ویندوز اینترنالز

«بخش ۱: مفاهیم پایه و ابزارها»

تاریخ تالیف: دوشنبه – ۲۴ شهریور ۱۳۹۹

تهیه شده توسط تیم فنی آزمایشگاه امنیت کی‌پاد

مفاهیم و ابزارها

در قسمت اول سلسله مقالات ویندوز اینترنالز، ما مفاهیم کلیدی و اصطلاحات پایه سامانه‌عامل ویندوز که در طول این سلسله مقالات استفاده خواهیم کرد، از قبیل رابط‌های برنامه‌نویسی ویندوز[1]، فناوری COM، فناوری WinRT، فریمورک .NET، مُد کاربر و مُد کرنل، مفهوم OneCore، و دیگر مفاهیم مهم ویندوزی را مورد بررسی قرار خواهیم داد. در قسمت‌های بعدی سری مقالات ویندوز اینترنالز، به مفاهیم دیگر مانند پروسه‌ها، تردها، جاب‌ها، حافظه مجازی، نقش MMU و ... خواهیم پرداخت تا مرحله به مرحله با معماری ویندوز آشنا شویم.

کلیدواژه:

معماری ویندوز، مفاهیم ویندوز، ویندوز اینترنالز

نسخه‌های سامانه‌عامل ویندوز

این سلسله مقالات جدیدترین نسخه سامانه‌عامل سرور و کلاینت مایکروسافت از قبیل ویندوز ویندوز ۱۰ (نسخه ۳۲ بیتی، نسخه آرم و نسخه ۶۴ بیتی)، ویندوز سرور ۲۰۱۲ (نسخه ۶۴ بیتی) را پوشش می‌دهند. مگر در مواردی خاص که متن این سلسله مقالات برای تمامی نسخه‌های سامانه‌عامل ویندوز کاربرد دارد. با این حال، به عنوان اطلاعات پایه، جدول ۱ نام‌ها، شماره نسخه داخلی و تاریخ انتشار محصولات ویندوز را لیست کرده است.

جدول ۱: نسخه‌های مختلف منتشر شده سامانه‌عامل ویندوز

نام محصول	شماره داخلی	تاریخ انتشار
ویندوز NT 3.1	3.1	July 1993
ویندوز NT 3.5	3.5	September 1994
ویندوز NT 3.5.1	3.5.1	May 1995
ویندوز NT 4.0	4.0	July 1996
ویندوز ۲۰۰۰	5.0	December 1999
ویندوز XP	5.1	August 2001
ویندوز سرور ۲۰۰۳	5.2	March 2003
ویندوز ویستا	6.0 (Build 6000)	January 2007
ویندوز سرور ۲۰۰۸	6.0 (Build 6001)	March 2008
ویندوز هفت	6.1 (Build 7600)	October 2009
ویندوز سرور 2008 R2	6.1 (Build 7600)	October 2009
ویندوز ۸	6.2	October 2012
ویندوز سرور 2012	6.2	October 2012
ویندوز ۸.۱	6.3	October 2013
ویندوز سرور 2012 R2	6.3	October 2013
ویندوز ۱۰	10.0 (Build 10240)	July 2015
ویندوز ۱۰ نسخه ۱۵۱۱	10.0 (Build 10586)	November 2015
ویندوز ۱۰ نسخه ۱۶۰۷	10.0 (Build 14393)	July 2016
ویندوز سرور ۲۰۱۶	10.0 (Build 14393)	October 2016

توجه کنید، شماره ۷ در نام محصول Windows 7 به شماره نسخه داخلی این سامانه‌عامل اشاره نمی‌کند، این عدد فقط نمایانگر شماره ایندکس تولیدی این محصول است. در واقع، به

[1] Windows API

منظور کاهش مسائل سازگاری[1] برنامه‌های‌کاربردی، شماره نسخه داخلی سامانه‌عامل ویندوز ۷ در اصل ۶.۱ است که در جدول ۱ نمایش داده شده است.

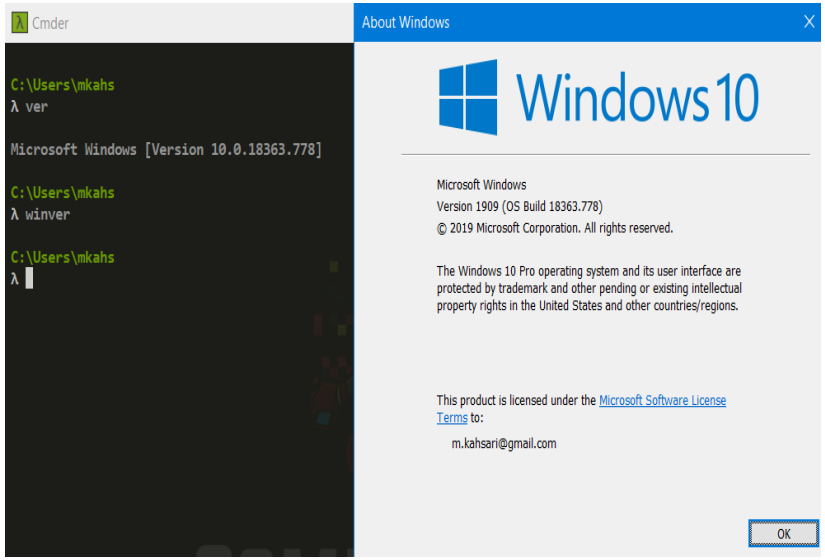
این ویژگی به برنامه‌های کاربردی ویندوز اجازه می‌دهد برای بررسی شماره نسخه اصلی همانطور در ویندوز ۷ رفتار کنند که در ویندوز ویستا رفتار می‌کنند، به عبارت دیگر این موضوع باعث می‌شود از پیچیدگی توسعه و عملکرد برنامه‌های‌کاربردی در پلتفرم سامانه‌عامل‌های مایکروسافت کاسته شود.

شایان ذکر است، ویندوز ۷ و ویندوز سرور 2008 R2 شماره ساخت و شماره نسخه مشابه‌ای دارند، زیرا آن‌ها با کد پایه[2] مشابه‌ای از پلتفرم سامانه‌عامل ویندوز ایجاد شده‌اند. با این حال، با شروع ویندوز ۱۰، نسخه داخلی هم به ۱۰ به روزرسانی شد.

نکته: با شروع ویندوز ۸، رابط برنامه‌نویسی GetVersionEx، به صورت پیش‌فرض شماره نسخه سامانه‌عامل را ٦.٢ بازگشت می‌داد (این تابع اکنون منسوخ اعلام شده است). همانطور که پیش از این ذکر شد، این مسئله به منظور حفظ سازگاری و مسائل مرتبط با آن رخ داده است. با این حال، در بسیاری از شرایط این رویکرد برای شناسایی نسخه و مدل سامانه‌عامل ویندوز یک رویکرد خوب به شمار نمی‌رود. با این وجود، اگر به نسخه اصلی سامانه‌عامل نیاز دارید، می‌توانید از توابع جدید مانند VerifyVersionInfo یا رابط‌های جدید دیگر کمکی ویندوز یا Helper API از قبیل IsWindows8OrGreater، IsWindows10OrGreater، IsWindowsServer و ... استفاده کنید.

شایان ذکر است، برای مشاهده نسخه سامانه عامل ویندوز، می توانید از ابزار تحت کنسول ver استفاده کنید. این ابزار همچنین دارای نسخه گرافیکی است که می توانید آن را با دستور winver اجرا کنید. تصویر ۱، خروجی این ابزار را در سامانه عامل ویندوز ۱۰ نمایش می دهند.

[1] Comptebility



تصویر ۱: خروجی فرمان ver و winver در ویندوز ۱۰

ویندوز ۱۰ و نسخه‌های آینده ویندوز

با ارائه ویندوز ۱۰، مایکروسافت اعلام کرد با سرعت بیشتری از این پس سامانه‌عامل ویندوز را به‌روزرسانی خواهد کرد. همچنین ویندوز ۱۱ رسمی وجود نخواهد داشت به جای آن، نسخه جاری سامانه‌عامل ویندوز ۱۰ را به یک نسخه جدید به‌روزرسانی خواهد کرد. در زمان نگاشت این سلسله مقالات، دو به‌روزرسانی از این جنس توسط مایکروسافت صورت گرفته است. اولین به‌روزرسانی با شماره ۱۵۱۱ صورت گرفت که اشاره‌گر به سال ۲۰۱۵ و ماه November است. دومین به‌روزرسانی با شماره ۱۶۰۷ صورت گرفت که اشاره‌گر به سال ۲۰۱۶ و ماه July است. این به‌روزرسانی همچنین با عنوان Anniversary Update شناخته می‌شود.

[2] Code Base

نکته: از نظر معماری داخلی، مایکروسافت هنوز سامانه‌عامل ویندوز را مبتنی بر یک قاعده مواج ایجاد می‌کند. به عنوان مثال، نسخه اول ویندوز ۱۰ دارای نام رمز **Threshold 1**، در حالیکه به‌روزرسانی **November 2015** دارای نام رمز **Threshold 2** بود. همچنین سه فاز بعدی به‌روزرسانی مایکروسافت با عنوان **Redstone 1** (نسخه ۱۶۰۷)، **Redstone 2** و همچنین **Redstone 3** شناخته می‌شود.

ویندوز ۱۰ و مفهوم یک هسته[1]

در طول سالیان، چندین نسخه متفاوت از سامانه‌عامل ویندوز توسط مایکروسافت برای سخت‌افزارها و شرایط گوناگون توسعه داده شد. به عنوان مثال، در کنار سامانه‌عامل اصلی ویندوز مایکروسافت که بر روی کامپیوترهای دسکتاپ اجرا می‌شود، یک انشعاب از سامانه‌عامل ویندوز ۲۰۰۰ توسط مایکروسافت ایجاد شد که بر روی کنسول بازی XBOX 360 اجرا شود.

همچنین یک نسخه از ویندوز وجود دارد که با عنوان ویندوز موبایلی[2] شناخته می‌شود که بر روی گوشی‌های موبایل قابل اجرا است. سامانه‌عامل ویندوز موبایلی در حقیقت یک انشعاب از Windows CE (سامانه‌عامل بلادرنگ مایکروسافت) است. به هر صورت، نگهداری و توسعه تمامی این سامانه‌های عامل توسط مایکروسافت واضح است که یک کار دشوار به حساب می‌آید. از همین روی، مایکروسافت بعدها تصمیم گرفت، پشتیبانی از کرنل‌ها و باینری‌های اصلی را یکی کند.

این تغییرات با انتشار ویندوز ۸ و ویندوز موبایلی ۸ شروع شد که دارای یک کرنل مشترک با یکدیگر بودند. سپس این حرکت با انتشار ویندوز ۸.۱ و ویندوز موبایلی ۸.۱ سرعت گرفت، چون دارای یک رابط برنامه‌نویسی زمان اجرای[3] همگرا بودند.

با انتشار ویندوز ۱۰، این همگرایی تکمیل شد. به هر صورت، این پلتفرم واحد در سامانه‌های عامل مایکروسافت اکنون با عنوان OneCore شناخته می‌شود که می‌تواند بر روی

[1] OneCore
[2] Windows Phone

کامپیوترهای خانگی، موبایل، تبلت، تلویزیون هوشمند، کنسول بازی XboxOne، کنسول HoloLens و تجهیزات اشیاء اینترنتی از قبیل Raspberry Pi 2 اجرا شود.

تصویر ۲: پوشش ویندوز OneCore از تجهیزات گوناگون

این مسئله البته کاملا واضح است، بسیاری از این تجهیزات (در تصویر ۲ برخی از آن را مشاهده می‌کنید) از نظر فاکتورهای سخت‌افزاری و نرم‌افزاری بسیار متفاوت از یکدیگر هستند. از همین روی، عدم وجود برخی از ویژگی‌های ویندوز بر روی برخی از این تجهیزات یک مورد کاملا طبیعی است.

به عنوان مثال، در تجهیزات HoloLens پشتیبانی از موس و کیبورد در ویندوز ۱۰ وجود ندارد، چون به آن‌ها نیاز نیست و نمی‌توان انتظار داشت بر روی ویندوز ۱۰ این ویژگی‌ها برای تجهیزات HoloLens وجود داشته باشد. با این حال، کرنل، درایورها و باینری‌های اصلی پلتفرم[4] در تمامی این تجهیزات مشابه یکدیگر هستند. در این سلسله مقالات، هدف ما ورود و تحلیل ساختار کرنل OneCore مایکروسافت است.

[3] Windows Runtime API
[4] Base platform binaries

اصطلاحات و مفاهیم بنیادی[1]

در این سلسله مقالات، ما به برخی از استراکچرها[2] و مفاهیم رجوع خواهیم کرد که ممکن است برای بعضی از خوانندگان ناشناخته باشند. به همین دلیل در این قسمت از سلسله مقالات معماری ویندوز، این اصطلاحات را تعریف می‌کنیم که در طول سلسله مقالات از آن‌ها استفاده خواهیم کرد. همچنین بهتر است قبل از پیشروی در مطالعه این سلسله مقالات، با این اصطلاحات و مفاهیم کاملا آشنا شوید.

رابط‌های برنامه‌نویسی کاربردی ویندوز

رابط برنامه‌نویسی کاربردی ویندوز (Windows API)[3]، در واقع یک رابط برنامه‌نویسی سیستمی در مُد کاربر برای سامانه‌عامل مایکروسافت است. شایان ذکر است، قبل از معرفی نسخه ۶۴ بیتی سامانه‌عامل ویندوز، در نسخه‌های ۳۲ بیتی سامانه‌عامل ویندوز، رابط‌های برنامه‌نویسی ۳۲ بیتی، Win32 API خوانده می‌شدند، زیرا برنامه‌نویس‌ها بتوانند میان رابط‌های ۳۲ بیتی و ۱۶ بیتی تفاوت قائل شوند. با این حال، در این سلسله مقالات ما فقط از واژه رابط‌های برنامه‌نویسی ویندوز (Windows API) به منظور اشاره به رابط‌های برنامه‌نویسی ۳۲ و ۶۴ بیتی ویندوز استفاده خواهیم کرد.

نکته: رابط‌های برنامه‌نویسی ویندوز در مستندات کیت توسعه نرم‌افزار ویندوز (**Windows SDK**) تشریح شده‌اند. این مستندات به منظور بازدید آنلاین در آدرس **www.msdn.microsoft.com** موجود و قابل دسترس هستند. همچنین شایان ذکر است، این مستندات شامل تمامی جزئیات **Microsoft Developer Network** می‌شوند که یک برنامه به منظور پشتیبانی از توسعه‌دهندگان ویندوز توسط شرکت مایکروسافت است. همچنین در سلسله مقالات **Windows via C/CPP** نوشته **Jeffrey Richter** و **Christophe Nasarre** از انتشارات **Microsoft** اطلاعات دقیقی از چگونگی برنامه‌نویسی با رابط‌های برنامه‌نویسی پایه ویندوز آورده شده است که آن را می‌توانید به منظور اطلاعات بیشتر مورد مطالعه قرار بدهید. همچنین برای مشاهده ویدیوهای آموزشی در ارتباط با **C++/WinAPI** به آدرس (لینک) رجوع کنید.

انواع رابط‌های برنامه‌نویسی ویندوز

رابط‌های برنامه‌نویسی ویندوز در حقیقت توابعی هستند که با زبان و قواعد زبان C نوشته شده‌اند. در حقیقت تفاوتی ما بین توابعی که در C توسط شما نوشته می‌شوند، از نظر ماهیت پیاده‌سازی با رابط‌های برنامه‌نویسی ویندوز وجود ندارد. شایان ذکر است، اکنون صدها هزار از چنین توابعی در ویندوز وجود دارند که توسعه‌دهندگان نرم‌افزار از آن‌ها می‌توانند برای توسعه نرم‌افزار، درایور و سرویس و کتابخانه و ... استفاده کنند.

قابل ذکر است، یکی از دلایلی که مهندسان مایکروسافت، زبان C را برای توسعه ویندوز و رابط‌های برنامه‌نویسی آن انتخاب کردند، سطح پایین بودن و سرعت فوق‌العاده و قابل حمل بودن این زبان است (در قسمت پنجم سری مقالات تجزیه و تحلیل بدافزار کی‌پاد در این باره صحبت کردیم. برای اطلاعات بیشتر به آن مقالات رجوع کنید). همچنین علاوه بر سطح پایین بودن، امکان دسترسی از دیگر زبان‌ها به کدهای نوشته شده توسط C وجود دارد.

با این حال، علاوه بر مزیت‌هایی که انتخاب زبان C داشت، این زبان دارای معایبی از قبیل عدم پشتیبانی از فضای نام (namespaces) و عدم قواعد نامگذاری یکپارچه است. برخی از نتایج این معایب، در رابط‌های برنامه‌نویسی جدیدی که در حال استفاده از مکانیزم متفاوتی مانند COM[4] هستند، ظاهر می‌شوند.

در حقیقت مکانیزم COM برای این شکل گرفت که برنامه‌های کاربردی مجموعه Office مایکروسافت بتوانند با یکدیگر تعامل داشته باشند و مابین یکدیگر داده معاوضه یا تبادل کنند

[1] Foundation concepts and terms
[2] Structures
[3] Windows application programming interface (API)
[4] Component Object Model (COM)

(از قبیل قرار دادن یک چارت Excel درون یک مستند Word). این توانایی در حقیقت نهفته‌سازی و پیوند آبجکت (OLE)[1] در ویندوز شناخته می‌شود.

مکانیزم نهفته‌سازی و پیوند آبجکت (OLE) در ویندوز مبتنی بر استفاده از یک مکانیزم پیام‌رسانی قدیمی در ویندوز که تبادل دینامیک داده (DDE)[2] خوانده می‌شود، پیاده‌سازی شد که ماهیتاً دارای محدودیت بسیار بود. از همین روی مایکروسافت تصمیم گرفت یک مکانیزم جدید برای ارتباط داخلی مولفه‌های ویندوز توسعه دهد که با عنوان COM اکنون شناخته می‌شود. مکانیزم COM در سال ۱۹۹۳ ابداع شد، و در ابتدا OLE 2 خوانده می‌شد.

مکانیزم COM بر اساس دو مفهوم است. اول، کلاینت‌ها می‌توانند با آبجکت‌ها از طریق رابط‌هایی (گاهی اوقات، آبجکت‌های COM Server شناخته می‌شوند) با یکدیگر ارتباط برقرار کنند. شایان ذکر است، این رابط‌ها، یک مجموعه از قراردادها به همراه متدهای منطقی هستند که تحت مکانیزم ارسال[3] جدول مجازی[4] که یک رویکرد برای پیاده‌سازی توابع مجازی توسط کامپایلرهای CPP است، با یکدیگر ارتباط برقرار کنند. این موجب سازگاری باینری‌ها و حذف مسائل مرتبط با Name Mangling کامپایلرها می‌شود. در نتیجه، فراخوانی چنین توابعی از دیگر زبان‌ها از قبیل C، C++، Visual Basic، C#، Delphi و ... ممکن خواهد شد.

اصطلاح COM Server، معمولا به یک کتابخانه از نوع DLL یا یک فایل اجرایی (EXE) اشاره دارد که کلاس های COM در آن پیاده‌سازی شده‌اند. شایان ذکر است، مکانیزم COM در ویندوز دارای ویژگی‌های مهم دیگر مرتبط با امنیت، هدایت مستقل پروسه[5]، مدل ترد، و ...است. به عنوان مثال، برخی از رابط‌های برنامه‌نویسی ویندوز که از طریق COM در دسترس قرار می‌گیرند، شامل DirectShow، Windows Media Foundation، DirectX، DirectComposition، Windows Imaging Comonent و Background Intelligent Transfer Service می‌شوند.

به هر صورت، مبحث COM بسیار پیچیده و گسترده است، از همین روی پرداختن به آن در این سلسله مقالات ممکن نیست. با این حال، وقتی ما وارد بحث برنامه‌نویسی سیستمی ویندوز می‌شویم، با هزاران رابط‌های برنامه‌نویسی قابل فراخوانی رو به رو خواهیم شد که در گروه‌های اصلی زیر تقسیم می‌شوند.

- سرویس‌های پایه
- سرویس‌های کامپوننتی
- سرویس‌های رابط کاربری
- سرویس‌های چندرسانه‌ای و گرافیکی
- سرویس‌های همکاری و پیام‌رسانی
- سرویس‌های شبکه‌ای
- سرویس‌های وب

شایان ذکر است، این سلسله مقالات ویندوز اینترنالز بر روی سرویس‌های پایه و کلیدی از قبیل پروسه‌ها[6]، تِرِدها[7]، مدیریت حافظه[8]، ورودی و خروجی[9]، و امنیتی متمرکز است.

ویژگی Windows Runtime

سامانه‌عامل ویندوز ۸ یک رابط برنامه‌نویسی و پشتیبانی از زمان اجرا[10] معرفی کرد که با عنوان Windows Runtime شناخته می‌شود. شایان ذکر است، گاهی اوقات Windows Runtime به صورت WinRT نوشته می‌شود که از مبحث Windows RT مجزا است و ارتباطی بین این دو وجود ندارد.

[1] Object Linking and Embedding
[2] Dynamic Data Exchange (DDE)
[3] Dispatch Mechanism
[4] Virtual Table
[5] Cross-Process Marshalling

[6] Process
[7] Threads
[8] Memory Mangement
[9] I/O
[10] Supporting Runtime

ویژگی Windows Runtime که در ویندوز ۸ معرفی شد، در حقیقت شامل سرویس‌های پلتفرمی می‌شود که به توسعه‌دهندگان نرم افزار ویندوزی کمک شایانی می‌کند. توسعه‌دهندگان برنامه‌های ویندوزی با عنوان Windows Apps که در گذشته با عنوان Metro Apps، Modern Apps، Immersive Apps و Windows Store Apps مورد ارجاع قرار می گرفتند، از WinRT نهایت بهره را می‌برند.

برنامه‌های کاربردی ویندوز ممکن است تجهیزات سخت‌افزاری گوناگونی از قبیل تجهیزات دسکتاپ، موبایل، و حتی دستگاه‌هایی از قبیل کنسول بازی XBOX یا Microsoft HoloLens را هدف قرار بدهند.

از منظر رابطهای برنامه‌نویسی، ویژگی WinRT بر روی مکانیزم COM ایجاد شده است و چندین افزونه به زیرساخت COM در ویندوز اضافه کرده است. به عنوان مثال، متادیتا کاملی در WinRT وجود دارد (در فایل‌های WINMD ذخیره می‌شود و مبتنی بر فرمت متادیتا Net. است) که یک مفهوم مشابه در COM با عنوان کتابخانه‌های نوع داده[1] را توسعه می‌دهد. از منظر طراحان رابطهای برنامه‌نویسی، WinRT بسیار منسجم‌تر از رابطهای برنامه‌نویسی کلاسیک ویندوز، به دلیل استفاده از سلسله مراتب فضای‌نام[2]، نامگذاری ثابت[3]، پترن‌های برنامه‌نویسی[4] است.

قابل ذکر است، اپ‌های ویندوزی (Windows Apps) سوژه قوانین جدید هستند، برخلاف برنامه‌های کاربردی ویندوز که اکنون Windows Desktop Application یا Classic Windows Application شناخته می‌شوند. به این نکته هم دقت داشت باشید، در ویندوز بین اپ‌ها (App) با برنامه‌هایی کاربردی (Application) تفاوت وجود دارد. به هر صورت، ارتباط بین رابطهای برنامه‌نویسی مختلف و برنامه‌های کاربردی ویندوز بدون چالش و سرراست نیست. برنامه‌های دسکتاپ می‌توانند قسمتی از رابطهای WinRT را مورد استفاده قرار بدهند. متعاقباً، اپ‌های ویندوزی می‌توانند بخشی از رابطهای Win32 و COM را استفاده کنند. برای مشاهده رابطهایی که برای هر پلتفرم در دسترس هستند، باید به MSDN رجوع کنید.

توجه کنید، در سطح کتابخانه‌ها، رابطهای WinRT هنوز بر بالای کتابخانه‌ها و رابطهای برنامه‌نویسی سنتی ویندوز تعریف شده‌اند، اگرچه موجودیت برخی رابطهای برنامه‌نویسی ممکن است، مستندسازی یا پشتیبانی نشده باشد. در هر حال، WinRT همانند Net. هنوز از بخش قابل توجه‌ای از رابطهای برنامه نویسی سنتی ویندوز استفاده می‌کند.

نکته مهم دیگر در ارتباط با WinRT وجود ویژگی منحصربفردی با عنوان پروژکتور زبان (Language Projector) است که این مکانیزم در WinRT اجازه می‌دهد، از آن در زبان‌هایی مانند C++، زبان‌های مبتنی بر Net. و حتی JavaScript استفاده کرد. به عنوان مثال، در C++، مایکروسافت یک افزونه غیراستانداردی ایجاد کرده است که با عنوان C++/CX شناخته می‌شود که استفاده از انواع داده WinRT را ساده‌سازی می‌کند. سطح قابلیت همکاری COM یا به عبارت دیگر COM interop برای محیط Net. (به همراه پشتیبانی از برخی افزونه‌های زمان اجرا) اجازه خواهد داد، هر زبان برنامه‌نویسی تحت چارچوب Net. بتواند از رابطهای برنامه‌نویسی WinRT به صورت طبیعی استفاده کند. برای توسعه‌دهندگان JavaScript یک افزونه توسعه داده شده بود که WinJS عنوان داشت. این افزونه می‌توانست به WinRT دسترسی بگیرد اگرچه توسعه‌دهندگان JavaScript باید هنوز از HTML برای ایجاد رابط کاربری برنامه‌های خود استفاده کنند.

نکته: اگرچه HTML می‌تواند در توسعه و پیاده‌سازی اپ‌های ویندوزی استفاده شود، اما این اپ‌ها هنوز یک برنامه کلاینتی ساده در سیستم لوکال هستند و امکان این وجود ندارد که مانند یک برنامه تحت وب از طریق یک سرور مورد دسترسی و فراخوانی قرار بگیرند.

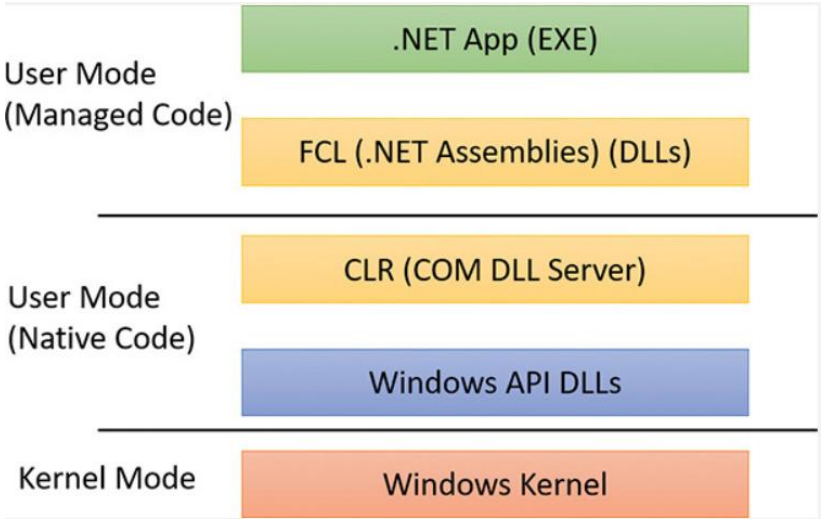
[1] Type Libraries
[2] Namespace Hierarchies
[3] Consistent Naming
[4] Programmatic Patterns

پلتفرم دات‌نت (.Net) چیست؟

پلتفرم دات‌نت (.Net) مایکروسافت شامل یک سلسله کتابخانه از کلاس‌های قابل فراخوانی است که Framework Class Library خوانده می‌شوند. این پلتفرم همچنین شامل یک زبان زمان اجرای مشترک (CLR)[1] است که یک محیط مدیریت شده برای اجرای کد با ویژگی‌هایی از قبیل کامپایل درجا (JIT)[2]، راستی آزمایی نوع[3]، زباله‌روب[4] و امنیت دسترسی به کد[5] را ارائه می‌کند.

با ارائه این ویژگی‌ها، زبان زمان اجرای مشترک یا به عبارت دیگر مولفه CLR یک محیط توسعه ارائه می‌کند که محصولات برنامه‌نویسان را بهبود می‌بخشد و خطاهای برنامه‌نویسی رایج را کاهش می‌دهد. با این حال، شما می‌توانید به منظور بدست آوردن اطلاعات دقیق در مورد پلتفرم دات‌نت (.Net) و ساختار درونی آن سلسله مقالات CLR via C# نوشته Jeffrey Richter از انتشارات مایکروسافت را مورد مطالعه قرار بدهید.

زبان زمان اجرای مشترک یا مولفه CLR به عنوان یک سرور COM کلاسیک پیاده‌سازی شده است که کدهایش در یک کتابخانه پیوندی پویا[6] قرار دارد. در واقع، تمامی کامپوننت‌های پلتفرم دات‌نت (.Net) مایکروسافت به عنوان کتابخانه‌های پیوندی پویا استاندارد در مُد کاربر بر روی رابطهای برنامه‌نویسی مدیریت نشده ویندوز پیاده‌سازی شده‌اند (هیچ چیزی از پلتفرم دات‌نت به صورت مستقیم در مُد کرنل سامانه‌عامل ویندوز اجرا نمی‌شود). تصویر ۳ ارتباط میان این مولفه‌ها را نشان می‌دهد.



تصویر ۳: ارتباط میان کامپوننت‌های پلتفرم دات‌نت (.Net)

روتین‌ها، توابع و سرویس‌ها[7]

چندین اصطلاح در مستندات برنامه‌نویسی سامانه‌عامل ویندوز وجود دارند که در کانتَکست‌های مختلف[8] دارای معانی متفاوتی هستند. به عنوان مثال، کلمه سرویس می‌تواند به یک روتین قابل فراخوانی[9] در سامانه‌عامل ویندوز، یک دیوایس درایور[10] یا یک پروسه سرور[11] اشاره داشته باشد. در لیست زیر معانی برخی از اصطلاحات استفاده شده در این سلسله مقالات تشریح شده است.

[1] Common Language Runtime
[2] Just-in-time compilation
[3] Type verification
[4] Garbage Collection
[5] Code Access Security
[6] Dynamic-link Library

[7] Services, Functions and Routines
[8] Different Context
[9] Callable Routine
[10] Device Driver
[11] Server Process

نکته: از این قسمت به بعد، به منظور تسهیل خواندن متن این سلسله مقالات ویندوز اینترنالز، به جای واژه رابط‌های برنامه‌نویسی کاربردی سامانه‌عامل ویندوز از رابط‌های برنامه‌نویسی ویندوز استفاده خواهیم کرد.

توابع رابط‌های برنامه‌نویسی ویندوز[1]

سابروتین‌های قابل فراخوانی در رابط‌های برنامه‌نویسی ویندوز را توابع رابط‌های برنامه‌نویسی ویندوز می‌گویند. به عنوان مثال CreateProcess، CreateFile و GetMessage از این جمله توابع هستند. این توابع توسط شرکت مایکروسافت کاملا مستندسازی شده‌اند و از طریق آدرس http://msdn.microsoft.com/library در دسترس برنامه‌نویسان و توسعه‌دهندگان نرم‌افزارهای ویندوزی هستند.

سرویس‌های نیتیو سیستم (فراخوانی‌های سیستمی)[2]

سرویس‌های اصلی سامانه‌عامل ویندوز که از مُد کاربر قابلیت فراخوانی دارند، سرویس‌های نِیتیو سیستم گویند. به عنوان مثال، NtCreateUserProcess یک سرویس سیستمی است که تابع CreateProcess به منظور ایجاد یک پروسه جدید آن را فراخوانی می‌کند. شایان ذکر است، سرویس‌های نِیتیو سامانه‌عامل ویندوز توسط شرکت مایکروسافت مستندسازی نشده‌اند.

توابع پشتیبانی کرنل (روتین‌ها)[3]

سابروتین‌های درون سامانه‌عامل ویندوز که می‌توانند فقط از مُد کرنل فراخوانی شوند را توابع پشتیبانی کرنل یا روتین‌های کرنل سامانه‌عامل ویندوز می‌نامند (در ادامه این سلسله مقالات تشریح خواهند شد). به عنوان مثال، ExAllocatePoolWithTag روتینی است که دیوایس درایورها به منظور تخصیص حافظه از هیپ سامانه‌عامل ویندوز آن را فراخوانی می‌کنند.

سرویس‌های ویندوز[4]

پروسه‌های شروع شده توسط مدیر کنترل سرویس‌های ویندوز[5] را سرویس‌های ویندوز می‌خوانند. به عنوان مثال، سرویس Task Scheduler در یک پروسه مُد کاربر اجرا می‌شود که از فرمان at پشتیبان می‌کند (این فرمان شبیه به فرمان cron سامانه‌عامل لینوکس است).

نکته: اگرچه رجیستری سامانه‌عامل ویندوز دیوایس درایورها را همچنین به عنوان سرویس تعریف می‌کند، اما در این سلسله مقالات به دیوایس درایورهای ویندوز با این مفهوم ارجاع داده نمی‌شود.

کتاب‌خانه‌های پیوندی پویا

به یک مجموعه از سابروتین‌های قابل فراخوانی که به عنوان یک فایل باینری به یکدیگر پیوند داده شده‌اند و می‌توانند به صورت پویا توسط برنامه‌های کاربردی سامانه‌عامل ویندوز بارگزاری شوند و مورد استفاده قرار گیرند، سلسله مقالات خانه‌های پیوندی پویا یا به اختصار DLL می‌گویند.

به عنوان مثال، Msvcrt.dll (کتاب‌خانه زمان اجرای C) و Kernel32.dll (یکی از کتاب‌خانه‌های سیستمی رابط‌های برنامه‌نویسی ویندوز) از این نوع کتاب‌خانه‌ها هستند.

مولفه‌های مُد کاربر ویندوز و برنامه‌های کاربردی از کتاب‌خانه‌های پیوندی پویا (DLL) به صورت گسترَده استفاده می‌کنند. مزئیت استفاده از کتاب‌خانه‌های پیوندی پویا به جای کتاب‌خانه‌های ایستا[6] این است که برنامه‌های کاربردی می‌توانند این کتاب‌خانه‌ها را بین خودشان به اشتراک بگذارند و سامانه‌عامل ویندوز می‌تواند همچنین اطمینان حاصل کند که فقط یک نمونه از کد کتاب‌خانه پیوندی پویا در حافظه میان برنامه‌های کاربردی وجود دارد.

[1] Windows API Functions
[2] Native System Services (or System Calls)
[3] Kernel support functions (or Routines)

[4] Windows Services
[5] Windows service control manager
[6] Static

توجه کنید: اسمبلی‌های غیرقابل اجرای پلتفرم دات‌نت[1] به عنوان کتاب‌خانه‌های پیوندی پویا، اما بدون هیچ سابروتین اکسپورت شده[2] کامپایل می‌شوند. به جای آن، CLR متاداده[3] کامپایل شده را به منظور دسترسی به اطلاعات مربوط پارس[4] می‌کند.

پایان

در قسمت اول سلسله مقالات ویندوز اینترنالز برخی از مفاهیم مهم برنامه‌نویسی و معماری ویندوز را مورد بررسی قرار دادیم. به عنوان مثال، متوجه شدیم معماری OneCore چیست و مبتنی بر چه ایده‌ای توسعه آن شروع شد. با فناوری COM و WinRT و فریمورک .NET آشنا شدیم و همچنین برخی از مفاهیم مهم دیگر مانند رابط‌های برنامه‌نویسی و ... را مورد بررسی قرار دادیم. در قسمت بعدی به مفاهیم دیگر مانند پروسه‌ها و تردها و جاب‌ها و ... خواهیم پرداخت.

[1] Nonexecutable .Net Assemblies
[2] Exported Subroutines
[3] Metadata
[4] در زبان پارسی، واژه لاتین Parse، دارای معنی تجزیه است. از همین روی می‌توان از واژه تجزیه هم به جای این کلمه لاتین استفاده کرد. اما در این مقالات، به دلیل تخصصی بودن بسیاری از مفاهیم، از ترجمه عموم کلمات لاتین ممانعت شده است. با این حال، افرادی که می‌خواهند معنی و مفهوم دقیق این واژگان را متوجه شوند، بهتر است به یک فرهنگ لغت معتبر مانند آکسفورد رجوع کنند و مفهوم این نوع واژگان را مورد مطالعه قرار بدهند. در این مقالات، به منظور یکپارچگی متن مقالات با مفاهیم اصلی و تخصصی، این واژگان تخصصی لاتین عیناً با حروف پارسی مورد استفاده قرار گرفته است.