Lekce 4

Kolekce II

Kolekce II

- Zatím jsme se seznámili s kolekcí Array (pole)
- Pro další kolekce budeme potřebovat přidat do záhlaví programu:
  using System.Collections.Generic;
- Tyto kolekce mají narozdíl od pole rozšířené možnosti a vlastnosti, které nám mohou usnadnit práci ...

Kolekce - List<Type>

- List<int>, List<string>, atd. = List, seznam hodnot
- Kolekce obsahující hodnoty jednoho datového typu
- Narozdíl od pole **nemá pevně definovaný počet prvků**
- List je také indexovaný od nuly
- Počet prvků v Listu získáme z vlastnosti Count

Kolekce - List<Type>

- Vytvoření Listu
  - List<string> texty = new List<string>();
- Přidání prvků metodou Add:
  - texty.Add("pes");
  - texty.Add("kočka");
  - texty.Add("kůň");
  - //atd. lze libovolně kdykoli přidávat nové prvky, přidají se vždy nakonec

- Vytvoření listu rovnou s přiřazením hodnot
  - List<string> texty = new List<string>() { "pes", "kocka", "kun" };

Kolekce - List<Type>

- Přepsání hodnoty prvku:
  - texty[0] = "krava";
- Vložení prvku na vybranou pozici - původní prvek nesmaže, ale všechny hodnoty posune o index
  - texty.Insert(0,"mys");
- Odstranění prvku metodou Remove a RemoveAt:
  - texty.Remove("krava"); //podle hodnoty
  - texty.RemoveAt(0); //podle indexu
  - v obou případech dojde k odstranění prvku a na jeho místo se posouvá následující prvek

Kolekce - List<Type>

- Podobně jako string má i List metodu Contains
- V listu vyhledá prvek s danou hodnotou
  - Console.WriteLine(texty.Contains("pes")); //false

Kolekce - List<Type>

- Výpis všech hodnot v listu

```
for (int i = 0; i < texty.Count; i++)
        {
            Console.WriteLine(texty[i]);
        }
```

Cyklus - Foreach

- Tento cyklus **projde všechny prvky v dané kolekci**
- Jednodušší zápis, není potřeba index vzhledem k tomu, že prochází postupně všechny prvky

```
foreach(string item in texty)
{
    Console.WriteLine(item);
}
```

- Hodí se např. k rychlému výpisu všech prvků

Kolekce - List<Type>

- Hromadný zápis do listu pomocí cyklu
  - např. zkopírování obsahu listu texty do nového listu a převedení obsahu na velká písmena

```
List<string> textyVelke = new List<string>();
//POZOR: tady určuje konec List texty, nový List má zatím délku nula!
for (int i = 0; i < texty.Count; i++)
{
    textyVelke.Add(texty[i].ToUpper());
}
```

Kolekce - Dictionary<Key Type, Value Type>

- Dictionary<int,string>, Dictionary<string,string>, atd.
  = slovník
- Kolekce obsahující dvojice hodnot: **klíč + hodnotu**
- Datový typ klíče a hodnoty může být různý
- Také **nemá pevně definovaný počet prvků**
- Počet prvků (dvojic klíč + hodnota) ve slovníku získáme z vlastnosti Count

Kolekce - Dictionary<Key Type, Value Type>

- Vytvoření slovníku
  - Dictionary<string, string> telSeznam = new Dictionary<string, string>()
- Přidání prvků metodou Add:
  - telSeznam.Add("Pepa Novak", "123 456 789");
  - telSeznam.Add("Karel Mrak", "432 567 098");
  - //atd. lze libovolně kdykoli přidávat nové prvky, přidají se vždy nakonec

Je to čím dál tím delší...

Klíčové slovo var

- Při vytváření proměnné lze na levé straně místo datového typu použít klíčové slovo var
- Překladač si datový typ odvodí z pravé strany výrazu
  - var telSeznam = new Dictionary<string, string>()
  - var texty = new List<string>() { "pes", "kocka", "kun" };

- Nemusíte to vůbec používat, ale je dobré to znát pro brouzdání na Stack Overflow a podobně :)

Kolekce - Dictionary<Key Type, Value Type>

- Přepsání hodnoty pod daným klíčem:
  - telSeznam["Pepa Novak"] = "000 000 000";
- Výpis hodnoty pod daným klíčem:
  - Console.WriteLine(telSeznam["Pepa Novak"])
- Vyhledání podle klíče
  - Console.WriteLine(telSeznam.ContainsKey("Karel Slunicko"));
    //vrátí false
- Vyhledání podle hodnoty
  - Console.WriteLine(telSeznam.ContainsValue("432 567 098"));
    //vrátí true

KeyValuePair<Type, Type>

- Je dobré si uvědomit, z jakých prvků se slovník vlastně skládá
- Není to jeden string, nebo jeden int, atd.
- Je to **dvojice prvků** různých datových typů
- Je to něco úplně nového…

Kolekce - Dictionary<Key Type, Value Type>

- Jak vyhledáme klíč k dané hodnotě?

```
foreach(KeyValuePair<string, string> item in telSeznam)
{
    if(item.Value == "123 456 789")
    {
        Console.WriteLine(item.Key);
    }
}
```

TIP: místo *KeyValuePair<string, string>* lze napsat *var*, ale musíme vědět, co děláme :)

Výčtový typ enum

Výčtový typ enum

- obsahuje omezenou sadu pojmenovaných hodnot
  - měsíce v roce, světové strany, povolené operace, atd.
- je nutné jej nejprve vytvořit (podobně jako třídu)
  - enum Pocasi {Slunecno, Dest, Mraz};
- Můžeme vytvořit např. pole předpovědí, které uživatel zadal
  - Pocasi[] tydenniPredpoved = new Pocasi[7];
- Zadání první hodnoty
  - tydenniPredpoved[0] = Pocasi.Dest;
  - atd.

Výčtový typ enum

- Dále můžeme použít switch a poradit uživateli, co si má vzít na sebe

```
switch(tydenniPredpoved[0])
{
    case  Pocasi.Slunecno:
            Console.WriteLine("Nezapomen slunecni bryle.");
            break;
    case  Pocasi.Dest:
            Console.WriteLine("Nezapomen destnik.");
            break;
    case  Pocasi.Mraz:
            Console.WriteLine("Teple se oblekni.");
            break;
}
```

Výčtový typ enum

- Hodnoty v enum jsou indexované od nuly (podobně jako pole)
- Je možné je přiřadit i pomocí indexu
  - Pocasi pocasiDnes = (Pocasi)0; //přiřadí hodnotu Slunecno