Операции Join и GroupJoin

Операции соединения (join) связывают вместе несколько последовательностей.

Join

Операция Join выполняет внутреннее соединение по эквивалентности двух последовательностей на основе ключей, извлеченных из каждого элемента этих последовательностей. Операция Join имеет один прототип, описанный ниже:

C#

```
public static IEnumerable<V> Join<T, U, K, V>(
     this IEnumerable<T> outer,
     IEnumerable<U> inner,
     Func<T, K> outerKeySelector,
     Func<U, K> innerKeySelector,
     Func<T, U, V> resultSelector);
```

Обратите внимание, что первый аргумент метода имеет имя outer. Поскольку это расширяющий метод, последовательность, на которой вызывается операция Join, будет называться внешней последовательностью.

Операция Join возвращает объект, который при перечислении сначала проходит последовательность inner элементов типа U, вызывая метод innerKeySelector по одному разу для каждого элемента и сохраняя элемент, на который ссылается его ключ, в хеш-таблице. Затем он проходит последовательность outer элементов типа T. По мере того, как возвращаемый объект перечисляет каждый объект последовательности outer, он вызывает *метод outerKeySelector* для получения ключа и извлекает соответствующий элемент последовательности inner из хеш-таблицы, используя этот ключ.

Для каждой соответствующей пары элементов из последовательности outer и inner возвращаемый объект вызывает метод resultSelector, передавая ему и элемент outer, и соответствующий ему элемент inner. Метод resultSelector вернет экземпляр объекта

типа V, который возвращаемый объект поместит в выходную последовательность типа V.

Порядок элементов последовательности outer сохраняется, равно как и порядок элементов inner в пределах каждого элемента outer.

Для того чтобы продемонстрировать пример применения этой операции, вместо массива cars, используемого в большинстве примеров, будут использоваться два общих класса — Employee и EmployeeOptionEntry (2_0.php).

Ниже приведен пример вызова операции Join с использованием этих классов. Чтобы повысить читабельность каждого аргумента Join, код построен немного иначе, чем обычно:

C#

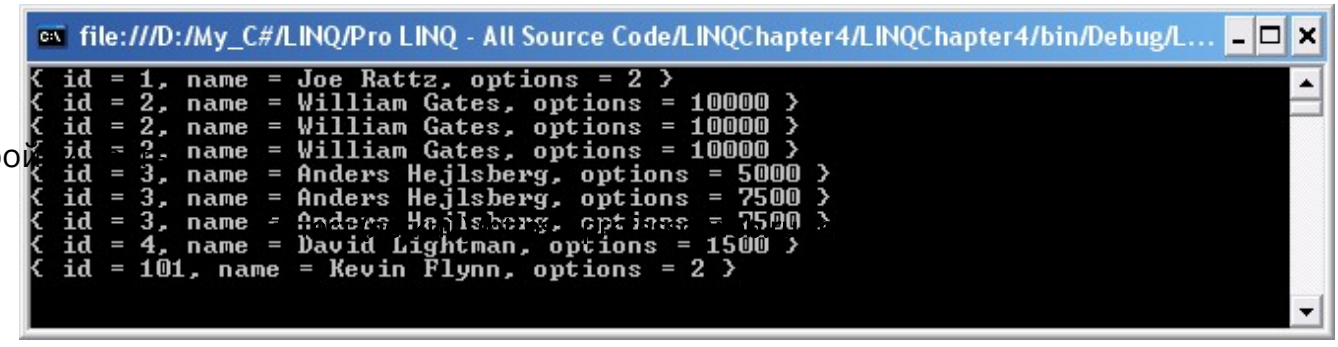
```
Employee[] employees = Employee.GetEmployeesArray();
      EmployeeOptionEntry[] empOptions = EmployeeOptionEntry.GetEmployeeOptionEntries();

      var employeeOptions = employees
        .Join(
          empOptions,         //  inner sequence
          e => e.id,              //  outerKeySelector
          o => o.id,              //  innerKeySelector
          (e, o) => new       //  resultSelector
          {
            id = e.id,
            name = string.Format("{0} {1}", e.firstName, e.lastName),
            options = o.optionsCount
          });

      foreach (var item in employeeOptions)
        Console.WriteLine(item);
```

В коде сначала получается пара массивов для соединения с использованием двух общих классов. Поскольку операция Join вызывается на массиве employees, он становится внешней (outer) последовательностью, а empOptions — внутренней (inner) последовательностью. Ниже показаны результаты вызова операции Join:

Прои



Обратите внимание, что resultSelector создает анонимный класс в качестве типа элемента для результирующей выходной последовательности. То, что это анонимный класс, можно понять по вызову new без указания имени класса. Поскольку тип анонимный, выходную последовательность необходимо сохранить в переменной, тип которой указан с использованием ключевого слова var. Его нельзя указать как IEnumerable<>, поскольку нет именованного типа, который послужил бы параметром для объявления IEnumerable.

GroupJoin

Операция GroupJoin выполняет групповое соединение двух последовательностей на основе ключей, извлеченных из каждого элемента последовательностей.

Операция GroupJoin работает очень похоже на Join, за исключением того, что Join передает один элемент внешней последовательности с одним соответствующим элементом внутренней последовательности методу resultSelector. Это значит, что множество элементов внутренней последовательности, соответствующих одному элементу внешней последовательности, приведут в результате к множеству вызовов resultSelector для этого элемента внешней последовательности. В случае операции GroupJoin все соответствующие элементы внутренней последовательности для определенного элемента внешней последовательности передаются в resultSelector как последовательность этого типа элемента, в результате чего метод resultSelector вызывается только по одному разу для каждого элемента внешней последовательности.

Эта операция имеет один прототип, описанный ниже:

C#

```
public static IEnumerable<V> GroupJoin<T, U, K, V>(
      this IEnumerable<T> outer,
      IEnumerable<U> inner,
      Func<T, K> outerKeySelector,
      Func<U, K> innerKeySelector,
      Func<T, IEnumerable<U>, V> resultSelector);
```

Обратите внимание, что первый аргумент метода имеет имя outer. Поскольку это расширяющий метод, последовательность, на которой вызывается операция Join, будет называться внешней последовательностью.

Операция GroupJoin возвращает объект, который при перечислении сначала проходит по последовательности inner элементов типа U, вызывая метод innerKeySelector по одному разу для каждого элемента и сохраняя элемент, на который

ссылается ключ, в хеш-таблице. Затем возвращенный объект выполняет перечисление последовательности outer элементов типа T. По мере того, как возвращаемый объект перечисляет каждый элемент последовательности outer, он вызывает метод outerKeySelector для получения его ключа и извлекает соответствующий элемент последовательности inner из хеш-таблицы по этому ключу.

Для каждого элемента последовательности outer возвращаемый объект вызывает метод resultSelector, передавая ему элемент outer и последовательность соответствующих элементов inner, так что resultSelector может вернуть экземпляр объекта типа V, куда возвращаемый объект поместит выходную последовательность типа V.

В примере использования операции GroupJoin применяются те же классы Employee и EmployeeOptionEntry, что и в примере с Join. Код, приведенный ниже, соединяет сотрудников с опционами и вычисляет сумму опционов для каждого сотрудника с помощью операции GroupJoin:

C#

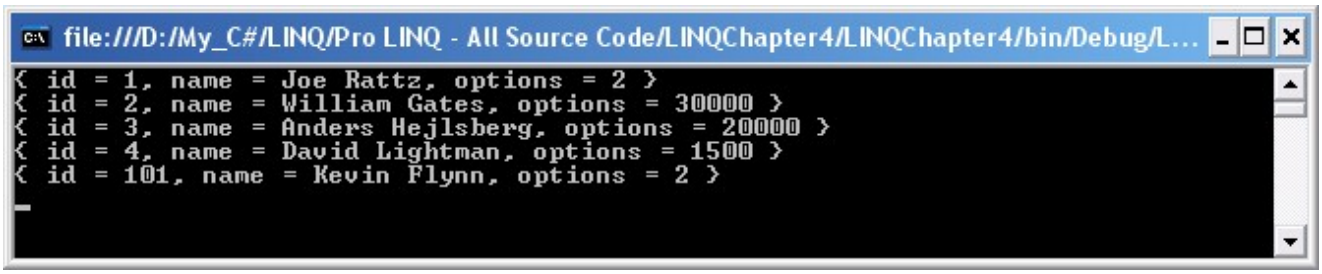
```
Employee[] employees = Employee.GetEmployeesArray();
      EmployeeOptionEntry[] empOptions = EmployeeOptionEntry.GetEmployeeOptionEntries();

      var employeeOptions = employees
        .GroupJoin(
          empOptions,
          e => e.id,
          o => o.id,
          (e, os) => new
          {
            id = e.id,
            name = string.Format("{0} {1}", e.firstName, e.lastName),
            options = os.Sum(o => o.optionsCount)
          });

      foreach (var item in employeeOptions)
        Console.WriteLine(item);
```

Приведенный код почти идентичен примеру с операцией Join. Однако здесь второй аргумент лямбда-выражения, переданного в качестве метода resultSelector, назван os, в отличие от o в примере с Join. Причина в том, что в примере с Join в этом аргументе передавался единственный объект опциона — o, а в примере с GroupJoin передается последовательность объектов опционов сотрудника — os. Затем последнему члену экземпляра анонимного объекта, optionsCount, присваивается сумма последовательности из объектов опционов сотрудника с использованием операции Sum, которая будет описана позже (поскольку это не отложенная операция запроса).

Пока достаточно знать, что операция Sum позволяет вычислить сумму каждого элемента или члена каждого элемента из входной последовательности. Этот код выдаст следующие результаты:



Обратите внимание, что в этих результатах одна запись для каждого сотрудника содержит сумму всех записей о его опционах. Отличие от примера применения операции Join состоит в том, что там были отдельные записи для каждой записи опциона сотрудника.

Назад (2_6.php)	6	7	8	Вперед (2_8.php)

Пройди тесты

C# тест (легкий) (https://professorweb.ru/test/c-sharp-test.html)

.NET тест (средний) (https://professorweb.ru/test/asp-test.html)