Лабораторна робота 5. Взаємодія керованого і некерованого коду

Вправа 1. Використання COM-компонента для створення PDF-застосування (3 бали)
Вправа 2. Виклик функції API (3 бали)

Мета роботи: Вивчення способів використання можливостей об'єднання старого коду з кодом, керованим середовищем CLR та отримання навичок по роботі із службами взаємодії керованого коду з некерованим.

Вправа 1. Використання COM-компонента для створення PDF-застосування (3 бали)

При створенні застосувань, які використовують платформу Microsoft .NET, виникає завдання використання у власних проектах вже готових бібліотек коду, написаних на інших мовах. Стислі терміни розробки і вже наявні програмні блоки не дозволяють відмовитися від готових рішень, тому їх доводиться використовувати, вбудовуючи в структуру власних проектів.

Код, який виконується під управлінням середовища виконання (у випадку платформи .NET - середовища **Common Language Runtime**), називається ***керованим***.

Код, який запускається не під управлінням зазначеного середовища, називається ***некерованим***. Прикладом некерованого коду можуть служити **COM-компоненти, Microsoft ActiveX інтерфейси** і **функції API Win32**.

Microsoft .NET Framework дозволяє взаємодіяти з COM-компонентами, COM+-службами, зовнішніми типами бібліотек і різними службами операційної системи.

Платформа .NET Framework пропонує дві служби ***взаємодії керованого коду з некерованим*** - **Platform Invoke** і **COM interoperability**, які використовують реальні можливості проекту.

1. Створіть нове Windows-застосування і назвіть його **PDFReader**.
2. Додайте на форму елементи **OpenFileDialog** і **MenuStrip**.
3. Встановіть наступні властивості форми **Form1**:

Властивість	Значення
Text	Перегляд документів у форматі PDF
WindowState	Maximized

4. Властивість **FileName** елемента **OpenFileDialog** зробіть порожньою.
5. Додайте пункт меню верхнього рівня **File** з командами **Open** та **Exit**.
6. Виберіть **Tools → Choose Toolbox Items** (**Сервис → Выбрать элементы панели элементов**).
7. На вкладці **COM Components** виберіть **Adobe Acrobat (PDF) Reader** (цей компонент з'являється після встановлення програми **Adobe Acrobat Reader**) і натисніть **OK**. Переконайтеся, що на панелі **Toolbox** цей елемент управління дійсно з'явився.
8. Перенесіть **Adobe Acrobat Reader** на форму і встановіть властивості **Dock** значення **Fill**, а властивості **(Name)** - **axAcroPDF1** (від імені об'єкта викликатимуться потрібні нам методи, в принципі ім'я може бути будь-яким).
9. Додайте обробник пункту меню **Open**:

```
private void openToolStripMenuItem_Click (object sender, System.EventArgs e)
```

```
{
        openFileDialog1.Filter = "Файли pdf|*.pdf";
        openFileDialog1.ShowDialog();
        axAcroPDF1.LoadFile(openFileDialog1.FileName);
}
```

10. Реалізуйте обробник події **Click** для пункту меню **Exit**.
11. Побудуйте і запустіть застосування. При відкритті документа у форматі **pdf** відбувається, по суті, вбудовування у форму інтерфейсу програми **Adobe Acrobat Reader**.

Вправа 2. Виклик функції API (3 бали)

Служби **Platform Invoke** дозволяють керованому коду запускати функції некерованого коду, які ***знаходяться у файлах бібліотек динамічного компонування (DLL)***.

Ці служби надають механізми виявлення і запуску некерованих функцій і перетворення типів даних вхідних і вихідних аргументів функції.

Коли керований код запускає функцію некерованого коду, локалізовану в DLL-файлі, служби **Platform Invoke** знаходять цей DLL-файл, завантажують його в оперативну пам'ять і знаходять адресу функції в пам'яті. Після цього служби передають вхідні аргументи функції, в стек, перетворюють дані, які потрібно перетворить, емулюють прибирання сміття і передають управління за адресою некерованої функції в пам'яті.

Першим кроком у запуску некерованої функції є оголошення функції. Функція має бути ***статичною*** (static) і ***зовнішньою*** (extern). Далі імпортується бібліотека, яка містить цю функцію.

Імпортувати бібліотеку треба, використовуючи атрибут **DllImport**, який знаходиться в просторі імен System.Runtime.InteropServices.

1. Створіть нове застосування і назвіть його **WinAnim**.
2. Розташуйте на формі три кнопки і встановіть наступні властивості форми і кнопок:

Властивість	**Значення**
Form1	
Text	Анімація форми
Button1	
Name	btnAW_BLEND
Location	30; 62
Size	232; 23
Text	Поява
Button2	
Name	btnHOR_AW_SLIDE
Location	30; 118
Size	232; 23
Text	Горизонтальна поява
Button3	
Name	btnCenter_AW_SLIDE
Location	30; 182
Size	232; 23
Text	Поява з центру

3. Додайте клас **WinAPIClass**:

```
using System;
using System.Runtime.InteropServices;
using System.Windows.Forms;
using AnimatedWindow;

namespace AnimatedWindow
{
	public class WinAPIClass
	{
		#region Анімація вікна
		/// <summary>
		/// Тип анімації вікна. Перерахування повертає тип int
		/// після приведення. Кожному елементу присвоєно
		/// своє значення типу int.
		/// </summary>
		[Flags]
		public enum AnimateWindowFlags: int
		{
			AW_HOR_POSITIVE = 1,
			AW_HOR_NEGATIVE = 2,
			AW_VER_POSITIVE = 4,
			AW_VER_NEGATIVE = 8,
			AW_CENTER = 16,
			AW_HIDE = 65536,
			AW_ACTIVATE = 131072,
			AW_SLIDE = 262144,
			AW_BLEND = 524288
		};
		/// <summary>
		/// Анімація вікна.
		/// </summary>
		/// <param name="hwnd">Вікно.</param>
		/// <param name="dwTime">Час.</param>
		/// <param name="dwFlags">Тип анімації.
		/// Якщо в некерованому коді використовується перерахування,
		/// то його треба конвертувати в тип даних int.
		/// </param>
		/// <returns></returns>
		[DllImportAttribute("user32.dll", EntryPoint="AnimateWindow", SetLastError=true)]
		public static extern bool AnimateWindow(IntPtr hwnd, int dwTime, int dwFlags);
		/// <summary>
		/// Анімація вікна.
		/// </summary>
		/// <param name="ctrl">Вікно.</param>
		/// <param name="dwTime">Час.</param>
		/// <param name="Flags">Прапорці.</param>
		/// <returns></returns>
		public static bool AnimateWindow(Control ctrl, int dwTime, AnimateWindowFlags Flags)
		{
			return AnimateWindow(ctrl.Handle, dwTime,(int) Flags);
		}
```

```
            #endregion
        }
    }
```

4. Створіть обробники кнопок:

```
private void btnAW_BLEND_Click(object sender, System.EventArgs e)
{
        // Приховуємо вікно
        this.Hide();
        // Запускаємо анімацію.
        // Другий параметр в дужках - час анімації в мілісекундах.
        WinAPIClass.AnimateWindow(this, 3000,
            WinAPIClass.AnimateWindowFlags.AW_ACTIVATE|
            WinAPIClass.AnimateWindowFlags.AW_BLEND);
        // Відображаємо кнопки після анімації
        this.btnAW_BLEND.Invalidate();
        this.btnHOR_AW_SLIDE.Invalidate();
        this.btnCenter_AW_SLIDE.Invalidate();
}
private void btnHOR_AW_SLIDE_Click(object sender, System.EventArgs e)
{
        this.Hide();
        WinAPIClass.AnimateWindow(this, 3000,
        WinAPIClass.AnimateWindowFlags.AW_HOR_POSITIVE|
            WinAPIClass.AnimateWindowFlags.AW_SLIDE);
        this.btnAW_BLEND.Invalidate();
        this.btnHOR_AW_SLIDE.Invalidate();
        this.btnCenter_AW_SLIDE.Invalidate();
}
private void btnCenter_AW_SLIDE_Click(object sender, System.EventArgs e)
{
        this.Hide();
        WinAPIClass.AnimateWindow(this, 3000,
        WinAPIClass.AnimateWindowFlags.AW_CENTER|
            WinAPIClass.AnimateWindowFlags.AW_SLIDE);
        this.btnAW_BLEND.Invalidate();
        this.btnHOR_AW_SLIDE.Invalidate();
        this.btnCenter_AW_SLIDE.Invalidate();
}
```

5. Побудуйте і запустіть додаток. Протестуйте три види анімації.