


Programování

Třída a struktura





Třída a struktura jako datový typ

- ▶ Nová instance třídy nebo struktury vyžaduje určitý prostor v paměti
 - ▶ Struktura je kompaktnější – její celková velikost se dá spočítat
 - ▶ Práce se strukturou je paměťově levnější
 - ▶ Objekt, který je ze třídy vytvořen slouží pouze jako reference na místo v paměti
 - ▶ Objekty třídy jsou sice paměťově dražší, ale přináší své vlastní výhody při používání
- 



Struktura

- ▶ Odlehčená verze třídy
- ▶ Svými vlastnostmi je blíže k předdefinovaným datovým typům
- ▶ Struktura slouží jako obalení základní datových typů, do kompaktního formátu
- ▶ Hodí se do malých programů a pro popis jednoduchých objektů
- ▶ Ideální použití struktur je v případech, kdy potřebujeme shlukovat několik informací spolu související

Struktura

- Proměnné, které struktura schraňuje stačí uvést pouze skrze vlastnosti (**properties**)
- Klíčová slova get, set, init nám určují jak s danými hodnotami můžeme pracovat
- Struktura může mít vlastní konstruktor i další metody

```
Počet odkazů: 3
] struct Souradnice
{
    Počet odkazů: 2
    int X { get; set; }
    Počet odkazů: 2
    int Y { get; set; }
    Počet odkazů: 2
    int Z { get; init; }

    Počet odkazů: 1
    public Souradnice(int _x, int _y, int _z)
    {
        X = _x; Y = _y; Z = _z;
    }

    Počet odkazů: 0
    public override string ToString()
    {
        return $"[{X},{Y},{Z}]";
    }
}
```



Třída

- Základní stavební prvek OOP
 - Umožňuje využívat vlastnosti dědičnosti, zapouzdření, polymorfismu
- Slouží jako šablona pro vytváření nových objektů
- Atributy (**fields**) jsou uváděny jako privátní, aby k nim nešlo přistupovat
- Pro zpřístupnění je třeba uvádět `public` dané třídy
- Stejně jako struktura má svůj konstruktor a může mít své vlastní metody
- C# umožňuje třídě rozšiřovat pouze jednu jinou třídu, ale může implementovat více rozhraní

Třída

- ▶ I u obyčejné třídy využíváme vlastnosti zapouzdření
- ▶ Zápis atributů můžeme nahradit užitím vlastností, ale pro další práci může být takový přístup svazující
- ▶ Stejně jako u struktury můžeme říci, které atributy budou dostupné

```
class Bezec
{
    private string jmeno;
    private int startovniCislo;

    Počet odkazů: 0
    public string Jmeno { get { return jmeno; } }
    Počet odkazů: 0
    public int StartovniCislo
    {
        get { return startovniCislo; }
        init { startovniCislo = value; }
    }

    Počet odkazů: 0
    public Bezec(string _jmeno)
    {
        jmeno = _jmeno;
        startovniCislo = new Random().Next(1, 101);
    }
}
```



Procvičování tvorby tříd a struktur

Vytvořte strukturu:

1. Bod v prostoru
2. Poštovní adresa
3. Komolý kužel
4. Elipsa
5. Nábytek
6. Kabát

Vytvořte třídu:

1. Zboží uložené ve skladu
2. Knížka v knihovně
3. Vozidlo
4. Film
5. Příspěvek na sociální síti
6. Herní charakter