LINQ

Language INtegrated Query

Веселин Георгиев

Съдържание

- Анонимни типове
- Анонимни методи
- Ламбда изрази
- Extension методи
- LINQ
  - Синтаксис
  - Разновидности
  - Изпълнение
  - Оператори
- LINQ To SQL

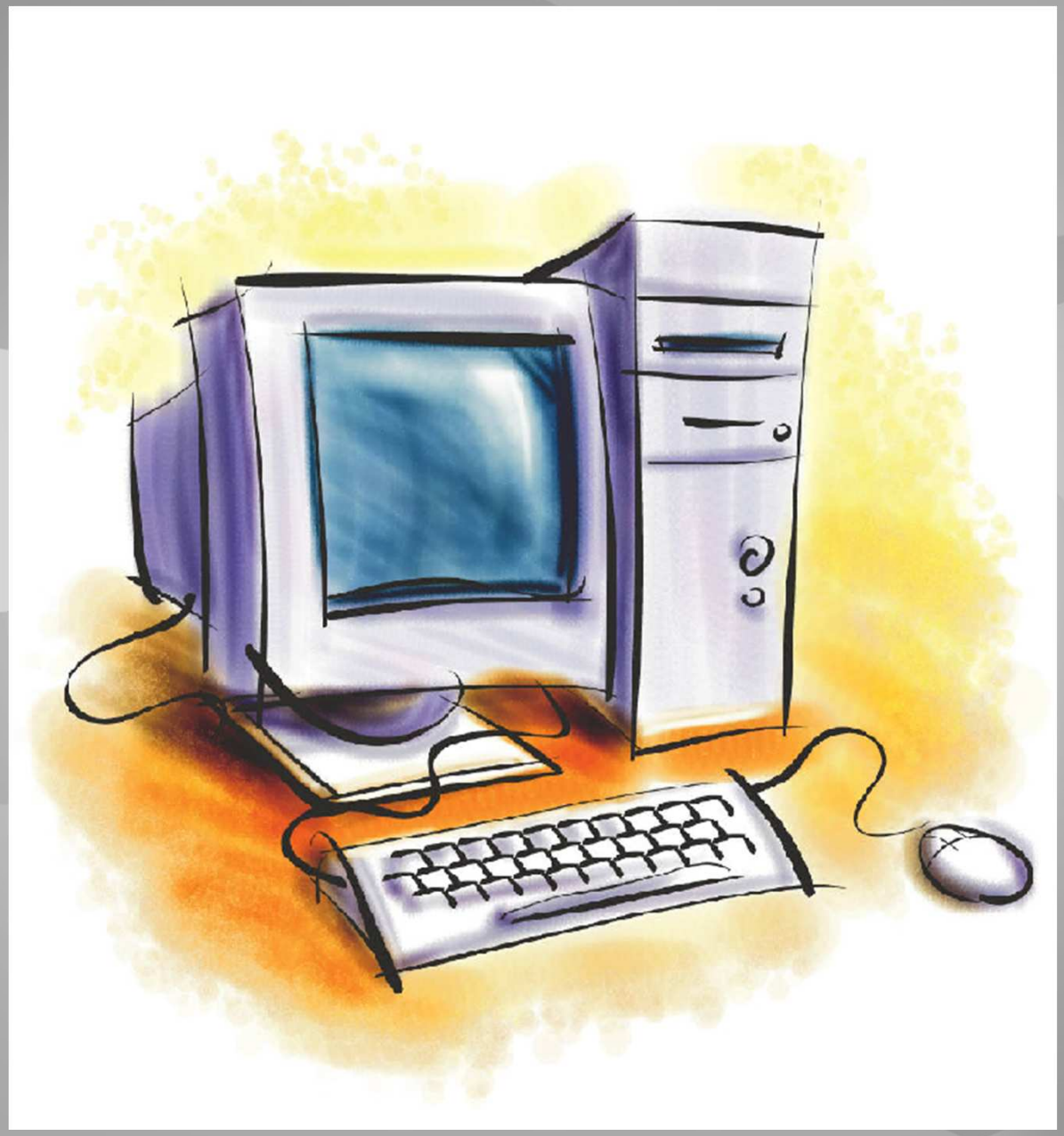
Анонимни типове

- Типове декларирани без име
- Декларират се директно на мястото, където ще се използват
- Синтактично улеснение
- Съдържа само readonly properties
- Референтен тип, наследява директно Object

```
var student = new { Name = "Joe", Age = 22 };
```

Анонимни типове

- При компилация се генерира стандартен клас с име избрано от компилатора
- Името е програмно недостъпно
- Използват се когато имаме нужда от клас за "еднократна употреба" с определени полета за данни и без логика
- Проблемни при нужда да върнем анонимен тип като резултат от метод (cast to object first)



Делегати

- Указатели към методи в .NET
- Single cast и multicast делегати
- Делегатите инстанцирани с delegate са multicast делегати
- Силно типизирани – задава се сигнатурата на методите към които могат да сочат

```
public delegate void Alert(string message);
```

- Характерен за C# 1.0

Анонимни методи

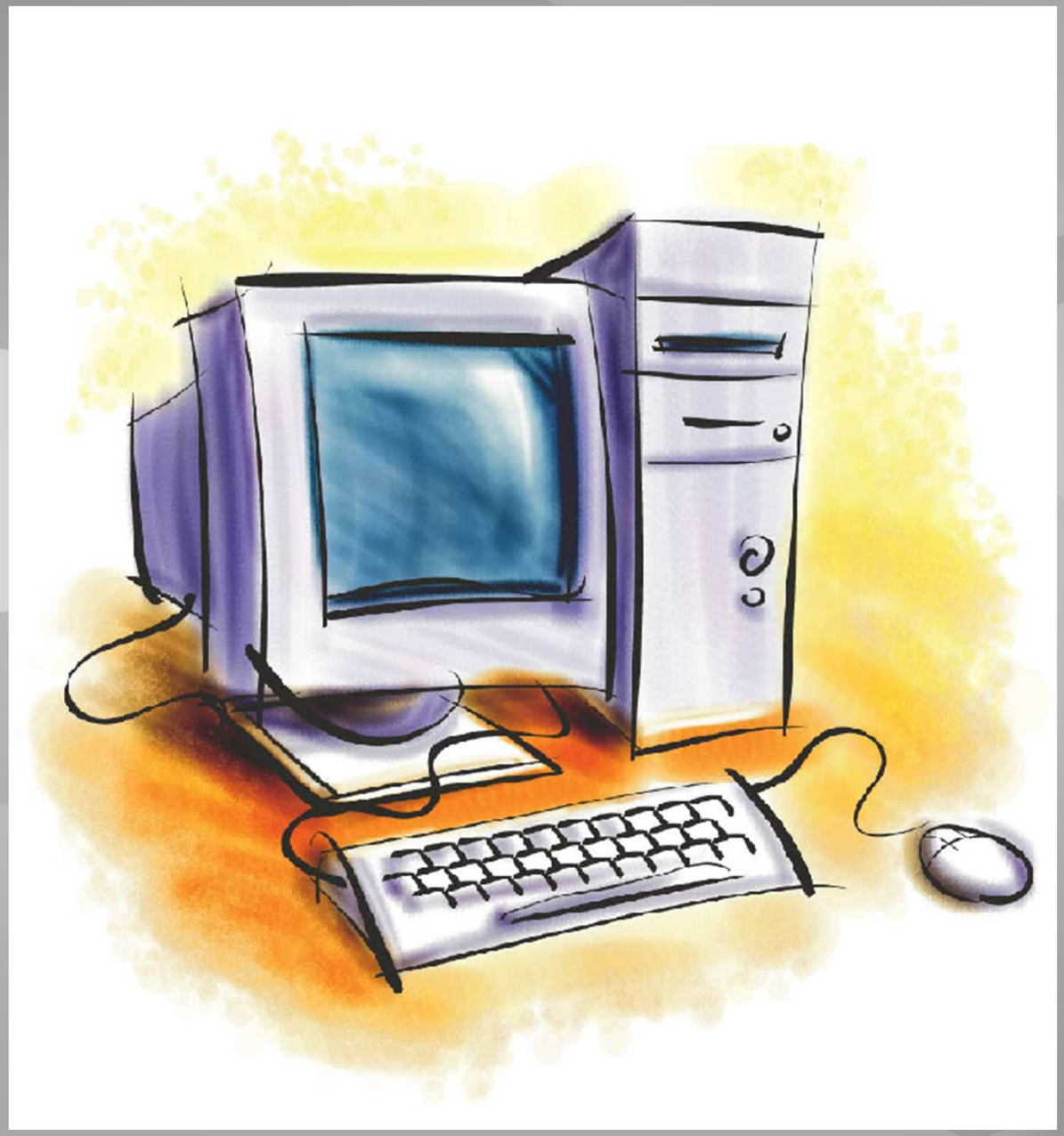
- Синтактично улеснение за създаване на делегат
- Създава делегат, който сочи към метод с определена функционалност
- Делегат "за еднократна употреба"
- Характерен за C# 2.0

```
Alert a = delegate(string message)
{
  Console.WriteLine(message);
};
```

Ламбда изрази

- Израз, който връща делегат
- Синтактично улеснение
- Типовете на параметрите и на върнатата стойност се определят автоматично при компилация
- Характерен за C# 3.0

```
Alert a = (message) => Console.WriteLine(message);
```

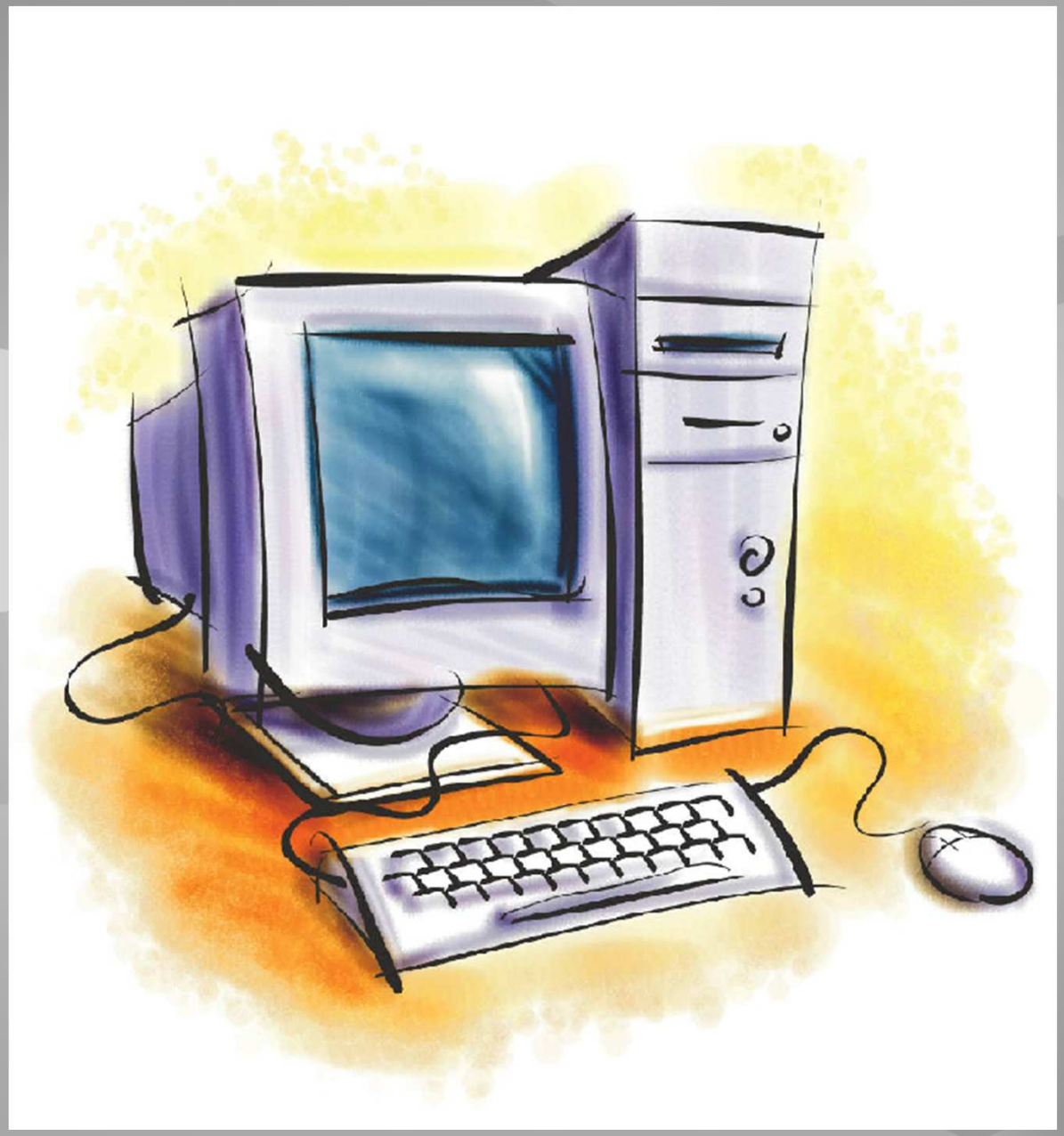


Extension методи

- Добавят нови методи на вече съществуващи класове
- Декларират като статични
- CLR се грижи да открие нужния метод
- Първият им параметър е this [тип]
  - Тип е типа на разширяваният клас

```
Public static void Cut(this string myString) {}
```

- Трябва да се използват внимателно



LINQ

- LINQ – Language INtegrated Query
- Множество от технологии
- Унифициран език за заявки към източник на данни
- Независим от източника на данни
- Вграден в .NET Framework 3.0
- Поддържа се от C# и Visual Basic
- Винаги работим с обекти
- Лесна манипулация на колекции
- ...

Предимства на LINQ

- Compile-time проверка на типовете
- IntelliSense
- Един език – много източници на данни
- Лесен и интуитивен за използване
- ...

Разновидности

- LINQ to Objects
- LINQ to SQL
- LINQ to Entities
- LINQ to XML
- LINQ to SharePoint
- LINQ to ADO.NET Data Services
- LINQ to DataSet
- …

LINQ Синтаксис

- Декларативен синтаксис (като SQL заявка)

```
int[] numbers = new int[] { 5, 18, 97, 92, 81, 60 };

var even = from num in numbers
           where num % 2 == 0
           select num;
```

- Синтаксис с извикване на методи

```
var even = numbers.Where(x => x % 2 == 0)
                  .Select(x => x);
```

Синтаксис на LINQ

- LINQ има специален синтаксис, интегриран с програмния език
- Изразите се преобразуват до извиквания на **extension методи** и **ламбда изрази**
- Като резултат получаваме **IEnumerable<T>** или **IQueriable<T> (queryable types)**
- CLR конструира **expression trees** за изразите

Синтаксис на LINQ

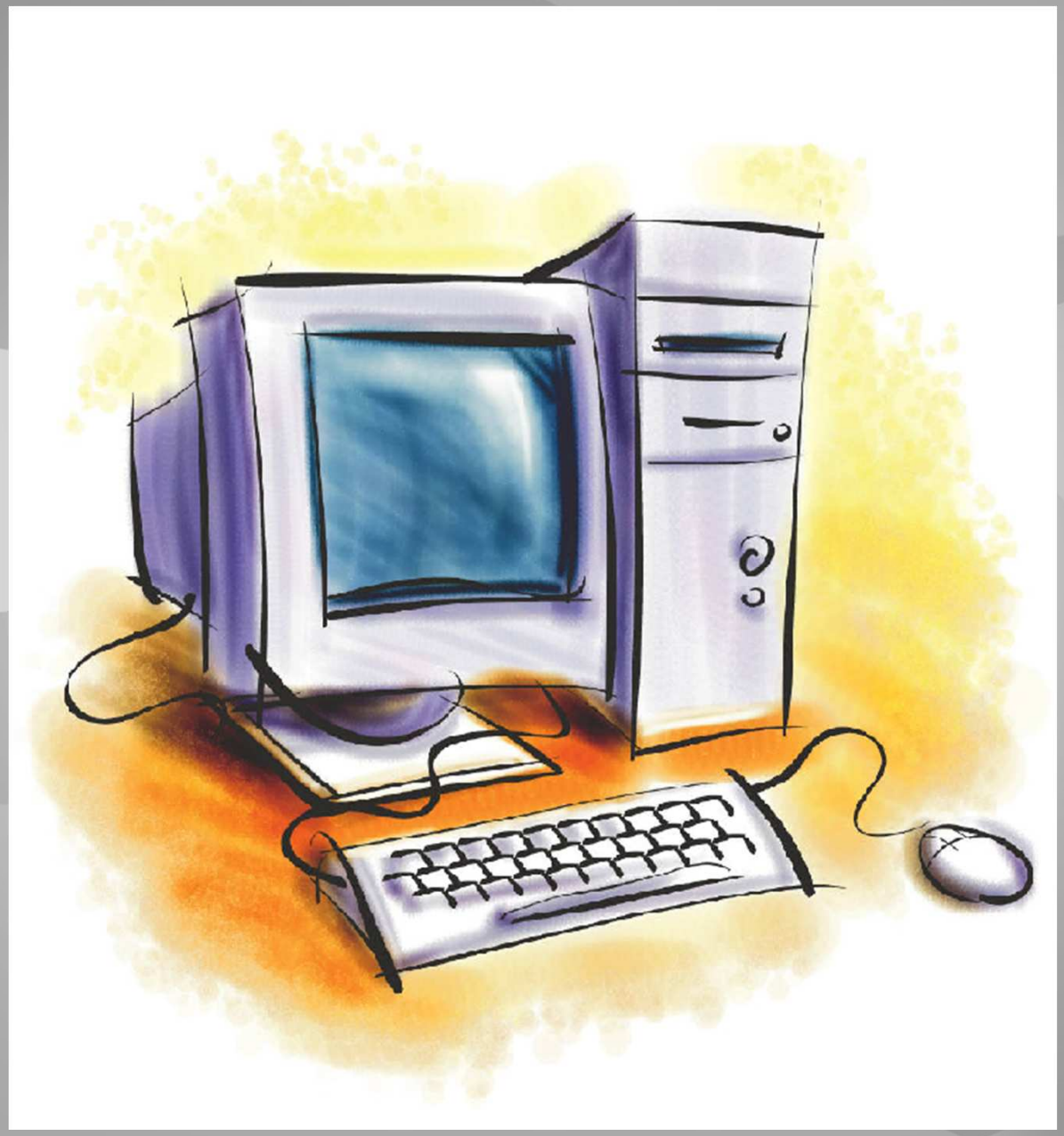
- Deferred loading (lazy loading) – данните **не се зареждат до първо поискване на конкретна стойност**
- Оператори
  - Отложени (deferred) - променят дървото от изрази (expression tree)
  - Незабавни (non-deferred) - предизвикват изчисляване на стойността на дървото от изрази

Deferred Operators

- Join
- OfType
- OrderBy
- OrderByDescending
- Range
- Repeat
- Reverse
- Select
- SelectMany
- Skip
- SkipWhile
- Take
- AsEnumerable
- AsQueryable
- Cast
- Concat
- DefaultIfEmpty
- Distinct
- Empty
- Except
- GroupBy
- GroupJoin
- Intersect

Non Deferred Operators

- Aggregate
- All
- Any
- Average
- Contains
- Count
- ElementAt
- ElementAtOrDefault
- First
- FirstOrDefault
- Last
- LastOrDefault
- LongCount
- Max
- Min
- SequenceEqual
- Single
- SingleOrDefault
- Sum
- ToArray
- ToDictionary
- ToList
- ToLookup



Филтриране

- Избор на елементи по дадено условие
  - where

```
var positiveEven = from num in numbers
                   where num > 0 && num % 2 == 0
                   select num;
```

  - Where()

```
var positiveEven =
      numbers.Where(x => x > 0 && x % 2 == 0);
```

Проекция

- Прилагаме операция върху всеки елемент
  - select

```
var squares = from num in numbers
              select num * num;
```

  - Select()

```
var squares = numbers.Select(x = > x * x);
```

Сортиране

- Сортира елементите по критерий
  - orderby

```
var sortedNumbers = from num in numbers
                    orderby num descending
                    select num;
```

  - OrderBy()

```
var sortedNumbers =
      numbers.OrderByDescending(x => x);
```

Групиране

- Образуване на групи по зададен критерий
  - group by

```
var numberGroups = from n in numbers
                   group n by n % 2 into g
                   select new
                   {
                        Remainder = g.Key,
                        Numbers = g
                   };
```

Оператори за агрегиране

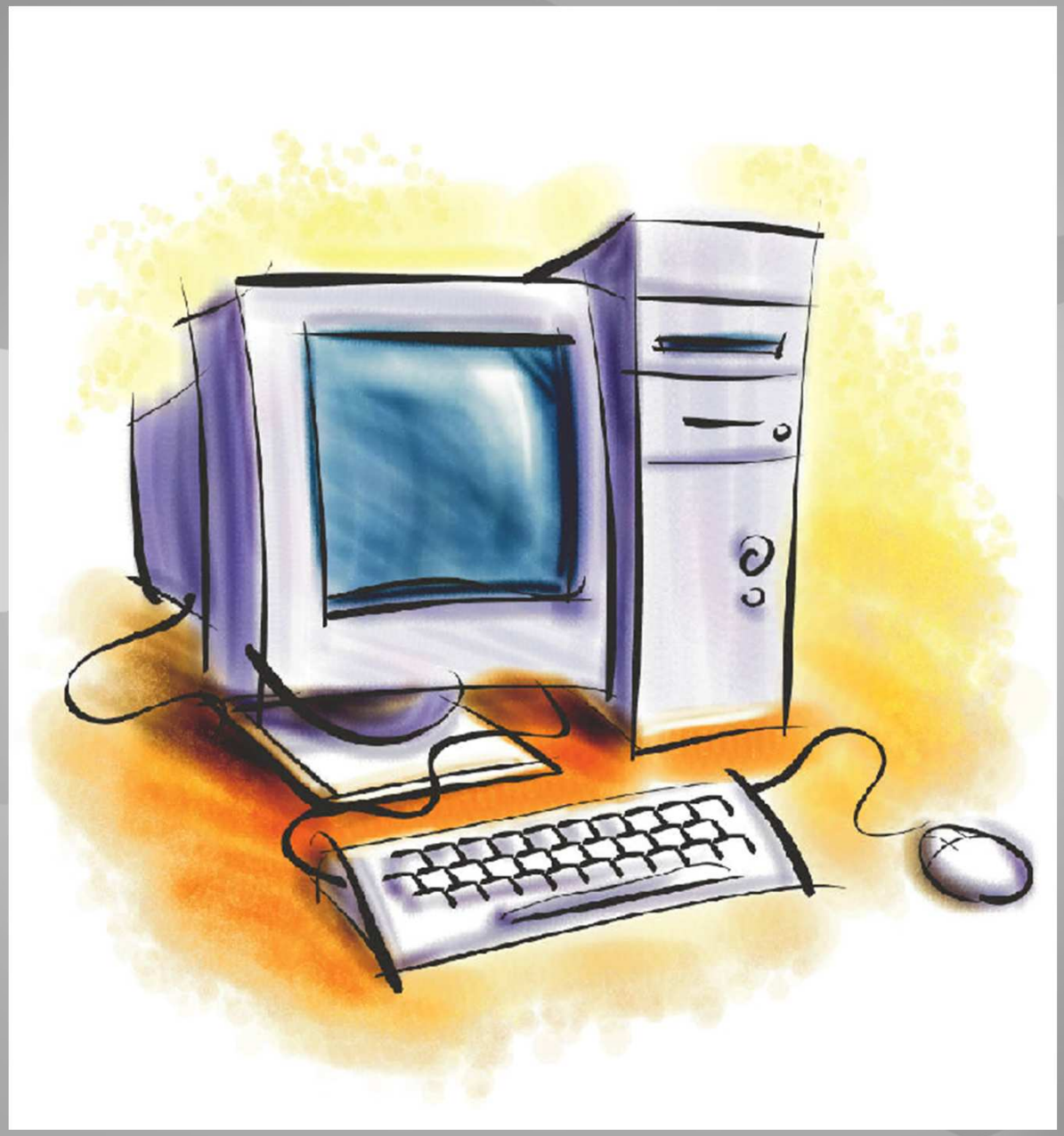
- Count
- Sum
- Min
- Max
- Average
- …

```
int sumOfSquares = numbers.Sum(x => x * x);
```

Оператори за преобразуване

- Преобразуват IEnumerable<T> или IQueriable<T> до нужния ни тип
- **Предизвикват изчисляване на израза**
- ToArray()
- ToList()
- ToDictionary()

```
int sumOfSquares = numbers.Sum(x => x * x).ToList();
```



LINQ to SQL

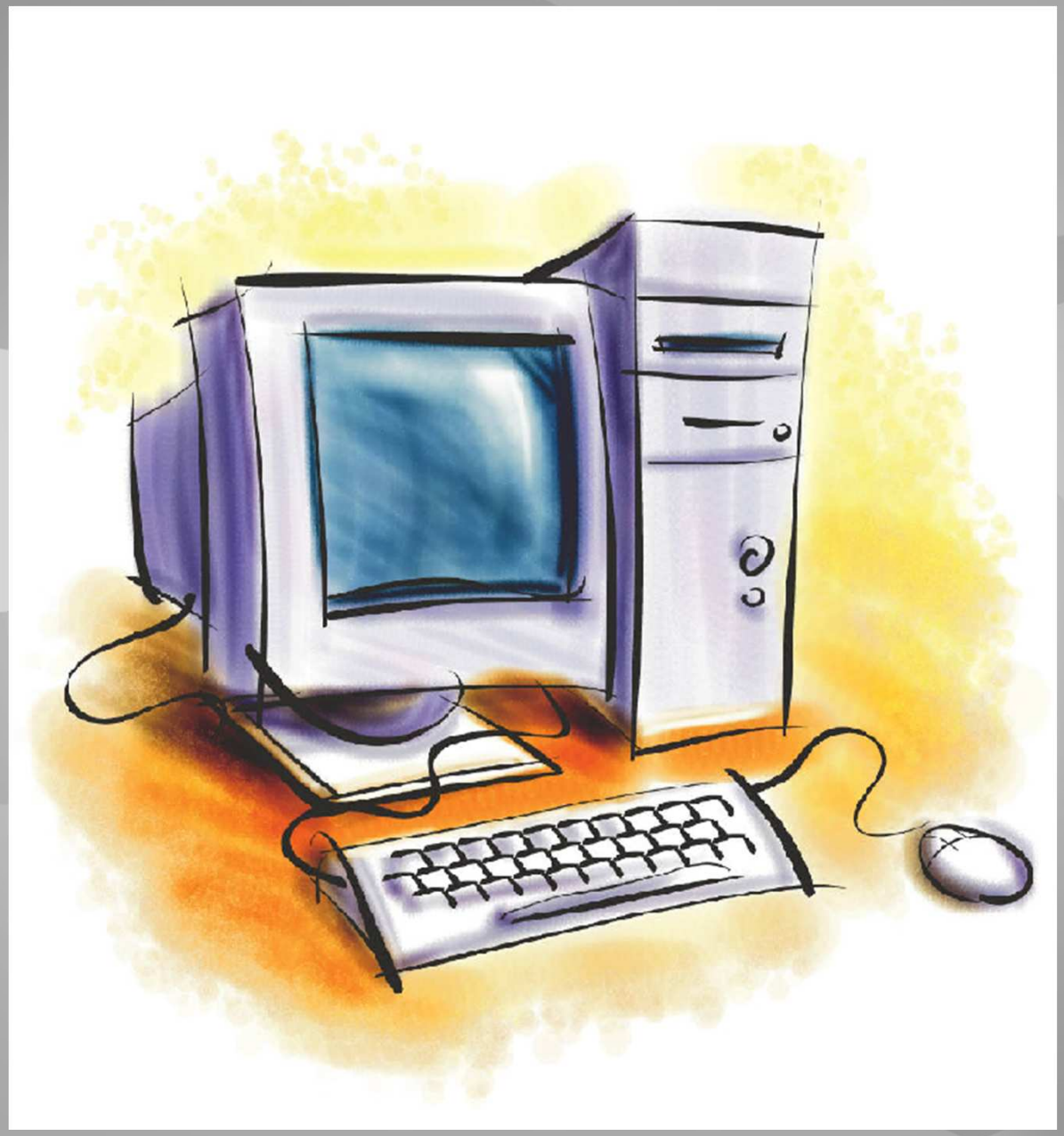
- ORM framework създаден специално за интеграция с LINQ
- Генерира класове съответстващи на SQL entities
- Изграден е на основата на декларативно задаване на връзки към базата данни в xml файлове

Data Context

- Абстракция на множеството от данни
- Усигурява достъп до таблиците и процедурите в базата
- Генерира се типизиран DataContext клас, въз основа на xml файл
- За достъп се инстанцира нов DataContext
- При промяна на данните е нужно да се извика SubmitChanges() на целия context обект

LINQ to SQL файлове

- DBML – DataBase Markup Language
- DBML съдържа информацията за типовете
- DBML е xml файл
- dbml.cs
  - Съдържа дефиниции на ORM класовете
  - Генерира се въз основа на xml-а
  - Съдържа connection string за връзка с базата



LINQPad

LINQPad

File Edit Query Help

Add connection

Nutshell.mdf in <ApplicationData>\L

My Queries | Samples

- C# 3.0 in a Nutshell
  - Before You Start
    - Readme.First()
    - Populate Demo Databas
    - A Note on AsQueryable
    - Query syntax diagram
  - Chapter 8 - LINQ Queries
    - Getting Started
      - Simple Filtering
      - Extension Methods
      - Basic Comprehensio
    - Introducing Lambda Qu
    - Introducing Compreher
      - A Basic Query
      - A Basic Query - Tran
      - Mixing Syntax
      - Query Syntax in its E
    - Deferred Execution
      - Introduction
      - Reevaluation
      - Defeating Reevaluati

Simple Filtering | Basic Comprehension Query | A Basic Query | Inside the foreach Statement | Multiple statements | Downloading update...

Language: C# Statement(s) | Database: <None> | Use Nutshell.mdf

```
// Setting the query language to "C# Statement(s)" permits multiple statements:

var words =
    from word in "The quick brown fox jumps over the lazy dog".Split()
    orderby word.ToUpper()
    select word;

var duplicates =
    from word in words
    group word.ToUpper() by word.ToUpper() into g
    where g.Count() > 1
    select new { g.Key, Count = g.Count() };

// The Dump extension method writes out queries:

words.Dump();
duplicates.Dump();

// Notice that we do need semicolons now!
```

Results | λ | SQL | IL

IOrderedEnumerable<String> (9 items)

brown
dog
fox
jumps
lazy
over
quick
The
the

IEnumerable<> (1 item)

Key	Count
THE	2

Query successful (00:00.094)

Въпроси?

Полезни линкове

- http://msdn.microsoft.com/en-us/library/bb397696.aspx
- http://msdn.microsoft.com/en-us/library/0yw3tz5k(VS.80).aspx
- http://msdn.microsoft.com/en-us/library/bb397687.aspx
- http://msdn.microsoft.com/en-us/library/bb383977.aspx

Полезни линкове

- http://msdn.microsoft.com/en-us/library/bb397933.aspx - Getting started with LINQ
- http://msdn.microsoft.com/en-us/vcsharp/aa336746.aspx - 101 examples with LINQ
- http://www.linqpad.net/ - awesome tool ☺