عنوان: **انتخاب پویای فیلد ها در LINQ**
نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۰۳/۰۳ ۸:۳۰
آدرس: www.dotnettips.info
گروه‌ها: C#, LINQ

LINQ یک DLS بر مبنای .NET می باشد که برای پرس و جو در منابع داده ای مانند پایگاه‌های داده ، فایل‌های XML و یا لیستی از اشیاء درون حافظه کاربرد دارد.

یکی از بزرگترین مزیت‌های آن Syntax آسان و خوانا آن می‌باشد.

LINQ از 2 نوع نمادگذاری پشتیبانی می‌کند:

LINQ Inline یا query expressions :

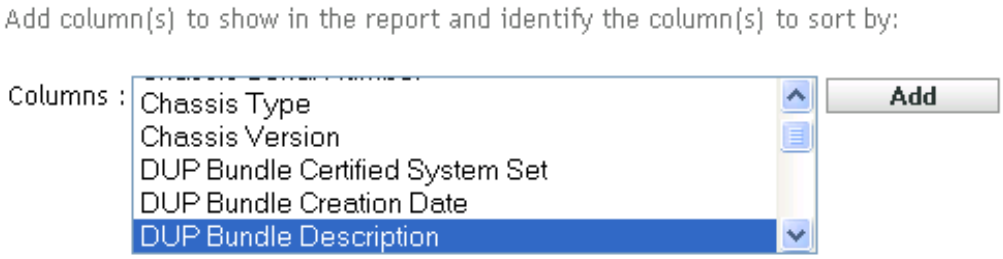
```
var result =
    from product in dbContext.Products
    where product.Category.Name == "Toys"
    where product.Price >= 2.50
    select product.Name;
```

Fluent Syntax :

```
var result = dbContext.Products
    .Where(p => p.Category.Name == "Toys" && p.Price >= 250)
    .Select(p => p.Name);
```

در پرس و جوهای بالا فیلدهای مورد نیاز در قسمت Select در زمان Compile شناخته شده هستند . اما گاهی ممکن است فیلدهای مورد نیاز در زمان اجرا مشخص شوند.

به عنوان مثال یک گزارش ساز پویا که کاربر مشخص می‌کند چه ستون هایی در خروجی نمایش داده شوند یا یک جستجوی پیشرفته که ستون‌های خروجی به اختیار کاربر در زمان اجرا مشخص می‌شوند.



این مدل را در نظر داشته باشید :

```
public class Student
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public string Field1 { get; set; }
```

```
        public string Field2 { get; set; }
        public string Field3 { get; set; }


        public static IEnumerable<Student> GetStudentSource()
        {
            for (int i = 0; i < 10; i++)
            {
                yield return new Student
                                {
                                    Id = i,
                                    Name = "Name " + i,
                                    Field1 = "Field1 " + i,
                                    Field2 = "Field2 " + i,
                                    Field3 = "Field3 " + i
                                };
            }
        }
    }
```

ستون‌های کلاس Student را در رابط کاربری برنامه جهت انتخاب به کاربر نمایش می‌دهیم. سپس کاربر یک یا چند ستون را انتخاب می‌کند که قسمت Select کوئری برنامه باید بر اساس فیلدهای مورد نظر کاربر مشخص شود.

یکی از روش هایی که می‌توان از آن بهره برد استفاده از کتاب خانه Dynamic LINQ معرفی شده در اینجا می باشد.

این کتابخانه جهت سهولت در نصب به کمک NuGet در این
آدرس قرار دارد.

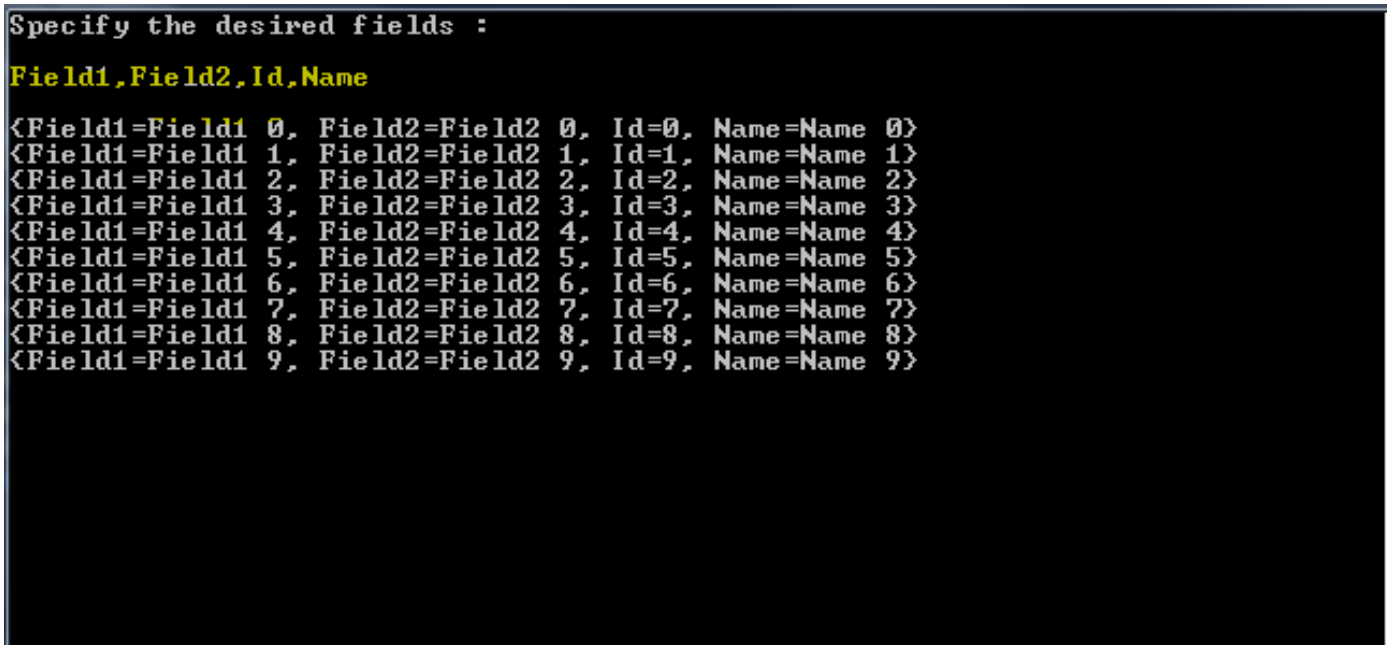
فرض بر این است که فیلدهای انتخاب شده توسط کاربر با " , " از یکدیگر جدا شده اند.

```
public class Program
    {
        private static void Main(string[] args)
        {
            System.Console.WriteLine("Specify the desired fields : ");
            string fields = System.Console.ReadLine();
            IEnumerable<Student> students = Student.GetStudentSource();
            IQueryable output = students.AsQueryable().Select(string.Format("new({0})", fields));
            foreach (object item in output)
            {
                System.Console.WriteLine(item);
            }

            System.Console.ReadKey();
        }

    }
```

همانطور که در عکس ذیل مشاهده می‌کنید پس از اجرای برنامه ، فیلدهای انتخاب شده توسط کاربر از منبع داده‌ی دریافت شده و در خروجی نمایش داده شده اند.



این روش مزایا و معایب خودش را دارد ، به عنوان مثال خروجی یک لیست از شیء Student نیست یا این Select فقط برای روی یک شیء IQueryable قابل انجام است.

روش دیگری که می‌توان از آن بهره جست استفاده از یک متد کمکی جهت تولید پویای عبارت Lambda ورودی Select می باشد :

```
public  class SelectBuilder <T>
    {
        public static Func<T, T> CreateNewStatement(string fields)
        {
            // input parameter "o"
            var xParameter = Expression.Parameter(typeof(T), "o");


            // new statement "new T()"
            var xNew = Expression.New(typeof(T));

            // create initializers
            var bindings = fields.Split(',').Select(o => o.Trim())
                .Select(o =>
                {

                    // property "Field1"
                    var property = typeof(T).GetProperty(o);

                    // original value "o.Field1"
                    var xOriginal = Expression.Property(xParameter, property);

                    // set value "Field1 = o.Field1"
                    return Expression.Bind(property, xOriginal);
                }
            ).ToList();

            // initialization "new T { Field1 = o.Field1, Field2 = o.Field2 }"
            var xInit = Expression.MemberInit(xNew, bindings);

            // expression "o => new T { Field1 = o.Field1, Field2 = o.Field2 }"
            var lambda = Expression.Lambda<Func<T, T>>(xInit, xParameter);

            // compile to Func<T, T>
            return lambda.Compile();
        }
    }
```

برای استفاده از متد CreateNewStatement باید اینگونه عمل کرد :

```
IEnumerable<Student> result = students.Select(SelectBuilder<Student>.CreateNewStatement("Field1,
Field2")).ToList();

            foreach (Student student in result)
            {
                System.Console.WriteLine(student.Field1);
            }
```

خروجی یک لیست از Student می باشد.

نحوه‌ی کارکرد CreateNewStatement :

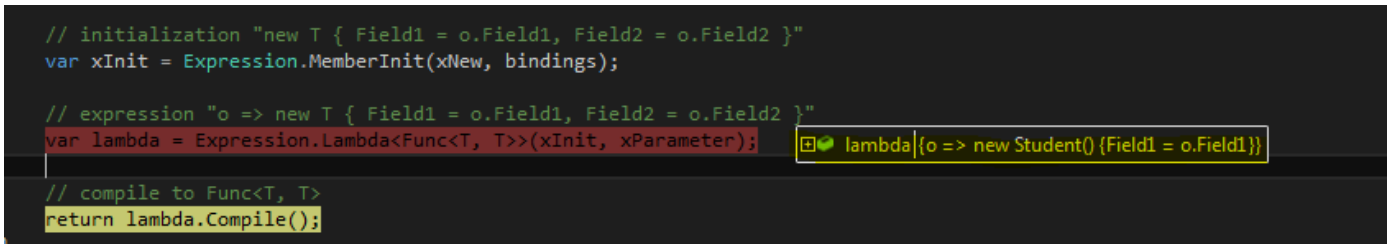
ابتدا فیلدهای انتخابی کاربر که با "," جدا شده اند به ورودی پاس داده می‌شود سپس یک statement خالی ایجاد می‌شود :

```
o=>new Student()
```

فیلدهای ورودی از یکدیگر تفکیک می‌شوند و به کمک Reflection پراپرتی معادل فیلد رشته ای در کلاس Student پیدا می‌شود :

```
var property = typeof(T).GetProperty(o);
```

سپس عبارت Select و تولید شیء جدید بر اساس فیلدهای ورودی تولید می‌شود و برای استفاده Compile به Func می‌شود. در نهایت Func تولید شده به Select پاس داده می‌شود و لیستی از Student بر مبنای فیلدهای انتخابی تولید می‌شود.



دریافت مثال : DynamicSelect.zip

نظرات خوانندگان

نویسنده: sorosh
تاریخ: ۱۳۹۲/۱۱/۱۲ ۷:۴۴

با سلام؛ با ایجاد ستون ردیف با Select new در LINQ مشکل دارم. طوریکه بصورت اتوماتیک یک ستون ردیف ایجاد نماییم:

```
var all = (from x in db.tblZones
select new
{
  RowNuber=???????????
  Code = x.xCode,
  Caption = x.xCaption,
  Comment = x.xComments,
  DT_RowId = "tr" + x.xCode.ToString(),

});
```

نویسنده: محسن خان
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۱:۳۲

یک متغیر count قبل از عبارتی که نوشتی ایجاد کن. اینبار جلوی RowNumber بنویس count++.

نویسنده: شاهین کیاست
تاریخ: ۱۳۹۲/۱۱/۱۲ ۱۱:۴۰

متد Select یک Overload دیگر دارد که Index را فراهم می‌کند :

```
string[] weekDays = { "Monday", "Tuesday", "Wednesday", "Thursday", "Friday" };
weekDays.Select((day, index) => new { Day = day, Index = index })
        .Where(x => x.Day.Contains("s"))
        .Select(x => x.Index)
        .ToArray();
```

البته در این کد از Lambda Syntax استفاده شده که برای کد شما هم ممکن است.

نویسنده: محمدرضا کنی
تاریخ: ۱۳۹۴/۰۷/۰۶ ۱۷:۲

سلام ، بسیار عالی بود
حالا همین کد رو چطور میشه برای قرار دادن ضابطه بصورت داینامیک گسترش داد

نویسنده: محسن خان
تاریخ: ۱۳۹۴/۰۷/۰۶ ۱۸:۴۰

یک نگاهی به پروژه جستجوی پویا با استفاده از Expression ها داشته باشید.