Изчертаване на графични примитиви

1. Компютърна графика

А) Растерна графика

- Растерното графично изображение(bitmap) е съставено от цветните стойности на всеки отделен пиксел.
- Растерното изображение е обемисто и след като веднъж е изработено не може лесно да се промени.
- При опит да се увеличи или намали, качеството му силно се влошава.
- Растерни изображения се създават и обработват, например, с познатата ни от уроците по ИТ програма Paint.

Б) Векторна графика

- Векторното изображение е съставено от отделни графични елементи – части от прави и криви линии, различни фигури и т.н., всеки от които е представен с математическо описание.
- Представянето е с малък обем.
- Когато векторното изображение трябва да се представи на екрана – математическото описание се трансформира в множество от цветни точки, които изменят съответните пиксели на екрана.
- Векторните изображения се увеличават или намаляват без да се губи качеството им.

2. Създаване на приложение с графика

Използва се класът Graphics, дефиниран в пространството от имена System.Drawing;

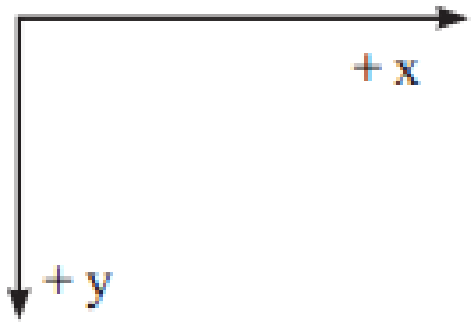
В класа Form1 се записва специален защитен и предефиниран метод за обработка на събитието OnPaint с един аргумент– обект e от класа PaintEventArgs:

```
protected override void OnPaint(PaintEventArgs e)
    {  Graphics g = e.Graphics;  ...}
```

3. Чертожно поле

- Правоъгълна част от екранния растер с r реда и c стълба, която заема вътрешността на екранната форма;

- Текущите размери на чертожното поле на формата се съдържат в свойствата ClientSize.Width– ширина и ClientSize.Height– височина на чертожното поле.



- Екранните точки са пикселите и координатите им са само цели положителни числа.

- Пикселът в горния ляв ъгъл на чертожното поле е началото на координатната система – координати(0,0).
- x-координати са целите числа от 0 до c – 1 и растат отляво надясно.
- y-координатите са целите числа от 0 до r – 1 и растат отгоре надолу (тъй като обновяването на изображението на монитора става с обхождане на пикселите отгоре надолу и отляво надясно).

4. Създаване на писалка

Инструментът, който избираме, се характеризира основно с цвета и дебелината на следата, която оставя в чертожното поле. Тези и други характеристики, са обобщени в класа Pen (писалка).

Затова, преди да започнем изчертаването, трябва да създадем обект от този клас:

Pen p = new Pen(<цвят>, <дебелина>);

- Цветът е обект от класа Color със140 свойства, всяко от които е някакъв цвят и се избира от списъчна кутия, която се отваря след написване на знака точка след името на класа.
- Дебелината на линията е число от тип float, и може да считеме, че закръглено до цяло число, то означава брой пиксели.
- Всяка стойност на дебелината, по-малка от 1, се приема за дебелина 1.
- Операторът Pen p = new Pen(Color.Red,-3);, например, създава писалка с червен цвят и дебелина 1 пиксел;

- Ако искаме да създадем писалка с цвят, който не е измежду включените като свойства на класа Color, трябва да използваме статичния метод Color.FromArgb(<червен>, <зелен>, <син>), който задава цвят, базиран на модела RGB (red-green-blue), т.е. чрез интензитетите (цели числа от0 до255) на трите образуващи цвята– червено, зелено и синьо:

```
Pen p = new Pen(Color.FromArgb(49,226,29),2);
```

- Трите интензитета на избрания цвят могат да се вземат от цветовата палитра на програмата Paint, като се отвори палитрата и се избере интересуващият ни цвят.

5. Фонов цвят на чертожното поле

- Фоновият цвят на чертожното поле се задава като стойност на свойството BackColor.
- Друг начин за смяна на фоновия цвят на полето е с метода Clear на класа Graphics.
- Той има един параметър, който е обект от класа Color:

```
g.Clear(Color.Red); или
g.Clear(Color.FromArgb(20,140,60));
```

6. Работа с компютър

Да се напише програма, която отваря екранна форма, трансформира я в чертожно поле и с опе-ратор за цикъл променя многократно цвета на фона с генериран по случаен начин цвят, при което се получава интересен ефект:

```
protected override void OnPaint(PaintEventArgs e)
{  Graphics g = e.Graphics;
   int a = 255,freq=10;
   Random c = new Random();
   for (int i = 1; i <= 40; i++)
   {  g.Clear(Color.FromArgb(c.Next()%a,c.Next()%a,c.Next()%a));
      System.Threading.Thread.Sleep(1000 / 10);
   }
}
```

- Такъв прием се използва при създаването на анимационни ефекти, например.
- Методът System.Threading.Thread.Sleep(<време>), който ще използваме при създаването на компютърна графика забавя изпълнението на следващия оператор за времето в милисекунди, зададено като параметър.
- Всяко извикване Next() на обекта с от класа Random пък връща едно случайно число, което превръщаме в интензитет(от 0 до 255) като вземем остатъка му по модул255.