

Jazyk C# 2

5. seminář

Večeřa J., Janoščík R.

Univerzita Palackého v Olomouci

21.3.2024

Reakce na úkoly

- Odevzdání
- Přehlednost kódu

Delegáti

• Delegate

- ▶ Nutné deklarovat signature předem
- ▶ Lepší opakované použití
- ▶ Pojmenované parametry

```
1 delegate string FuncDelegate(int input1, out int input2);
```

• Func

- ▶ In: 0-16, Out: 1
- ▶ Není potřeba deklarovat předem

```
1 Func<string, char> MethodFunc
```

• Action

- ▶ In: 0-16, Out: 0
- ▶ Vrací vždy void

```
1 Action<string> MethodAction
```

Lambda výrazy

- = anonymní funkce pro vytváření delegátů či výrazových stromů

```
1 delegate int somedelegate(int blah);
2 static void Main(string[] args) {
3     somedelegate del = (i) => i + 5;
4 }
5
6 Func<int, int> fce = ((int i) => i + 2);
7 Console.WriteLine(fce.Invoke(5));
```

- Mohou být předány jako parametr metody
- Mohou být vráceny metodou
- Při přiřazení však staticky typované

LINQ

- Zkratka pro Language INtegrated Query
- Jednotný přístup k datům
- Typová kontrola, našeptávače
- Přístup např. ke:
 - ▶ Kolekcím (vše co splňuje IEnumerable)
 - ▶ XML
 - ▶ Databázovým objektům
 - ▶ Webovým službám
- Rozšiřitelné

Syntaxe

- Textová „ukecaná“ syntaxe:

```
1 int[] numbers = { 1, 2, 3, 4, 5, 6, 7 };  
2 IEnumerable<int> bigNumbers = from num in numbers where num > 5  
    select num;
```

- ▶ Podobná přirozenému jazyku, může být matoucí

- Fluent API (preferovaná):

```
1 int[] numbers = { 1, 2, 3, 4, 5, 6, 7 };  
2 IEnumerable<int> bigNumbers2 = numbers.Where(p => p > 5);
```

- ▶ Přehlednější, „programátorštější“

Dotazovací operace (1/3)

- Restrikce

```
1 int[] numbers = { 1, 2, 3, 4, 5, 6, 7 };  
2 IEnumerable<int> bigNumbers2 = numbers.Where(p => p > 5);
```

Dotazovací operace (1/3)

- Restrikce

```
1 int[] numbers = { 1, 2, 3, 4, 5, 6, 7 };  
2 IEnumerable<int> bigNumbers2 = numbers.Where(p => p > 5);
```

- Selekcce

```
1 public class Person {  
2     public string Name { get; set; }  
3     public string Surname { get; set; }  
4     public int Age { get; set; }  
5     public string Address { get; set; }  
6 }  
7 List<Person> persons = new List<Person>();  
8 var names= persons.Select(p => new { p.Name, p.Surname });
```


Dotazovací operace (2/3)

- Řazení

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name);  
2 persons.OrderByDescending(p => p.Surname).ThenBy(p => p.Name);
```

Dotazovací operace (2/3)

- Řazení

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name);  
2 persons.OrderByDescending(p => p.Surname).ThenBy(p => p.Name);
```

- Omezení počtu

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name).Take(10);
```

Dotazovací operace (2/3)

- Řazení

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name);  
2 persons.OrderByDescending(p => p.Surname).ThenBy(p => p.Name);
```

- Omezení počtu

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name).Take(10);
```

- Přeskočení prvních n

```
1 persons.OrderBy(p => p.Age).ThenBy(p =>  
    p.Name).Skip(20).Take(10);
```

Dotazovací operace (2/3)

- Řazení

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name);  
2 persons.OrderByDescending(p => p.Surname).ThenBy(p => p.Name);
```

- Omezení počtu

```
1 persons.OrderBy(p => p.Age).ThenBy(p => p.Name).Take(10);
```

- Přeskočení prvních n

```
1 persons.OrderBy(p => p.Age).ThenBy(p =>  
    p.Name).Skip(20).Take(10);
```

- Poslední s vlastností

```
1 persons.Last(p => p.Age == 30);  
2 persons.LastOrDefault(p=>p.Name=="Pepa");
```

- ▶ Analogicky First, FirstOrDefault

Dotazovací operace (3/3)

- Unikátnost

```
1 persons.Distinct ();
```

Dotazovací operace (3/3)

- Unikátnost

```
1 persons.Distinct();
```

- Obsahuje nějaký prvek s vlastností

```
1 persons.Any(p=>p.Name=="Karel");
```

Dotazovací operace (3/3)

- Unikátnost

```
1 persons.Distinct ();
```

- Obsahuje nějaký prvek s vlastností

```
1 persons.Any (p=>p.Name=="Karel" );
```

- Počet prvků s vlastností

```
1 persons.Count (p => p.Age == 30);  
2 persons.Where (p=>p.Age==30) .Count ();
```

- Množinové operace

- ▶ Průnik – `persons.Intersect (anotherPersons) ;`
- ▶ Sjednocení – `persons.Union (anotherPersons) ;`
- ▶ Rozdíl – `persons.Except (anotherPersons) ;`

Úkol

- 1 Stáhnout soubor
`https://apollo.inf.upol.cz/~janostik/slides/Database.cs` a
includovat do projektu
- 2 Nejprve „po staru“ abecedně vzestupně seřadit osoby dle příjmení (Mergesort, dodělat `IComparable` a komparátor) a vypsát 6 unikátních dvojic jmen a příjmení, které jsou 3. - 8. v pořadí
- 3 Udělat totéž pomocí LINQ
- 4 Pomocí LINQ vypsát první osobu, která je v 5. ročníku či výše
- 5 Pomocí LINQ zjistit počet lidí s `OborKomb` rovnou „INF-PVS“
- 6 Pomocí LINQ zjistit, zda existuje osoba se jménem „Lukáš“
- 7 Vše vypsát do konzole v pořadí úkolů