عنوان: **استثنای Sequence contains no elements در حین استفاده از LINQ**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۸/۰۶/۰۸ ۲۱:۲۵:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: LINQ

در ابتدا مثال‌های زیر را در نظر بگیرید:

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

namespace testWinForms87
{
  public class Data
  {
      public int id { get; set; }
      public string name { get; set; }
  }

  class CLinqTests
  {
      public static int TestGetListMin1()
      {
          var lst = new List<Data>
          {
                  new Data{ id=1, name="id1"},
                  new Data{ id=2, name="id2"},
                  new Data{ id=3, name="name3"}
          };

          return (from c in lst
                  where c.name.Contains("id")
                  select c.id).Min();
      }

      public static int TestGetListMin2()
      {
          var lst = new List<Data>();

          return (from c in lst
                  where c.name.Contains("id")
                  select c.id).Min();
      }
  }
}
```

در متد TestGetListMin1 قصد داریم کوچکترین آی دی رکوردهایی را که نام آن‌ها حاوی id است، از لیست تشکیل شده از کلاس Data بدست آوریم (همانطور که مشخص است سه رکورد از نوع Data در لیست lst ما قرار گرفته‌اند).
محاسبات آن کار می‌کند و مشکلی هم ندارد. اما همیشه در دنیای واقعی همه چیز قرار نیست به این خوبی پیش برود. ممکن است همانند متد TestGetListMin2 ، لیست ما خالی باشد (برای مثال از دیتابیس، رکوردی مطابق شرایط کوئری‌های قبلی بازگشت داده نشده باشد). در این حالت هنگام فراخوانی متد Min ، استثنای Sequence contains no elements رخ خواهد داد و همانطور که در مباحث defensive programming عنوان شد، وظیفه‌ی ما این نیست که خودرو را به دیوار کوبیده (یا منتظر شویم تا کوبیده شود) و سپس به فکر چاره بیفتیم که خوب، عجب! مشکلی رخ داده است!
اکنون چه باید کرد؟ حداقل یک مرحله بررسی اینکه آیا کوئری ما حاوی رکوردی می‌باشد یا خیر باید به این متد اضافه شود (به صورت زیر):

```
public static int TestGetListMin3()
      {
          var lst = new List<Data>();
          var query = from c in lst
                      where c.name.Contains("id")
                      select c.id;

          if (query.Any())
              return query.Min();
```

```
            else
                return -1;
        }
```

البته می‌شد اگر هیچ رکوردی بازگشت داده نمی‌شد، یک استثنای سفارشی را ایجاد کرد، اما به شخصه ترجیح می‌دهم عدد منهای یک را بر گردانم (چون می‌دانم رکوردهای من عدد مثبت هستند و اگر حاصل منفی شد نیازی به ادامه‌ی پروسه نیست).

شبیه به این مورد در هنگام استفاده از تابع Single مربوط به LINQ نیز ممکن است رخ دهد (تولید استثنای ذکر شده) اما در اینجا مایکروسافت تابع SingleOrDefault را نیز پیش بینی کرده است. در این حالت اگر کوئری ما رکوردی را برنگرداند، SingleOrDefault مقدار نال را برگشت داده و استثنایی رخ نخواهد داد (نمونه‌ی دیگر آن متدهای First و FirstOrDefault هستند).
در مورد متدهای Min و Max ، متدهای MinOrDefault یا MaxOrDefault در دات نت فریم ورک وجود ندارند. می‌توان این نقیصه را با استفاده از extension methods برطرف کرد.

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

  public static class LinqExtensions
  {
      public static T MinOrDefault<T>(this IEnumerable<T> source, T defaultValue)
      {
          if (source.Any<T>())
              return source.Min<T>();

          return defaultValue;
      }

      public static T MaxOrDefault<T>(this IEnumerable<T> source, T defaultValue)
      {
          if (source.Any<T>())
              return source.Max<T>();

          return defaultValue;
      }
  }
```

اکنون با استفاده از extension methods فوق، کد ما به صورت زیر تغییر خواهد کرد:

```
public static int TestGetListMin4()
      {
          var lst = new List<Data>();
          return (from c in lst
                  where c.name.Contains("id")
                  select c.id).MinOrDefault(-1);
      }
```