

LINQ to SQL

в реальном ASP.NET проекте



team meeting

Инструменты*

- Microsoft Visual Studio Web Developer Express 2008 SP1 (.NET 3.5 SP1)
- Microsoft SQL Server 2005 Express Edition

Опционально:

- SQL Server Management Studio Express
- LINQpad

* Microsoft Web Platform вышла чуть позже :)

Доступ к данным в .NET

- DataReader'ы
- Нетипизированные DataSet'ы
- Типизированные DataSet'ы
- Собственные классы для сущностей
- **LINQ to SQL**
- ADO.NET Entity Framework

Жизнь до LINQ (небольшой пример)

```
public DataRow GetOne(int id)
{
    using (SqlConnection connection = MsSqlHelper.GetConnection(_connectionString))
    {
        SqlCommand command = new SqlCommand(@"
            SELECT *
            FROM [dbo].[products]
            WHERE [id] = @id
        ");
        command.Parameters.AddWithValue("id", id);
        return MsSqlHelper.ExecuteSelectCommandToDataTable(command, connection).Rows[0];
    }
}

//...

DataRow row = GetOne(100);

string productName = row["name"].ToString();
int productCode = Convert.ToInt32(row["id"]);
```

Почему это плохо?

- Сложно и долго «заворачивать» сущности базы в объекты
- Нет *type safety*
- Необходимость конвертации форматов
- Необходимо конструировать SQL-запросы

Факты о LINQ to SQL

- LINQ to SQL — это облегчённый ORM
- Все сущности БД видны в C# коде как классы и объекты (строго типизированные!)
- Благодаря поддержке LINQ в C# и VB.NET нет необходимости писать запросы на SQL (почти нет :))
- В комплекте есть инструменты для генерации классов по имеющейся БД (визуальные и command-line)
- Любую операцию можно выполнить на SQL, если это необходимо

Ограничения LINQ to SQL

- В настоящий момент полноценно поддерживается только MS SQL Server
- Классы модели данных полностью повторяют таблицы БД
- Остальные ограничения можно обойти :)

Условия по условию :)

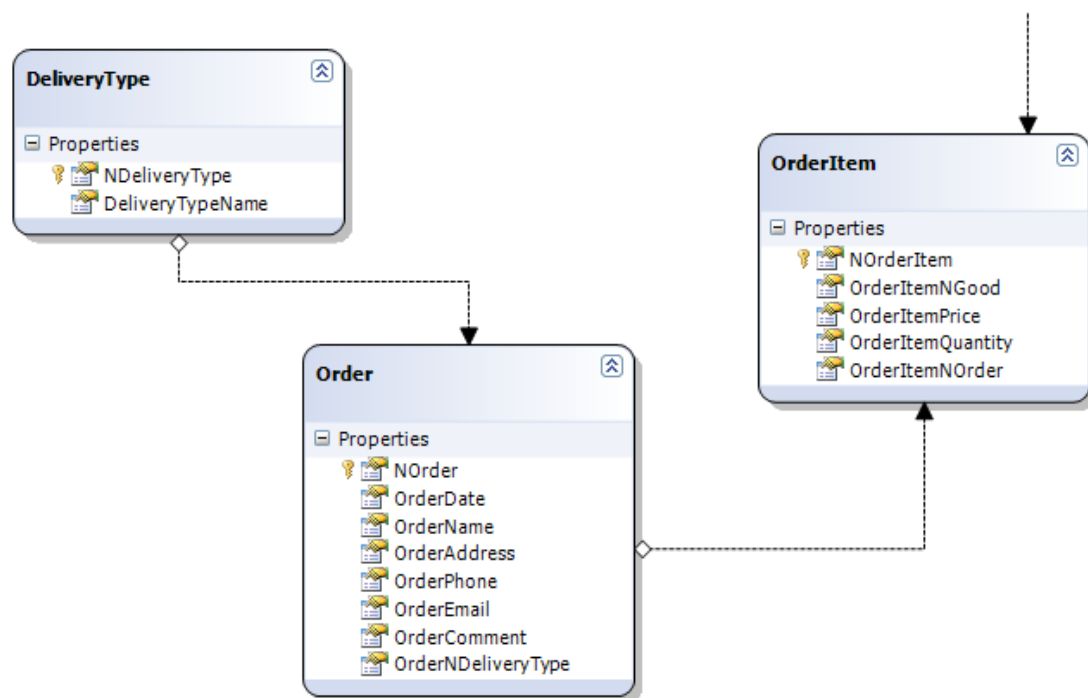
```
var result = from n in Db.News
              select n;

if (FilterByDate)
{
    result = result.Where(n => n.NewsDate < DateTime.Now);
}

if (FilterByTitle)
{
    result = result.Where(n => n.NewsTitle == "Просто новость");
}

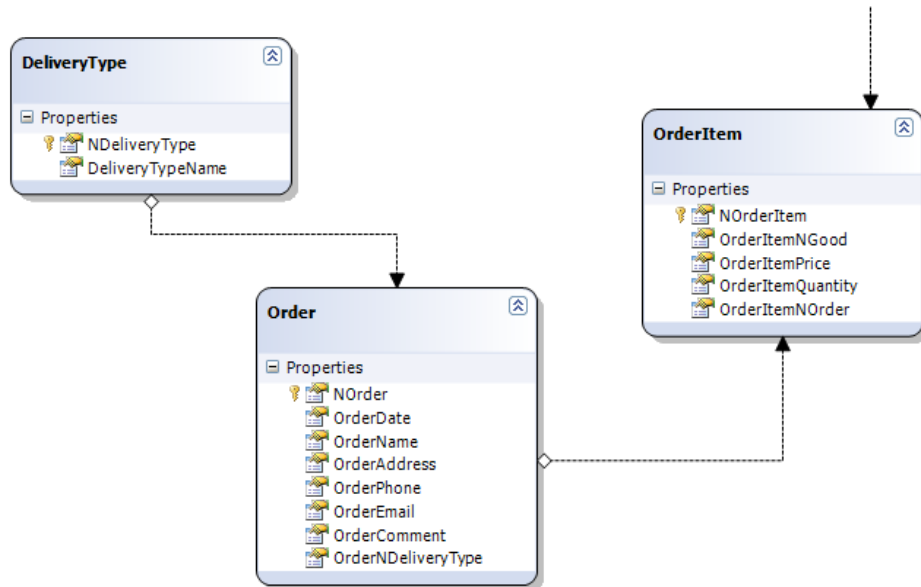
var newsList = result.ToList();
```

Связи между таблицами



```
var temp = (from o in UtilsLinq.Db.Orders
            where o.DeliveryType.DeliveryTypeName == "Супер Доставка"
            select o.NOrder).Count();
```

Что-то поинтереснее?



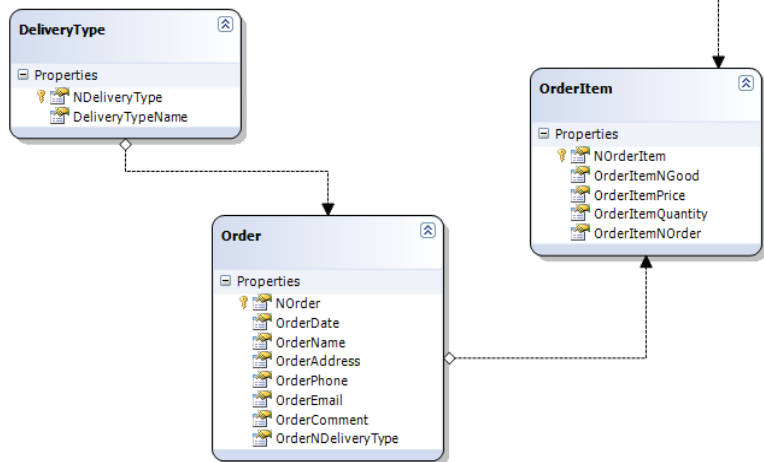
```
var result = from o in UtilsLinq.Db.Orders
              where o.OrderDate > startTime.AddDays(-1) && o.OrderDate < endTime.AddDays(+1)
              select new OrderCustom
              {
                  NOrder = o.NOrder,
                  OrderName = o.OrderName,
                  OrderAddress = o.OrderAddress,
                  OrderPhone = o.OrderPhone,
                  OrderEmail = o.OrderEmail,
                  OrderDate = o.OrderDate,
                  OrderComment = o.OrderComment,
                  DeliveryType = o.DeliveryType,
                  OrderSum = o.OrderItems.Sum(oi => oi.OrderItemPrice * oi.OrderItemQuantity)
              };
```

```
totalCount = result.Count();
```

```
totalSumm = result.Sum(oc => oc.OrderSum);
```

```
return result.Skip(pageSize * (pageNumber - 1)).Take(pageSize).ToList();
```

Вставка в master-detail?



```
using (SkladContext dataContext = UtilsLinq.GetContext())
{
    var tempOrder = new Order
    {
        OrderAddress = "ул. Ленина 22",
        OrderDate = DateTime.Now,
        OrderComment = "Привезите мой заказ быстро!",
        OrderEmail = "example@example.com",
        OrderName = "Иванов Иван Иванович",
        OrderPhone = "22-33-44",
        OrderNDeliveryType = 1
    };

    dataContext.Orders.InsertOnSubmit(tempOrder);

    var tempItemsList = new List<OrderItem>();

    foreach (var item in Cart.GetItems())
    {
        var tempItem = new OrderItem
        {
            OrderItemNGood = item.Key,
            OrderItemPrice = item.Value.Price,
            OrderItemQuantity = item.Value.Quantity,
            Order = tempOrder
        };
        tempItemsList.Add(tempItem);
    }

    dataContext.OrderItems.InsertAllOnSubmit(tempItemsList);
    dataContext.SubmitChanges();
}
```

Welcome to the real world.

Morpheus

Можно SQL посмотреть?

The screenshot shows a Visual Studio IDE with a LINQ query in a C# class. The code is as follows:

```
public partial class News
{
    public static IEnumerable<NewsCustom> GetPage(int pageNumber, int pageSize, out int totalCount)
    {
        var result = from n in UtilsLinq.Db.News
                     orderby n.NewsDate descending
                     select new NewsCustom
                     {
                         NNNews = n.NNNews,
                         NewsTitle = n.NewsTitle,
                         NewsLead = n.NewsLead,
                         NewsDate = n.NewsDate
                     };

        totalCount = result.Count();
        return result.Skip(pageSize * (pageNumber - 1)).Take(pageSize).ToList();
    }
}
```

The SQL translation for the highlighted LINQ query is shown in the Output window:

```
{SELECT [t0].[NNNews], [t0].[NewsTitle], [t0].[NewsDate], [t0].[NewsLead] FROM [dbo].[News] AS [t0] ORDER BY [t0].[NewsDate] DESC}
```

The Locals window shows the following variables:

Name	Value	Type
pageNumber	1	int
pageSize	20	int
totalCount	3	int
result	{SELECT [t0].[NNNews], [t0].[NewsTitle], System.Li	System.Li

The Output window shows the SQL query and its context:

```
SELECT TOP (20) [t0].[NNNews], [t0].[NewsTitle], [t0].[NewsDate], [t0].[NewsLead]
FROM [dbo].[News] AS [t0]
ORDER BY [t0].[NewsDate] DESC
-- Context: SqlProvider(Sql2005) Model: AttributedMetaModel Build: 3.5.30729.1
```

Как сделать WHERE IN

```
var newsIds = new List<int> { 1, 2, 3, 4 };

var result = from n in Db.News
              where newsIds.Contains(n.NNews)
              select n;

var temp = result.ToList();
```

Output

Show output from: Debug

```
SELECT [t0].[NNews], [t0].[NewsTitle], [t0].[NewsDate], [t0].[NewsLead], [t0].[NewsText]
FROM [dbo].[News] AS [t0]
WHERE [t0].[NNews] IN (@p0, @p1, @p2, @p3)
-- @p0: Input Int (Size = 0; Prec = 0; Scale = 0) [1]
-- @p1: Input Int (Size = 0; Prec = 0; Scale = 0) [2]
-- @p2: Input Int (Size = 0; Prec = 0; Scale = 0) [3]
-- @p3: Input Int (Size = 0; Prec = 0; Scale = 0) [4]
-- Context: SqlProvider(Sql2005) Model: AttributedMetaModel Build: 3.5.30729.1
```

Операции над несколькими строками таблицы

Вариант 1: Использовать SQL

```
dataContext.ExecuteCommand(@"  
    DELETE FROM [dbo].[News]  
    WHERE ([NewsDate] < {0})",  
    DateTime.Now.AddMonths(-1));
```

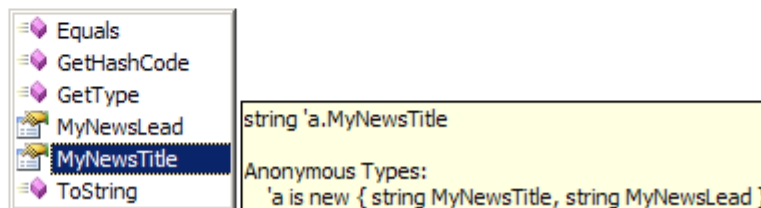
Вариант 2: Написать свой LINQ SQL оператор
(нетривиальная задача)

Выбор набора полей из таблицы

Вариант 1: Использование анонимного типа

```
var simpleResult = from n in Db.News
                   select new
                   {
                       MyNewsTitle = n.NewsTitle,
                       MyNewsLead = n.NewsLead
                   };

string firstTitle = simpleResult.First().
```



Вариант 2: Использование собственных классов (их можно унаследовать от классов сущностей — не совсем красиво, но работает :))

Сортировка по заранее неизвестному полю

Вариант 1: Написать кучу условий

Вариант 2: Сконструировать оператор вручную
(не работает для связанных таблиц)

Вариант 3: Использовать Dynamic LINQ

```
totalCount = result.Count();

result = result.OrderBy(String.Format("{0} {1}", order, orderByOrder));
return result.Skip(pageSize * (pageNumber - 1)).Take(pageSize).ToList();
}
```

Использование SQL + ПОЛНОТЕКСТОВЫЙ ПОИСК

```
CREATE FUNCTION [dbo].[fn_GetGoodsByFullText]
(
    @search_string varchar(250)
)
RETURNS TABLE
AS
RETURN
(
    SELECT *
    FROM [Goods]
    WHERE FREETEXT( ([GoodCode], [GoodName], [GoodLead], [GoodDescription]), @search_string)
)
```

Теперь опишем...

The screenshot displays the Visual Studio IDE with several components:

- Left Panel:** Shows a tree view of project items including **News** (with properties: NNews, NewsTitle, NewsDate, NewsLead, NewsText), **Block** (with properties: NBlo, Block, Block), **TextPage** (with properties: NTex, TextP, TextP, TextP), and **Folder** (with properties: NFolder, FolderTitle).
- Right Panel:** Shows the **Properties** window for the **GetGoodsByFullText (System.String search_string)** Data Function.

The **Properties** window details are as follows:

GetGoodsByFullText Data Function	
Code Generation	
Access	Public
Inheritance Modifier	(None)
Name	GetGoodsByFullText
Return Type	Good
MISC	
Method Signature	GetGoodsByFullText (System.String search_string)
Source	dbo.fn_GetGoodsByFullText

Below the table, the **Name** property is described as "Name of the method."

И воспользуемся...

```
var searchResult = (from g in Db.GetGoodsByFullText("Nokia")
                    select g).Take(50);
```

Output

Show output from: Debug

```
SELECT TOP (50) [t0].[NGood], [t0].[GoodCode], [t0].[GoodName], [t0].[GoodPrice], [t0].[GoodMeasure], [t0].[GoodCount]
FROM [dbo].[fn_GetGoodsByFullText](@p0) AS [t0]
-- @p0: Input VarChar (Size = 5; Prec = 0; Scale = 0) [Nokia]
-- Context: SqlProvider(Sql2005) Model: AttributedMetaModel Build: 3.5.30729.1
```