عنوان: **مثال‌هایی بیشتر از عملکرد پشته‌ای MSIL**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۲/۰۵/۱۳ ۱۵:۱۶
آدرس: www.dotnettips.info
برچسب‌ها: C#, CIL, CLR, IL, MSIL, Reflection

بررسی عملکرد و کدهای IL یک متد

```
ldarg.0
        stloc_0
L_0000:
        ldloc_0
        ldc_i4 5
        add
        stloc_0
        ldloc_0
        ldc_i4 15
        blt L_0000
        ldloc_0
        ret
```

به کدهای IL فوق دقت کنید. در ادامه قصد داریم عملکرد این متد را بررسی کرده و سپس سعی کنیم تا معادل سی شارپ آن‌را حدس بزنیم. (البته سعی کنید طوری مطلب را مطالعه کنید که ادامه بحث را در ابتدا مشاهده نکنید!)
این متد، یک مقدار int را دریافت کرده و با انجام محاسباتی بر روی آن، مقدار int دیگری را بازگشت می‌دهد.
کار با ldarg.0 شروع می‌شود. به این ترتیب آرگومان موجود در ایندکس صفر، بر روی پشته بارگذاری خواهد شد. فرض کنید ورودی 5 را به این متد ارسال کرده‌ایم.
سپس stloc_0 این مقدار را از پشته pop کرده و در یک متغیر محلی ذخیره می‌کند.
در ادامه برچسب L_0000 تعریف شده است. از برچسب‌ها برای انتقال جریان اجرایی برنامه استفاده می‌کنیم.
ldloc_0 به معنای بارگذاری متغیر محلی از ایندکس صفر است. به این ترتیب عدد 5 بر روی پشته ارزیابی قرار می‌گیرد.
توسط ldc_i4 5، یک i4 یا 32 بیتی یا int ایی با 4 بایت، به عنوان یک عدد ثابت بارگذاری می‌شود. این عدد نیز بر روی پشته ارزیابی قرار می‌گیرد.
در ادامه با فراخوانی Add، دو مقدار قرار گرفته بر روی پشته pop شده و نتیجه 10؛ مجددا بر روی پشته قرار می‌گیرد.
stloc_0 سبب می‌شود تا این عدد 10 در یک متغیر محلی در ایندکس صفر ذخیره شود.
با فراخوانی ldloc_0، این متغیر محلی به پشته ارزیابی منتقل می‌شود.
به کمک ldc_i4 15، یک عدد صحیح 4 بایتی با مقدار ثابت 15 بارگذاری می‌شود.
در ادامه blt بررسی می‌کند که اگر 10 کوچکتر است از 15 ایی که بر روی پشته قرار گرفته، آنگاه جریان عملیات را به برچسب L_0000 منتقل می‌کند (پرش به برچسب صورت خواهد گرفت).
اگر با سایر زبان‌های اسمبلی کار کرده باشید با lt، gt و امثال آن به طور قطع آشنایی دارید. در اینجا blt به معنای branch less than equal است.
بنابراین در ادامه مجددا همین اعمال فوق تکرار خواهند شد تا به ارزیابی blt جهت دو مقدار 15 با 15 برسیم. از آنجائیکه اینبار 15 کوچکتر از 15 نیست، سطر پس از آن یعنی ldloc_0 اجرا می‌شود که معادل است با بارگذاری 15 به پشته ارزیابی و سپس فراخوانی return می‌شود تا این مقدار را بازگشت دهد.

خوب؛ آیا می‌توانید کدهای معادل سی‌شارپ آن‌را حدس بزنید؟!

```
public static int Calculate(int x)
        {
            for (; x < 15; x += 5)
            {
            }
            return x;
        }
```

بله. متد محاسباتی که در ابتدای بحث کدهای IL آن‌را ملاحظه نمودید، یک چنین معادل سی‌شارپی دارد.

فراخوانی متدها در MSIL

برای فراخوانی متدها در کدهای IL از OpCode ایی به نام call استفاده می‌شود:

```
ldstr "hello world"
call void [mscorlib]System.Console::WriteLine(string)
```

در این مثال توسط ldstr، یک رشته بارگذاری شده و سپس توسط call اطلاعات متدی که باید فراخوانی شود، ذکر می‌گردد. همانطور که ملاحظه می‌کنید، امضای کامل متد نیاز است ذکر گردد؛ متدی از نوع void قرار گرفته در mscorlib با ذکر فضای نام و نام متد مورد نظر.

بررسی یک الگوریتم بازگشتی در MSIL

ابتدا متد بازگشتی ذیل را درنظر بگیرید:

```
public static int Calculate(int n)
        {
            if (n <= 1) return n;
            return n * Calculate(n - 1);
        }
```

اگر کد دی‌کامپایل شده‌ی آن را در ILSpy بررسی کنیم، به دستورات ذیل خواهیم رسید:

```
.method public hidebysig static
int32 Calculate (
int32 n
) cil managed
{
// Method begins at RVA 0x3c0d
// Code size 17 (0x11)
.maxstack 8

IL_0000: ldarg.0
IL_0001: ldc.i4.1
IL_0002: bgt.s IL_0006

IL_0004: ldarg.0
IL_0005: ret

IL_0006: ldarg.0
IL_0007: ldarg.0
IL_0008: ldc.i4.1
IL_0009: sub
IL_000a: call int32 FastReflectionTests.Test::Calculate(int32)
IL_000f: mul
IL_0010: ret
} // end of method Test::Calculate
```

در اینجا ابتدا مقدار آرگومان متد، توسط ldarg بارگذاری می‌شود. سپس مقدار ثابت یک بارگذاری می‌شود. توسط bgt بررسی خواهد شد اگر مقدار آرگومان (عدد اول) بزرگتر است از مقدار عدد ثابت یک (عدد دوم)، آنگاه به برچسب IL_0006 پرش صورت گیرد و قسمت ضرب بازگشتی انجام شود. در غیراینصورت آرگومان متد بارگذاری شده و در سطر IL_0005 کار خاتمه می‌یابد.
در سطرهای IL_0006 و IL_0007، دوبار کار بارگذاری آرگومان متد انجام شده است؛ یکبار برای عملیات ضرب و بار دیگر برای عملیات تفریق از یک.
سپس عدد ثابت یک بارگذاری شده و از مقدار آرگومان، توسط sub کسر خواهد شد. در ادامه متد Calculate به صورت بازگشتی فراخونی می‌گردد. در این حالت دوباره به سطر IL_0000 هدایت خواهیم شد و عملیات ادامه می‌یابد.