

МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ
ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО
ОБРАЗОВАНИЯ
«ИЖЕВСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ
УНИВЕРСИТЕТ
ИМЕНИ М.Т. КАЛАШНИКОВА»
ФАКУЛЬТЕТ «ИНЖЕНЕРНО-ЭКОНОМИЧЕСКИЙ»
КАФЕДРА «ИНФОРМАЦИОННЫЕ СИСТЕМЫ»

Реферат на тему: «Linq to entities»
(по дисциплине «Проектный практикум 2»)

Проверил
к.ф.-м.н., доцент

А.В. Корепанов

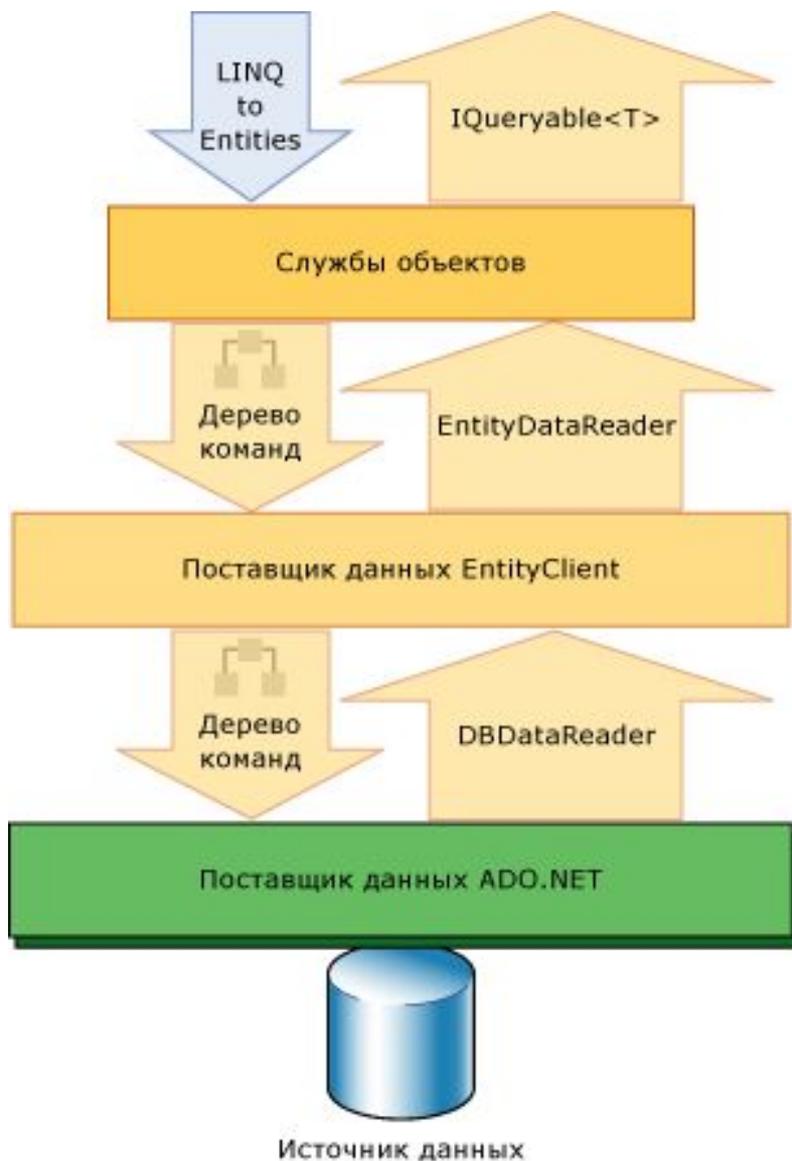
Выполнил
студент группы Б07-021-1

О.Ю. Светлаков

LINQ to entities

Это часть платформы ADO.NET Entity Framework, альтернативный интерфейс LINQ API, используемый для обращения к базе данных. Он отделяет сущностную объектную модель данных от физической базы данных, вводя логическое отображение между ними.

Технология LINQ to Entities



LINQ to Entities позволяет разработчикам создавать запросы к базе данных с помощью того же языка, который был использован для создания бизнес-логики. На схеме показана связь между LINQ to Entities и платформой Entity Framework, ADO.NET 2.0 и источником данных.

Пример работы программы

- `public class Company`
- `{`
- `public int Id { get; set; }`
- `public string Name { get; set; }`
- `public ICollection <Phone> Phones { get; set; }`
- `public Company()`
- `{`
- `Phones = new List<Phone>();`
- `}`
- `}`
- `public class Phone`
- `{`
- `public int Id { get; set; }`
- `public string Name { get; set; }`
- `public int Price { get; set; }`
- `public int CompanyId { get; set; }`
- `public Company Company { get; set; }`
- `}`

Пример работы программы

```
• class PhoneContext : DbContext
• {
•     static PhoneContext()
•     {
•         Database.SetInitializer(new MyContextInitializer());
•     }
•     public PhoneContext() :base("DefaultConnection")
•     {}
•
•     public DbSet<Company> Companies { get; set; }
•     public DbSet<Phone> Phones { get; set; }
• }
•
• class MyContextInitializer : DropCreateDatabaseAlways<PhoneContext>
• {
•     protected override void Seed(PhoneContext db)
•     {
•         Company c1 = new Company { Name = "Samsung" };
•         Company c2 = new Company { Name = "Apple" };
•         db.Companies.Add(c1);
•         db.Companies.Add(c2);
•
•         db.SaveChanges();
•
•         Phone p1 = new Phone {Name="Samsung Galaxy S5", Price=20000, Company = c1};
•         Phone p2 = new Phone {Name="Samsung Galaxy S4", Price=15000, Company = c1};
•         Phone p3 = new Phone {Name="iPhone5", Price=28000, Company = c2};
•         Phone p4 = new Phone {Name="iPhone 4S", Price=23000, Company = c2};
•
•         db.Phones.AddRange(new List<Phone>(){p1, p2, p3, p4});
•         db.SaveChanges();
•     }
• }
```

ЗАПРОСЫ В LINQ TO ENTITIES

Например, используем некоторые операторы LINQ:

- `using(PhoneContext db = new PhoneContext())`
- `{`
- `var phones = from p in db.Phones`
- `where p.CompanyId == 1`
- `select p;`
- `}`

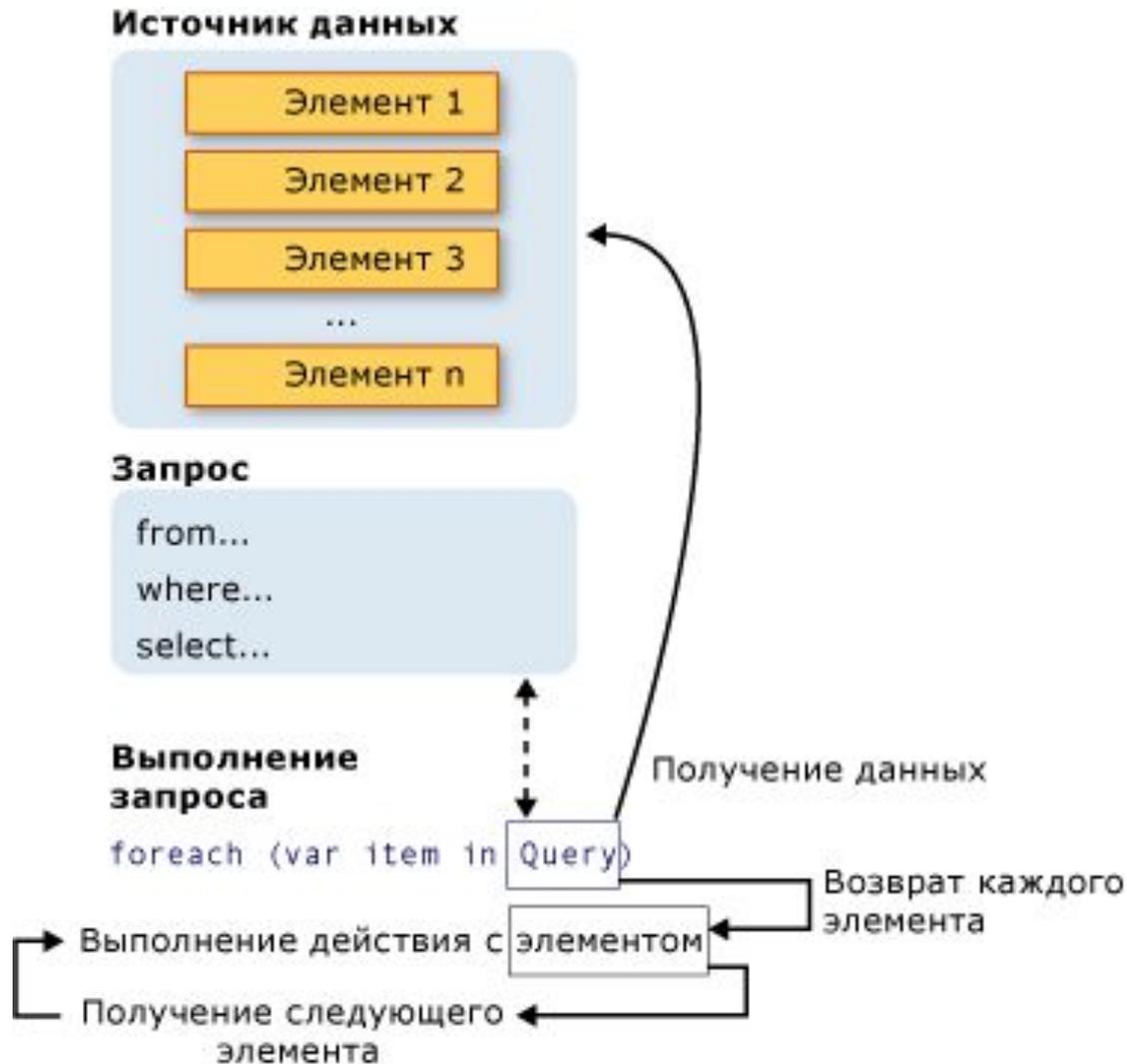
И тот же запрос с помощью методов расширений LINQ:

- `using(PhoneContext db = new PhoneContext())`
- `{`
- `var phones = db.Phones.Where(p=> p.CompanyId == 1);`
- `}`

Оба запроса в итоге транслируются в одно выражение sql:

- `SELECT [Extent1].[Id] AS [Id],`
- `[Extent1].[Name] AS [Name],`
- `[Extent1].[Price] AS [Price],`
- `[Extent1].[CompanyId] AS [CompanyId]`
- `FROM [dbo].[Phones] AS [Extent1]`
- `WHERE 1 = [Extent1].[CompanyId]}`

Выполнение запроса



Результат запроса

```
var contacts =  
  from c in customers  
  where c.State == "WA"  
  select new { c.Name, c.Phone };
```

Интегрированный
запрос

Типизирование
переменной

```
var contacts =  
  customers  
  .Where(c => c.State == "WA")  
  .Select(c => new { c.Name, c.Phone });
```

Лямбда-
выражения

Методы-
расширения

Анонимные
типы

Упрощенная
инициализация
объекта