

חזרה והרחבה של תכנות מונחה עצמים

בסיכום שיעור של 1.12.2014 הזכרנו את המושג: class (מחלקה או טיפוס) שבעזרתו ניתן להגדיר עצמים (objects). הייתה לנו דוגמה של מחלקה מסוג: תלמיד.

אמרנו שעצם הוא מקרה פרטי של מחלקה, כלומר ממחלקה בשם תלמיד, ניתן לבנות עצמים, כאשר כל עצם הוא תלמיד ספציפי, ותהליך זה ניקרא: בניית עצם או: instantiation of an object

אמרנו שעצם מורכב לרוב מתכונות (attributes) ומפעולות או מתודות (methods)

היום נראה דוגמה לעצם שיהיה עיגול ונגדיר מחלקה שניקראת: Circle - נישתמש במחלקה לבנות מספר עצמים ונראה איך זה עובד.

צעד אחד אחורה - עבור כל פעולה או method שניכתוב בשפת C# בד"כ בכותרת של הפעולה נראה מילות מפתח כמו: int, public, void וכד'. כעת הגיע הזמן להסביר מה הכוונה.

public - יש גם private למרות שעדיין לא ראינו. ב-public כל פעולה אחרת יכולה להשתמש בפעולה ש מוגדרת עם public. למשל פעולת ה-Main תוכל לקרוא לפעולה אחרת אם האחרת מוגדרת כ-public.

void או טיפוס משתנה כמו int או string או long או כל טיפוס אחר - מקדים הגדרה של פעולה באופן שהוא מציין איזה סוג ערך 'מחזירה' הפעולה.

למשל: public void print() - זוהי כותרת של פעולה בשם print שניתן להשתמש בה על ידי פעולות אחרות כי היא public והיא לא מחזירה שום ערך ולכן: void.

עוד דוגמה: public int show_grade(1) - זוהי כותרת של פעולה בשם: show_grade שגם ניתן להשתמש בה בכל מקום והיא מחזירה ערך שלם (int), היא גם מקבלת פרמטר (בסוגריים). הפעולה הקודמת print לא קיבלה שום פרמטר ולכן הסוגריים היו ריקים.

בתוך המחלקה, כפי שכבר הסברנו בעבר, ישנם משתנים (מאפיינים) ופעולות - כמו בדוגמת התלמידים. ישנה גם פעולה שניקראת: פעולה בונה או באנגלית: Constructor (עליה עדיין לא דיברנו) מהי פעולה בונה? זו פעולה שמגדירה את העצם ההתחלתי שאותו נירצה ליצור במהלך התוכנית. הפעולה הבונה תהיה בעלת אותו שם כמו שם המחלקה ולא יהיה לה ערך מוחזר (אפילו לא void)

דוגמת הגדרת המחלקה של העיגול: לעיגול תהיינה 3 תכונות - קואורדינטת x, קואורדינטת y ורדיוס r. הפעולות יהיו: שינוי רדיוס וציור העיגול.

```
public class Circle {
    private int r, x, y;
    public Circle(int radius, int xcoord, int ycoord) //Constructor
    {
        r=radius;
        x=xcoord;
        y=ycoord;
    }
    public void draw()
    {
        Console.WriteLine("Circle with radius: {0} is drawn at: {1}:{2}", r,x,y);
    }
    public void change_radius(int c)
    {
        r=c;
    }
}
```

שימו לב שתכונות המעגל הוגדרו כמשתנים מסוג `private`. זה כדי להגן עליהם מפני שינוי מחוץ למחלקה - מה שניקרא: `Data encapsulation` - אחת ממטרות התכנות מונחה עצמים זה שיהיו כמה שפחות אפשרויות לטעות ולכן אם מגינים על משתנים ונותנים להם תכונת `private` - באם יהיה בהם שינוי בלתי מובן, נוכל לחפשו רק בתוך המחלקה שבה הוגדר ולא בכל התוכנית.

שימו לב שפעולת הציור היא רק מתודית ולא ממש ציור (כי לא למדנו גרפיקה בשפה).

בואו נעשה שימוש מייד במחלקה שהגדרנו בדוגמה של תוכנית:

```
using System;
using System.Collections.Generic;
namespace Circles
{
    /* Here comes the definition of Circle as we outlined above */
    class MainClass
    {
        public static void Main(string[] args)
        {
            Circle c1= new Circle(4,2,3); //Instantiating one object
            Circle c2=new Circle(7,5,2); //Instantiating second object
            c1.draw();
            c2.draw();
            c1.change_radius(6);
        }
    }
}
```