**Преобразования данных с помощью LINQ (C#)**

* Статья
* 28.09.2021

LINQ используется не только для получения данных. Это эффективный инструмент для их преобразования. С помощью запросов LINQ можно использовать исходную последовательность в качестве входных данных и изменять ее разными способами для создания новой выходной последовательности. Можно изменить саму последовательность, не изменяя элементы, с помощью сортировки и группировки. Но самой интересной функцией запросов LINQ можно назвать возможность создания новых типов. Это выполняется в предложении [select](https://docs.microsoft.com/ru-ru/dotnet/csharp/language-reference/keywords/select-clause). Например, можно выполнить следующие задачи.

* Объединение нескольких входных последовательностей в одну выходную последовательность, имеющую новый тип.
* Создание выходных последовательностей, элементы которых состоят из одного или нескольких свойств каждого элемента в исходной последовательности.
* Создание выходных последовательностей, элементы которых состоят из результатов операций, выполняемых с источником данных.
* Создание выходных последовательностей в другом формате. Например, можно преобразовать данные из строк SQL или текстовых файлов в XML.

Это лишь несколько примеров. Очевидно, что эти преобразования могут сочетаться различными способами в одном запросе. Более того, выходную последовательность одного запроса можно использовать как входную последовательность для нового запроса.

## Объединение нескольких входных последовательностей в одну выходную

Запрос LINQ можно использовать для создания выходной последовательности, содержащей элементы из нескольких входных последовательностей. Следующий пример демонстрирует объединение двух структур данных в памяти, однако те же принципы могут применяться для объединения данных из источников XML, SQL или DataSet. Рассмотрим следующие два типа классов: см. прогу.

## Выбор подмножества каждого исходного элемента

Существует два основных способа выбора подмножества каждого элемента в исходной последовательности:

1. Выбор только одного члена исходного элемента с помощью операции dot. В следующем примере предполагается, что объект Customer содержит несколько открытых свойств, включая строку с именем City. При выполнении этот запрос создаст выходную последовательность строк.

C#Копировать

var query = from cust in Customers

select cust.City;

1. Для создания элементов, содержащих более одного свойства исходного элемента, можно использовать инициализатор объектов с именованным объектом или анонимным типом. В следующем примере показано использование анонимного типа для инкапсуляции двух свойств из каждого элемента Customer:

C#Копировать

var query = from cust in Customer

select new {Name = cust.Name, City = cust.City};