Entity Framework

Slides Roadmap

- Intro Linq
- Lambda expressions
- Intro Entity Framework

Linq: Language Integrated Query

- Language **extensions** that enable **query operations** over data

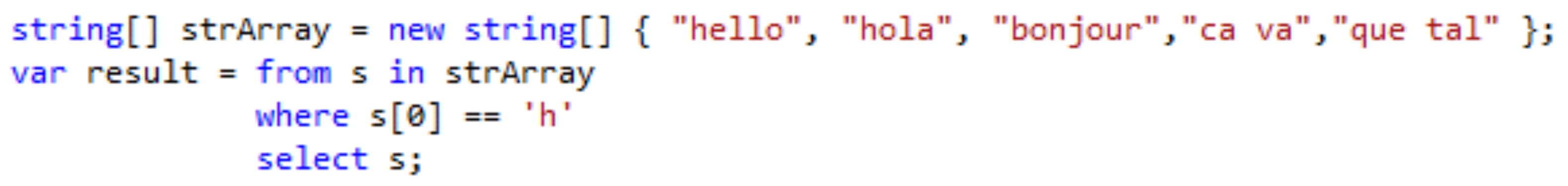
Linq: Language Integrated Query

- **System.Core.dll** : IEnumerable<T>
- **System.Data.DataExtensions.dll** LINQ for DataSet
- **System.Data.Linq.dll** : LINQ for SQL Server
- **System.Xml.Linq.dll** : LINQ for XML.

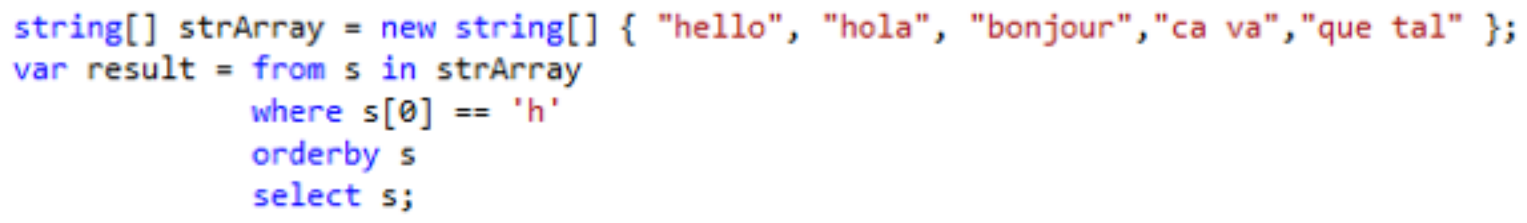
Linq: Language Integrated Query

- **System.Linq** namespace contains different extensions under **System.Linq.Enumerable**.
- Commonly used operators: **OrderByDescending/ OrderBy/ Take/ Average/ Sum/ Distinct/ Count/ First/ FirstOrDefault** etc…

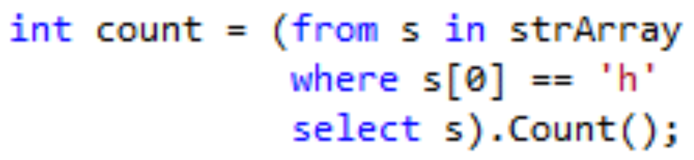
Linq: Language Integrated Query



```
string[] strArray = new string[] { "hello", "hola", "bonjour","ca va","que tal" };
var result = from s in strArray
             where s[0] == 'h'
             select s;
```



```
string[] strArray = new string[] { "hello", "hola", "bonjour","ca va","que tal" };
var result = from s in strArray
             where s[0] == 'h'
             orderby s
             select s;
```

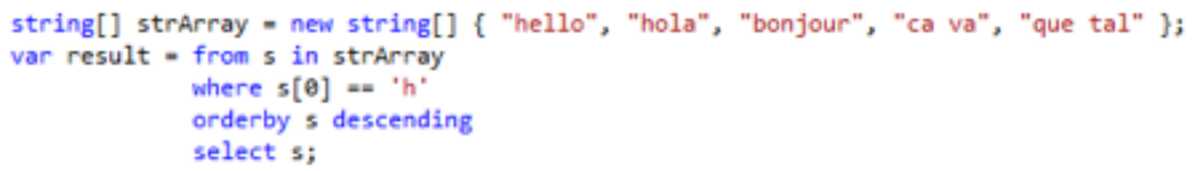


```
int count = (from s in strArray
             where s[0] == 'h'
             select s).Count();
```

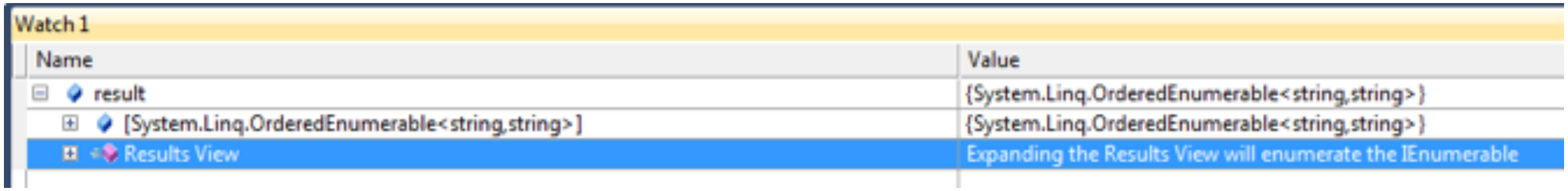
Linq: Language Integrated Query

- **Conversion** operators: **ToArray**/ **ToList**/ **ToDictionary**
- Enforce the **immediate execution** of queries
- Convert to different collection types **List**/ **Enumerable**/ **Queryable**/ etc..

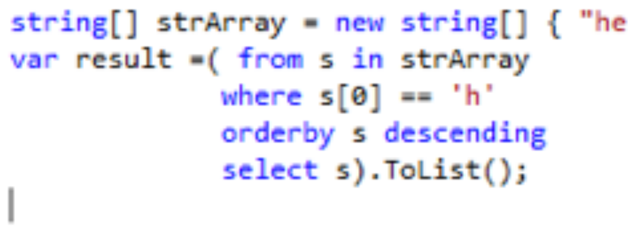
Linq: Language Integrated Query



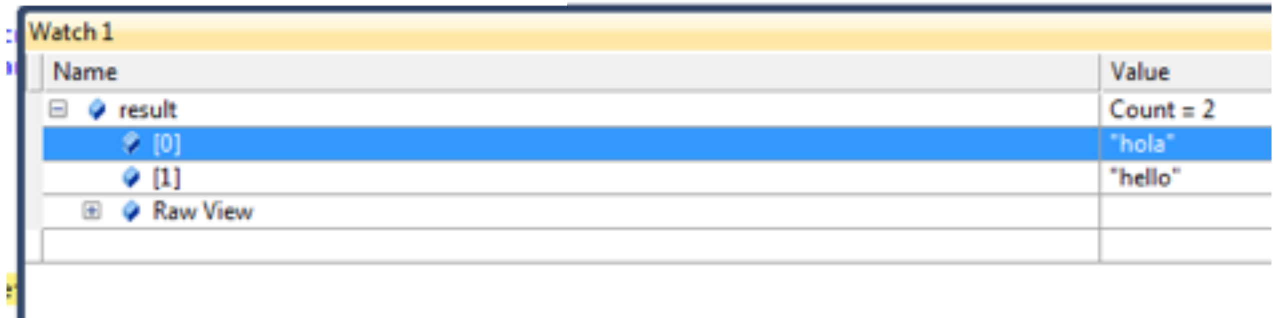
```
string[] strArray = new string[] { "hello", "hola", "bonjour", "ca va", "que tal" };
var result = from s in strArray
             where s[0] == 'h'
             orderby s descending
             select s;
```



```
Watch 1
Name                                                  Value
⊟ result                                              {System.Linq.OrderedEnumerable<string,string>}
   ⊞ [System.Linq.OrderedEnumerable<string,string>]   {System.Linq.OrderedEnumerable<string,string>}
   ⊞ Results View                                     Expanding the Results View will enumerate the IEnumerable
```



```
string[] strArray = new string[] { "he
var result =( from s in strArray
             where s[0] == 'h'
             orderby s descending
             select s).ToList();
```

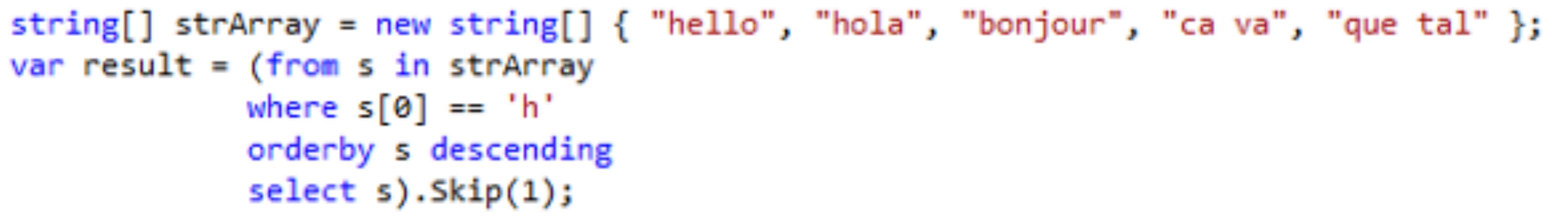


```
Watch 1
Name                   Value
⊟ result               Count = 2
     [0]               "hola"
     [1]               "hello"
   ⊞ Raw View
```

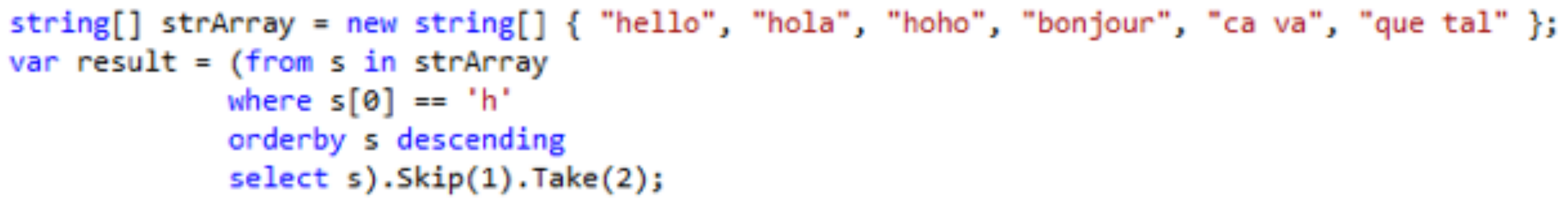
Linq: Language Integrated Query

- **Partitioning** operators: **Take/ Skip/ TakeWhile/ SkipWhile**
- Allows partitioning the results of query into a specific sequence.

Linq: Language Integrated Query



```
string[] strArray = new string[] { "hello", "hola", "bonjour", "ca va", "que tal" };
var result = (from s in strArray
              where s[0] == 'h'
              orderby s descending
              select s).Skip(1);
```



```
string[] strArray = new string[] { "hello", "hola", "hoho", "bonjour", "ca va", "que tal" };
var result = (from s in strArray
             where s[0] == 'h'
             orderby s descending
             select s).Skip(1).Take(2);
```

Demo

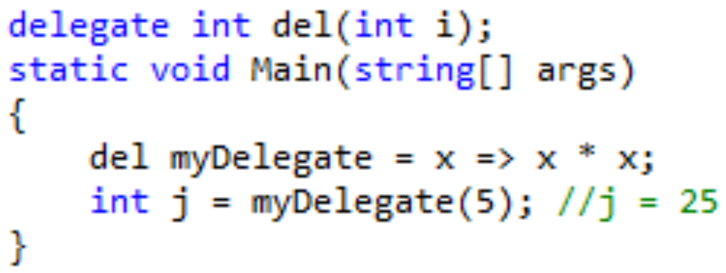
- LINQ query objects

Linq: Lambda expressions

- Anonymous function that can contain expressions and statements.
- Can be used to create delegates or expression tree types.
- => “goes to”
- All restrictions that apply to anonymous methods also apply to lambda expressions.

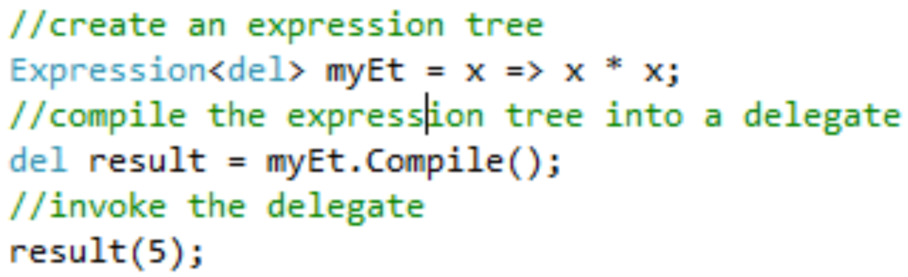
Linq: Lambda expressions

- Example 1



```
delegate int del(int i);
static void Main(string[] args)
{
    del myDelegate = x => x * x;
    int j = myDelegate(5); //j = 25
}
```

- Example 2

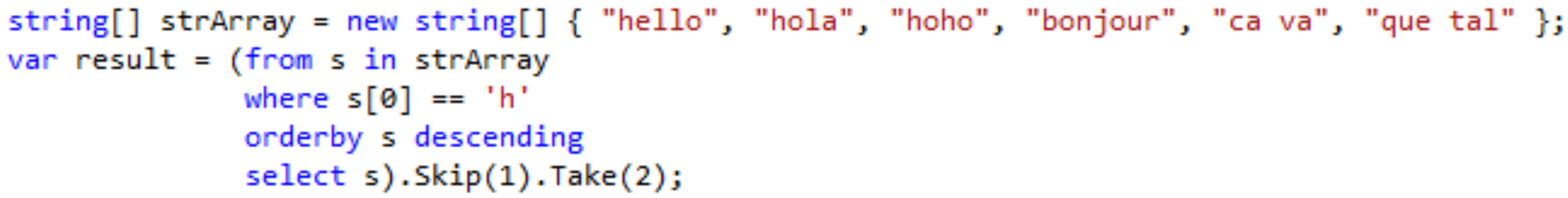


```
//create an expression tree
Expression<del> myEt = x => x * x;
//compile the expression tree into a delegate
del result = myEt.Compile();
//invoke the delegate
result(5);
```

Linq: Lambda expressions

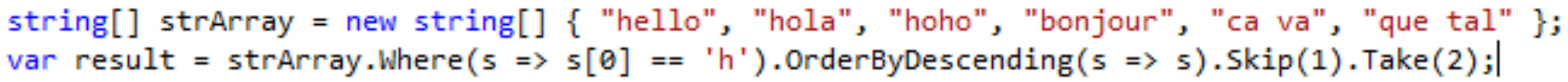
- Linq and Lambda expressions

before



```
string[] strArray = new string[] { "hello", "hola", "hoho", "bonjour", "ca va", "que tal" };
var result = (from s in strArray
              where s[0] == 'h'
              orderby s descending
              select s).Skip(1).Take(2);
```

after



```
string[] strArray = new string[] { "hello", "hola", "hoho", "bonjour", "ca va", "que tal" };
var result = strArray.Where(s => s[0] == 'h').OrderByDescending(s => s).Skip(1).Take(2);
```

Linq: Lambda expressions

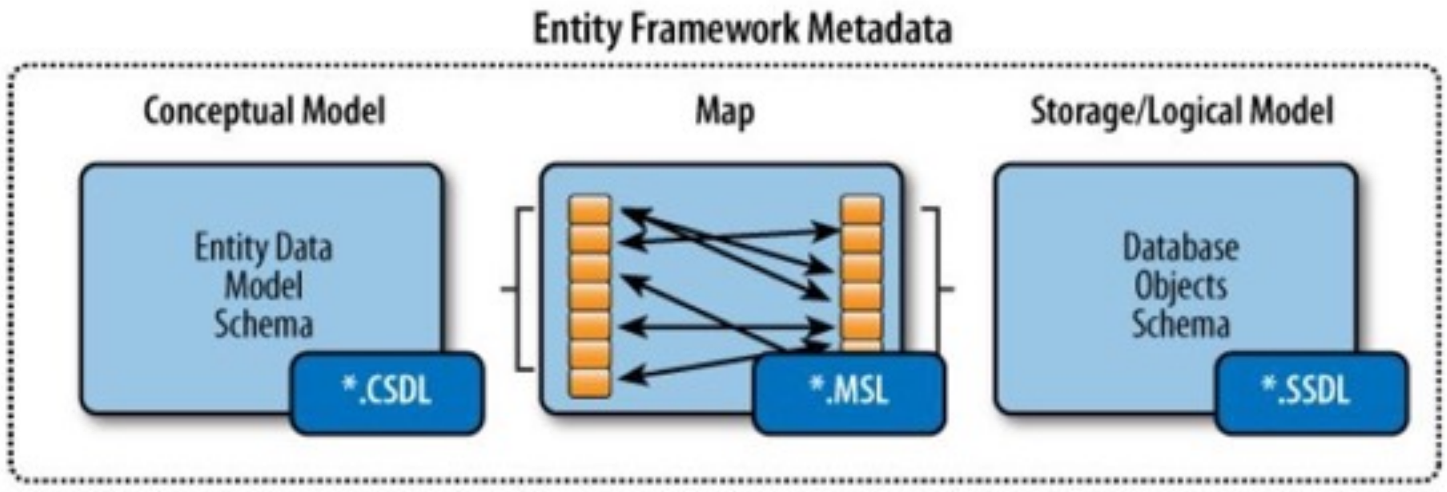
- Demo

Entity Framework: Components

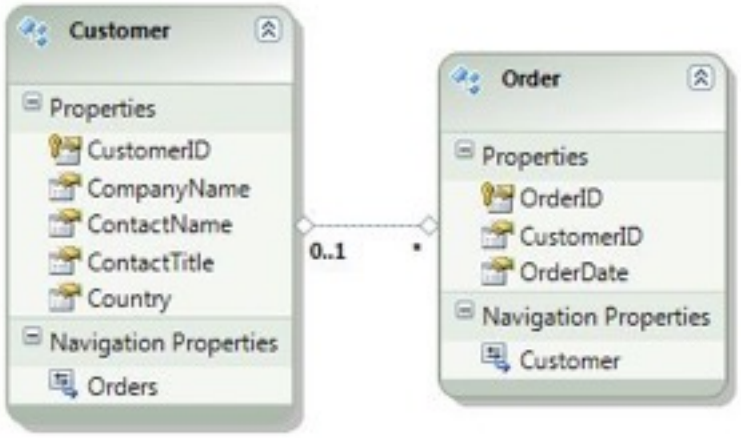
- Edmx (**Entity Data Model** – X for XML)
  - Modeling Entities
  - Define Mapping with Database

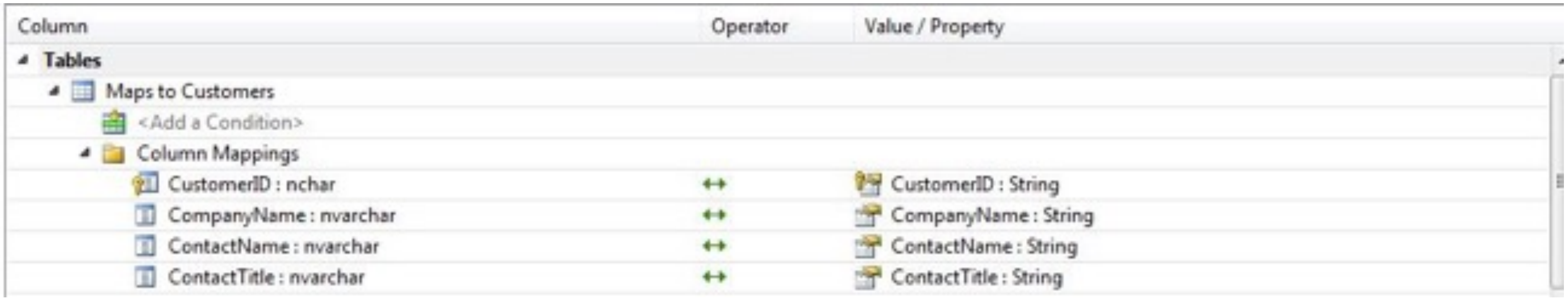
- SSDL (**Store Schema Definition Language**):
  - Database description
- CSDL (**Conceptual Schema Definition Language**) :
  - Entities description
- MSL (**Mapping Specification Language**) :
  - Mapping description

Entity Framework: Components



Entity Framework: Components





Column	Operator	Value / Property
Tables		
Maps to Customers		
<Add a Condition>		
Column Mappings		
CustomerID : nchar	↔	CustomerID : String
CompanyName : nvarchar	↔	CompanyName : String
ContactName : nvarchar	↔	ContactName : String
ContactTitle : nvarchar	↔	ContactTitle : String

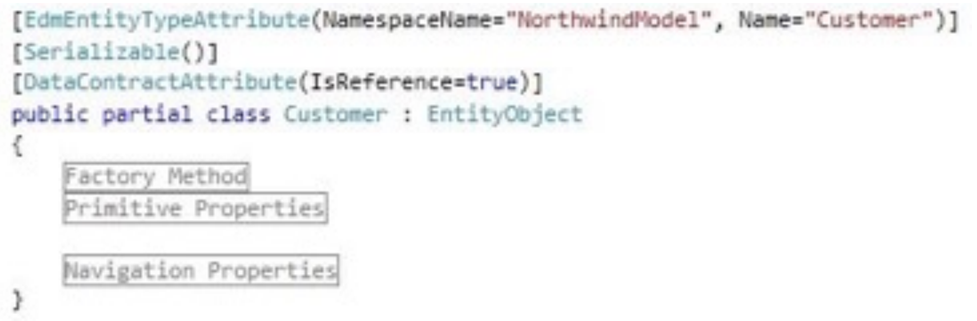
Entity Framework: Strategies

- **Database First** (generate Model from database)
- **Model First** (generate database from Model)
- **Code First** (No model, create code only)

Entity Framework: Template T4

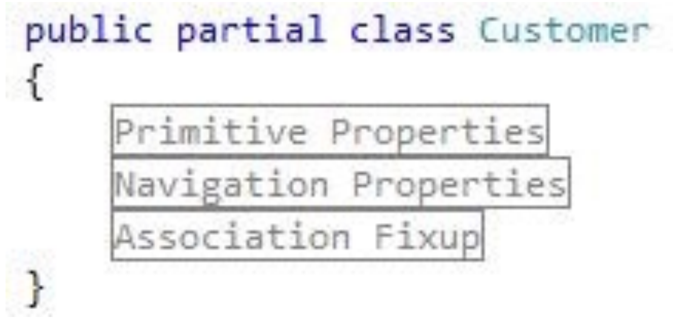
- Template **generation code** for :
  - Entities
  - Context
- Template examples:
  - POCO
  - Self Traking Entities

Entity Framework: Template T4



```
[EdmEntityTypeAttribute(NamespaceName="NorthwindModel", Name="Customer")]
[Serializable()]
[DataContractAttribute(IsReference=true)]
public partial class Customer : EntityObject
{
    Factory Method
    Primitive Properties

    Navigation Properties
}
```



```
public partial class Customer
{
    Primitive Properties
    Navigation Properties
    Association Fixup
}
```

Entity Framework: Key benefits

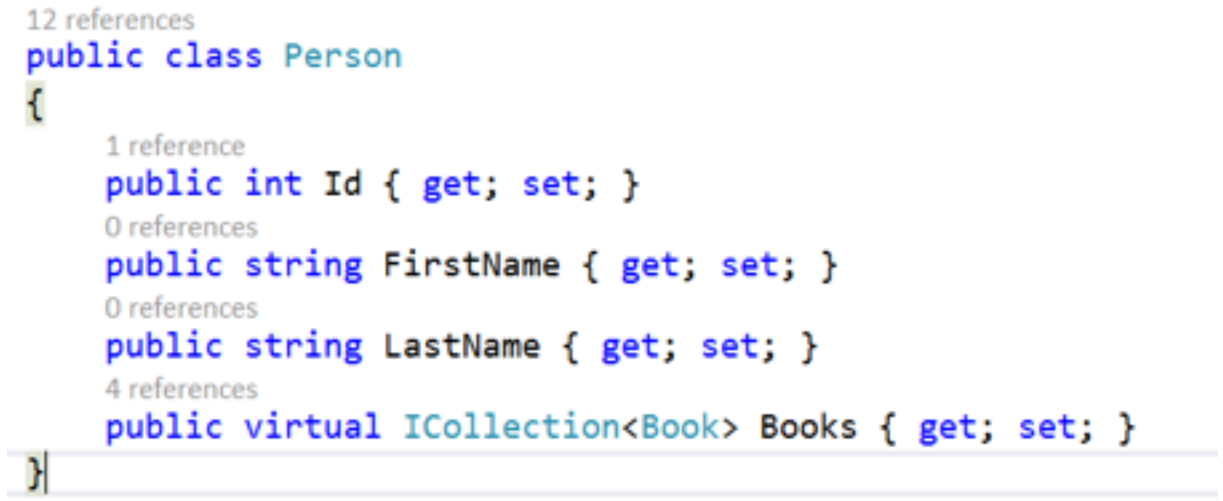
- **Integration** with WCF DataService and Scaffolding
- **Powerful mapping**
- Entity framework use the EDM to **build SQL queries**
- **Low coupling** between objects of the data access layer and elements of the database

Entity Framework: ObjectContext

- Main class for interacting with objects
- Contains :
  - **EntityConnection** (Manage connections to the database)
  - **MetadataWorkspace** (Get Metadata from edm at runtime)
  - **ObjectStateManager** (track entities)
- Any entity retrieved from the ObjectContext is attached

Entity Framework: Code First

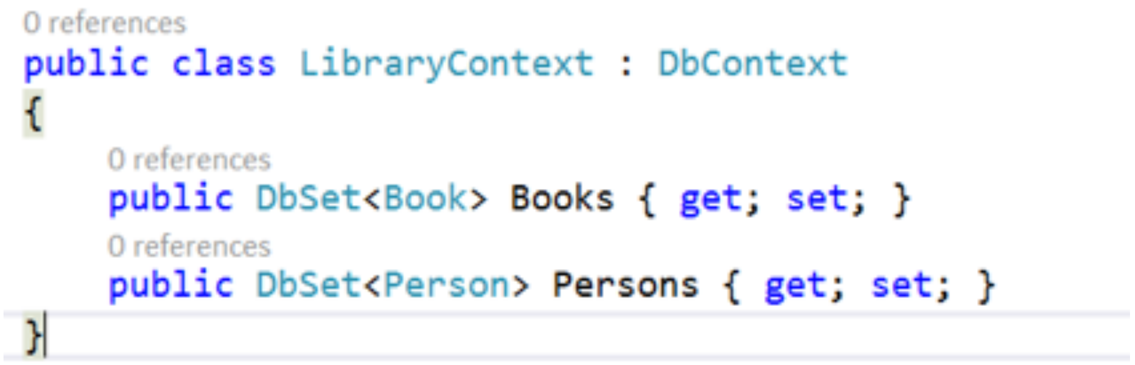
- Create Model Classes
- POCO
- No data code
- Navigation property virtual



```
12 references
public class Person
{
    1 reference
    public int Id { get; set; }
    0 references
    public string FirstName { get; set; }
    0 references
    public string LastName { get; set; }
    4 references
    public virtual ICollection<Book> Books { get; set; }
}
```

Entity Framework: Code First

- Create the context
- Get/Persist data



```
0 references
public class LibraryContext : DbContext
{
    0 references
    public DbSet<Book> Books { get; set; }
    0 references
    public DbSet<Person> Persons { get; set; }
}
```

Entity Framework: Code First

- Conventions
  - DbContext properties => Tables
  - POCO properties => Columns
  - Property Id will be primary key
  - http://blogs.msdn.com/b/efdesign/archive/2010/06/01/conventions-for-code-first.aspx

Entity Framework: Code First

- DataAnnotations (simple configuration)
  - KeyAttribute
  - ForeignKeyAttribute
  - RequiredAttribute
  - MaxLengthAttribute
  - MinLengthAttribute
  - TableAttribute
  - ColumnAttribute
  - DatabaseGeneratedAttribute (example Computed)
  - IndexAttribute

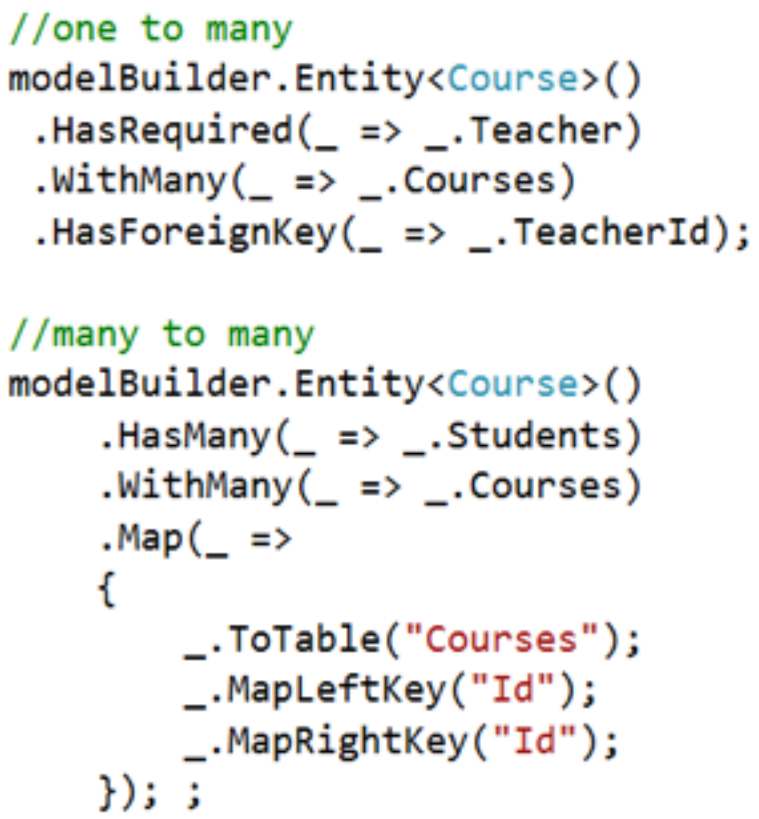
Entity Framework: Code First

- Fluent API (advanced configuration)
  - DbContext: override OnModelCreating
  - modelBuilder.HasDefaultSchema: change schema
  - modelBuilder.Entity<>: Entity Conventions
    - HasKey: primary key
    - ToTable
    - Property: get property convention
      - ex: Ignore, HasColumnName, HasColumnType, Required, HasMaxLength…

```
modelBuilder.Entity<Course>().Property(_ => _.Title).HasMaxLength(100);
```

Entity Framework: Code First

- Fluent API (advanced configuration)



```
//one to many
modelBuilder.Entity<Course>()
 .HasRequired(_ => _.Teacher)
 .WithMany(_ => _.Courses)
 .HasForeignKey(_ => _.TeacherId);

//many to many
modelBuilder.Entity<Course>()
    .HasMany(_ => _.Students)
    .WithMany(_ => _.Courses)
    .Map(_ =>
    {
        _.ToTable("Courses");
        _.MapLeftKey("Id");
        _.MapRightKey("Id");
    }); ;
```

Entity Framework: Code First

- Custom conventions
  - ex: new primary key convention

```
modelBuilder.Properties()
        .Where(p => p.Name.EndsWith("Key"))
        .Configure(p => p.IsKey().HasDatabaseGeneratedOption(DatabaseGeneratedOption.None));
```

  - ex: custom attribute

```
modelBuilder.Properties()
            .Having(x =>x.GetCustomAttributes(false).OfType<IsUnicode>().FirstOrDefault())
            .Configure((config, att) => config.IsUnicode(att.Unicode));
```

Entity Framework: Code First

- example Convention class

```
public class DateTime2Convention : Convention
{
    public DateTime2Convention()
    {
        this.Properties<DateTime>()
            .Configure(c => c.HasColumnType("datetime2"));
    }
}
```

- Register with modelBuilder

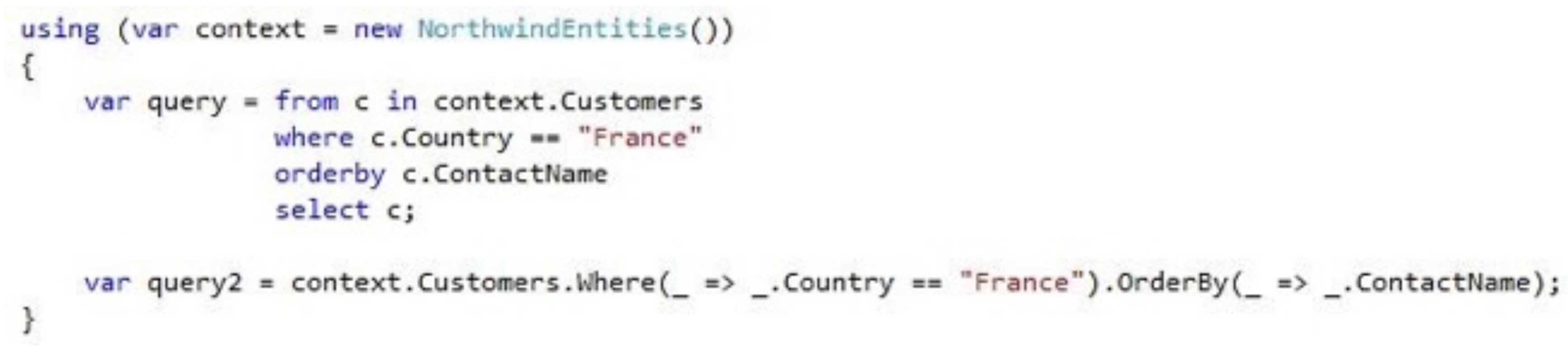
```
modelBuilder.Conventions.Add(new DateTime2Convention());
```

Entity Framework: Code First

- Migrations
  - Enable-Migrations
  - Add-Migration «MyMigration»
  - Update-Database
  - http://msdn.microsoft.com/en-US/data/jj591621

- Tips
  - Local server: (localdb)\v11.0
  - Connectionstring in App.config/web.config

Entity Framework: ObjectContext / DbContext



```
using (var context = new NorthwindEntities())
{
    var query = from c in context.Customers
                where c.Country == "France"
                orderby c.ContactName
                select c;

    var query2 = context.Customers.Where(_ => _.Country == "France").OrderBy(_ => _.ContactName);
}
```

Entity Framework: Demo

- Mapping + query