מבוא לתכנות II

הרצאה 5 – OOP II – רב צורתיות

סמסטר 2

רב צורתיות (פולימורפיזם) - הגדרה

- היכולת לממש פונקציה בעלת אותה שם במחלקה הנגזרת, כאשר בזמן ריצה הפונקציה תעבוד בהתאם לסוג האובייקט ולא לסוג ההפניה.
- ??? ☺
- הורשה הינה הכרח למימוש רב צורתיות.
- לפי ברירת המחדל, הגישה לפונקציה נקבעת לפי ההפניה, אולם ברב צורתיות נעשה את ההפך: ניגש לפונקציות דרך האובייקט שנוצר בזמן הריצה במקום דרך ההפניה.

Virtual

- דרך שימוש במילות המפתח *virtual* ו-*override* נממש את רב הצורתיות.
- המילה *virtual* תופיע במחלקת הבסיס והמילה *override* תופיע במחלקה הנגזרת.

```
class Base
{
    public int x;

    public virtual void Print()
    {
        Console.WriteLine("Base print");
    }
}

class Derived : Base
{
    public int y;

    public override void Print()
    {
        Console.WriteLine("Derived print");
    }
}
```

המשך בדף הבא...

המשך Virtual

```
class Program
{
    static void Main(string[] args)
    {
        Base b = new Derived();
        b.Print();
    }
}
```

Derived print

- מה יקרה ללא השימוש במילים *virtual* ו-*override*?
- המילה *override* תעבוד רק אם פונקציית הבסיס מוגדרת בתור *virtual*.
- פונקציה המוגדרת כ *override* אוטומטית מוגדרת בתור *virtual* עבור המחלקות היורשות בהמשך שושלת ההורשה.
- אם פונקצית הבסיס לא *virtual* אז הפונקציה הנגזרת תוגדר בתור *new* והיא לא תיקרא בהתאם לאובייקט שנוצר.
- הפונקציה הנגזרת לא יכולה להיות ברמת נגישות נמוכה משל פונקציית הבסיס. לדוגמה, היא לא יכולה להיות פרטית.

גישה לשדות

- כאשר ההפניה היא מסוג מחלקת הבסיס והאובייקט הוא מסוג המחלקת הנגזרת לא נוכל לגשת לשדות של המחלקה הנגזרת – שגיאת קומפילציה.
- הפתרון הוא ליצור הפניה מתאימה (של המחלקה הנגזרת) עבור האובייקט למשל ע"י המרה (casting).

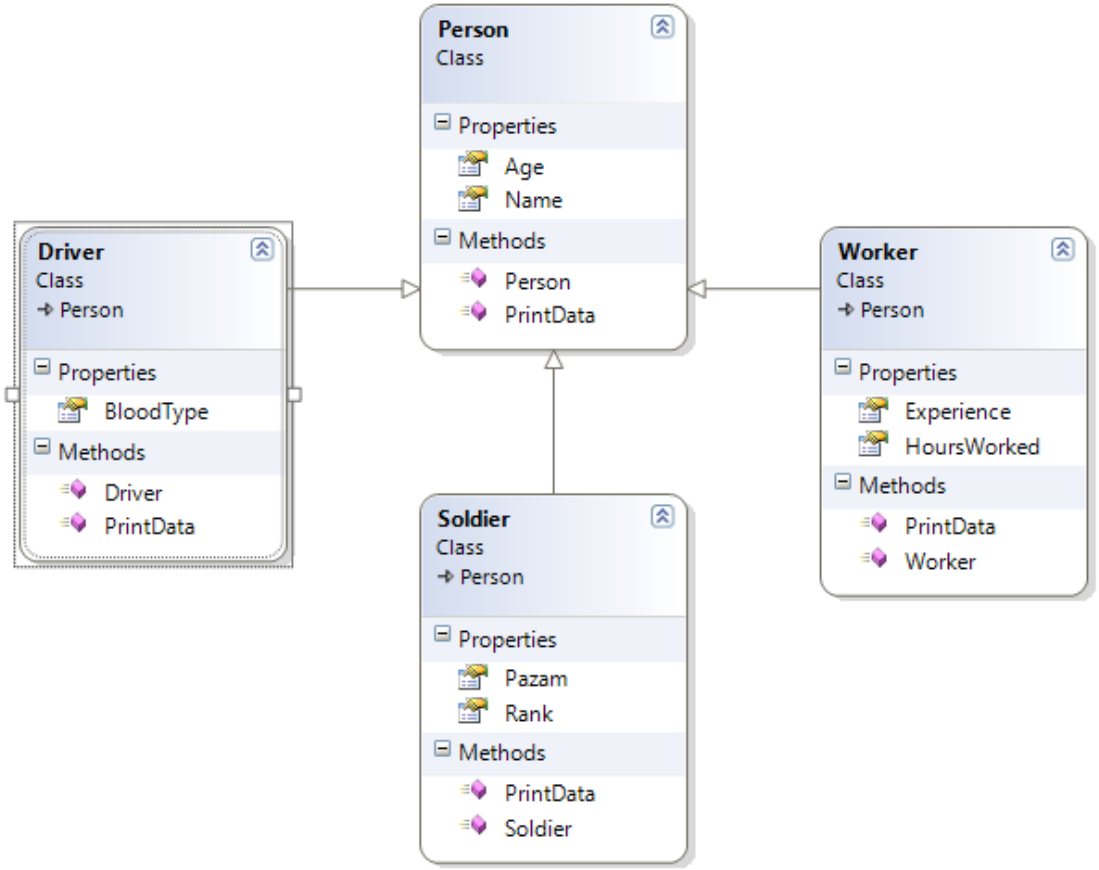
```
b.y = 10;

Derived d = (Derived)b;
d.y = 10;

((Derived)b).y = 10;      //shorter version - the same as above
```

Error!

שימושים

- מערך הטרוגני – יצירת מערך מסוג הבסיס אשר מסוגל להכיל אובייקטים מכל הסוגים היורשים. במעבר על המערך נוכל לקרוא לפונקציה אחת שתתאים את עצמה לפי סוג האובייקט, ללא צורך מצדנו למצוא עבור כל איבר מאיזה סוג הוא.
- כפרמטר לפונקציה – ארגומנט הפונקציה יהיה מסוג הבסיס, ואנו נוכל לשלוח אובייקטים מכל הסוגים היורשים בידיעה שהפרמטר יוכל לקבל את כולם ולהפעיל את הפונקציות הרלוונטיות לכל אובייקט.
- ללא שימוש ברב צורתיות ניאלץ לעשות שימוש מופרז בtry-catch כדי למצוא את האובייקט המתאים כדי לקרוא לפונקציה המתאימה עבורו. לעתים גם זה לא פתרון אפשרי מכיוון שלא תמיד נדע אילו מחלקות יורשות קיימות, כי הן נכתבו על ידי מישהו אחר! (למשל DLL)

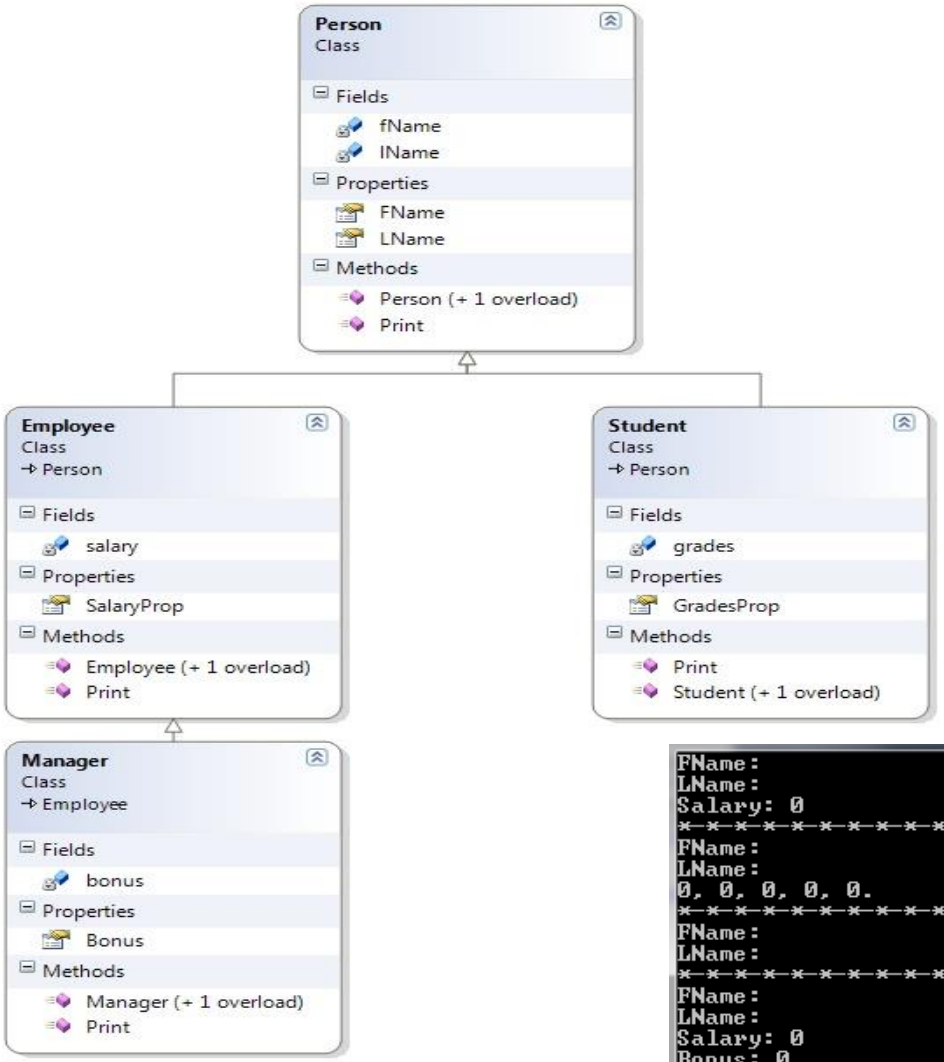
Exercise 1 - 02 Person Driver Soldier Worker




```
//Worker worker = new Worker("Nir", 33, 200, 9);
//Driver driver = new Driver("Karin", 29, "A+");

//worker.PrintData();
//Console.WriteLine();
//driver.PrintData();

Person[] persons = new Person[4];
persons[0] = new Worker("Ilan", 20, 300, 10);
persons[1] = new Driver("Moti", 24, "AB");
persons[2] = new Person(22, "Stam");
persons[3] = new Soldier("Rambo", 27, 12, "major");

for (int i = 0; i < persons.Length; i++)
{
    persons[i].PrintData();
    Console.WriteLine();
}

Soldier s = (Soldier)persons[3];
s.PrintData();
```

```
Name: Ilan, Age: 20
Hours: 300, Exp: 10...

Name: Moti, Age: 24
Blood type: AB!

Name: Stam, Age: 22

Name: Rambo, Age: 27

Name: Rambo, Age: 27
Rank: major, Pazam: 12!
Press any key to continue . . .
```

Exercise 2 - 03 Person Student Employee Manager

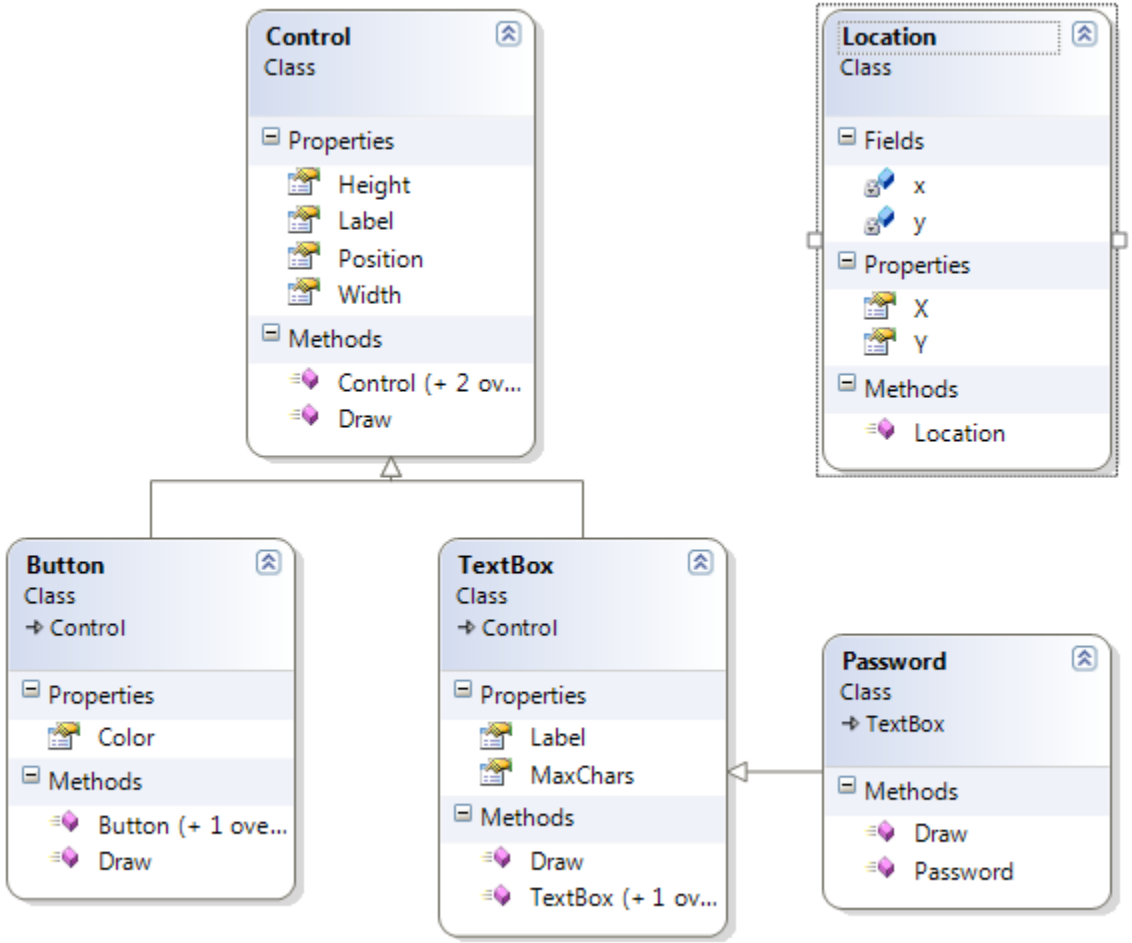


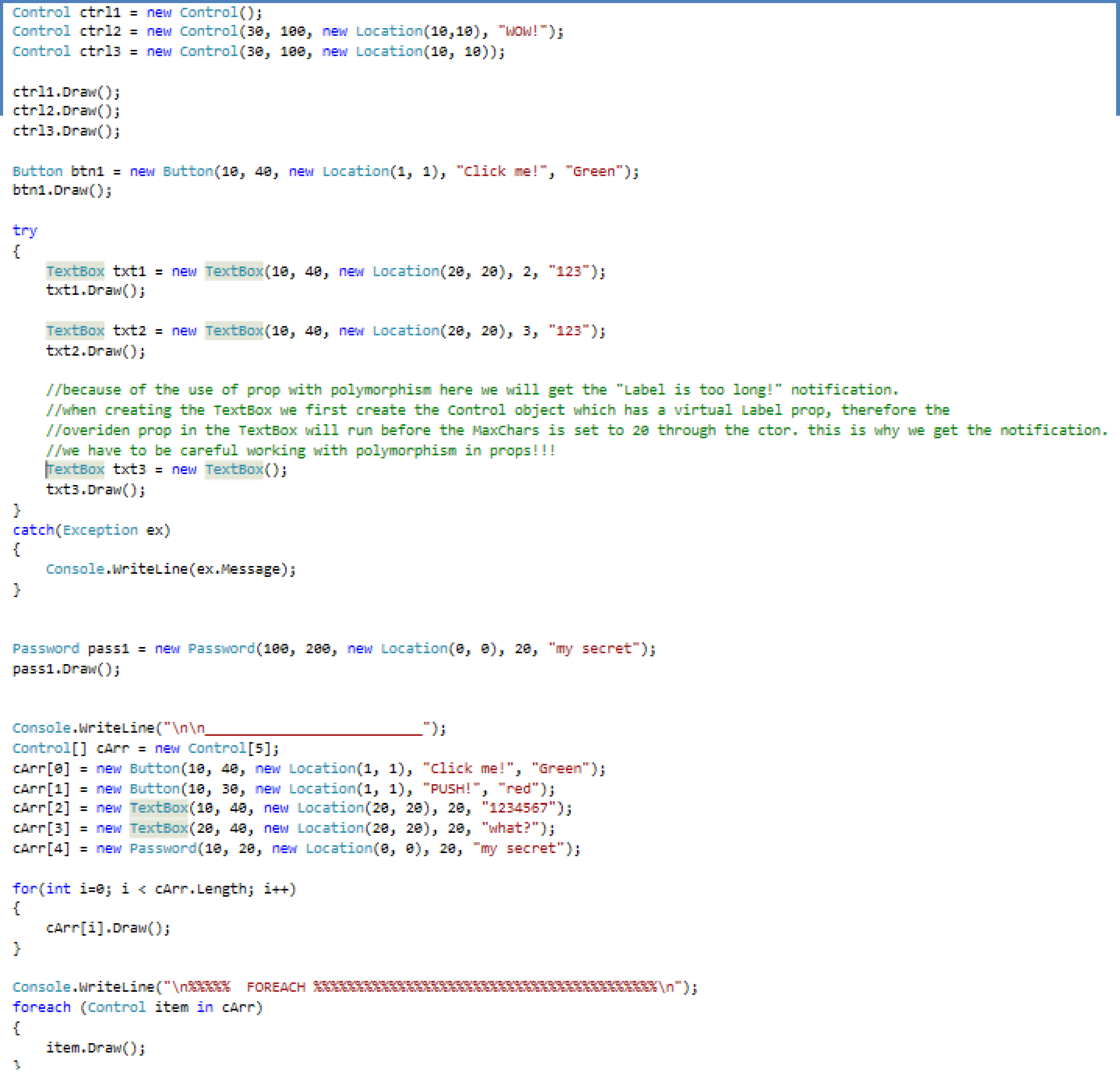

```
static void Test1()
{
    Person[] pArr = new Person[5];
    pArr[0] = new Person();
    pArr[1] = new Employee();
    pArr[2] = new Student();
    pArr[3] = new Person();
    pArr[4] = new Manager();

    for (int i = 0; i < pArr.Length; i++)
    {
        pArr[i].Print();
        Console.WriteLine("*-*-*-*-*-*-*-*-*-*-*");
    }
}

static void Test2(Person p)
{
    p.Print();
}
static void Main(string[] args)
{
    Test1();
    Person p1 = new Person();
    Manager m1 = new Manager();

    Test2(p1);
    Console.WriteLine("----------------------");
    Test2(m1);
    Console.WriteLine("----------------------");
}
```

```
FName:
LName:
Salary: 0
*-*-*-*-*-*-*-*-*-*-*-*
FName:
LName:
0, 0, 0, 0, 0.
*-*-*-*-*-*-*-*-*-*-*-*
FName:
LName:
*-*-*-*-*-*-*-*-*-*-*-*
FName:
LName:
Salary: 0
Bonus: 0
*-*-*-*-*-*-*-*-*-*-*-*
FName:
LName:
----------------------
FName:
LName:
Salary: 0
Bonus: 0
----------------------
Press any key to continue .
```

Exercise 3
04 Exercise Winix

- Read the instruction in the directory of the exercise
- !!! Here is good example (and explanation) for polymorphism in props and the tricky stuff about it, see the creation of TextBox txt3 , line 35 !!!



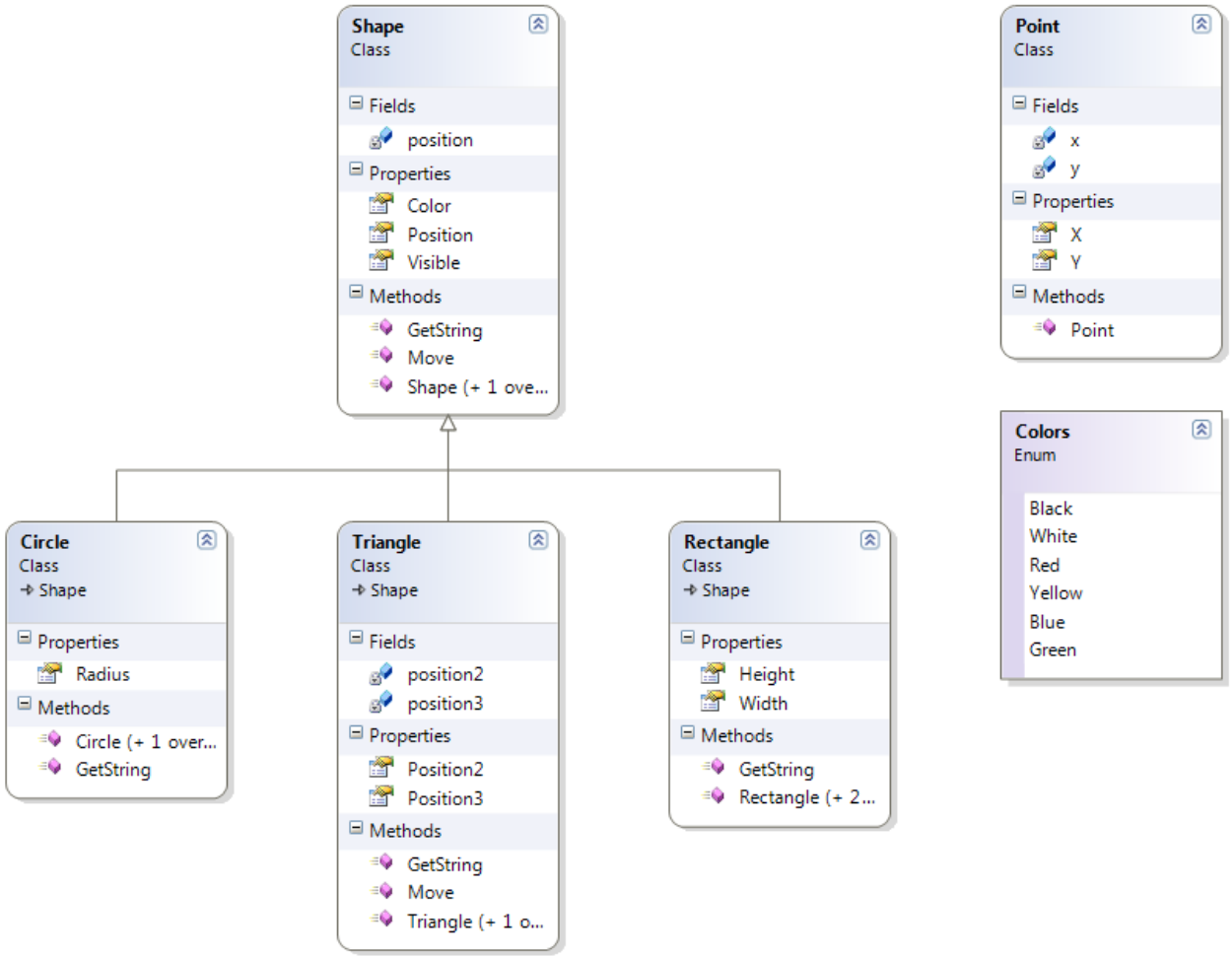


```
Control ctrl1 = new Control();
Control ctrl2 = new Control(30, 100, new Location(10,10), "WOW!");
Control ctrl3 = new Control(30, 100, new Location(10, 10));

ctrl1.Draw();
ctrl2.Draw();
ctrl3.Draw();

Button btn1 = new Button(10, 40, new Location(1, 1), "Click me!", "Green");
btn1.Draw();

try
{
    TextBox txt1 = new TextBox(10, 40, new Location(20, 20), 2, "123");
    txt1.Draw();

    TextBox txt2 = new TextBox(10, 40, new Location(20, 20), 3, "123");
    txt2.Draw();

    //because of the use of prop with polymorphism here we will get the "Label is too long!" notification.
    //when creating the TextBox we first create the Control object which has a virtual Label prop, therefore the
    //overiden prop in the TextBox will run before the MaxChars is set to 20 through the ctor. this is why we get the notification.
    //we have to be careful working with polymorphism in props!!!
    TextBox txt3 = new TextBox();
    txt3.Draw();
}
catch(Exception ex)
{
    Console.WriteLine(ex.Message);
}


Password pass1 = new Password(100, 200, new Location(0, 0), 20, "my secret");
pass1.Draw();


Console.WriteLine("\n\n__________________________");
Control[] cArr = new Control[5];
cArr[0] = new Button(10, 40, new Location(1, 1), "Click me!", "Green");
cArr[1] = new Button(10, 30, new Location(1, 1), "PUSH!", "red");
cArr[2] = new TextBox(10, 40, new Location(20, 20), 20, "1234567");
cArr[3] = new TextBox(20, 40, new Location(20, 20), 20, "what?");
cArr[4] = new Password(10, 20, new Location(0, 0), 20, "my secret");

for(int i=0; i < cArr.Length; i++)
{
    cArr[i].Draw();
}

Console.WriteLine("\n%%%%%%  FOREACH %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%\n");
foreach (Control item in cArr)
{
    item.Draw();
```

```
Empty
WOW!

**************
* Click me! *
**************
Label is too long!
*********

*********
***
123
***
Label is too long!
*********

*********
**********
**********
**********


**************
* Click me! *
**************
**********
* PUSH! *
**********
********
1234567
********
*****
what?
*****
**********
**********
**********

%%%%%  FOREACH %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

**************
* Click me! *
**************
**********
* PUSH! *
**********
********
1234567
********
*****
what?
*****
**********
**********
**********
Press any key to continue . . .
```

Exercise 4
05 Exercise Shapes

- Read the instruction in the directory of the exercise



```
static void Main(string[] args)
{
    Shape[] shapes = new Shape[3];
    //shapes[0] = new Rectangle();
    //shapes[1] = new Circle();
    //shapes[2] = new Triangle();

    //for (int i = 0; i < shapes.Length; i++)
    //{
    //    Console.WriteLine(shapes[i]);
    //}

    string output = "";
    int i = 0;
    while (output != "4")
    {
        Console.WriteLine("Please select:");
        Console.WriteLine("1) Rectangle");
        Console.WriteLine("2) Circle");
        Console.WriteLine("3) Triangle");
        Console.WriteLine("4) Exit");
        Console.WriteLine("5) Print array!");
        Console.WriteLine("6) move 100 to the right!");
        Console.WriteLine("7) Print array type!");

        output = Console.ReadLine();

        switch (output)
        {
            case "1":
                Console.Write("Enter width:");
                int width = Convert.ToInt32(Console.ReadLine());
                Console.Write("Enter height:");
                int height = Convert.ToInt32(Console.ReadLine());
                shapes[i] = new Rectangle(width, height);
                i++;
                break;
            case "2":
                Console.Write("Enter radius:");
                int radius = Convert.ToInt32(Console.ReadLine());
                shapes[i] = new Circle(radius);
                i++;
                break;
            case "3":
                Console.Write("Enter x:");
                int x = Convert.ToInt32(Console.ReadLine());
                Console.Write("Enter y:");
                int y = Convert.ToInt32(Console.ReadLine());
                Console.Write("Enter x2:");
                int x2 = Convert.ToInt32(Console.ReadLine());
                Console.Write("Enter y2:");
                int y2 = Convert.ToInt32(Console.ReadLine());
                shapes[i] = new Triangle(new Point(x, y), new Point(x2, y2));
                break;
```

```
            case "4":
                Console.WriteLine("Bye!");
                return;
            case "5":
                for (int j = 0; j < shapes.Length; j++)
                {
                    if (shapes[j] != null)
                    {
                        Console.WriteLine(shapes[j].GetString());
                    }
                }
                break;
            case "6":
                for (int j = 0; j < shapes.Length; j++)
                {
                    if (shapes[j] != null)
                    {
                        shapes[j].Move(100,0);
                    }
                }
                break;
            case "7":
                for (int j = 0; j < shapes.Length; j++)
                {
                    if (shapes[j] != null)
                    {
                        Console.WriteLine(shapes[j].GetType());
                    }
                }
                break;
```

```
Please select:
1) Rectangle
2) Circle
3) Triangle
4) Exit
5) Print array!
6) move 100 to the right!
7) Print array type!
1
Enter width:2
Enter height:4
Please select:
1) Rectangle
2) Circle
3) Triangle
4) Exit
5) Print array!
6) move 100 to the right!
7) Print array type!
2
Enter radius:5
Please select:
1) Rectangle
2) Circle
3) Triangle
4) Exit
5) Print array!
6) move 100 to the right!
7) Print array type!
5
Position: (0,0), Visible: True, Color: Red, Width: 2, Height: 4
Position: (0,0), Visible: True, Color: Red, Radius: 5
Please select:
1) Rectangle
2) Circle
3) Triangle
4) Exit
5) Print array!
6) move 100 to the right!
7) Print array type!
6
Please select:
1) Rectangle
2) Circle
3) Triangle
4) Exit
5) Print array!
6) move 100 to the right!
7) Print array type!
5
Position: (100,0), Visible: True, Color: Red, Width: 2, Height: 4
Position: (100,0), Visible: True, Color: Red, Radius: 5
Please select:
1) Rectangle
2) Circle
3) Triangle
4) Exit
5) Print array!
6) move 100 to the right!
7) Print array type!
```