Лабораторная работа 11

Взаимодействие управляемого и неуправляемого кода

Цель работы: Изучение способов использования возможностей объединения старого кода с кодом, управляемым средой CLR и получение навыков по работе со службами взаимодействия управляемого кода с неуправляемым.

Упражнение *1*
Использование *COM*-компонента для создания *PDF*-приложения

При создании приложений, использующих платформу **Microsoft .NET**, возникает задача применения в собственных проектах уже готовых библиотек кода, написанных на других языках. Сжатые сроки разработки и уже имеющиеся программные блоки не позволяют отказаться от готовых решений, поэтому их приходится использовать, встраивая в структуру собственных проектов.

Код, выполняющийся под управлением среды выполнения (в случае платформы **.NET** — среды **Common Language Runtime**), называется управляемым.

Код, запускаемый не под управлением среды, называется неуправляемым. Примером неуправляемого кода могут служить **COM** компоненты, **Microsoft ActiveX** интерфейсы и функции **API Win32**.

Microsoft .NET Framework позволяет взаимодействовать с **COM** компонентами, **COM+** - службами, внешними типами библиотек и разными службами операционной системы.

Платформа **.NET Framework** предлагает две службы взаимодействия управляемого кода с неуправляемым — **Platform Invoke** и **COM interoperability**, которые используют реальные возможности проекта.

1. Создайте новое Windows-приложение и назовите его PDF Reader.
2. Добавьте на форму элементы OpenFileDialog и MenuStrip.
3. Установите следующие свойства формы Form1:

Свойство	Значение
Text	Обозреватель документов в формате PDF
WindowState	Maximized

4. Свойство FileName элемента OpenFileDialog сделайте пустым.
5. Добавьте пункт меню верхнего уровня File с командами Open и Exit.
6. Выберите Tools | Choose Toolbox Items…
7. На вкладке COM Components выберите Adobe Acrobat (PDF) Reader (этот компонент появляется после установки программы Adobe Acrobat Reader) и нажмите OK. Убедитесь, что на панели Toolbox этот ЭУ
   действительно появился.
8. Перенесите Adobe Acrobat Reader на форму и установите свойству Dock значение Fill, а свойству (Name) – axAcroPDF1 (от имени объекта будет вызываться нужные нам методы, в принципе имя может быть любым).
9. Добавьте обработчик пункта меню Open:

```
private void openToolStripMenuItem_Click (object sender,
System.EventArgs e)
{
  openFileDialog1.Filter = "Файлы pdf|*.pdf";
  openFileDialog1.ShowDialog();
  axAcroPDF1.LoadFile(openFileDialog1.FileName);
}
```

10. Реализуйте обработчик события Click для пункта меню Exit.
11. Постройте и запустите приложение. При открытии документа в формате pdf происходит, по сути, встраивание в форму интерфейса программы Adobe Acrobat Reader.

Упражнение 2
Вызов функции API

Службы Platform Invoke позволяют управляемому коду запускать функции неуправляемого кода, которые находятся в файлах библиотек динамической компоновки (**DLL**).

Эти службы предоставляют механизмы обнаружения и запуска неуправляемых функций и преобразование типов данных входящих и исходящих аргументов функции.

Когда управляемый код запускает функцию неуправляемого кода, локализованную в **DLL-файле**, службы **Platform Invoke** находят этот **DLL** файл, загружают его в оперативную память и находят адрес функции в памяти. После этого службы передают входящие аргументы функции в стек, преобразовывают данные, которые необходимо перевести, эмулируют сборку мусора и передают управление по адресу неуправляемой функции в памяти.

Первым шагом в запуске неуправляемой функции является объявление функции. Функция должна быть статической (static) и внешней (extern). Далее следует импорт библиотеки, содержащей эту функцию. Импортировать библиотеку нужно, используя атрибут **DllImport**, который находится в пространстве имен **System.Runtime.InteropServices**.

1. Создайте новое приложение и назовите его WinAnim.
2. Расположите на форме три кнопки и установите следующие свойства формы и кнопок:

Свойство	**Значение**
Form1	
Text	Анимация формы
Button1	
Name	btnAW_BLEND
Location	30; 62
Size	232; 23
Text	Проявление
Button2	
Name	btnHOR_AW_SLIDE
Location	30; 118
Size	232; 23
Text	Горизонтальное появление
Button3	
Name	btnCenter_AW_SLIDE
Location	30; 182
Size	232; 23
Text	Появление из центра

3. Добавьте класс WinAPIClass:

```
using System;
using System.Runtime.InteropServices;
using System.Windows.Forms;
namespace AnimatedWindow
{
public class WinAPIClass
{
#region Анимация окна
/// <summary>
/// Тип анимации окна. Перечисление возвращает тип int
/// после приведения. Каждому элементу присвоено
/// свое значение типа 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>
/// Анимация окна.
/// </summary>
/// <param name="hwnd">Окно.</param>
/// <param name="dwTime">Время.</param>
/// <param name="dwFlags">Тип анимации. Если в
/// неуправляемом коде используется перечисление, то его
/// нужно конвертировать в тип данных int.
</param>
/// <returns></returns>
[DllImportAttribute("user32.dll",EntryPoint="AnimateWindow",Se
t
LastError=true)]
public static extern bool AnimateWindow(IntPtr hwnd,int
dwTime,int dwFlags);
/// <summary>
/// Анимация окна.
/// </summary>
/// <param name="ctrl">Окно.</param>
/// <param name="dwTime">Время.</param>
/// <param name="Flags">Флаги.</param>
/// <returns></returns>
public static bool AnimateWindow(Control ctrl,int dwTime,
AnimateWindowFlags Flags)
{
  return AnimateWindow(ctrl.Handle,dwTime,(int)Flags);
}
#endregion
}
}
```

4. Создайте обработчики кнопок:

```
private void btnAW_BLEND_Click(object sender,System.EventArgs e)
{
                // Скрываем окно
      this.Hide();
                // Запускаем анимацию.
                // Второй параметр в скобках – время анимации в
                // миллисекундах.
      WinAPIClass.AnimateWindow(this, 3000,
      WinAPIClass.AnimateWindowFlags.AW_ACTIVATE|
      WinAPIClass.AnimateWindowFlags.AW_BLEND);
                // Отображаем кнопки после анимации
      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. Постройте и запустите приложение. Протестируйте три вида анимации