**Dezvoltarea aplicațiilor pe platforma .NET**

Pelle Remus-Nicolae

SDI, anul II

**Grafică 2D**

**Introducere**

Pentru a putea aplica elemente de grafică 2D în .Net, cel mai simplu proiect pe care îl putem crea este *Windows Forms App (.Net Framework).* După creare, vom vedea designer-ul formularului, unde putem adăuga diferite componente folosind toolbox-ul. Haideți să adăugăm un PictureBox și un Button

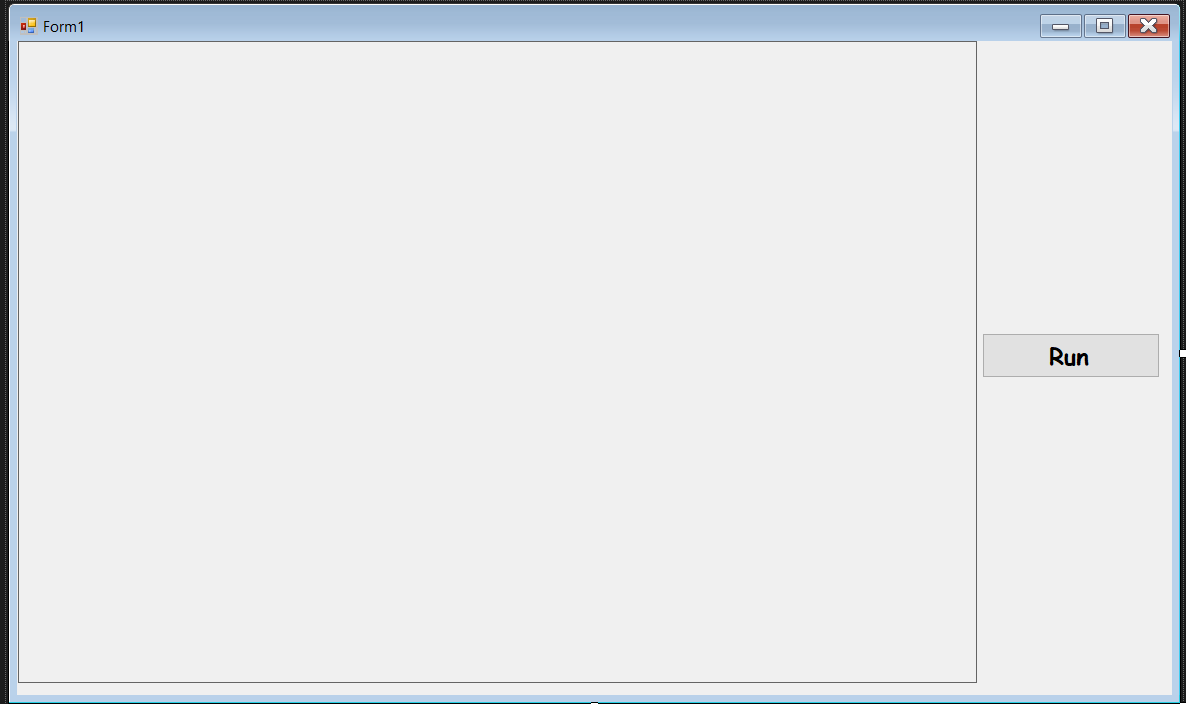


Figura 1: Formularul inițial

Pentru a scrie cod în cele mai comune evenimente ale fiecărei componente, putem da dublu click pe aceasta, și acest cod se va genera automat. Pentru formular în sine, se creează metoda *Form1\_Load*, iar pentru buton se creează metoda *button1\_Click*. Cele mai folosite obiecte pentru desenarea în pictureBox sunt un obiect *Graphics* și un obiect *Bitmap*. Obiectul bitmap va fi folosit ca și imaginea pictureBox-ului, iar obiectul graphics pentru a desena în acesta. Le vom inițializa când se încarcă formularul.

|  |
| --- |
| Graphics graphics;  Bitmap bitmap;  private void Form1\_Load(object sender, EventArgs e)  {  bitmap = new Bitmap(pictureBox1.Width, pictureBox1.Height);  graphics = Graphics.FromImage(bitmap);  } |

Pentru a desena ceva în pictureBox, întâi trebuie să înțelegem care sunt regulile de desenare.

**Sistemul de coordonate**

În grafica 2D a calculatorului, originea (punctul de coordonate 0, 0) este colțul din stânga sus. Un punct de coordonate (x, y) se află la x pixeli în dreapta acestuia, respectiv la y pixeli în jos față de acesta. Deci se comportă ca și cadranul IV din sistemul de coordonate matematic.

Putem desena linii între diferite puncte folosind metoda *DrawLine*. Dacă nu specificăm coordonatele unui punct, acesta va fi inițializat în origine. Cum am specificat obiectului graphics să deseneze pe obiectul bitmap, nu vom vedea niciun rezultat. Așa că la final, trebuie să-i spunem la pictureBox1 că imaginea sa este bitmap.

Vom desena aceste linii la apăsarea butonului.

|  |
| --- |
| Random random = new Random();  private void button1\_Click(object sender, EventArgs e)  {  graphics.DrawLine(Pens.Black, new Point(), new Point(100, 100));  graphics.DrawLine(Pens.Black, new Point(50, 60), new Point(150, 80));  graphics.DrawLine(Pens.Black,  new Point(random.Next(pictureBox1.Width), random.Next(pictureBox1.Height)),  new Point(random.Next(pictureBox1.Width), random.Next(pictureBox1.Height)));  pictureBox1.Image = bitmap;  } |



Figura 2: Desenarea unor linii

Putem schimba culoarea și grosimea creionului folosit pentru desenare folosind un obiect din clasa *Pen*. Dar acum, observăm că dacă apăsăm de mai multe ori pe buton, vom avea mai multe linii aleatorii.

|  |
| --- |
| private void button1\_Click(object sender, EventArgs e)  {  graphics.DrawLine(new Pen(Color.Black, 2), new Point(), new Point(100, 100));  graphics.DrawLine(new Pen(Color.Blue, 3), new Point(50, 60), new Point(150, 80));  graphics.DrawLine(  new Pen(Color.FromArgb(random.Next(255), random.Next(255), random.Next(255)), random.Next(10)),  new Point(random.Next(pictureBox1.Width), random.Next(pictureBox1.Height)),  new Point(random.Next(pictureBox1.Width), random.Next(pictureBox1.Height)));  pictureBox1.Image = bitmap;  } |

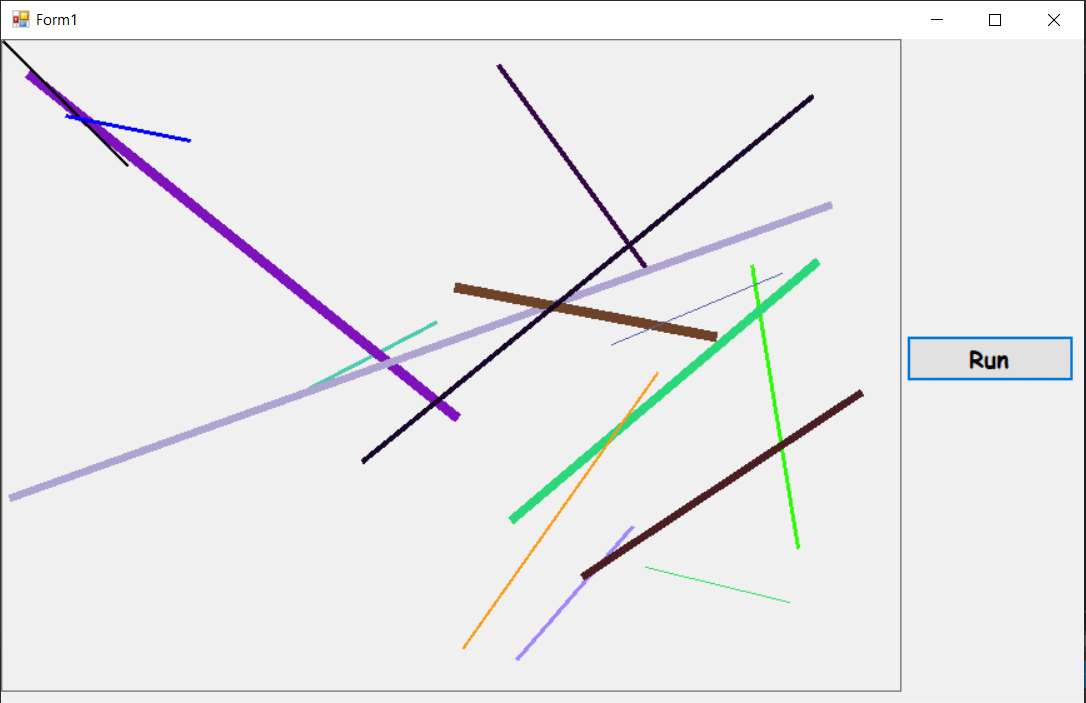


Figura 3: Linii de culori și grosimi aleatorii, dar avem mai multe decât ne așteptam

Soluția este simplă: înainte să desenăm liniile, folosim comanda *Clear*, care „șterge” conținutul deja desenat de graphics

|  |
| --- |
| graphics.Clear(Color.White); |

Dacă am dori să lucrăm cu un grafic matematic, astfel având nevoie de sistemele de coordonate în centrul ecranului, iar y-ul să mearga in sus, și nu în jos? Cea mai simplă sooluție ar fi să nu luăm punctele direct din clasa Point, ci să apelăm o functie creată de noi care mapează punctul dorit la noul punct, considerând centrul pictureBox-ului ca fiind originea. Putem și crea o funcție ce deseneaza axele de coordonate în sine.

|  |
| --- |
| // ...  Point center = new Point();  private void Form1\_Load(object sender, EventArgs e)  {  // ...  center = new Point(pictureBox1.Width / 2, pictureBox1.Height / 2);  }  private void button1\_Click(object sender, EventArgs e)  {  graphics.Clear(Color.White);  DrawXoYAxes();  graphics.DrawLine(new Pen(Color.Black, 2), GetMappedPoint(), GetMappedPoint(100, 100));  // ...  }  private void DrawXoYAxes()  {  graphics.DrawLine(new Pen(Color.Black, 2), GetMappedPoint(-center.X, 0), GetMappedPoint(center.X, 0));  graphics.DrawLine(new Pen(Color.Black, 2), GetMappedPoint(0, -center.Y), GetMappedPoint(0, center.Y));  }  private Point GetMappedPoint(int x, int y) => new Point(center.X + x, center.Y - y);  private Point GetMappedPoint() => GetMappedPoint(0, 0); |

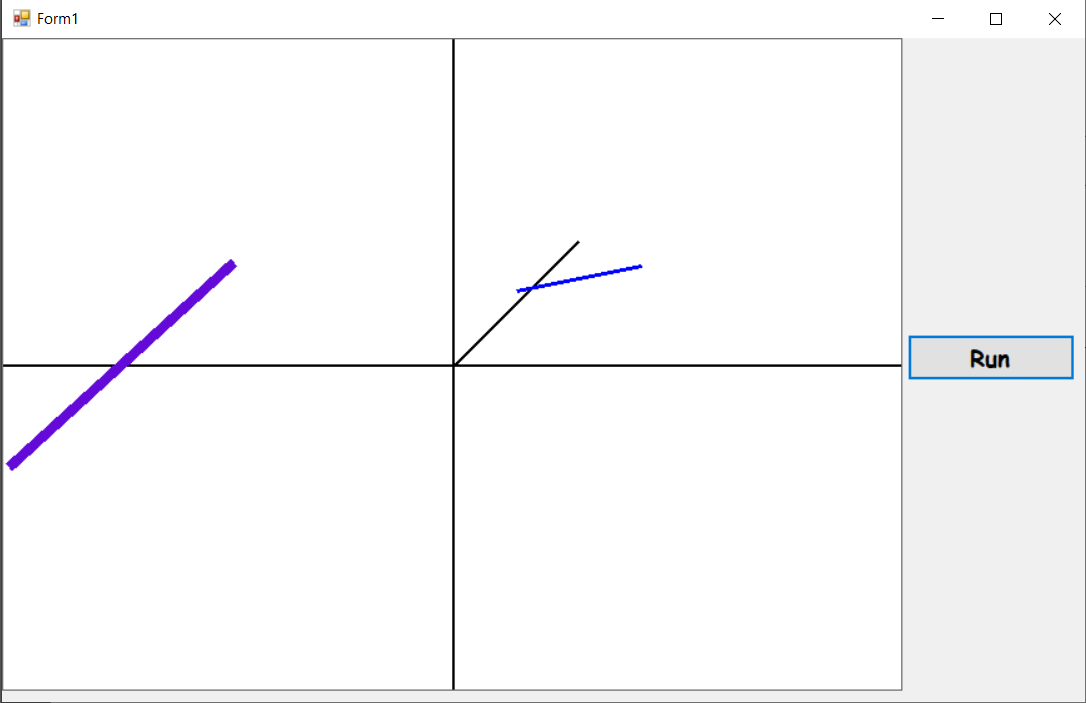


Figura 4: Sistem de coordonate propriu

Parcă este puțin greu să ne dăm seama că aceste linii reprezintă axele de coordonate. Pentru un simț familiar, am putea „desena” scrisul X, O si Y în locurile potrivite. De asemenea, putem desena aceste linii ca săgeți.

|  |
| --- |
| private void DrawXoYAxes()  {  Pen pen = new Pen(Color.Black, 2);  Brush brush = new SolidBrush(Color.Black);  Font font = new Font("Arial", 12, FontStyle.Bold);  graphics.DrawLine(pen, GetMappedPoint(-center.X, 0), GetMappedPoint(center.X, 0));  graphics.DrawLine(pen, GetMappedPoint(0, -center.Y), GetMappedPoint(0, center.Y));  graphics.DrawLine(pen, GetMappedPoint(center.X-20,10), GetMappedPoint(center.X,0));  graphics.DrawLine(pen, GetMappedPoint(center.X-20,-10),GetMappedPoint(center.X,0));  graphics.DrawLine(pen, GetMappedPoint(10,center.Y-20), GetMappedPoint(0,center.Y));  graphics.DrawLine(pen, GetMappedPoint(-10,center.Y-20),GetMappedPoint(0,center.Y));  graphics.DrawString("X", font, brush, GetMappedPoint(center.X - 20, -10));  graphics.DrawString("O", font, brush, center);  graphics.DrawString("Y", font, brush, GetMappedPoint(10, center.Y));  } |

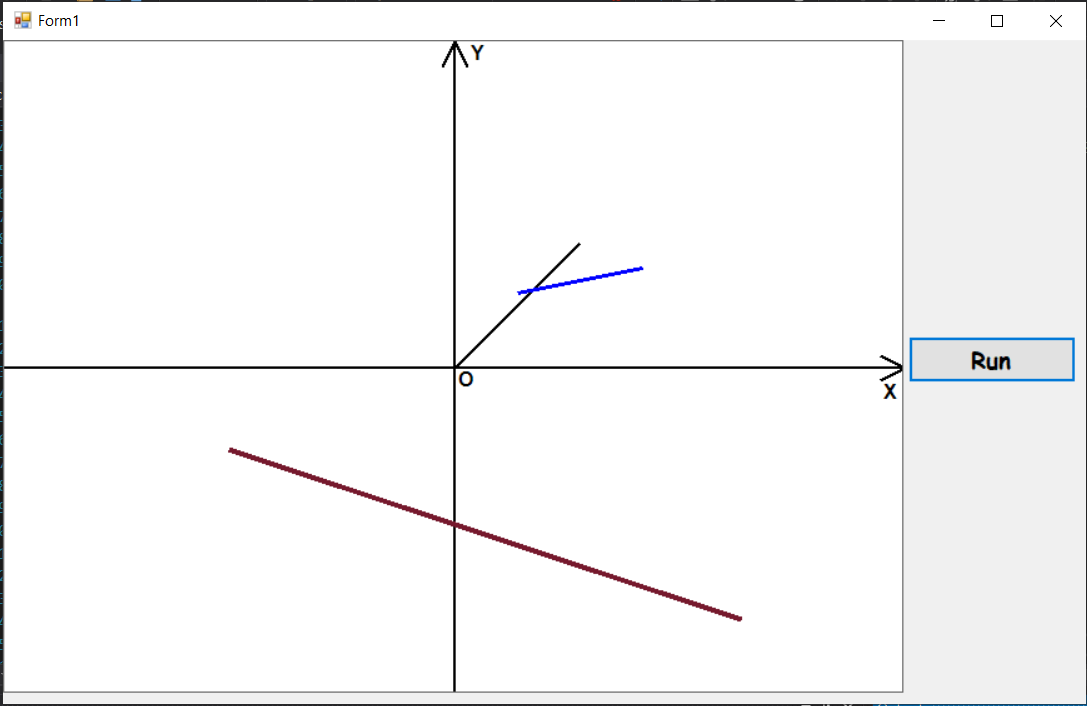


Figura 5: Axele de coordonate desenate într-un mod mai familiar, ușor de recunoscut

**Alte posibile forme de desenat folosind Graphics**

Pe lângă linii și text, sunt multe alte forme ce se pot desena:Elipsă, Dreptunghi, Poligon, etc. Acestea se pot și umple dacă așa dorim.

|  |
| --- |
| private void button1\_Click(object sender, EventArgs e)  {  // ...  graphics.FillRectangle(new SolidBrush(Color.ForestGreen), 0,0, center.X, center.Y);  Point[] points = new Point[] { GetRandomMappedPoint(), GetRandomMappedPoint(), GetRandomMappedPoint(), GetRandomMappedPoint(), GetRandomMappedPoint() };  graphics.FillPolygon(  new SolidBrush(Color.FromArgb(random.Next(255), random.Next(255), random.Next(255))),  points);  graphics.DrawEllipse(Pens.Red, center.X-(center.Y/2),center.Y/2,center.Y,center.Y);  pictureBox1.Image = bitmap;  }  // ...  private Point GetRandomMappedPoint() =>  GetMappedPoint(random.Next(-center.X, center.X), random.Next(-center.Y, center.Y)); |

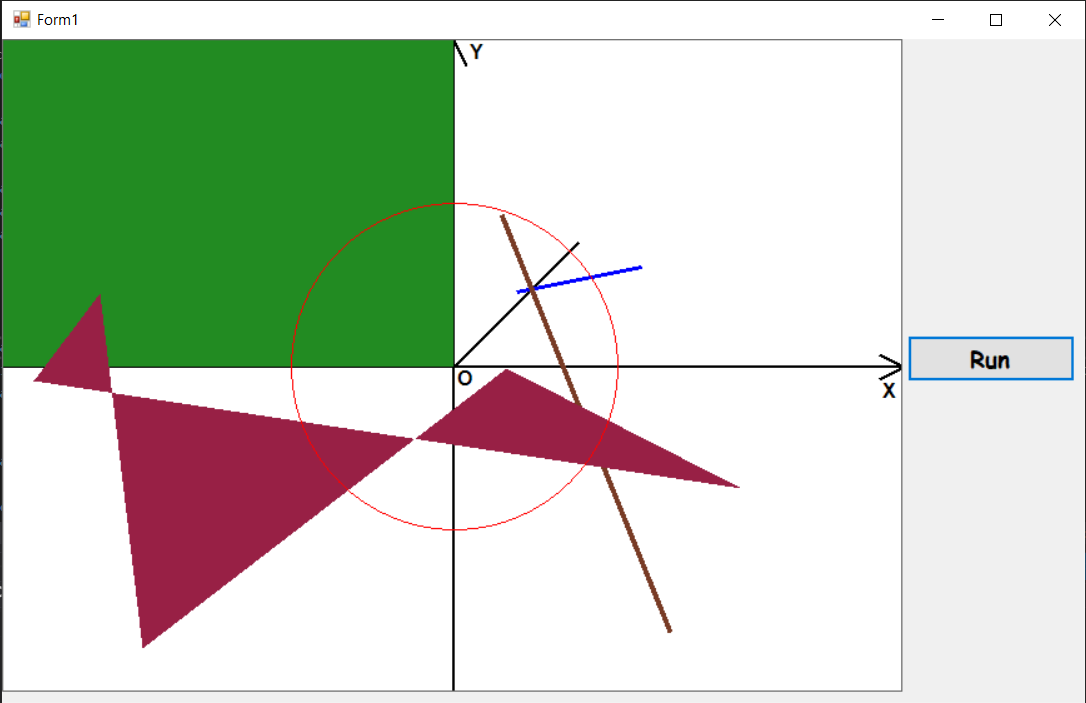


Figura 6: Diferite forme desenate folosind Graphics

**Aplicare practică a graficii 2D: Jocuri 2D**

Dupa teorie, este bine să știm și cum sau la ce să punem în practică. Crearea unui joc 2D mai simplu este un mod bun de a învăța mai bine grafica 2D și alte lucruri ce se pot folosi împreună cu aceasta din interiorul aplicațiilor WFA.