**Section07 Adapter Design Pattern**

**Lesson01 What is the Adapter Design Pattern**

**Notes: -**

**1-this is the first pattern for the Structural pattern**

**2-Adapter design pattern is talk about getting the interface you want from the interface you have**

**(A construct which adapt an existing interface X to conform to the required interface Y)**

**Example: -**

**1-some systems require interface A and other systems require interface B and you want to interact between these systems, so you need to use Adapter design pattern**

**Lesson02 Vector / Raster Demo**

**Notes: -**

**1-the bad side of the Adapter design pattern is that it will generate a lot of temporary information**

**2-in this example adapter is just class inherit from Collection<Point> which is used to convert list of lines that represent shape into list of points**

**Example:-**

**using System.Collections.ObjectModel;**

**namespace AdapterSimplePro{**

**//the base class of multiple shapes**

**public class VectorObject : Collection<Line>{}}**

**namespace AdapterSimplePro{**

**public class VectorRectangle : VectorObject{**

**public VectorRectangle(int x, int y, int width, int height){**

**Add(new Line(new Point(x, y), new Point(x + width, y)));**

**Add(new Line(new Point(x+width, y), new Point(x + width, y+height)));**

**Add(new Line(new Point(x, y), new Point(x , y + height)));**

**Add(new Line(new Point(x, y), new Point(x + width, y + height)));}}}**

**namespace AdapterSimplePro{**

**public class Point{**

**public int X,Y;**

**public Point(int x, int y){X = x;Y = y;}**

**public override string ToString(){**

**return $"{nameof(X)}: {X}, {nameof(Y)}: {Y}";}}}**

**namespace AdapterSimplePro{**

**//Line contains two proeprty Point of Start , End**

**public class Line{**

**public Point Start;**

**public Point End;**

**public Line(Point start, Point end){**

**this.Start = start;**

**this.End = end;}}}**

**using System;**

**using System.Collections.ObjectModel;**

**using static System.Console;**

**namespace AdapterSimplePro{**

**//we using LineToPointAdapter as adapter to convert line into collection of points**

**public class LineToPointAdapter : Collection<Point>{**

**private static int count = 0;**

**public LineToPointAdapter(Line line){**

**WriteLine($"{++count}: Generating points for line [{line.Start.X},{line.Start.Y}]-[{line.End.X},{line.End.Y}] (no caching)");**

**int left = Math.Min(line.Start.X, line.End.X);**

**int right = Math.Max(line.Start.X, line.End.X);**

**int top = Math.Min(line.Start.Y, line.End.Y);**

**int bottom = Math.Max(line.Start.Y, line.End.Y);**

**int dx = right - left;**

**int dy = line.End.Y - line.Start.Y;**

**if (dx == 0){**

**for (int y = top; y <= bottom; ++y){Add(new Point(left, y));}}**

**else if (dy == 0){**

**for (int x = left; x <= right; ++x){Add(new Point(x, top));}}}}}**

**on the main entry point**

**using MoreLinq;**

**using System.Collections.Generic;**

**using static System.Console;**

**namespace AdapterSimplePro{**

**//we will use adapter design pattern which convert the vector object which is rectangle**

**//into list of lines that will converted into collection of points**

**//we have VectorObject which inherit from Collection<Line>**

**//we have VectorRectangle which inherit from VectorObject**

**//we have Line > Point Start,Point End**

**class Program{**

**public static readonly List<VectorObject> vectorObjects**

**= new List<VectorObject>(){**

**new VectorRectangle(1,1,10,10),**

**new VectorRectangle(3,3,6,6)};**

**public static void DrawPoint(Point p){Write(".");}**

**static void Main(string[] args){**

**Draw();**

**Draw();**

**ReadLine();**

**WriteLine("Hello World!");}**

**private static void Draw(){**

**foreach (var vo in vectorObjects){**

**foreach (var line in vo){**

**var adapter = new LineToPointAdapter(line);**

**adapter.ForEach(pt => {WriteLine($"({pt.X},{pt.Y})");});}}**

**ReadLine();}}}**