Обзор языка программирования C#

(продолжение)

Гутников С.Е.

Библиотека Windows Forms

Обращаем внимание, что библиотека `Windows Forms` требует, чтобы метод `Main()` приложения был обязательно помечен атрибутом `[STAThread]`, иначе приложение может работать неверно.

Более подробно смотрите в MSDN описание критической ошибки CA2232 (поиск по слову: `MarkWindowsFormsEntryPointsWithStaThread`)

Библиотека Windows Forms

Вывод изображений

Библиотека содержит класс `Graphics` для вывода изображений и графических примитивов таких как линии, многоугольники, эллипсы и т.п.

Рассмотрим пример использования класса `Graphics` для вывода изображений в клиентской области формы со скроллингом.

Библиотека Windows Forms

```
using System;
using System.IO;
using System.Drawing;
using System.Drawing.Imaging;
using System.Windows.Forms;
namespace ExamGraph
{
    static class Program
    {   // Application object:
        [STAThread] //<-Required for WinForms
        static void Main(string[] args)
        {
            string fileimg = args.Length < 1 ?
                strDefImage : args[0];
```

Библиотека Windows Forms

```
Image img = null;
do try
{
    img = Image.FromFile(fileimg, true);
}
catch (FileNotFoundException e)
{
    MessageBox.Show(
      "Error: file not found " + e.Message,
        strAppTitle,
        MessageBoxButtons.OK,
        MessageBoxIcon.Error
    );
```

Библиотека Windows Forms

```
fileimg = ChooseFile(
        "Select image to show",
        "Jpeg images (*.jpg)|*.jpg", "jpg",
        "*.jpg", strDefImage );
    }
    while ( img == null );
    Application.EnableVisualStyles();
    Application.Run(new AppWindow(
        strAppTitle, img));
    } // Select file by std-dialog:
    static string ChooseFile(
        string title, string filter,
        string defExt, string fname,
        string defFile ) {
```

Библиотека Windows Forms

```
using (OpenFileDialog openFileDialog =
            new OpenFileDialog())
{
openFileDialog.Title = title;
openFileDialog.InitialDirectory = ".";
openFileDialog.Filter = filter;
openFileDialog.FilterIndex = 0;
openFileDialog.DefaultExt = defExt;
openFileDialog.FileName = fname;
openFileDialog.RestoreDirectory = true;
openFileDialog.Multiselect = false;
```

Библиотека Windows Forms

```
            if (openFileDialog.ShowDialog()
                    == DialogResult.OK)
            {
                return openFileDialog.FileName;
            }
        }
        return defFile;
    }
    static readonly string strAppTitle =
        "Graphics sample #3";
    static readonly string strDefImage =
        "imagex.jpg";
}
```

Библиотека Windows Forms

```
// class of application Window:
class AppWindow : Form
{
    Size NcSize;
    ImageWindow ImgWnd;
    public AppWindow(
        string title, Image Aimg)
    {
        // set application title,
        //   image, width, height:
        Text = title;
        // set required client area and size
        NcSize = new Size(
```

Библиотека Windows Forms

```
        Width - ClientSize.Width,
        Height - ClientSize.Height);
MaximumSize = new Size(
        NcSize.Width + Aimg.Width,
        NcSize.Height + Aimg.Height);
ClientSize = new Size(640, 480);
// ImgWnd
AutoScroll = true;
ImgWnd = new ImageWindow(Aimg);
ImgWnd.SetBounds(0, 0,
        Aimg.Width, Aimg.Height);
Controls.Add(ImgWnd);
// Center window:
```

Библиотека Windows Forms

```
            CenterToScreen();
        }
}

class ImageWindow : Control
{
        Image Img;
        public ImageWindow(Image Aimg)
        {
            // set application title,
            //   image, width, height:
            Img = Aimg;
            // set required client area and size
            ClientSize = new Size(
```

Библиотека Windows Forms

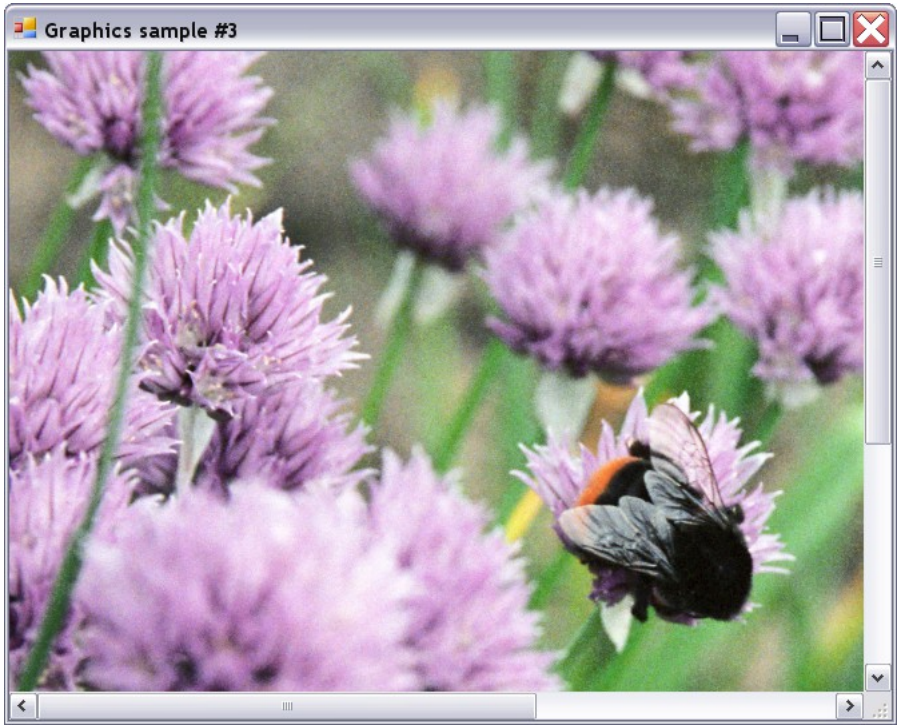
```
            Aimg.Width, Aimg.Height);
        MinimumSize = new Size(
            Aimg.Width, Aimg.Height);
        MaximumSize = MinimumSize;
        // setup event handler:
        Paint += ImageWindow_Paint;
    }
    private void ImageWindow_Paint(
        object sender, PaintEventArgs pe)
    {
        Graphics g = pe.Graphics;
        Draw(g);
    }
```

Библиотека Windows Forms

```
        private void Draw(Graphics g)
        {
            g.DrawImage(Img, 0, 0,
                Img.Width, Img.Height);
        }
    }
}
```

Библиотека Windows Forms

Результат работы:



Библиотека Windows Forms

Двумерная графика

Рассмотрим пример использования класса `Graphics` для рисования в клиентской области формы.

Будем использовать внеэкранный буфер – растровое изображение. Случайным образом в изображение рисуем отрезки при помощи `Graphics`, при необходимости перерисовки окна – выводим изображение.

Библиотека Windows Forms

```
using System;
using System.Drawing;
using System.Windows.Forms;
namespace ExamGraph2
{
    static class Program
    {
        private const int GraphX = 400;
        private const int GraphY = 400;
        [STAThread] //<-Required for WinForms
        static void Main(string[] args)
        {
```

Библиотека Windows Forms

```
try
{
    Application.EnableVisualStyles();
    Application.Run(
       new AppWindow(strAppTitle,
           GraphX, GraphY));
}
catch (Exception e)
{
    MessageBox.Show(
         "Error: " + e.Message,
         strAppTitle,
         MessageBoxButtons.OK,
         MessageBoxIcon.Error
```

Библиотека Windows Forms

```
                );
            }
        }
        static readonly string strAppTitle =
            "Graphics sample #4";
}
// class of application Window:
class AppWindow : Form
{
        Size NcSize;
        Random Rand;
        Bitmap Bmp;
        Graphics Buffer;
```

Библиотека Windows Forms

```
public AppWindow(string title,
                 int width, int height)
{
    Rand = new Random();
    // Buffer:
    Bmp = new Bitmap(width, height);
    Buffer = Graphics.FromImage(Bmp);
    Buffer.FillRectangle(
        new SolidBrush(Color.Black),
        0, 0, Bmp.Width, Bmp.Height);
    // set application title:
    Text = title;
    // set required client area and size
    NcSize = new Size(
```

Библиотека Windows Forms

```
        Width - ClientSize.Width,
        Height - ClientSize.Height);
    ClientSize = new Size(width, height);
    MinimumSize = new Size(
        NcSize.Width + width,
        NcSize.Height + height);
    MaximumSize = MinimumSize;
    BackColor = Color.Black;
    InitDraw();
    // setup event handler:
    Paint += AppWindow_Paint;
    Click += AppWindow_Click;
    // Center window:
    CenterToScreen();
```

Библиотека Windows Forms

```
}
private void AppWindow_Paint(
    object sender, PaintEventArgs pe)
{
    Graphics g = pe.Graphics;
    Draw(g);
}
private void AppWindow_Click(
    object sender, System.EventArgs e)
{
    DrawRandomLine(Buffer, true);
}
```

Библиотека Windows Forms

```
private void Draw(Graphics g)
{
    Buffer.Flush();
    g.DrawImage(Bmp, 0, 0,
        Bmp.Width, Bmp.Height);
}
private void InitDraw()
{
    for (int i = 0; i < 10; i++)
    {
        DrawRandomLine(Buffer, false);
    }
}
```

Библиотека Windows Forms

```
private void DrawRandomLine(
    Graphics g, bool bInvalidate)
{
    int x = Rand.Next(0, ClientSize.Width);
    int y = Rand.Next(0, ClientSize.Height);
    int x1 = Rand.Next(0, ClientSize.Width);
    int y1 =
        Rand.Next(0, ClientSize.Height);

    using (Pen p = new Pen(
                Color.FromArgb(
```

Библиотека Windows Forms

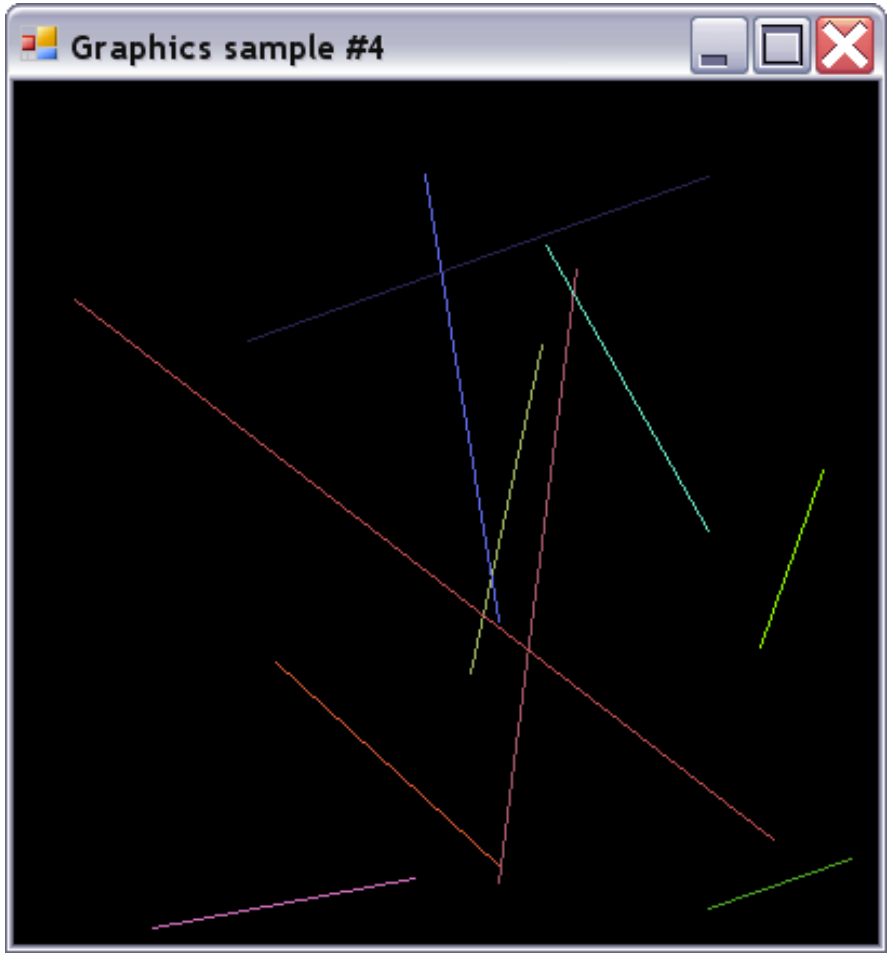
```
                    Rand.Next(0, 0xFF),
                    Rand.Next(0, 0xFF),
                    Rand.Next(0, 0xFF))))
{
    g.DrawLine(p, x, y, x1, y1);
}
if (bInvalidate)
{
    int n;
    if (x > x1)
    {
        n = x; x = x1; x1 = n;
    }
```

Библиотека Windows Forms

```
                if (y > y1)
                {
                    n = y; y = y1; y1 = n;
                }
                Invalidate(new Rectangle(
                    x, y, x1 - x, y1 - y));
            }
        }
    }
}
```

Библиотека Windows Forms

Результат работы:



Библиотека Windows Forms

Диалоговые окна.

Формы и диалоговые окна могут быть модальными и немодальными.

Модальная форма или диалоговое окно должно быть закрыто или спрятано перед продолжением работы с другими формами приложения.

Библиотека Windows Forms

Диалоговые окна, в которых отображаются важные сообщения, всегда должны быть модальными.

В приведенных выше примерах, при выводе сообщений об ошибках, `MessageBox` отображается в виде модального диалогового окна.

Библиотека Windows Forms

Немодальные формы или диалоговые окна позволяют переключаться между формами без необходимости закрывать начальную форму.

Пока отображается форма, пользователь может продолжать работу в любом другом месте приложения.

Библиотека Windows Forms

Чтобы отобразить форму как модальное диалоговое окно:

Вызовите метод `ShowDialog`. В методе `ShowDialog` существует необязательный аргумент `owner`, который можно использовать в форме для определения отношения "главный-подчиненный".

Библиотека Windows Forms

Например, когда в главной форме отображается диалоговое окно, в качестве `owner` можно передать значение `this`, чтобы определить, что главная форма является владельцем:

```
// Display frmAbout as a modal dialog
Form frmAbout = new Form();
frmAbout.ShowDialog(this);
```

Библиотека Windows Forms

При отображении модальной формы, код, следующий после метода `ShowDialog`, не выполняется до тех пор, пока не будет закрыто диалоговое окно.

Библиотека Windows Forms

Чтобы отобразить форму как немодальное диалоговое окно:

Вызовите метод `Show`.

```
//Display f as a modeless dialog
Form f= new Form();
f.Show();
```

Библиотека Windows Forms

При отображении немодальной формы код, следующий после метода `Show`, выполняется немедленно после отображения формы.

Пример для самостоятельного изучения, находится в файле `ExampleDlg.cs`

Задача

Разработайте приложение `Windows Forms` при помощи `MS Visual Studio C#`.

При реализации интерфейса следуйте рекомендациям стандарта CUI (Common User Interface).

Программа должна иметь главное меню, строку состояния и клиентскую область (область рисования).

Главное меню приложения обязательно должно содержать следующие команды (в формате Подменю/Команда):

Задача

- `File/Open` - выбор файла
- `File/Quit` - завершение приложения после подтверждения пользователя
- `Help/About` - вывод диалогового окна которое печатает условие задачи и информацию о разработчике (ФИО студента, группа, курс, факультет, ВУЗ)
- `(команды)` - команды необходимые для решения вашей задачи, например, `View/Fonts` для выбора шрифта, `View/Colors` для настройки цвета и т.п.

Задача

Постройте диаграмму опроса в сети отображающую процентное соотношение проголосовавших.

Входные данные хранятся в текстовом файле. Первая строка файла содержит наименование опроса. Каждая последующая строка файла содержит количество проголосовавших, затем, через пробел и до конца строки текст, соответствующий опции опроса.

Для выбора файла входных данных использовать стандартный диалог.

Задача

Пример файла входных данных:

Автомобиль какой фирмы вы используете?
800 Toyota
966 Mazda
1024 Ford
321 ВАЗ
166 Другой

<the end>