<https://www.linkedin.com/posts/amr-elshnhab-3ab12624b_%D8%A7%D9%84%D9%8A%D9%88%D9%85-%D8%B3%D8%A7%D8%AA%D9%83%D9%84%D9%85-%D8%B9%D9%86-%D8%A7%D9%84%D9%80-string-%D9%81%D9%8A-%D9%84%D8%BA%D8%A9-c-activity-7268357366566301696-BXvU?utm_source=share&utm_medium=member_desktop>

5- what meant by user defined constructor and its role in initialization

**المُنشئ المُعرَّف من قِبَل المُستخدم** (User-Defined Constructor) هو طريقة خاصة تُستخدم في البرمجة لتحديد كيفية تهيئة (تهيئة القيم) لكائن من صنف معين عند إنشائه. في لغة C#، يمكن للمبرمجين تعريف منشئ خاص بهم لتخصيص عملية تهيئة الكائنات بالطريقة التي تناسبهم. يكون هذا المنشئ بمثابة وظيفة يتم استدعاؤها تلقائيًا عند إنشاء الكائن من الصنف (الفئة).

**دور المنشئ المُعرَّف من قِبَل المُستخدم في التهيئة:**

**تهيئة القيم الأولية**:1

* + عندما يتم إنشاء كائن جديد من الصنف، فإن المنشئ يقوم بإعطاء القيم الأولية للخصائص (المتغيرات) داخل الكائن. بدلاً من أن تترك الخصائص فارغة أو غير مُعرفة، يُمكنك تخصيص قيم لها من خلال المنشئ.

**تخصيص عملية التهيئة**:2

* باستخدام المنشئ المُعرَّف من قِبَل المُستخدم، يمكن تخصيص كيف ومتى يتم إعطاء قيم للخصائص. كما يمكن تضمين منطق إضافي مثل التحقق من القيم المدخلة أو إجراء حسابات قبل التهيئة.

**استخدام أكثر من منشئ**:3

* يمكن تعريف أكثر من منشئ واحد داخل نفس الصنف، مما يسمح بمرونة أكبر في كيفية إنشاء الكائنات. يمكن للمنشئات أن تحمل معايير (Parameters) مختلفة لتناسب حالات استخدام متنوعة.

**التحقق من البيانات**:4

* يمكن استخدام المنشئ للتحقق من صحة البيانات التي يتم تمريرها. إذا كانت البيانات غير صالحة، يمكن للمنشئ أن يعيد خطأ أو يحدد قيمًا افتراضية.

3- what are scenarios to use string Vs StringBuilder?

في C#، يُستخدم كل من string و StringBuilder لإدارة النصوص، ولكن كل منهما له خصائص وأداء مختلفين، لذلك يعتمد الاختيار بينهما على متطلبات الأداء وسيناريوهات الاستخدام. إليك متى يجب استخدام كل منهما:

**string:. استخدام**

string هو نوع غير قابل للتغيير (immutable) في C#. يعني ذلك أن أي عملية تعديل على النص، مثل دمج نص جديد أو استبدال جزء منه، تُنتج سلسلة نصية جديدة. يمكن أن يكون استخدام string أكثر كفاءة في الحالات التالية:

* **عند التعامل مع نصوص ثابتة أو غير متغيرة:**  
  إذا كان النص لا يحتاج إلى تعديلات متعددة أو إذا كنت تعرف حجم النص مسبقًا، فاستخدام string يعد الخيار الأفضل. على سبيل المثال، إذا كنت بحاجة إلى تخزين اسم أو عنوان ثابت لن يتغير بعد ذلك.
* **أداء جيد في حالات الاستخدام البسيطة:**  
  العمليات التي تشمل الوصول العشوائي إلى النصوص أو مقارنة السلاسل قد تكون أسرع باستخدام string لأنها مصممة لهذا الغرض.
* **نصوص قصيرة أو غير متكررة التعديل:**  
  إذا كنت بحاجة إلى إجراء تعديل بسيط على النص مثل دمج نصوص بشكل قليل أو تعديل محتوى السلسلة في مكان واحد، فاستخدام string سيكون فعالًا.

**StringBuilder:. استخدام**

StringBuilder هو هيكل بيانات مُعدل (mutable)، مما يعني أنه يمكن تعديل النصوص فيه دون الحاجة إلى إنشاء كائنات جديدة في كل مرة. يجب استخدام StringBuilder في الحالات التي يتطلب فيها التعديل المستمر أو المتكرر على النصوص.

* **عند الحاجة إلى تعديل النصوص بشكل متكرر:**  
  إذا كنت تقوم بدمج نصوص كثيرة أو إجراء تعديلات متعددة على النص (مثل إضافة أو إزالة أجزاء من النص داخل حلقة تكرار)، فإن StringBuilder هو الخيار الأمثل لأنه يقلل من تكلفة إنشاء الكائنات الجديدة.
* **عند التعامل مع نصوص طويلة:**  
  عند العمل مع نصوص كبيرة، حيث أن استخدام string قد يؤدي إلى مشاكل في الأداء بسبب كثرة النسخ، بينما يمكن لـ StringBuilder تعديل النصوص داخل نفس الكائن بكفاءة.
* **أداء ممتاز في الحلقات والتكرار:**  
  إذا كنت بحاجة إلى بناء نص عبر حلقة متعددة، مثل بناء نص من مجموعة من العناصر أو مع بيانات مدخلات StringBuilder هو الأنسب.