عنوان: **آشنایی با CLR: قسمت بیستم**
نویسنده: علی یگانه مقدم
تاریخ: ۱۳۹۴/۰۶/۰۴ ۱:۳۵
آدرس: www.dotnettips.info
گروه‌ها: CLR, .NET Framework

در قسمت قبلی با نحوه انتشار برنامه‌ها آشنا شدیم. در این قسمت نحوه پیکربندی یا تغییر پیکربندی برنامه را مشخص می‌کنیم. کاربر یا مدیر سیستم بهتر از هر کسی می‌تواند جنبه‌های مختلف اجرای برنامه را مشخص کند. به عنوان نمونه ممکن است مدیر سیستم بخواهد فایل‌های یک برنامه را سمت هارد دیسک سیستم کاربر انتقال دهد یا اطلاعات مانیفست یک اسمبلی را رونویسی کند و مباحث نسخه بندی که در آینده در مورد آن صحبت می‌کنیم.
با ارائه یک فایل پیکربندی در شاخه برنامه می‌توان به مدیر سسیتم اجازه داد تا کنترل بیشتر بر روی برنامه داشته باشد. ناشر برنامه می‌تواند این فایل را همراه دیگر فایل‌های برنامه پکیج کند تا در شاخه برنامه نصب شود تا بعدا مدیر یا کاربر سیستم بتوانند آن را تغییر و ویرایش کنند. CLR هم محتوای این فایل را تفسیر کرده و قوانین بارگیری اسمبلی‌ها و ... را تغییر می‌دهد. این فایل پیکربندی می‌تواند به صورت XML هم ارائه شود. مزیت قرار دادن یک فایل جداگانه نسبت به رجیستری این مزیت را دارد که هم قابل جابجایی و پشتیبانی گیری است و هم اینکه تغییر آن ساده‌تر است.
البته در آینده بیشتر در مورد این فایل صحبت می‌کنیم ولی در حال حاضر بهتر است اندکی طعم آن را بچشیم. فرض را بر این می‌گذاریم که ناشر می‌خواهد فایل‌های اسمبلی MultiFileLibrary را در دایرکتوری جداگانه‌ای قرار دهد و چیزی شبیه به ساختار زیر را در نظر دارد:

```
AppDir directory (contains the application’s assembly files)
Program.exe
Program.exe.config (discussed below)


AuxFiles subdirectory (contains MultiFileLibrary’s assembly files)
MultiFileLibrary.dll
FUT.netmodule
RUT.netmodule
```

حال با تنظیم بالا به دلیل اینکه CLR انتظار دارد این اسمبلی را در دایرکتوری برنامه بیابد و با این جابجایی قادر به انجام این کار نیست، استثنای زیر را صادر می‌کند:

```
System.IO.FileNotFoundException
```

برای حل این مشکل، ناشر یک فایل XML را ایجاد کرده و در مسیر دایرکتوری برنامه قرار می‌دهد. این فایل باید همنام اسمبلی اصلی برنامه با پسوند config. باشد. به عنوان مثال، نام فایل می‌شود: Program.exe.config و فایل پیکربندی هم چیزی شبیه فایل زیر می‌شود:

```
<configuration>
<runtime>
<assemblyBinding xmlns="urn:schemas-microsoft-com:asm.v1">
<probing privatePath="AuxFiles" />
</assemblyBinding>
</runtime>
</configuration>
```

حالا هر موقع CLR در جست و جوی یک اسمبلی باشد که نتواند آن را در دایرکتوری مربوطه بیابد مسیر Auxfiles را هم بررسی خواهد کرد. با قرار دادن کارکتر « , » هم می‌توان برای خصوصیت PrivatePath، دایرکتوری‌های زیادتری را معرفی کرد. البته مسیردهی این خصوصیت باید به طور نسبی باشد نه مطلق یا اینکه یک مسیر نسبی خارج از دایرکتوری برنامه. ایده اصلی اینکار این است که برنامه کنترل بیشتری روی دایرکتوری خود و دایرکتوری‌های زیرمجموعه داشته باشد نه خارج از آن.

نحوه عملکرد اسکن CLR برای بارگزاری اسمبلی ها

موقعی که CLR قصد بارگزاری اسمبلی‌های خنثی را دارد، شاخه‌های زیر را به طور خودکار اسکن خواهد نمود (مسیرهای FirstPrivatePath و SecondPrivatePath توسط فایل پیکربندی مشخص شده است)

```
AppDir\AsmName.dll
AppDir\AsmName\AsmName.dll
```

```
AppDir\firstPrivatePath\AsmName.dll
AppDir\firstPrivatePath\AsmName\AsmName.dll
AppDir\secondPrivatePath\AsmName.dll
AppDir\secondPrivatePath\AsmName\AsmName.dll
...
```

این نکته ضروری است که اگر اسمبلی شما در یک دایرکتوری همنام خودش (در مثال ما MultiFileLibrary) قرار بگیرد، نیازی نیست این مسیر را در فایل پیکربندی ذکر کنید؛ زیرا CLR در صورت نیافتن دایرکتوری با این نام را اسکن خواهد نمود. بعد از آن اگر به هر نحوی CLR نتواند اسمبلی را در هیچ کدام از دایرکتوری‌های گفته شده بیابد، با همان قوانین گفته شده اینبار به دنبال فایلی با پسوند exe خواهد بود و اگر باز هم جست و جوی آن نتیجه‌ای را در بر نداشته باشد، استثنای زیر را صادر می‌کند:

```
FileNotFoundException
```

برای اسمبلی‌های ماهواره‌ای همان قوانین بالا دنبال می‌شود؛ با این تفاوت که انتظار می‌رود اسمبلی داخل یک زیر دایرکتوری با تگ‌های RFC1766 مطابقت داشته باشد. به عنوان مثال اگر اسمبلی با فرهنگ و منطقه En-US مشخص شده باشد، دایرکتوری‌های زیر اسکن خواهند شد:

```
C:\AppDir\en-US\AsmName.dll
C:\AppDir\en-US\AsmName\AsmName.dll
C:\AppDir\firstPrivatePath\en-US\AsmName.dll
C:\AppDir\firstPrivatePath\en-US\AsmName\AsmName.dll
C:\AppDir\secondPrivatePath\en-US\AsmName.dll
C:\AppDir\secondPrivatePath\en-US\AsmName\AsmName.dll
C:\AppDir\en-US\AsmName.exe
C:\AppDir\en-US\AsmName\AsmName.exe
C:\AppDir\firstPrivatePath\en-US\AsmName.exe
C:\AppDir\firstPrivatePath\en-US\AsmName\AsmName.exe
C:\AppDir\secondPrivatePath\en-US\AsmName.exe
C:\AppDir\secondPrivatePath\en-US\AsmName\AsmName.exe
C:\AppDir\en\AsmName.dll
C:\AppDir\en\AsmName\AsmName.dll
C:\AppDir\firstPrivatePath\en\AsmName.dll
C:\AppDir\firstPrivatePath\en\AsmName\AsmName.dll
C:\AppDir\secondPrivatePath\en\AsmName.dll
C:\AppDir\secondPrivatePath\en\AsmName\AsmName.dll
C:\AppDir\en\AsmName.exe
C:\AppDir\en\AsmName\AsmName.exe
C:\AppDir\firstPrivatePath\en\AsmName.exe
C:\AppDir\firstPrivatePath\en\AsmName\AsmName.exe
C:\AppDir\secondPrivatePath\en\AsmName.exe
C:\AppDir\secondPrivatePath\en\AsmName\AsmName.exe
```

نحوه اسکن کردن CLR میتواند به ما بگوید که عمل اسکن می‌تواند گاهی اوقات با زمان زیادی روبرو شود (به خصوص که قابلیت اسکن روی شبکه را هم دارد). برای محدود کردن ناحیه یا نواحی اسکن می‌توانید یک یا چند المان culture را در فایل پیکربندی مشخص کنید. همچنین مایکروسافت ابزاری به نام FUSLogVw.exe را ارائه داده است که می‌تواند نواحی اسکن را در حین اجرای برنامه به ما گزارش دهد.

نام و محل فایل پیکربندی بسته به نوع برنامه می‌تواند متغیر باشد:
برای فایل‌های اجرایی EXE فایل پیکربندی باید در شاخه فایل اجرایی باشد و باید نام فایل پیکربندی همانند فایل exe بوده و یک پسوند config. را به آن اضافه کرد.

برای برنامه‌های وب فرم، فایل web.config موجود است که در ریشه شاخه مجازی وب اپلیکیشن قرار میگیرد و هر زیر دایرکتوری هم می‌تواند یک web.config جداگانه داشته باشد که میتواند از web.config ریشه هم تنظیماتش را ارث بری کند. برای نمونه آدرس زیر را در نظر بگیرید:

```
http://www.dotnettips.info/newsarchive
```

یک فایل کانفیگ در ریشه قرار می‌گیرد و یکی هم در زیر شاخه newsArchive می‌تواند قرار بگیرد.

فصل دوم « *نحوه ساخت و توزیع اسمبلی ها* » از بخش اول « *اصول اولیه CLR* » پایان یافت. فصل بعدی در مورد اسمبلی‌های اشتراکی است که بعد از آماده شدن این فصل، قسمت‌های بعدی در دسترس عزیزان قرار خواهد گرفت.