.NET 8

.NET 8 is Microsoft’s flagship open-source runtime. You can write web and console applications that run on Windows, Linux, and macOS; rich-client applications that run on Windows 10+ and macOS; and mobile apps that run on iOS and Android. This book focuses on the .NET 8 CLR and BCL. Unlike .NET Framework, .NET 8 is not preinstalled on Windows machines. If you try to run a .NET 8 application without the correct runtime being present, a mes sage will appear directing you to a web page where you can download the runtime. You can avoid this by creating a self-contained deployment, which includes the parts of the runtime required by the application

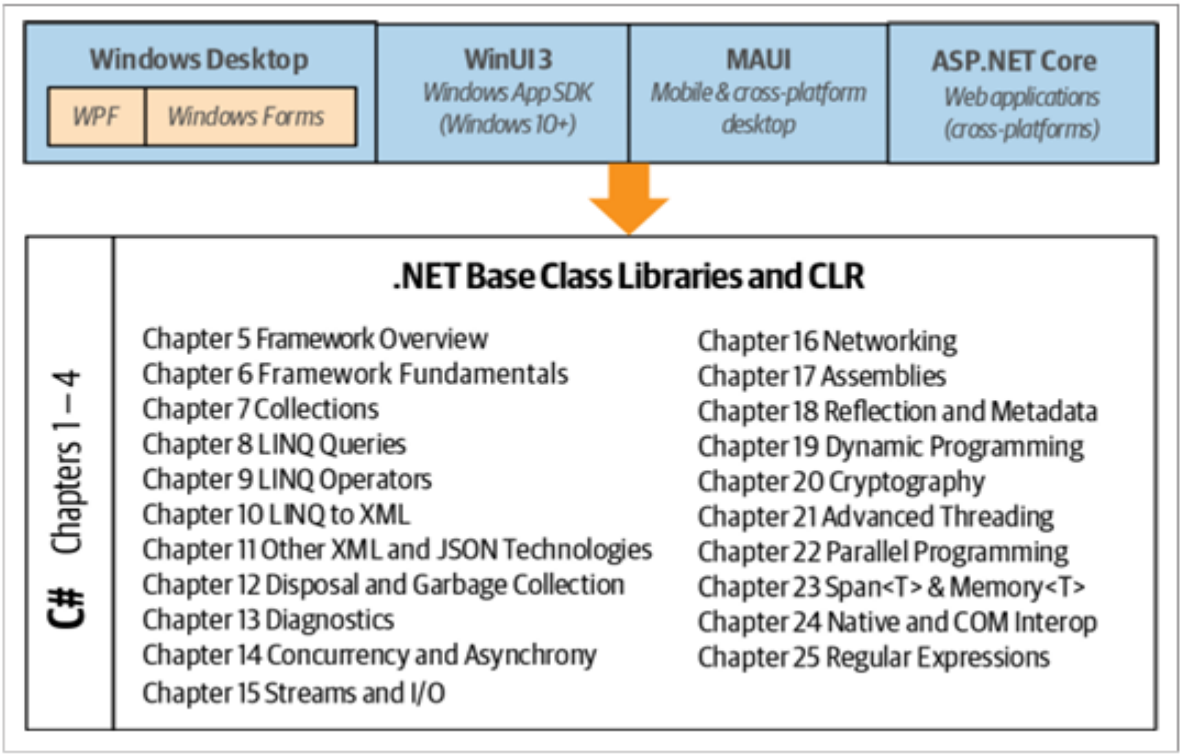

Figure 1-2. Runtimes for C#

ترجمه پاراگراف

.NET 8
.NET 8 جدیدترین و اصلی‌ترین Runtime متن‌باز مایکروسافت است.
شما می‌توانید با آن:

- اپلیکیشن‌های وب و کنسول بنویسید که روی ویندوز، لینوکس و macOS اجرا شوند.
- اپلیکیشن‌های rich-client بسازید که روی Windows 10+ و macOS اجرا شوند.

- اپلیکیشن‌های موبایل ایجاد کنید که روی iOS و Android اجرا شوند.

این کتاب بر روی CLR و BCL در NET 8. تمرکز دارد.

بر خلاف NET Framework.، NET 8.به‌طور پیش‌فرض روی ماشین‌های ویندوز نصب نیست. اگر تلاش کنید یک اپلیکیشن NET 8. را بدون وجود Runtime مناسب اجرا کنید، پیغامی ظاهر می‌شود که شما را به صفحه‌ای وب هدایت می‌کند تا Runtime را دانلود کنید.

شما می‌توانید این مشکل را با ایجاد یک self-contained deploymentحل کنید، که شامل بخش‌های لازم Runtime برای اجرای اپلیکیشن است.

🔷توضیحات تکمیلی و نکات ارائه

1. ویژگی‌های کلیدی:NET 8.
   - متن‌باز.(Open-Source)
   - چند سکویی :(Cross-platform) ویندوز، لینوکس، macOS، iOS، .Android
   - پشتیبانی از انواع اپلیکیشن‌ها: وب، کنسول، دسکتاپ(Rich-client) ، موبایل.
2. تفاوت با:NET Framework.
   - :NET Framework.فقط روی ویندوز → قدیمی، preinstalledروی ویندوز.
   - :NET 8.مدرن، cross-platform، پیش‌فرض نصب نیست → باید دستی نصب بشه یا همراه اپ بیاد.
3. :Self-contained Deployment
   - وقتی اپ رو منتشر می‌کنی، می‌تونی Runtime رو همراه اپلیکیشن قرار بدی.
   - اینطوری کاربر نیاز نداره جداگانه Runtime نصب کنه.
   - مزیت: اپلیکیشن بدون وابستگی خارجی اجرا میشه.
   - عیب: حجم نهایی اپ بزرگ‌تر میشه.
4. کاربرد عملی:
   - برای اپ‌های سازمانی یا جایی که کنترل روی محیط کاربر نداری، self-containedخیلی مفیده.
   - برای اپ‌هایی که حجم مهم نیست (مثلاً سرویس‌های داخلی)، این انتخاب امن‌تره.

جمع‌بندی ارائه برای این بخش:
NET 8.“پرچمدار جدید مایکروسافته: متن‌باز، چند سکویی، و مناسب برای همه نوع اپ — وب، دسکتاپ، موبایل و کنسول. تفاوتش با NET Framework. اینه که پیش‌فرض نصب نیست. برای حل این مشکل میشه اپ رو به شکل self-contained منتشر کرد تا runtime همراه خودش بیاد”.

.NET’s update history runs as follows: .NET Core 1.x → NET Core 2.x → .NET Core 3.x → .NET 5 → .NET 6 → .NET 7 → .NET 8. After .NET Core 3, Microsoft removed “Core” from the name and skipped version 4 to avoid confusion with .NET Framework 4.x, which precedes all of the preceding runtimes but is still supported and in popular use. This means that assemblies compiled under .NET Core 1.x → .NET 7 will, in most cases, run without modifica tion under .NET 8. In contrast, assemblies compiled under (any version of) .NET Framework are usually incompatible with .NET 8.

ترجمه پاراگراف

تاریخچه‌ی به‌روزرسانی‌هایNET.
تاریخچه‌ی به‌روزرسانی .NET به این صورت است:

.NET Core 1.x → .NET Core 2.x → .NET Core 3.x → .NET 5 → .NET 6 → .NET 7 → .NET 8.

بعد از3 .NET Core ، مایکروسافت واژه‌ی "Core" را از نام حذف کرد و نسخه‌ی ٤ را هم رد کرد تا با .NET Framework 4.x) که قدیمی‌تر از همه‌ی اینRuntime هاست اما همچنان پشتیبانی می‌شود و پرکاربرد است (اشتباه نشود.

این یعنی اسمبلی‌هایی که تحت .NET Core 1.x تا .NET 7 کامپایل شده‌اند، در بیشتر موارد بدون نیاز به تغییر تحت .NET 8 اجرا خواهند شد.
اما اسمبلی‌هایی که تحت (هر نسخه‌ای از) .NET Framework.کامپایل شده‌اند، معمولاً با .NET 8 ناسازگار هستند.

🔻توضیحات تکمیلی و نکات ارائه

1. تاریخچه نسخه‌ها:
   - شروع با .NET Core 1 →.اولین نسخه cross-platform
   - .NET Core 2 و ۳ →تکامل‌یافته‌تر، پشتیبانی از Windows Desktop در Core 3 اضافه شد.
   - بعد از اون) .NET 5, 6, 7, 8 → دیگه Core توی اسم نبود.(
   - دلیل حذف نسخه ٤ → جلوگیری از قاطی شدن با.NET Framework 4.x.
2. سازگاری عقبگرد:(Backward Compatibility)
   - .NET 8: → .NET Core.در اکثر موارد بدون مشکل.
   - .NET 8: → .NET Framework.معمولاً ناسازگار → چون Framework قدیمی‌تره و APIهاش متفاوت هستن.
3. چرا .NET Framework هنوز مهمه؟
   - هنوز در خیلی از شرکت‌ها و پروژه‌های قدیمی استفاده میشه.
   - مایکروسافت همچنان ازش پشتیبانی می‌کنه (ولی توسعه‌ی جدید دیگه روش انجام نمیشه).
4. نکته برای ارائه:
   - می‌تونی روی تفاوت مهم تاکید کنی:

> "امروز .NET 8 ادامه‌ی مسیر Core هست، نه .Framework اگر پروژه‌ای روی Coreنوشته باشین، تقریباً مطمئنید روی .NET 8 هم کار می‌کنه. ولی پروژه‌های قدیمی Framework معمولاً مستقیم قابل انتقال نیستن".

👉جمع‌بندی برای ارائه:
"مایکروسافت مسیر .NET رو از Core شروع کرد و بعد از نسخه ۳ اسم Core رو برداشت تا با Framework 4.xقاطی نشه. الان مسیر رسمی از .NET 5 تا .NET 8 ادامه پیدا کرده. پروژه‌هایی که روی Core بودن با ۸ سازگارن، ولی پروژه‌های قدیمی Framework معمولاً ناسازگار هستن".