Language-Integrated Query
LINQ

Katedra za informatiku
Fakultet tehničkih nauka
Univerzitet u Novom Sadu

Uvod

- LINQ je jezik za upite nad podacima
- Integrisan je u .NET kao deo sintakse jezika
- LINQ univerzalnom sintaksom omogućuje upite nad različitim izvorima podataka:
  - kolekcijom u memoriji
  - relacionom bazom
  - XML dokumentom
  - tokom podataka
  - ...

LINQ upit

- LINQ upit
  1. Definisanje izvora podataka
  2. Kreiranje upita
  3. Izvršavanje upita
- Primer

```
//definisanje izvor podataka
int[] nums = new int[] {2, 0, 1, 6};
//kreiranje upita
var res = from a in nums where a < 3
                  orderby a select a;
//izvršavanje upita
foreach (int i in res)
   Console.WriteLine(i);
```

LINQ upit

- Izvor podataka – "*queryable type*" - bilo koji tip koji implementira interfejs **`IEnumerable<T>`** ili od njega izvedeni interfejs **`IQueryable<T>`**
- Definisanje upita – koje podatke želimo iz izvora podataka i kako da ti podaci budu organizovani
- Upit sadrži delove:
  - **`from`** – izvor podataka
  - **`where`** – filtriranje podataka
  - **`group`** – grupisanje podataka
  - **`join`** – veze između podataka koje nisu eksplicitno modelovane u izvoru podataka
  - **`select`** – tip elemenata koji su rezultat upita
- Upit se izvršava odloženo
  - pri korišćenju promenljive koja predstavlja rezultat upita
  - svako korišćenje promenljive inicira izvršavanje novog upita – uvek se dobijaju aktuelni podaci iz izvora podataka

Povratna vrednost upita

- Povratna vrednost je **`IEnumerable<T>`** ili **`IQueryable<T>`**
- **`Primer`**

```
int[] nums = new int[] {2, 0, 1, 6};
IEnumerable<int> res = from a in nums
   where a < 3 orderby a select a;
```

- promenljiva **`res`** će nakon izvršavanja upita sadržati sekvencu vrednosti tipa **`int`**
- Kompajler može automatski da odredi tip korišćenjem ključne reči **`var`**

```
var res = from a in nums
   where a < 3 orderby a select a;
```

Struktura LINQ upita

- from
  - sadrži izvor podataka i "*range*" promenljivu
  - *"range"* promenljiva – redom ukazuje na svaki element u izvoru podataka. Koristi se za referenciranje elemenata u izvoru podataka u drugim delovima upita
- where
  - sadrži logički izraz
  - rezultat upita sadrži samo one elemente iz izvora podataka za koje je definisani logički izraz istinit

Struktura LINQ upita

- select
  - specificira objekte koji su rezultat upita (npr. element kolekcije definisane u **`from`** delu, jedan atribut elementa, podskup atributa elementa, ili neka nova promenljiva nastala kao rezultat izračunavanja u upitu)
  - Primer (preuzimanje jednog atributa elementa)

```
var rez = from s in studenti select s.Ime;

foreach (string ime in rez)
  Console.WriteLine(ime);
```

Struktura LINQ upita

- orderby
  - sadrži elemente po kojima se kolekcija sortira i smer sortiranja – **`ascending`** ili **`descending`**
  - Primer

```
var res = from a in nums where a < 3
 orderby a descending select a;
```

Struktura LINQ upita

- group
  - specificira koji element se grupiše po kojem kriterijumu
  - ako se u ostatku upita vrše operacije nad grupom, grupa se može imenovati
  - Primer

```
//studenti je ranije kreiran objekat tipa
   //List<Student>
var rez = from s in studenti
   group s by s.Prezime into g select g;

foreach (IGrouping<string, Student> g in rez) {
   Console.WriteLine(g.Key);
    foreach(Student s in g)
       Console.WriteLine("\t" + s.Ime);
}
```

Struktura LINQ upita

- join
  - navodi se kolekcija sa kojom se vrši spajanje i "*range*" promenljiva za kolekciju
  - Korišćenjem ključne reči **on** definiše se uslov spajanja kolekcija
  - Primer

```
//studenti i gradovi su ranije kreirani
// objekti tipa List<Student> i List<Grad>
var rez = from s in studenti
  join g in gradovi on s.Grad equals g.Naziv
  select new {Ime = s.Ime,Prezime=s.Prezime,
              PttGrada = g.PttBroj};

  foreach (var r in rez)
    Console.WriteLine(r.Ime + " " +
        r.Prezime + " " + r.PttGrada);
```