عنوان: **Any یا Count**
نویسنده: وحید نصیری
تاریخ: ۱۳۸۹/۰۸/۰۵ ۱۸:۲۴:۰۰
آدرس: www.dotnettips.info
برچسب‌ها: LINQ

با وجود امکانات مهیای توسط LINQ ، یک سری از عادات متداول حین کار با گروهی از اشیاء باید کنار گذاشته شوند؛ برای مثال چگونگی بررسی این مطلب که آیا شیء IEnumerable ما حاوی عنصری هست یا خیر.
روش متداول انجام اینکار استفاده از متد Count است. چون این متد پیش از تدارک امکانات LINQ نیز وجود داشته، بنابراین اولین موردی که جهت بررسی آن به ذهن خطور می‌کند، استفاده از متد Count می‌باشد؛ برای مثال:

```
void Method(IEnumerable<Status> statuses)
{
 if (statuses != null && statuses.Count() > 0)
 // do something...
}
```

این روش بهینه نیست زیرا کار متد Count بررسی تک تک عناصر شیء IEnumerable و سپس بازگرداندن تعداد آن‌ها است. این مورد خصوصا در حالت‌های کار با بانک اطلاعاتی و تنظیمات lazy-loading آن و یا تعداد بالای عناصر یک لیست، بسیار هزینه‌بر خواهد شد.
ولی در اینجا هدف ما این است که آیا شیء IEnumerable دارای حداقل یک عنصر است یا خیر؟ بنابراین بجای استفاده از متد Count بهتر است از یکی از extension methods فراهم شده توسط LINQ به نام Any استفاده شود.
کار متد Any ، پس از بررسی اولین عنصر یک مجموعه، خاتمه خواهد یافت و بدیهی است که نسبت به متد Count بسیار سریعتر و کم هزینه‌تر خواهد بود. علاوه بر آن حین کار با بانک‌های اطلاعاتی برای مثال توسط LINQ to Entities ، در SQL نهایی تولیدی به EXISTS ترجمه خواهد شد.

```
void Method(IEnumerable<Status> statuses)
{
 if (statuses != null && statuses.Any())
 // do something...
}
```

خلاصه‌ی بحث:
از این پس حین استفاده از انواع و اقسام لیست‌ها، آرایه‌ها، IEnumerable ها و امثال آن‌ها، جهت بررسی خالی بودن یا نبودن آن‌ها تنها از متد Any فراهم شده توسط LINQ استفاده نمائید.

```
if (myArray != null && myArray.Any())
// do something...
```

نظرات خوانندگان

نویسنده: Meysam
تاریخ: ۱۳۸۹/۰۸/۰۵ ۲۰:۲۷:۰۸

نکته ای که گفتین، زمانی که از dotTrace استفاده بکنید، به وضوح میبینید.

نویسنده: علی اقدم
تاریخ: ۱۳۸۹/۰۸/۰۶ ۰۰:۲۲:۵۳

نکته قابل توجهی بود،ممنون

نویسنده: Hosein Khoshraftar
تاریخ: ۱۳۸۹/۰۸/۱۰ ۰۷:۴۲:۱۳

خیلی نکته جالبی بود
ممنونم . من هم خودم همیشه از کانت استفاده می کردم