What's the difference between Compiled and Interpreted Languages?

Compiled Language:

- The code is **translated all at once** into machine code **before running**.
- Needs a **compiler**.
- Result: a fast, standalone **executable file**.

Fast at runtime
Compilation must happen first

Examples: C, C++, Go, Rust

Interpreted Language:

- The code is **translated line-by-line at runtime**.
- Needs an **interpreter** to run the code every time.

Easy to test/edit quickly
Slower at runtime

Examples: Python, JavaScript, Ruby

So, What About C#?

C# is **neither fully compiled nor fully interpreted**. It uses a **hybrid model**:

C# is compiled to Intermediate Language (IL):

1. You write C# code.
2. The **compiler** (e.g., Roslyn) compiles it into **IL (Intermediate Language)**.
3. At runtime, the **.NET runtime (CLR)** uses a **JIT compiler** (Just-In-Time) to convert IL → **machine code**.

Fast like compiled languages
Flexible like interpreted ones
You get the best of both worlds

Summary Table:

Language Type	How It Works	C# Case
Compiled	Translates all at once before running	❌
Interpreted	Translates line-by-line at runtime	❌
Hybrid (C#)	Compiles to IL, then JIT at runtime	✅

نواع التحويل (Casting) في C#

النوع	وصف سريع	مثال	هل ممكن يفشل؟	يحتاج try/catch؟
✅ **Implicit**	تحويل تلقائي من نوع أصغر إلى نوع أكبر (بدون فقد بيانات)	int i = 100; long l = i;	❌	لا
⚠️ **Explicit**	تحويل يدوي من نوع أكبر لأصغر – قد يتم فقد بيانات	double d = 5.5; int i = (int)d;	✅ مثلاً فقد الكسور)	لا (لكن راقب النتيجة)
🔁 **Convert**	يستخدم دوال في System.Convert – يتعامل مع null ويفشل لو غير قابل للتحويل	int i = Convert.ToInt32("123");	✅ (لو string غير رقمي)	✅
🔍 **Parse**	يحوّل string إلى نوع معين – **لا يقبل null**	int i = int.Parse("123");	✅	✅

متى تستخدم كل نوع؟

الحالة	الأفضل استخدام
التحويل التلقائي الآمن	Implicit
التحويل من نوع كبير لأصغر	Explicit
التعامل مع null أو أنواع مختلفة	Convert
تحويل string مضمون رقميًا	Parse

What is meant by "C# is managed code"?

المعنى 🔷:

Managed code يعني إن الكود بتاعك بيشتغل تحت إشراف وإدارة الـ **CLR** (Common Language Runtime) في .NET.

معنى "Managed" ✅:

- يتم **إدارة الذاكرة تلقائيًا**
- يتم تشغيل **Garbage Collector** لحذف الكائنات غير المستخدمة
- يتم توفير **أمان** و**تحكم بالأخطاء**
- يمنع **الوصول العشوائي** للذاكرة

باختصار 🧠:

كود C# زي **unmanaged code** اللي تعتبر (C++ أو C بيشتغل في بيئة آمنة ومدارة، مش محتاج تقلق من إدارة الذاكرة بنفسك).

What is meant by "struct is considered like class before"?

🔷 **المعنى المقصود غالبًا:**

الـ struct و class في C# **يبدوا متشابهين** لأنهم:

- الاتنين بيسمحوا بتعريف خصائص (properties) ودوال(methods)
- الاتنين ممكن فيهمconstructors
- الاتنين أنواع مخصصة(custom types)

لكن في فرق كبير بينهم:

الخاصية	struct	class
نوع البيانات	**Value Type**	**Reference Type**
مكان التخزين	Stack	Heap
الأداء	أسرع في بعض الحالات	أبطأ نسبيًا
الوراثة	لا تدعم وراثة (inheritance)	تدعم الوراثة الكاملة
استخدام مثالي لـ	بيانات بسيطة وثابتة	كائنات معقدة ومتغيرة

🧠 **باختصار:**

struct بيشبه class في الشكل، لكنه أخف وأبسط، ومناسب للحالات اللي مش محتاجة وراثة أو إدارة ذاكرة معقدة.