عنوان: **پیدا کردن آیتم‌های تکراری در یک لیست به کمک LINQ**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۸/۰۶ ۲۱:۴۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: LINQ

گاهی از اوقات نیاز می‌شود تا در یک لیست، آیتم‌های تکراری موجود را مشخص کرد. به صورت پیش فرض متد Distinct برای حذف مقادیر تکراری در یک لیست با استفاده از LINQ موجود است که البته آن‌هم اما و اگرهایی دارد که در ادامه به آن پرداخته خواهد شد، اما باز هم این مورد پاسخ سؤال اصلی نیست (نمی‌خواهیم موارد تکراری را حذف کنیم).

برای حذف آیتم‌های تکراری از یک لیست جنریک می‌توان متد زیر را نوشت:

```
public static List<T> RemoveDuplicates<T>(List<T> items)
{
   return (from s in items  select s).Distinct().ToList();
}
```

برای مثال:

```
public static void TestRemoveDuplicates()
{
   List<string> sampleList =
       new List<string>() { "A1", "A2", "A3", "A1", "A2", "A3" };
   sampleList = RemoveDuplicates(sampleList);
   foreach (var item in sampleList)
       Console.WriteLine(item);
}
```

این متد بر روی لیست‌هایی با نوع‌های اولیه مانند string و int و امثال آن درست کار می‌کند. اما اکنون مثال زیر را در نظر بگیرید:

```
public class Employee
{
   public int ID { get; set; }
   public string FName { get; set; }
   public int Age { get; set; }
}

public static void TestRemoveDuplicates()
{
   List<Employee> lstEmp = new List<Employee>()
   {
      new  Employee(){ ID=1, Age=20, FName="F1"},
      new  Employee(){ ID=2, Age=21, FName="F2"},
      new  Employee(){ ID=1, Age=20, FName="F1"},
   };

   lstEmp = RemoveDuplicates<Employee>(lstEmp);

   foreach (var item in lstEmp)
       Console.WriteLine(item.FName);
}
```

اگر متد TestRemoveDuplicates را اجرا نمائید، رکورد تکراری این لیست جنریک حذف نخواهد شد؛ زیرا متد distinct بکارگرفته شده نمی‌داند اشیایی از نوع کلاس سفارشی Employee را چگونه باید با هم مقایسه نماید تا بتواند موارد تکراری آن‌ها را حذف کند.
برای رفع این مشکل باید از آرگومان دوم متد distinct جهت معرفی وهله‌ای از کلاسی که اینترفیس IEqualityComparer را پیاده سازی می‌کند، کمک گرفت.

```
public static IEnumerable<TSource> Distinct<TSource>(this IEnumerable<TSource> source,
IEqualityComparer<TSource> comparer);
```

که نمونه‌ای از پیاده سازی آن به شرح زیر می‌تواند باشد:

```
public class EmployeeComparer : IEqualityComparer<Employee>
{
 public bool Equals(Employee x, Employee y)
 {
   //آیا دقیقا یک وهله هستند؟
   if (Object.ReferenceEquals(x, y)) return true;

   //آیا یکی از وهله‌ها نال است؟
   if (Object.ReferenceEquals(x, null) ||
       Object.ReferenceEquals(y, null))
       return false;

   return x.Age == y.Age && x.FName == y.FName && x.ID == y.ID;
 }

 public int GetHashCode(Employee obj)
 {
   if (Object.ReferenceEquals(obj, null)) return 0;
   int hashTextual = obj.FName == null ? 0 : obj.FName.GetHashCode();
   int hashDigital = obj.Age.GetHashCode();
   return hashTextual ^ hashDigital;
 }
}
```

اکنون اگر یک overload برای متد RemoveDuplicates با درنظر گرفتن IEqualityComparerتهیه کنیم، به شکل زیر خواهد بود:

```
public static List<T> RemoveDuplicates<T>(List<T> items, IEqualityComparer<T> comparer)
{
   return (from s in items select s).Distinct(comparer).ToList();
}
```

به این صورت متد آزمایشی ما به شکل زیر (که وهله‌ای از کلاس EmployeeComparer به آن ارسال شده) تغییر خواهد کرد:

```
public static void TestRemoveDuplicates()
{
   List<Employee> lstEmp = new List<Employee>()
   {
      new  Employee(){ ID=1, Age=20, FName="F1"},
      new  Employee(){ ID=2, Age=21, FName="F2"},
      new  Employee(){ ID=1, Age=20, FName="F1"},
   };

   lstEmp = RemoveDuplicates(lstEmp, new EmployeeComparer());

   foreach (var item in lstEmp)
       Console.WriteLine(item.FName);
}
```

پس از این تغییر، حاصل این متد تنها دو رکورد غیرتکراری می‌باشد.

سؤال: برای یافتن آیتم‌های تکراری یک لیست چه باید کرد؟
احتمالا مقاله " روش‌هایی برای حذف رکوردهای تکراری " را به خاطر دارید. اینجا هم می‌توان کوئری LINQ ایی را نوشت که رکوردها را بر اساس سن، گروه بندی کرده و سپس گروه‌هایی را که بیش از یک رکورد دارند، انتخاب نماید.

```
public static void FindDuplicates()
{
   List<Employee> lstEmp = new List<Employee>()
   {
      new  Employee(){ ID=1, Age=20, FName="F1"},
      new  Employee(){ ID=2, Age=21, FName="F2"},
      new  Employee(){ ID=1, Age=20, FName="F1"},
```

```
    };

    var query = from c in lstEmp
                group c by c.Age into g
                where g.Count() > 1
                select new { Age = g.Key, Count = g.Count() };

    foreach (var item in query)
    {
        Console.WriteLine("Age {0} has {1} records", item.Age, item.Count);
    }
}
```