Огляд просторів імен GDI+

Клакович Л.М.

GDI (Graphic Device Interface)

- інтерфейс графічних пристроїв. Це підсистема Windows, призначена для виведення графічних зображень незалежно від пристрою
- GDI+ набір класів .NET Framework, які інкапсулюють GDI

  - абстрагування від графічного пристрою
  - **DC** (контекст пристрою) - базовий елемент абстракції
  - API виведення графіки на графічний пристрій
  - синхронізація сумісного використання різними WA графічних пристроїв

<u>призначення DC:</u>

- **інформування GDI про необхідний драйвер графічного пристрою та його налаштування**
- **задання атрибутів графічних об’єктів:**
  - **пера (pen)** для проведення ліній певного типу та товщини
  - **пензлика(brush)** для зафарбовування областей фоном певної структури і кольору
  - **палітри кольорів**
  - **підтримка областей** вирізки та оновлення

Найважливіші простори імен GDI+

GDI+ - набір класів .NET Framework, які інкапсулюють GDI (**Graphics Device Interface)**

- **System.Drawing** – основні типи для виведення графіки (робота зі шрифтами, перами, пензликами і ін.), а також тип **Graphics**.
- **System.Drawing.Drawing2D** – типи для виконання операцій з двовимірною графікою (градієнтна заливка, геометричні перетворення і ін.).
- **System.Drawing.Imaging** – типи, для роботи з графічними зображеннями (зміна палітри, отримання метаданих зображень, операції з метафайлами і ін.).
- **System.Drawing.Printing** – типи для виведення графіки на принтер і взаємодії з принтером.
- **System.Drawing.Text** – типи для роботи з системними шрифтами.

Простір імен System.Drawing

Bitmap	Інкапсулює bitmap, який містить піксельні дані для малюнку і його атрибути
Brush, SolidBrush TextureBrush	Визначає об’єкти, що використовуються для заповнення вмісту фігур. Одноколірні пензлики. Пензлики із заповненням малюнком
Brushes	Пензлики для всіх стандартних кольорів.
Font, FontFamily	
Graphics	Інкапсулює об’єкт для виведення графіки.
Icon	
Image	Абстрактний клас, що надає функціональності для класів **Bitmap** і **Metafile**.
ImageAnimator	
Pen, Pens	Визначає об’єкт, що використовується для малювання ліній.
Region	Внутрішня область геометричної фігури
SystemBrushes, SystemColors SystemFonts, SystemIcons SystemPens	Об’єкти системних пензликів, кольорів, фонтів, іконок, пер

Структури System.Drawing

CharacterRange	Визначає ранг символів.
Color	Колір ARGB
Point, PointF	Пара цілих (float): X- та Y-координати точки на площині. **Оператори** (+, -, ==,!=), **Offset**() зміщення точки відносно вихідної позиції.
Rectangle, RectangleF	Зберігає 4 цілі (float) значення, що представляють розміщення та розміри прямокутника. **Властивості**: X, Y, Width, Height, **Оператори** (= =, !=), **Inflate(), Intersect(), Union()** – створення нових прямокутників шляхом збільшення, розділення та об'єднання, **Contains**(Point or Rect);
Size, SizeF	Зберігає пару цілих (float), як правило ширина та висота прямокутника

Клас Graphics

Клас Graphics надає функції для малювання на таких трьох основних типах поверхонь:

- Форма і керуючі елементи
- Сторінки, які відправляються на друк,
- Растрові зображення в пам”яті

Виведення графіки здійснюється:

1) Перевизначивши метод **OnPaint**() або перехопивши подію **Paint** з визначенням свого методу малювання. Об”єкт **Graphics** отримується з властивості аргументу **PaintEventArgs**.

2) В будь-якому іншому методі. Об”єкт **Graphics** отримується з методу **CreateGraphics()** даної форми, або іншими методами: **Graphics.FromHwnd(), FromHdc(), FromImage().**

```
protected override void OnPaint(PaintEventArgs e)
{
    Graphics g = e.Graphics;
    g.DrawRectangle(new Pen(Color.Red, 3), 0, 0, 200, 100);
}
```

```
protected void Form1_Click(object sender, EventArgs e)
{
        Graphics g = this.CreateGraphics();
        g.DrawRectangle(new Pen(Color.Red, 3), 0, 0, 200, 100);
        g.Dispose();
}
```

Обов”язково!

```
public void FromHwndHwnd()
{
        IntPtr hwnd = this.Handle;
        Graphics g = Graphics.FromHwnd(hwnd);
        g.DrawRectangle(new Pen(Color.Red, 3), 0, 0, 200, 100);
        g.Dispose();
}
```

Можливості класу Graphics

- **Аналог DC (контексту пристрою)**
- **Виведення тексту, геометричних фігур і ін. Методами класу:**

  `DrawBeziers(), DrawCurve() DrawEllipse(), DrawImage() DrawLine(), DrawPath(), DrawPie(), DrawPolygon() DrawRectangle(), DrawString() FillClosedCurve() FillEllipse() FillPath() FillPie() FillPolygon() FillRectangle() FillRectangles() FillRegion() і ін.`
- **Властивості для налаштовування параметрів графічних об”єктів:**

  Clip, ClipBounds, VisibleClipBounds, IsVisibleClipEmpty, IsClipEmpty – налаштовування параметрів відсічення об’єктів SmoothingMode, PixelOffsetMode, TextRenderingHint – задавати плавність переходів для геометричних об’єктів і ін.

Система координат

- **За замовчуванням**:
  - одиниці вимірювання – **пікселі**,
  - початок системи координат – верхній лівий куток форми (елементу керування, рисунку і ін.) координати (0, 0)
- **Зміна одиниць вимірювання** через властивість **PageUnit** класу **Graphics**, яка повертає одне із значень з

  перелічення **GraphicsUnit:**

Display	1/100 дюйма
Document	1/300 дюйма
Inch	дюйм
Millimeter	міліметр
Pixel	піксель
Point	1/72 дюйма

• **Зміна початку системи координат** здійснюється методом **TranslateTransform**() з класу **Graphics**.

```
e.Graphics.PageUnit=GraphicsUnit.Inch;
e.Graphics.TranslateTransform(100,100);
```

Колір (структура Color)

- Структура **Color** класу **Graphics** задає колір в системі **ARGB** (alpha-red-green-blue, альфа-канал, що відповідає за прозорість).
- Отримання об’єкту типу **Color:**

  1) через статичні властивості структури

  ```
  Color c=Color.PapayaWhip;
  ```

  2) з допомогою методів **FromArgb**(), **FromName**(),

  FromKnownColor()

  3) з діалогу **ColorDialog** через властивість **ColorDialog.Color**

- Система **HSB** (Hue-Saturation-Brightness, відтінок-насиченість-яскравість). Методи, які повертають ці характеристики для об’єкту **Color** : **GetBrightness(), GetHue(), GetSaturation().**

Робота зі шрифтами

- Клас **System.Drawing.Font**. Утворення фонту конструктором:

```
Font f=new Font("Times New Roman", 12);
Font f2=new Font("WingDings", 50, FontStyle.Bold|
                    FontStyle.Underline);
```

- **Клас FontFamily** – сімейство шрифтів.
- Простір імен **System.Drawing.Text** визначає типи для отримання інформації про системні шрифти та властивості, що стосуються вирівнювання тексту, відстані між рядками, якості виведення тексту і ін.
- Отримання шрифту з діалогу **FontDialog**

Простір імен System.Drawing.Drawing2D

- створення спеціальних наконечників для пер (**pen caps**),
- пензлики з текстурним заповненням,
- векторні маніпуляції з графічними об’єктами.

Основні класи:

AdjustableArrowCap **CustomLineCap**	Визначають наконечники для пер
Blend **ColorBlend**	Використовується для змішування кольорів, разом з LinearGradientBrush
GraphicsPath **GraphicsPathIterator**	Представляє набір зв’язаних ліній (прямих і кривих).
HatchBrush **LinearGradientBrush** **PathGradientBrush**	Екзотичні типи пензликів

Робота з перами (Pen)

- **Пера використовуються** для малювання ліній, контурів фігур і ін. графічних об’єктів класу **Graphics.** Клас **Pen.**
- **Пера характеризується**
  - кольором (властивість **Color**),
  - товщиною (**Width**),
  - стилем ліній (**PenStyle**, **DashStyle**),
  - наконечником (**CustomStartCap**, **CustomEndCap**, **DashCap**).
- **Pens** – містить колекцію пер, товщиною 1

```
Pen pen1=new Pen(Color.Blue, 20);
pen1.DashStyle=DashStyle.DashDotDot;
pen1.EndCap = Drawing2D.LineCap.ArrowAnchor;

Pen pen2=Pens.Firebrick;
```

Робота з пензликами (Brush)

- **Пензлики призначені** для замальовування простору між лініями:
  - виведення текстових рядків **Graphics.DrawString**()
  - заповнення вмісту фігур – **FillXXX**()
  - створення об’єкта **Pen**.
- **Пензлики характеризуються** кольором, текстурою або зображенням.
- **Brush** – абстрактний клас, похідні класів: **SolidBrush**, **HatchBrush**, **LinearGradientBrush** і ін.
- **Типи-колекції** **Brushes** і **System.Brushes**.

```
SolidBrush bluebrush=new SolidBrush(Color.Blue);
SolidBrush blackbrush=(SolidBrush)Brushes.Black;
```

```
HatchBrush br= new HatchBrush (HatchStyle.Cross,
                               Color.White, Color.Black);
```

- **Текстурні пензлики TextureBrush.** Замальовують область растровим зображенням **(*.bmp, *.gif, *.jpg**)

```
Image im=new Bitmap(“image1.bmp”);
TextureBrush tb= new  TextureBrush(im);
```

- **Градієнтний пензлик LinearGradientBrush.** Плавне змішування кольорів для отримання градієнтного переходу.

Виведення зображень

- Простір імен **System.Drawing.Imaging** (для проведення складних перетворень зображень).
  - робота з метафайлами
  - класи кодування/декодування (підтримка будь-якого графічного формату)
- Клас **System.Drawing.Image (**абстрактний) та похідні **Bitmap** і **Metafile**.
- Отримання растрових зображень:
  - з файлу (**FromFile**());
  - з потоку (**FromStream**());
  - з вже існуючого **Image**;
  - як порожній рисунок, в якому буде здійснено малювання.
- метод **DrawImage**() класу **Graphics -** виведення рисунку

```
Image im= new Bitmap("im1.bmp");
Graphics g=e.Graphics;
g. DrawImage(im, 10, 10, 90,90);
```

- елемент керування **PictureBox**

```
PictureBox pb= new PictureBox();
pb.SizeMode= PictureBoxSizeMode.StretchImage;
pb.Image=new Bitmap(“im1.bmp”);
Controls.Add(pb);
```