



UPPSALA
UNIVERSITET

. NET-programmering

LINQ



- Language-Integrated Query
- Definierar ett antal extension methods
- Syntaktiskt socker
- Förenklar dataåtkomst
- "Ersätter" bl.a. loopar
- Samma princip oavsett datakälla (t.ex. array/fil/databas)



UPPSALA
UNIVERSITET

LINQ

Finns några olika varianter, bl.a:

- LINQ to Objects
- LINQ to XML
- LINQ to SQL



UPPSALA
UNIVERSITET

1. Hämta datakällan

2. Skapa en fråga

3. Exekvera frågan

Processen

// The Three Parts of a LINQ Query:

// 1. Data source.

```
int[ ] numbers = new int[7] { 0, 1, 2, 3, 4, 5, 6 };
```

// 2. Query creation.

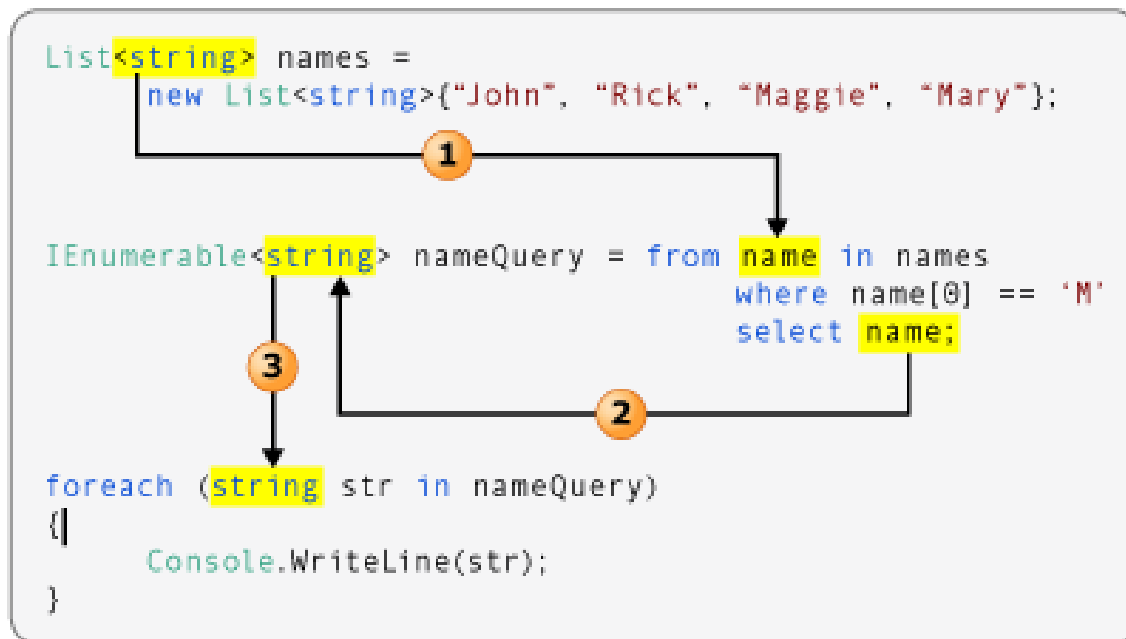
// numQuery is an IEnumerable<int>

```
var numQuery =  
    from num in numbers  
    where (num % 2) == 0  
    select num;
```

// 3. Query execution.

```
foreach (int num in numQuery)  
{  
    Console.WriteLine("{0,1} ", num);  
}
```

Ingen transformering av datat





Transformering av data

```
Table<Customer> Customers = db.GetTable<Customers>();  
  
IQueryable<string> custNameQuery =  
    from cust in Customers  
    where cust.City == "London"  
    select cust.Name;  
  
foreach (string str in custNameQuery)  
{  
    Console.WriteLine(str);  
}
```

Diagram illustrating the transformation of data:

- 1. Data is retrieved from the `Customers` table.
- 2. The data is transformed into an `IQueryable<string>` query (`custNameQuery`) using LINQ.
- 3. The transformed data is iterated over using a `foreach` loop to process each string.



Anonym typ

```
Table<Customer> Customers = db.GetTable<Customers>();
```

1

```
var namePhoneQuery =  
    from cust in Customers  
    where cust.City == "London"  
    select new { name = cust.Name,  
                phone = cust.Phone };
```

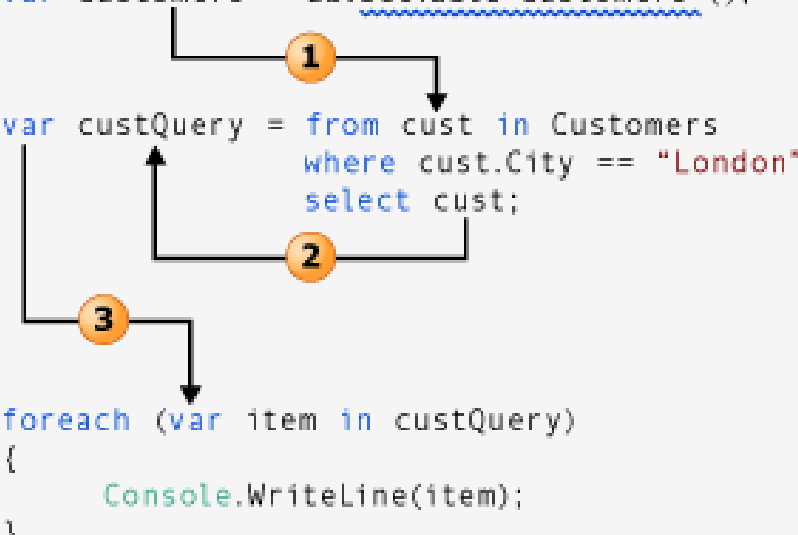
2

3

```
foreach (var item in namePhoneQuery)  
{  
    Console.WriteLine(item);  
}
```

Låt kompilatorn göra jobbet

```
var Customers = db.GetTable<Customers>();  
var custQuery = from cust in Customers  
                 where cust.City == "London"  
                 select cust;  
foreach (var item in custQuery)  
{  
    Console.WriteLine(item);  
}
```



1

2

3



IEnumerable<T>

- Flesta klasser och interfaces inom Collections ärver av denna
- Enkel hämtning av data och skickar vidare
- Hämtar hela tabellen och går sedan igenom den
- Fungerar bäst med interna källor

IQueryable<T>

- Ärver av IEnumerable<T>
- Skapar ett SQL anrop
- Fungerar bäst med externa källor



LINQ

```
int[] myNumbers = {1, 2, 3, 4, 5};
```

```
var myEvenNumbers =  
    from n in myNumbers  
    where n % 2 == 0  
    orderby n descending  
    select n;
```



```
int[] myNumbers = {1, 2, 3, 4, 5};  
  
var myEvenNumbers =  
    myNumbers  
        .Where(n => n%2 == 0)  
        .OrderByDescending(n => n)  
        .Select(n => n);
```



UPPSALA
UNIVERSITET

Nyckelord

- From
- Where
- Select
- Group (by)
- Into
- Orderby (ascending, descending)
- Join (in, on, equals)



UPPSALA
UNIVERSITET

Uttryck

NamnPåSamling =

from *itemAlias* **in** *datasource*

hämta med eller utan sortering,
filtrering, gruppering



Exempel

- Enkel lista med studenter

```
var students = backend.getStudentList();  
var matches =  
    from student in students  
    where student.LastName.StartsWith("D")  
    select student;
```

(Nyckelorden i LINQ är markerade)

- Listan **matches** kan sedan användas för olika ändamål, som att visa upp i GUI



UPPSALA
UNIVERSITET

Hämta data (select)

```
var matches =  
    from student in students  
    select student.FirstName;
```

```
var matches =  
    from student in students  
    select student.FirstName +  
        student.LastName;
```



UPPSALA
UNIVERSITET

Filtrering (where)

```
var matches =  
  from student in students  
  where student.LastName.StartsWith("D")  
  select student;
```

&&, ||, <, <=, >, >= kan också användas



UPPSALA
UNIVERSITET

Sortering (orderby)

```
var matches =  
    from student in students  
    orderby student.LastName,  
             student.FirstName  
    select student;
```

Vi kan också skriva descending för att få motsatt ordning

```
orderby student.LastName descending
```