

◆ Asynchronous Programming أولاً: إمتى نحتاج نستخدم

يعني الكود يقدر يكمل شغله من غير ما يستنى حاجة تخلص، وده بيساعد Asynchronous البرمجة الـ "البرنامج يفضل سريع وميروحش في حالة" مش بيرد.

async أكثر 3 حالات بنحتاج فيها الـ

1. الاتصال بقاعدة البيانات:
لما البرنامج بيعث استعلام كبير أو يضيف بيانات، ممكن العملية تاخذ وقت زي async بدل ما التطبيق يفضل مستنى، بنخليها:
`await db.SaveChangesAsync();`
2. قراءة أو كتابة الملفات:
لو الملف كبير، القراءة ممكن تبطأ التطبيق فبنستخدم:
`await File.ReadAllTextAsync("data.txt");`
3. أو الإنترنت API الاتصال بـ:
الطلبات دي بتاخذ وقت بسبب الشبكة، فبنستخدم:
`await httpClient.GetAsync(url);`

كده البرنامج يكمل شغله عادي لحد ما الرد بييجي.

◆ Thread و Task ثانيًا: الفرق بين

Thread و Task: عندنا طريقتين لتشغيل كود في الخلفية C# في الاثنين بيشغلوا كود في نفس الوقت، بس الفرق بينهم في الطريقة والإدارة.

- Thread:
دي الطريقة القديمة، وإننت اللي بتتحكم فيها بنفسك
مثال:
`Thread t = new Thread(MyMethod);`
`t.Start();`
- Task:
دي الطريقة الحديثة والأسهل.
، وإننت بس بتقول Thread Pool هو اللي بيدبرها عن طريق .NET.
`Task.Run(() => MyMethod());`

هتشتغل لوحدها من غير ما تشيل هم الإدارة Task والـ await و async كمان بتشتغل حلو جدًا مع.

الخلاصة:

من جوا Thread بتستخدم Task كل

You said:

عايزنا مختلفه بقا خالص

ChatGPT said:

تمام يا محمد

فيها نفس المعلومات بس مكتوبة بطريقة روائية — دي نسخة ريبورت جديدة ومختلفة تماماً في الأسلوب وحوكائية كأنك بتحكي لحد فاهم شوية كود، وعايز تفهمه المفاهيم بطريقة خفيفة ومرتبطة اللغة عامية بس مرتبة ومناسبة للتسليم برضو

وال Task وال Thread وال Asynchronous Programming ريبورت عن الـ ? Delegates في C#

Asynchronous Programming أول جزء: فكرة الـ ⚙️

تخيل إنك فاتح برنامج، ودوست على زرار بيعمل عملية ثقيلة زي جلب بيانات من السيرفر أو قراءة ملف كبير ، هيقف يستنى العملية تخلص وبعدين يكمل — يعني (Synchronous) لو البرنامج شغال بطريقة عادية المستخدم هيحس إن التطبيق هنج.

، اللي بتخلي التطبيق يشتغل "كأنه بيعمل كذا حاجة في Asynchronous Programming وهنا بييجي دور الـ "نفس الوقت".
يعني العملية الطويلة تشتغل في الخلفية، والبرنامج يكمل باقي الشغل عادي

⚙️:أمثلة بسيطة على الحالات دي

- ثقيلة Query ويستنى نتيجة Database لما التطبيق يتصل بـ
- لما يقرأ أو يكتب ملف كبير من الهارد
- API لما بيعث أو يستقبل بيانات من الإنترنت أو

بالتالي، المستخدم يفضل شايف التطبيق شغال وسريع، حتى لو في كود بياخد وقت في الخلفية ووده بيخلي تجربة الاستخدام أحسن بكثير

Task وال Thread ثاني جزء: الفرق بين الـ □

بص كده، الاتنين شبه بعض من حيث إنهم بيشتغلوا كود في الخلفية، بس طريقة الشغل مختلفة شوية

◆ ال Thread

هو وحدة تنفيذ ييشغلها النظام نفسه.
إنت اللي بتعملها وتتحكم فيها بنفسك، زي مثلاً

```
Thread t = new Thread(MyMethod) ;  
t.Start() ;
```

بس دي طريقة متعبة شوية لأنك مسؤول عن كل حاجة فيها: إدارة الموارد، التوقف، المزامنة... إلخ

◆ ال Task

دي بقى طريقة أذكى وأحدث.
Thread Pool هو اللي بيتكفل بإدارة الشريدات عن طريق حاجة اسمها NET.
بتخلي النظام هو اللي يختار ويضبط كل حاجة Task كتير بإيدك، ال Threads يعني بدل ما تعمل

مثلاً:

```
Task.Run ( () => MyMethod () ) ;
```

كده الميثود هتشتغل في الخلفية على طول، وانت مش محتاج تتحكم في أي تفاصيل

باختصار:

زي إنك تسوق عربية مانيوال، ال Thread
زي إنك تسوق أوتوماتيك — نفس الفكرة، بس أسهل وأنصف في الإدارة ال Task وال

□ ال C# الجاهزة في Delegates ثالث جزء: أنواع ال

هو زي متغير يقدر يشاور على ميثود، ال Delegate في C#
يعني بدل ما تبع اسم الميثود، تبع المتغير اللي شايلها
parameter. وده مفيد لما تحب تمرر ميثود كـ

علشان نسهل على نفسنا delegates بقى جهّز لنا 3 أنواع جاهزة من ال C#:

1. Action

(void) لما الميثود ما بترجعش حاجة

مثال:

```
2. Action<string> sayHi = name => Console.WriteLine($"Hi {name}");
```

```
3. sayHi ("Mohamed") ;
```

4. Func

لما الميثود بترجع قيمة

return آخر نوع في البراميتير هو نوع ال

مثال:

```
5. Func<int, int, int> add = (a, b) => a + b;
```

```
6. Console.WriteLine(add(3, 5)); // 8
```

7. Predicate

(يعني اختبار أو شرط) `true` أو `false` لما الميثود بترجع
مثال:

```
8. Predicate<int> isEven = x => x % 2 == 0;  
9. Console.WriteLine(isEven(4)); // True
```

← الخلاصة

بتخلي التطبيق أسرع ومايهنّش وقت العمليات البطيئة `Asynchronous Programming` الـ.
إن الأولى محتاجة إدارة يدوية، والثانية أذكى وأسهل `Task` و `Thread` الفرق بين
فهي طريقة مرنة نقدر نستخدم بيها الميثودز كقيم، `Delegates` أما الـ
بتخلينا نكتب كود أنصف وأوضح `Predicate` و `Action`, `Func` والأنواع الجاهزة زي