Laborator 01

Proiect 01

Să se implementeze o ierarhie de clase `Shape` (figuri geometrice: primitive grafice + picturi) împreună cu o interfaţă grafică cu utilizatorul pentru desenarea figurilor geometrice.

Se vor oferi următoarele facilităţi:

- desenare;
- redimensionare;
- mutare;
- grupare;
- ştergere;
- salvare pe disk (informaţiile despre figurile geometrice curente vor fi salvate pe disk în format `Json` pentru o recuperare ulterioară);

Observaţii:

1. Aplicaţia va fi proiectată astfel încât la adăugarea de noi tipuri de primitive grafice, modificările codului scris deja să fie minime (aplicaţie extensibilă).
2. Pentru aceasta vom abstractiza noţiunea de primitivă grafică / pictura prin intermediul unei clase abstracte `Shape`.
3. Primitivele grafice (linie, cerc, dreptunghi, triunghi) vor fi clase derivate din `Shape` - cel puţin 4 primitive grafice diferite.
4. O pictură va fi o compunere (colecţie) de obiecte derivate din `Shape` - derivată ea însăşi din `Shape` (şi o pictură este - comportamental - o figură geometrică).
5. Interfaţa grafică (`UI`) va interacţiona cu obiectele ierarhiei doar prin intermediul unui container de obiecte grafice `GraphicTool`. Acesta conţine o colecţie de obiecte `Shape` (fără să cunoască detalii despre obiectele grafice concrete).

```
public abstract class Shape {
  public string Name { get; set; }
  public Point2D Origin { get; set; }
  // Utilizat in metodele de desenare / redimensionare / mutare...
  public Graphics GraphicContext { get; set; }
  // Metode abstracte comune (comportamentele comune ierarhiei de clase)
  public abstract void Draw();
  public abstract void Resize(double factor);
  public abstract void MoveTo(Point2D newOrigin);
}

public class Point2D {
  int X { get; set; }
  int Y { get; set; }
}
```

```
public class Line: Shape {
  // Atribute specifice si alte metode utile
  public override void Draw() {
    throw new NotImplementedException();
  }
  public override void MoveTo(Point2D newOrigin) {
    throw new NotImplementedException();
  }
  public override void Resize(double factor) {
    throw new NotImplementedException();
  }
}

public class Circle: Shape {
  // Atribute specifice si alte metode
  public override void Draw() {
    throw new NotImplementedException();
  }
  public override void MoveTo(Point2D newOrigin) {
    throw new NotImplementedException();
  }
  public override void Resize(double factor) {
    throw new NotImplementedException();
  }
}

public class Picture: Shape {
  public List<Shape> Components { get; private set; }
  public Picture() {
    Components = new List<Shape> ();
  }
  public override void Draw() {
    foreach(var shape in Components)
      shape.Draw(); // Apel polimorfic
  }
  public override void MoveTo(Point2D newOrigin) {
    throw new NotImplementedException();
  }
  public override void Resize(double factor) {
    throw new NotImplementedException();
  }
  public void Add(Shape s) {
    Components.Add(s);
  }
  public void Remove(Shape s) {
    Components.Remove(s);
  }
}

// Container (colectie) de obiecte grafice (primitive sau picture)
// Aceasta clasa va fi utilizata intr-o interfata grafica cu utilizatorul (UI):
proiect separat de tip WindowForms
public class GraphicTool {
  public List<Shape> Shapes { get; private set;
  // Adaugare, eliminare, salvare pe disk, incarcare de pe disk, deseneaza
  public void DrawAll() {
    foreach(var shape in Shapes)
```

```
      shape.Draw();
  }
  public void GroupShapes(List<string> shapes) {
    // Se grupeaza figurile cu numele din lista shapes astfel:
    // - sunt cautate in lista 'Shapes' figurile cu numele din parametrul
'shapes'
    // - se creaza un nou obiect 'Picture' (cu un nume unic)
    // - toate figurile gasite la pasul 2 sunt adaugate la Picture si eliminate
din lista 'Shapes'
  }
  public void UngroupShapes(string pictureGroupName) {
    // Se cauta in 'Shapes' obiectul (de tip Picture) cu numele pictureGroupName
    // Daca exista, va fi eliminat din lista, iar figurile care il compun vor fi
adaugate separat in 'Shapes'
  }
  public void SaveOnDisk() {
    // In format json, colectia Shapes este salvata pe disk
  }
  public void LoadFromDisk() { }
  public void Add(Shape s) { }
  public void Remove(Shape s) { }
}
```

Timp de lucru: 2 saptamani.