Objektové programování I (C#) – 2

RNDr. Michal Pobucký

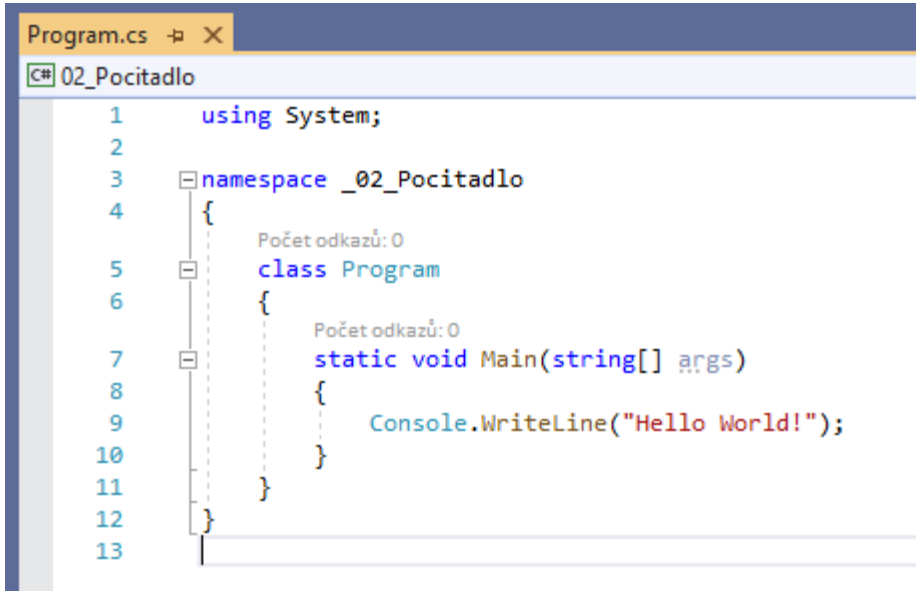
michal.pobucky@fpf.slu.cz

Ústav informatiky – zima 2020

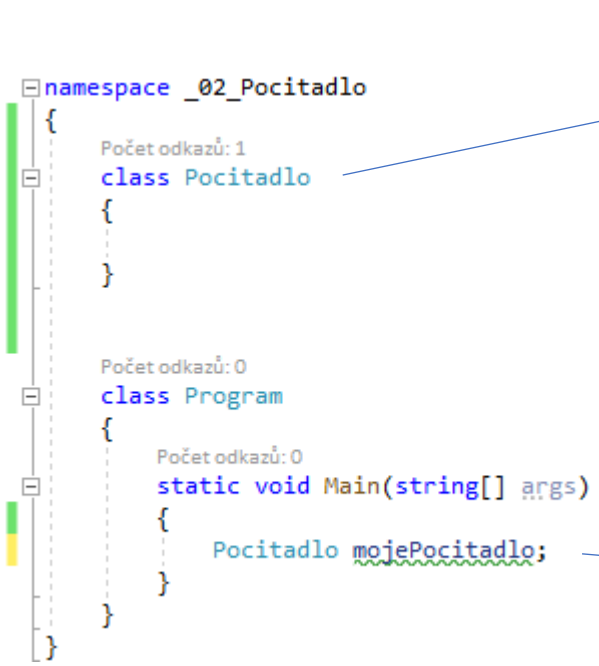
SLEZSKÁ UNIVERZITA
FILOZOFICKO-PŘÍRODOVĚDECKÁ FAKULTA V OPAVĚ

Třídy

- Vytvořte konzolovou aplikaci (.NET Core), název projektu 02_Pocitadlo



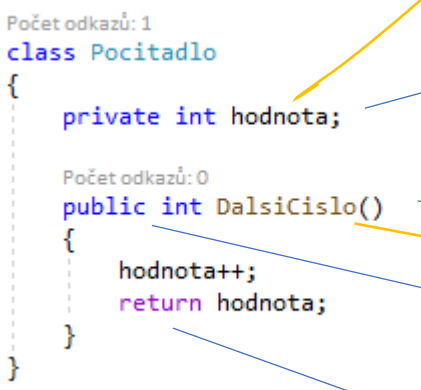
Třídy



Název třídy

Můžeme používat jako ostatní typy

Třídy



Základní jednotkou OP je objekt. Objekt obsahuje atributy (vlastnosti) a metody (schopnosti), které umí vykonávat.

Proměnná není veřejná, není vidět „z venku“, nedá se k ní přímo přistupovat

Metoda DalsiCislo

Je veřejná, viditelná „z venku“, vrací hodnotu typu int.

Musí obsahovat příkaz return pro vrácení hodnoty.

Třídy

Třída je vzor, podle kterého se objekty vytváří. Instance je objekt vytvořený podle třídy.

```
Počet odkazů: 0
static void Main(string[] args)
{
    Pocitadlo mojePocitadlo = new Pocitadlo();
    var tvojePocitadlo = new Pocitadlo();

    Console.WriteLine("Moje:  " + mojePocitadlo.DalsiCislo());
    Console.WriteLine("Tvoje: " + tvojePocitadlo.DalsiCislo());
}
```

Třídy

- Základem OP je
  - Zapouzdření
  - Dědičnost
  - Polymorfismus

Třídy – zapouzdření

- Umožňuje skrýt atributy a metody
- Private, public, protected

```
class Osoba
{
    public string Jmeno, Prijmeni;
    private DateTime datumNarozeni;

    Počet odkazů: 1
    public void Vypis()
    {
        Console.WriteLine("Já jsem " + Jmeno + " " + Prijmeni + ".");
    }
}
```

```
Osoba prvni = new Osoba();
prvni.Jmeno = "Karel";
prvni.Prijmeni = "Novák";
prvni.Vypis();
```

Třídy – dědičnost

```
Počet odkazů: 3
class Osoba
{
    public string Jmeno, Prijmeni;
    private DateTime datumNarozeni;

    Počet odkazů: 3
    public virtual void Vypis()
    {
        Console.WriteLine("Já jsem " + Jmeno + " " + Prijmeni + ".");
    }
}
```

```
Počet odkazů: 2
class Student:Osoba
{
    Počet odkazů: 3
    public override void Vypis()
    {
        Console.WriteLine("Já jsem student " + Jmeno + " " + Prijmeni + ".");
    }
}
```

```
Osoba prvni = new Osoba();
prvni.Jmeno = "Karel";
prvni.Prijmeni = "Novák";
prvni.Vypis();

Student druhy = new Student();
druhy.Jmeno = "Pavel";
druhy.Prijmeni = "Svoboda";
druhy.Vypis();
```

Třídy – konstruktor

```
Počet odkazů: 1
public Student()
{
    Jmeno = "Jana";
    Prijmeni = "Nováková";
}
Počet odkazů: 3
```

```
Student druhy = new Student();
//druhy.Jmeno = "Pavel";
//druhy.Prijmeni = "Svoboda";
druhy.Vypis();
```

Zkuste v konstruktoru změnit i proměnnou datumNarozeni. Co se stane a proč?

Třídy – konstruktor – přetěžování

```
Počet odkazů: 3
class Osoba
{
    public string Jmeno, Prijmeni;
    protected DateTime datumNarozeni;

    Počet odkazů: 3
    public virtual void Vypis()
    {
        Console.WriteLine("Já jsem " + Jmeno + " " + Prijmeni + ".");
    }
}
```

```
Počet odkazů: 1
public Student()
{
    Jmeno = "Jana";
    Prijmeni = "Nováková";
    datumNarozeni = DateTime.Now;
}
```

```
Počet odkazů: 1
public Student()
{
    Jmeno = "Jana";
    Prijmeni = "Nováková";
    datumNarozeni = DateTime.Now;
}
Počet odkazů: 0
public Student(string vstupJmeno, string vstupPrijmeni, DateTime vstupNarozen)
{
    Jmeno = vstupJmeno;
    Prijmeni = vstupPrijmeni;
    datumNarozeni = vstupNarozen;
}
```

```
Student treti = new Student("Alois", "Jirásek", new DateTime(1851, 8, 23));
treti.Vypis();
```

Třídy

- Všechny třídy jsou odvozeny od třídy object, od které dědí některé metody a atributy

```
Student treti = new Student("Alois", "Jirásek", new DateTime(1851, 8, 23));
treti.Vypis();
Console.WriteLine(treti);
```

```
Student treti = new Student("Alois", "Jirásek", new DateTime(1851, 8, 23));
treti.Vypis();
Console.WriteLine(treti.ToString());
```

Co se stane?

Třídy – přetěžování metod

```
class Student:Osoba
{
    Počet odkazů: 1
    public Student()
    {
        Jmeno = "Jana";
        Prijmeni = "Nováková";
        datumNarozeni = DateTime.Now;
    }
    Počet odkazů: 1
    public Student(string vstupJmeno, string vstupPrijmeni, DateTime vstupNarozen)
    {
        Jmeno = vstupJmeno;
        Prijmeni = vstupPrijmeni;
        datumNarozeni = vstupNarozen;
    }
    Počet odkazů: 0
    public override string ToString()
    {
        return "Já jsem student " + Jmeno + " " + Prijmeni + ".";
    }
}
```

```
Student treti = new Student("Alois", "Jirásek", new DateTime(1851, 8, 23));
Console.WriteLine(treti);
```

Třídy – vlastnosti

```
Počet odkazů: 3
class Osoba
{
    public string Jmeno, Prijmeni;
    protected DateTime datumNarozeni;
    private int vyska;

    Počet odkazů: 0
    public int Vyska
    {
        get
        {
            return vyska;
        }
        set
        {
            if (value > 0) vyska = value;
        }
    }
```

```
Student treti = new Student("Alois", "Jirásek", new DateTime(1851, 8, 23));
Console.WriteLine(treti);
Console.WriteLine(treti.Vyska);
treti.Vyska = -5;
Console.WriteLine(treti.Vyska);
treti.Vyska = 182;
Console.WriteLine(treti.Vyska);
```

Třídy – statická metoda

```
Počet odkazů: 0
static void Main(string[] args)
{
    Pocitadlo mojePocitadlo = new Pocitadlo();
    var tvojePocitadlo = new Pocitadlo();

    Console.WriteLine("Moje:   " + mojePocitadlo.Da
    Console.WriteLine("Tvoje: " + tvojePocitadlo.D
```

Označení static určuje, že nemusíme mít vytvořenou žádnou instanci třídy a můžeme přistupovat k metodám, atributům nebo vlastnostem. To už známe.

Kde jsme již použili metodu aniž bychom měli instanci třídy?

Třídy – statická metoda

```
Počet odkazů: 0
static void Main(string[] args)
{
    Pocitadlo mojePocitadlo = new Pocitadlo();
    var tvojePocitadlo = new Pocitadlo();

    Console.WriteLine("Moje:  " + mojePocitadlo.Da
    Console.WriteLine("Tvoje: " + tvojePocitadlo.D
```

Označení static určuje, že nemusíme mít vytvořenou žádnou instanci třídy a můžeme přistupovat k metodám, atributům nebo vlastnostem. To už známe.

Kde jsme již použili metodu aniž bychom měli instanci třídy?

```
Console.WriteLine(treti.Vyska);
```

Třídy – práce s pamětí

```
int promenna = 42;
int kopiePromenne = promenna;
kopiePromenne++;
Console.WriteLine(kopiePromenne);
Console.WriteLine(promenna);
```

Jaký bude výstup a proč?

```
Student ctvrty;
ctvrty = treti;
Console.WriteLine(ctvrty);

ctvrty.Jmeno = "Lojza";
Console.WriteLine(ctvrty);
Console.WriteLine(treti);
```

Jaký bude výstup a proč?

Třídy – práce s pamětí

```
int promenna = 42;
int kopiePromenne = promenna;
kopiePromenne++;
Console.WriteLine(kopiePromenne);
Console.WriteLine(promenna);
```

Jaký bude výstup a proč?

```
Student ctvrty;
ctvrty = treti;
Console.WriteLine(ctvrty);

ctvrty.Jmeno = "Lojza";
Console.WriteLine(ctvrty);
Console.WriteLine(treti);
```

Jaký bude výstup a proč?