FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

4 – LINQ

Obsah přednášky

- Language integrated query
- Operátory LINQ
- Jak LINQ funguje?
- LINQ to XML
- LINQ to SQL / LINQ to Entities
- LINQ to ...

Language integrated query

- Podpora pro dotazování se do kolekcí a zdrojů dat (SQL, XML, JSON atd.)
- LINQ byl přidán až ve verzi 3.5 .NET framework
- LINQ je typově bezpečný
- Odložené vyhodnocování
- Syntaxe:
  - Fluent syntaxe – pomocí extension metod
  - Query syntaxe

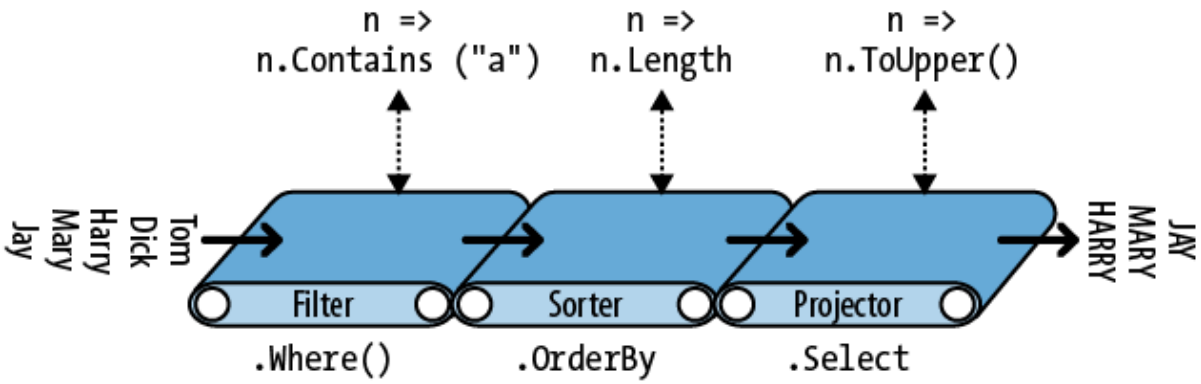
Fluent syntaxe

- Fluent interface
- Tvořeno pomocí extension metod nad `IEnumerable`
  - Vstupem `IEnumerable`
  - Specifický vstup pro jednotlivé operátory - lambda výraz / `Func`
  - Výstup `IEnumerable, scalar value`

Fluent syntaxe - příklad

```
string[] names = { "Tom", "Dick", "Harry", "Mary", "Jay" };
IEnumerable<string> query = names
  .Where(n => n.Contains("a"))
  .OrderBy(n => n.Length)
  .Select(n => n.ToUpper());

foreach (string name in query) Console.WriteLine(name);
```



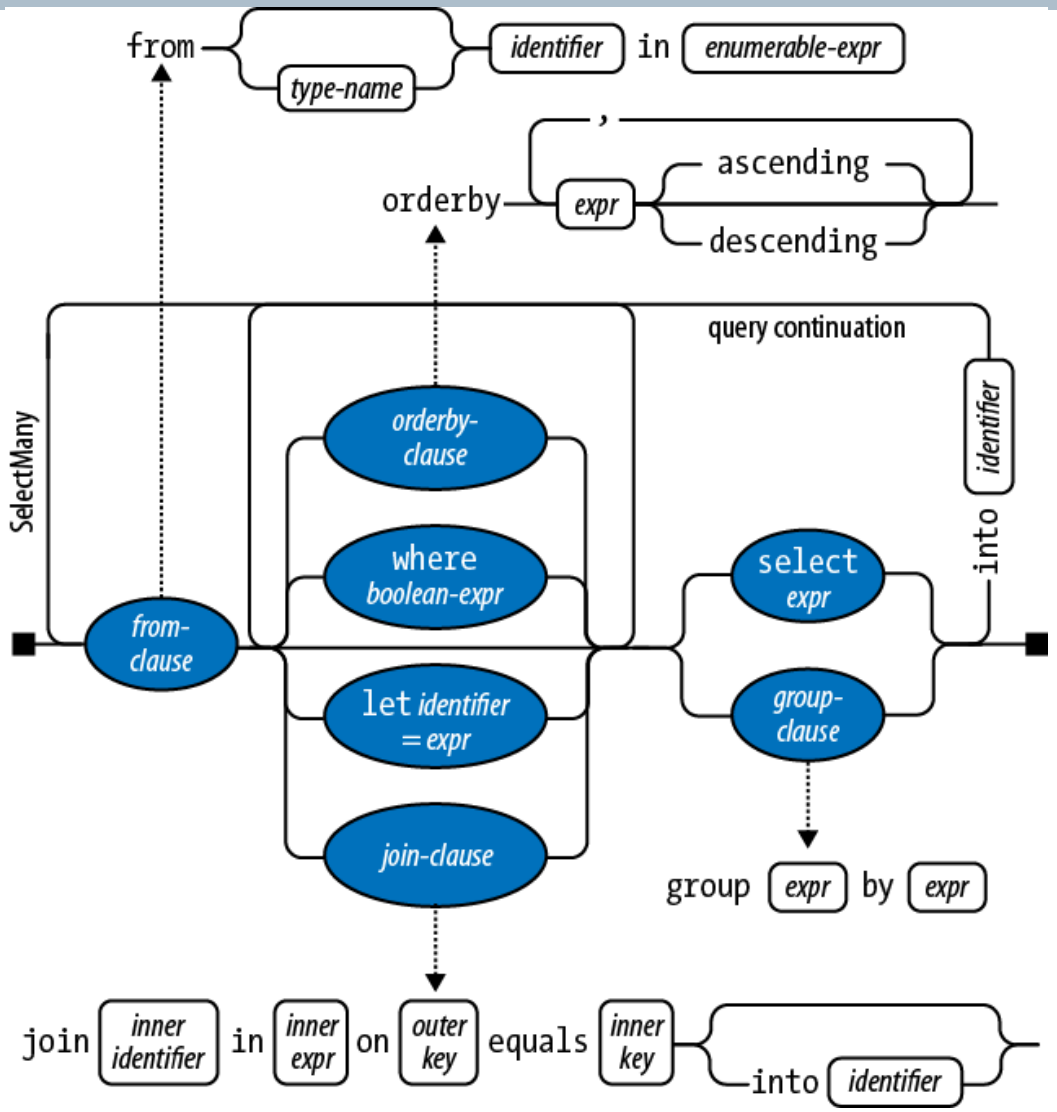
Query syntaxe

- Tvořeno pomocí klíčových slov
  - Seznam klíčových slov najdeme v dokumentaci:
  - https://msdn.microsoft.com/en-us/library/bb310804.aspx

```
string[] names = { "Tom", "Dick", "Harry", "Mary", "Jay" };

IEnumerable<string> query =
  from n in names
  where n.Contains("a") // Filter elements
  orderby n.Length // Sort elements
  select n.ToUpper(); // Translate each element (project)
```

Query syntaxe



Fluent syntax vs Query syntax

- Obě syntaxe mají své výhody i nevýhody
- Je možné je kombinovat
- **Query syntaxe** má snazší zápis v případech:
  - Při použití let pro vytvoření nové proměnné v rámci dotazu
  - Při SelectMany, Join, GroupJoin pokud potřebujeme použít vnější proměnnou
- **Fluent syntaxe** je snazší v případě:
  - Kdy používáme pouze jeden operátor

- Některé operátory existují pouze v **Fluent syntaxi**.

Poddotazy - subqueries

- LINQ dotazy je možné použít i uvnitř LINQ dotazů např. v lambda výrazu

```
var musos = { "David Gilmour", "Roger Waters", "Rick Wright",
       "Nick Mason" };

IEnumerable<string> query = musos.OrderBy(m => m.Split().Last());
var query = from m in musos
       orderby m.Split().Last()
       select m;
```

Operátory LINQ

- Operátory můžeme rozdělit dle vstupu a výstupu na tři kategorie:
  - sequence in → sequence out
  - sequence in → single element or scalar value out
  - nothing in → sequence out

- Dokumentace pro všechny LINQ operátory:

https://msdn.microsoft.com/enus/library/vstudio/system.linq.enumerable_methods(v=vs.100).aspx

LINQ operátory

- **Filtrování**

Metoda	Popis
Where	Vrací podmnožinu splňující danou podmínku
Take	Vrací podmnožinu o daném počtu prvků
Skip	Přeskočí prvních x prvků a vrací zbývající prvky
TakeWhile	Vrací prvky dokud splňují danou podmínku
SkipWhile	Přeskakuje prvky dokud je splňena daná podmínka a vrací zvývající prvky
Distinct	Vrátí podmnožinu, ve které nejsou duplicitní objekty

LINQ operátory

- **Projekce**

Metoda	Popis
Select	Transformuje každý prvek na vstupu
SelectMany	Transformuje každý prvek na vstupu a provede konkatenaci vstupních kolekcí

- **Joining**

Metoda	Popis
Join	Spojí dvě sekvence do ploché struktury
GroupJoin	Spojí dvě sekvence do hierarchické struktury
Zip	Prochází obě sekvence a provádí funkci na prvkcích z obou sekvencí

LINQ operátory

- **Řízení**

Metoda	Popis
OrderBy, ThenBy	Seřadí sekvenci vzestupně
OrderByDescending, ThenByDescending	Seřadí sekvenci sestupně
Reverse	Otočí pořadí sekvence

- **Seskupování - Grouping**

Metoda	Popis
GroupBy	Seskupí sekvenci do podsekvencí

LINQ operátory

- **Množinové operace**

Metoda	Popis
Concat	Spojí dvě sekvence – druhou dá za první
Union	Spojí dvě sekvence s vynecháním duplicitních prvků
Intersect	Vrací prvky obsažené v obou sekvencích
Except	Vrací prvky obsažené v první, ale ne v druhé sekvenci

LINQ operátory

- **Konverze**

Metoda	Popis
OfType	Převede typ prvků sekvence, vynechává chybně
Cast	Převede typ prvků sekvence, vyvolá výjimku v případě chyby
ToArray	Převede IEnumerable<T> na pole T[]
ToList	Převede IEnumerable<T> na List<T>
ToDictionary	Převede IEnumerable<T> na IDictionary<Tkey, TElement>
ToLookup	Převede IEnumerable<T> na ILookup<Tkey, TElement>
AsEnumerable	Provede downcast na IEnumerable<T>
AsQueryable	Provede přetypování nebo konverzi na IQueryable<T>

LINQ operátory

- **Element operators**

Metoda	Popis
First, FirstOrDefault	Vrací první element sekvence
Last, LastOrDefault	Vrací poslední element sekvence
Single, SingleOrDefault	Vrací první element sekvence, vyvolá výjimku vpřípadě, kdy kolekce obsahuje více prvků
ElementAt, ElementAtOrDefault	Vrací prvek na dané pozici
DefaultIfEmpty	Vrací sekvenci s jedním prvkem default(T) pokud je vstupní sekvence prázdná, jinak vrací původní sekvenci

LINQ operátory

- **Agregace**

Metoda	Popis
Count, LongCount	Vrací počet prvků v sekvenci
Min, Max	Vrací minimální / maximální prvek ze sekvence
Sum, Average	Spočítá sumu / průměr prvků v sekvenci
Aggregate	Spustí námi specifikovanou agregaci

- **Podmínky**

Metoda	Popis
Contains	Vrací true pokud sekvence obsahuje daný prvek
Any	Vrací true pokud min. jeden prvek splňuje danou podmínku
All	Vrací true pokud všechy prvky splňují danou podmínku
SequenceEquals	Vrací true pokud je sekvence shodná s druhou sekvencí

LINQ operátory

- **Generování**

Metoda	Popis
Empty	Vytvoří prázdnou sekvenci
Repeat	Vytvoří sekvenci opakující daný prvek
Range	Vytvoří sekvenci celočíselných hodnot

Jak LINQ funguje?

- Ukázka toho jak lze vytvořit vlastní implementaci LINQ:
  - http://codeblog.jonskeet.uk/category/eduling/

- Jak naimplementovat např. **Where**?

```
public static IEnumerable<TSource> Where<TSource>(
        this IEnumerable<TSource> source, Func<TSource, bool> predicate)
{
  //... Check of input parameters ...
  foreach (TSource item in source)
  {
    if (predicate(item))
    {
      yield return item;
    }
  }
}
```

LINQ to XML

- Přístup k XML v .NET
  - `XmlReader` / `XmlWriter` – čte / zapisuje přímo do streamu
  - `XmlDocument`
  - LINQ to XML

- Načtení XML pomocí **LINQ to XML**

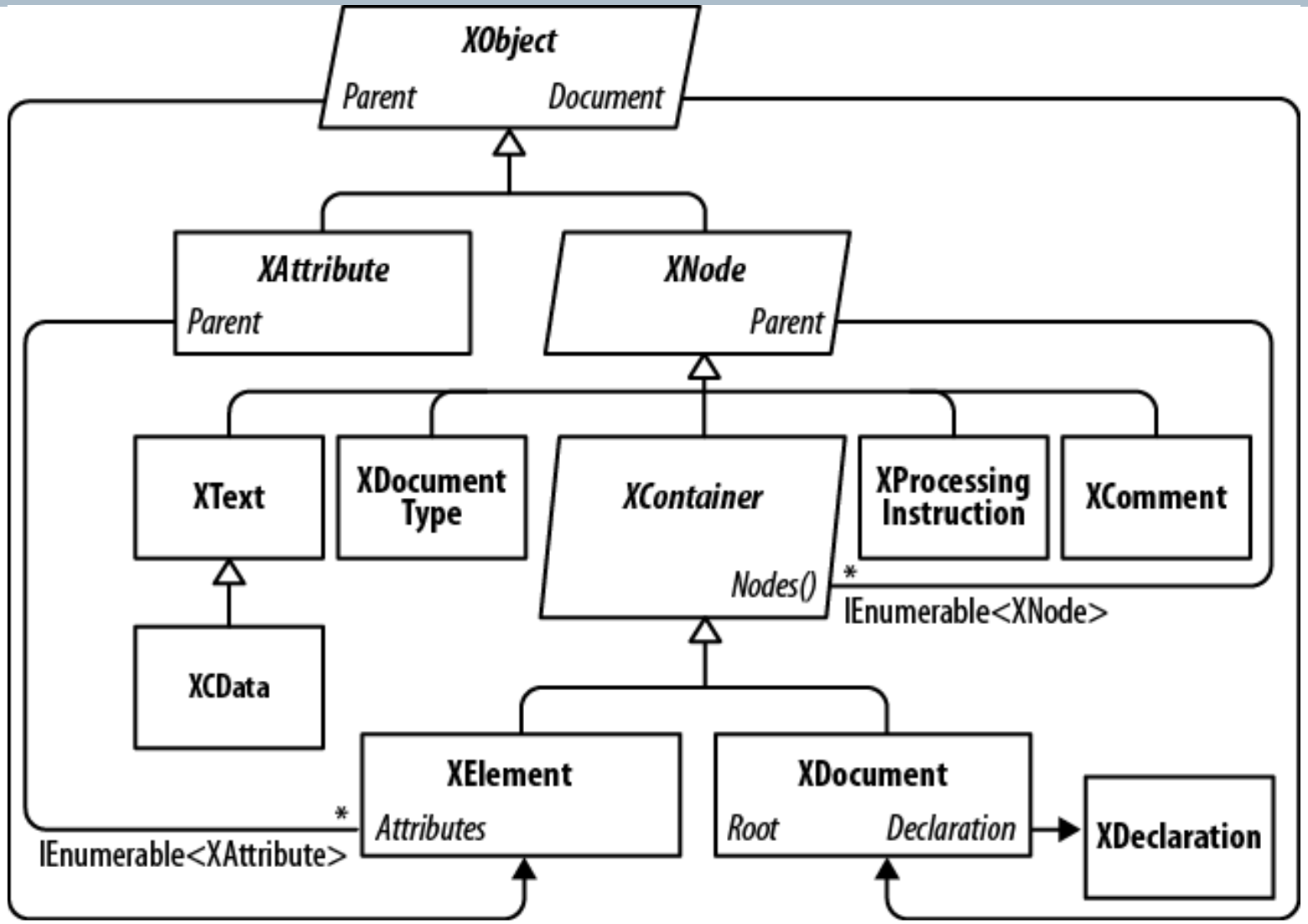
```
var xmlDocument = XDocument.Load("Customers.xml")
```

- Vytvoření XML pomocí **LINQ to XML**

```
var element = new XElement ("customer",
    new XAttribute ("id", 123),
    new XElement ("firstname", "joe"),
    new XElement ("lastname", "bloggs",
    new XComment ("nice name"))
);
```

```
<customer id="123">
 <firstname>joe</firstname>
  <lastname>bloggs
<!--nice name--></lastname>
</customer>
```

LINQ to XML – přehled tříd



LINQ to SQL, LINQ to Entities (Entity Framework)

- ORM – Objektově relační mapování
- Převádí LINQ dotazu na SQL dotaz
- **LINQ to SQL**
  - Vygenerované třídy pro přístup k databázi
- **Entity Framework**
  - Možnost vygenerování tříd pro přístup k databázi
  - Vlastní třídy namapované na tabulky v databázi
  - Code first přístup

LINQ to ...

- PLINQ – Paralel LINQ
- LINQ to NHibernate
- LINQ to Sharepoint
- LINQ to ActiveDirectory
- LINQ to JSON
- LINQ to SNMP
- LINQ to Wikipedia
- LINQ to Twitter
- LINQ to FQL (Facebook Query Language)
- ...

Reference

- http://www.amazon.com/5-0-Nutshell-The-Definitive-Reference/dp/1449320104
- https://msdn.microsoft.com/en-us/library/bb310804.aspx
- https://msdn.microsoft.com/enus/library/vstudio/system.linq.enumerable_methods(v=vs.100).aspx
- http://codeblog.jonskeet.uk/category/edulinq/
- https://code.msdn.microsoft.com/101-LINQ-Samples-3fb9811b
- http://en.wikipedia.org/wiki/Language_Integrated_Query