSel класу CListBox не викликає нотифікаційного повідомлення LBN_SELCHANGE батьківському вікну, тому реалізація нестандартної обробки клавіатури має самостійно викликати це повідомлення, наприклад: ::SendMessage (::GetParent(m_hWnd), WM_COMMAND, GetDlgCtrlID(), MAKELONG(m_hWnd, LBN_SELCHANGE))

Друга проблема виникла при обробці натискання клавіші Enter. Обробка VK_RETURN у методі OnKeyDown не викликається, поки не перевизначити метод OnGetDlgCode. Останній повертає, яку обробку клавіатури бере на себе елемент з вікна діалогу. У нашому випадку єдино можливим значенням, що повертається, має бути DLGC_WANTALLKEYS (обробка всіх клавіш). Але при цьому Outline втрачає можливість автоматичної обробки Tab і Shift + Tab, що не є задовільним. Тому обробку табуляції треба реалізувати самостійно, викликаючи для батьківського вікна відповідно методи NextDlgCtrl і PrevDlgCtrl.

Перелічимо деякі основні методи класу COutline:

int AddChild(LPCSTR lpszItem, int nodeIndex = OUTLINE_ROOT);

int AddChildObject(LPCSTR lpszItem, void* data, int nodeIndex = OUTLINE_ROOT);

void Collapse(int nodeIndex = OUTLINE_ROOT);

```
void DeleteItem(LPDELETEITEMSTRUCT lpDeleteItemStruct);
void DrawItem(LPDRAWITEMSTRUCT lpDIS);
virtual void Expand(int nodeIndex = OUTLINE_ROOT) = 0;
void ExpandAll(int nodeIndex = OUTLINE_ROOT);
virtual CString GetDrawText(int nodeIndex);
void* GetItemPtr(int nodeIndex);
int GetLevel(int nodeIndex);
CString GetCurPath(char delim = '.');
BOOL HasSubLevel(int nodeIndex);
afx_msg void OnDblClk();
void ResetContent();
void SetItemPtr(int nodeIndex, void* data).
```

Наприкінці наведемо основні властивості деяких існуючих реалізацій Outline.

1. Borland Delphi. Outline реалізований повністю самостійно. Програма SPY повідомила, що вікно Outline має зареєстрований Windows-клас TOutline. Зрозуміло, що нормальне використання цієї реалізації з Visual C++ практично неможливе.

2. File Manager. Використовуючи SPY, з'ясувалось, що Directory Tree реалізовано як owner-draw list box, тобто фактично методом, описаним у статті. Таким самим методом реалізований каталог у Books Online.

3. MSOUTLIN.VBX. Автору стало відомо про реалізацію Outline фірмою Microsoft як Visual Basic Control (з усіма недоліками і перевагами, що випливають з цього). Проте у поставку Visual C++ 1.51 цей VBX-елемент не входить, у зв'язку з чим виникають проблеми у легальному використанні цієї реалізації.

4. Є інформація, що у нових 32-бітових операційних системах фірми Microsoft вирішено зробити Outline стандартним елементом.

Стаття надійшла до редколегії 12.09. 95