عنوان: **استفاده از LINQ جهت تهیه کدهایی کوتاه‌تر و خواناتر**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۰۴ ۲۰:۱۸:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: LINQ

با کمک امکانات ارائه شده توسط LINQ ، می‌توان بسیاری از اعمال برنامه نویسی را در حجمی کمتر، خواناتر و در نتیجه با قابلیت نگهداری بهتر، انجام داد که تعدادی از آن‌ها را در ادامه مرور خواهیم کرد.

الف) تهیه یک یک رشته، حاوی عناصر یک آرایه، جدا شده با کاما.

```
using System.Linq;

public class CLinq
{
 public static string GetCommaSeparatedListNormal(string[] data)
 {
     string items = string.Empty;

     foreach (var item in data)
     {
         items += item + ", ";
     }

     return items.Remove(items.Length - 2, 1).Trim();
 }

 public static string GetCommaSeparatedList(string[] data)
 {
     return data.Aggregate((s1, s2) => s1 + ", " + s2);
 }
}
```

همانطور که ملاحظه می‌کنید در روش دوم با استفاده از LINQ Aggregate extension method ، کد جمع و جورتر و خواناتری نسبت به روش اول حاصل شده است.

ب) پیدا کردن تعداد عناصر یک آرایه حاوی مقداری مشخص
برای مثال آرایه زیر را در نظر بگیرید:

```
var names = new[] { "name1", "name2", "name3", "name4", "name5", "name6", "name7" };
```

قصد داریم تعداد عناصر حاوی name را مشخص سازیم.
در تابع GetCountNormal زیر، این کار به شکلی متداول انجام شده و در GetCount از LINQ Count extension method کمک گرفته شده است.

```
using System.Linq;

public class CLinq
{
 public static int GetCountNormal()
 {
     var names = new[] { "name1", "name2", "name3", "name4", "name5", "name6", "name7" };
     var count = 0;
     foreach (var name in names)
     {
         if (name.Contains("name"))
             count += 1;
     }
     return count;
 }

 public static int GetCount()
```

```
 {
     var names = new[] { "name1", "name2", "name3", "name4", "name5", "name6", "name7" };
     return names.Count(name => name.Contains("name"));
 }
}
```

به نظر شما کدام روش خواناتر بوده و نگهداری و یا تغییر آن در آینده ساده‌تر می‌باشد؟

ج) دریافت لیستی از عناصر شروع شده با یک عبارت
در اینجا نیز دو روش متداول و استفاده از LINQ بررسی شده است.

```
using System.Linq;
using System.Collections.Generic;

public class CLinq
{
 public static List<string> GetListNormal()
 {
     List<string> sampleList = new List<string>() { "A1", "A2", "P1", "P10", "B1", "B@", "J30", "P12"
};
     List<string> result = new List<string>();
     foreach (var item in sampleList)
     {
         if (item.StartsWith("P"))
             result.Add(item);
     }
     return result;
 }

 public static List<string> GetList()
 {
     List<string> sampleList = new List<string>() { "A1", "A2", "P1", "P10", "B1", "B@", "J30", "P12"
};
     return sampleList.Where(x => x.StartsWith("P")).ToList();
 }
}
```

و در حالت کلی، اکثر حلقه‌های foreach متداول را می‌توان با نمونه‌های خواناتر کوئری‌های LINQ معادل، جایگزین کرد.

نظرات خوانندگان

نویسنده: افشار محبی
تاریخ: ۱۳۸۸/۰۸/۰۵ ۰۹:۱۰:۳۵

آیا LINQ در زبان‌های دیگر مثل جاوا معادلی دارد؟

نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۰۵ ۱۱:۵۳:۲۳

سلام
تلاش‌هایی به صورت مستقل برای جاوا هم شده (یکپارچه با زبان نیست)
http://xircles.codehaus.org/projects/quaere
اما باز هم پیاده سازی آن در بسیاری از موارد type safety دات نت را ندارد و از رشته‌ها کمک گرفته.
در کل جاوا به دلیل نداشتن معادلی برای lambda expressions که پایه و اساس LINQ را تشکیل می‌دهد، هنوز در این زمینه کار پایه‌ای را انجام نداده است و در کل قسمت LI مربوط به LINQ را ندارد (language integrated)

نویسنده: Majid
تاریخ: ۱۳۸۸/۱۱/۲۲ ۲۱:۳۸:۰۹

ضمن تشکر از مطالب خوبتان.
بد نیست نگاهی به این add-in بیاندازید، برای من که جالب بود
http://code.msdn.microsoft.com/vlinq