C# programming language. The beginning

Course

Programming Languages

Semester 2, FIIT

Mayer S.F.

Mikhalkovich S.S

LECTURE # 17. Classes

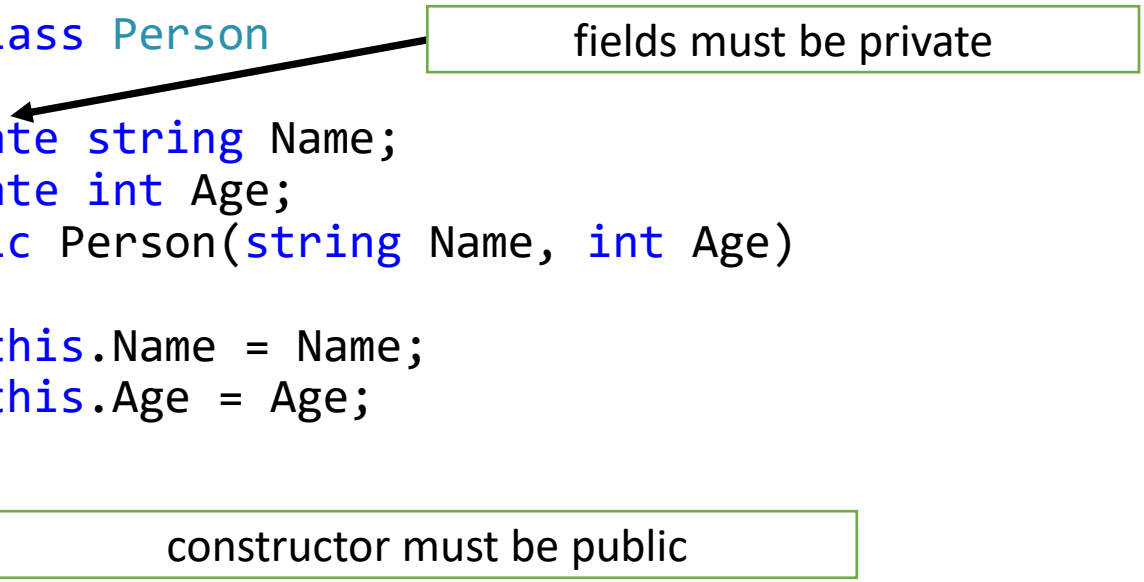
Classes

Lesson # 17

Class definition

private modifier – is the default modifier for members

```
public class Person
{
    private string Name;
    private int Age;
    public Person(string Name, int Age)
    {
        this.Name = Name;
        this.Age = Age;
    }
}
```



- How to access to the fields outside of a class?

Class definition 2

```
public class Person
{
    private string Name;
    private int Age;
    public Person(string Name, int Age) // constructor
    {
        this.Name = Name;
        this.Age = Age;
    }
    public void Print()
    {
        Console.WriteLine($"{Name} {Age}");
    }
}
class Program
{
    static void Main(string[] args)
    {
        Person p1 = null;
        p1 = new Person("Name1", 20);
        Person p2 = new Person("Name2", 21);
        var p3 = new Person("Name3", 19);

        p1.Print(); p2.Print(); p3.Print();
    }
}
```

main goal of a constructor is to initialize class fields

Getters and Setters:

To control the access to attributes. And to have possibility to check if data is valid for exact attribute

```
 public int age
        {
            // to read the field: p1.name
            get { return Age; }
            // to change the field: p1.name = ..
            set { Age = value; }
        }
public int grade
        {
            get { return Grade; }
            set
            {
                if (value < 1 || value > 5)
                    grade = 0;
                else
                    grade = value;
            }
        }
```

```
p1.age = 2;
```

Static attribute

```
private string Name;
private int Age;
private int Grade;
// static attribute:
public static int pupilsCount = 0; // with initial value


// constructor:
public Pupils(string name, int age, int grade)
{
    Name = name;
    Age = age;
    Grade = grade;
    // whenewhere we create an object, we should count it:
    pupilsCount++;
}
```

We should apply this static attribute to class, not to the object

```
Console.WriteLine($"total number ={Pupils.pupilsCount}");
```

Properties and methods

- Properties allow to restrict access to the fields only by reading their values. Methods can be created to change a value of property

```
public class Person
{
    public string Name { get; }
    public int Age { get; private set; }
    public Person(string Name, int Age)
    {
      this.Name = Name; // Only in constructor
      this.Age = Age;
    }
    public void IncAge()
    {
      Age += 1; // Only in methods
    }
}
...
var p = new Person("Name",18);
p.Name = "Name2"; // Error!
p.Age = 100; // Error!
p.IncAge(); // OK
```

To increase the Age property, we can create a method to do it

Lecture tasks

Lesson #17

https://labs-org.ru/c-sharp17-eng/

Q & A