**أولاً: إصدارات .NET (DotNet Versions)**

.NET مر بمراحل تطوير كبيرة، وتطور من بيئة مغلقة إلى منصة مفتوحة ومتعددة الأنظمة. إليك ملخص سريع:

| **الإصدار** | **الاسم** | **الوصف** |
| --- | --- | --- |
| .NET Framework | 1.0 – 4.8 | الإصدار الأصلي. يعمل فقط على نظام Windows. يستخدم في تطبيقات سطح المكتب مثل WinForms وWPF. |
| .NET Core | 1.0 – 3.1 | منصة مفتوحة المصدر. تدعم Windows, Linux, MacOS. أكثر سرعة وخفة من الـ Framework. |
| .NET 5 | بداية الدمج | أول إصدار موحد بعد دمج .NET Framework و .NET Core. |
| .NET 6 (LTS) | إصدار طويل الدعم | أسرع وأكثر استقرارًا. دعم كامل لـ Web API, MVC, Blazor, Desktop Apps. |
| .NET 7 | غير LTS | تحديث سنوي بمزايا إضافية. لكن بدون دعم طويل الأمد. |
| .NET 8 (LTS) | أحدث إصدار طويل الدعم حتى الآن (2024–2027). |  |

✅ **LTS** تعني Long-Term Support: دعم طويل الأجل (3 سنوات عادةً).

**🧭 ثانيًا: ما هو الـ namespace؟**

الـ namespace في C# و .NET هو طريقة لتنظيم الكود داخل مجالات منطقية، ولتجنّب تضارب الأسماء بين الكلاسات.

**🧠 مثال:**

namespace MyProject.Services

{

class UserService

{

public void PrintUser() => Console.WriteLine("User printed");

}

}

**✅ الفوائد:**

* ترتيب الكود (modular structure).
* تجنب التعارض بين الأسماء المتكررة.
* تسهيل القراءة والبحث عن الكلاسات داخل المشاريع الكبيرة.

**🏗️ ثالثًا: ما الفرق بين .NET Framework و .NET Core؟**

| **المقارنة** | **.NET Framework** | **.NET Core / .NET 5+** |
| --- | --- | --- |
| نظام التشغيل | Windows فقط | Windows, Linux, macOS |
| الأداء | أبطأ نسبيًا | أسرع وأخف |
| المصدر | مغلق جزئيًا | مفتوح المصدر بالكامل |
| التوزيع | مثبت مع النظام غالبًا | يمكن تضمينه مع التطبيق |
| التطوير الجديد | لا ينصح به | هو المستقبل لكل تطبيقات .NET |

🔥 **حاليًا، مايكروسوفت توصي باستخدام .NET 6 أو 8 لأي تطبيق جديد.**

**🧩 رابعًا: ما هو الـ Solution في Visual Studio / .NET؟**

الـ Solution هو **حاوية منطقية** لمجموعة مشاريع (Projects) في Visual Studio أو .NET.

**🧱 مكوناته:**

* **.sln file**: ملف يحتوي على إعدادات المشروع والمراجع.
* يمكن أن يحتوي على:
  + مشروع ويب (Web App)
  + مشروع كونسول (Console App)
  + مكتبة كود (Class Library)
  + مشروع اختبارات (Unit Testing)

**🧠 مثال:**

MySolution.sln

│

├── MyApp.Web (Web Project)

├── MyApp.Core (Class Library)

└── MyApp.Tests (Test Project)

**✅ الفائدة:**

* تنظيم المشاريع ذات الصلة معًا.
* تسهيل الإدارة في الفرق الكبيرة.
* دعم بناء المشاريع بالتسلسل (dependencies).