عنوان: **آشنایی با CLR: قسمت دوم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۴/۰۳/۲۳ ۰:۵۰
آدرس: www.dotnettips.info
گروه‌ها: CLR, .NET Framework

متادیتاهای یک ماژول مدیریت شده Managed Module

در قسمت قبلی به اصل وجودی CLR پرداختیم. در این قسمت تا حدودی به بررسی ماژول مدیریت شده managed module که از زبان‌های دیگر، کامپایل شده و به زبان میانی تبدیل گشته است صحبت می‌کنیم.

یک ماژول مدیریت شده شامل بخش‌های زیر است:

نام بخش	توضیح
هدر PE32 یا +PE32	CLR باید بداند که برنامه‌ی نوشته شده قرار است روی چه پلتفرمی و با چه معماری، اجرا گردد. این برنامه یک برنامه‌ی 32 بیتی است یا 64 بیتی. همچنین این هدر اشاره می‌کند که نوع فایل از چه نوعی است؛ GUI,CUI یا DLL. به علاوه تاریخ ایجاد یا کامپایل فایل هم در آن ذکر شده است. در صورتیکه این فایل شامل کدهای بومی native CPU هم باشد، اطلاعاتی در مورد این نوع کدها نیز در این هدر ذکر می‌شود و اگر ماژول ارائه شده تنها شامل کد IL باشد، قسمت بزرگی از اطلاعات این هدر در نظر گرفته نمی‌شود.
CLR Header	اطلاعاتی را در مورد CLR ارائه می‌کند. اینکه برای اجرا به چه ورژنی از CLR نیاز دارد. منابع مورد استفاده. آدرس و اندازه جداول و فایل‌های متادیتا و جزئیات دیگر.
metadata	هر کد یا ماژول مدیریت شده‌ای، شامل جداول متادیتا است که این جداول بر دو نوع هستند. اول جداولی که نوع‌ها و اعضای تعریف شده در کد را توصیف می‌کنند و دومی جداولی که نوع‌ها و اعضایی را که در کد به آن ارجاع شده است، توصیف می‌کنند.
IL Code	اینجا محل قرار گیری کدهای میانی تبدیل شده است که در زمان اجرا، CLR آن‌ها را به کدهای بومی تبدیل می‌کند.

کامپایلرهایی که بر اساس CLR کار می‌کنند، وظیفه دارند جداول متادیتاها را به طور کامل ساخته و داخل فایل نهایی embed کنند. متادیتاها مجموعه‌ی کاملی از فناوری‌های قدیمی چون فایل‌های COM یا Component Object Model و همچنین IDL یا Interface Definition (Description) Language هستند. گفتیم که متادیتاها همیشه داخل فایل IL که ممکن است DLL باشد یا EXE، ترکیب یا Embed شده‌اند و جدایی آن‌ها غیر ممکن است. در واقع کامپایلر در یک زمان، هم کد IL و هم متادیتاها را تولید کرده و آن‌ها را به صورت یک نتیجه‌ی واحد در می‌آورد.

متادیتاها استفاده‌های زیادی دارند که در زیر به تعدادی از آنان اشاره می‌کنیم:
موقع کامپایل نیاز به هدرهای C و C++ از بین می‌رود؛ چرا که فایل نهایی شامل تمامی اطلاعات ارجاع شده می‌باشد. کامپایلرها می‌توانند مستقیما اطلاعات را از داخل متادیتاها بخوانند.
ویژوال استودیو از آن‌ها برای کدنویسی راحت‌تر بهره می‌گیرد. با استفاده از قابلیت IntelliSense، متادیتاها به شما خواهند گفت چه متدهایی، چه پراپرتی‌هایی، چه رویدادهایی و ... در دسترس شماست و هر متد انتظار چه پارامترهایی را از شما دارد.
CLR Code Verification از متادیتا برای اینکه اطمینان کسب کند که کدها تنها عملیات type Safe را انجام می‌دهند، استفاده می‌کند.

متادیتاها به فیلد یک شیء اجازه می‌دهند که خود را به داخل بلوک‌های حافظ انتقال داده و بعد از ارسال به یک ماشین دیگر، همان شیء را با همان وضعیت، ایجاد نماید.
متادیتاها به GC اجازه می‌دهند که طول عمر یک شیء را رصد کند. GC برای هر شیء موجود می‌تواند نوع هر شیء را تشخیص داده و از طریق متادیتاها می‌تواند تشخیص دهد که فیلدهای یک شیء به اشیاء دیگری هم متصل هستند.

در آینده بیشتر در مورد متادیتاها صحبت خواهیم کرد.