

## الفرق بين Stack و Heap في تخزين المتغيرات في .NET

في عالم البرمجة باستخدام C# وبيئة DotNet#، من المهم جداً نفهم إزاي بيتم تخزين المتغيرات (Variables) في الذاكرة، وخصوصاً الفرق بين Value Types و Reference Types، وإزاي كل نوع بيتخزن في Stack أو Heap.

### Stack: الذاكرة المؤقتة والسريعة

الـ **Stack** هو نوع من الذاكرة بيتم فيه تخزين البيانات بطريقة منظمة جداً وسريعة. كل ما يتم استدعاء دالة (Method)، بيتم إنشاء مساحة في الـ Stack لتخزين المتغيرات المؤقتة الخاصة بيها.

#### مميزات Stack:

- السرعة العالية في التخزين والاسترجاع.
- إدارة أوتوماتيكية بدون الحاجة إلى تدخل المبرمج.
- مساحة صغيرة ومحدودة مقارنة بالـ Heap.

### Value Types في Stack

المتغيرات اللي نوعها من الـ **Value Types** (زي `int`, `float`, `bool`, `struct`) بيتم تخزينها مباشرة في الـ **Stack**، وده معناه إن:

- كل نسخة من المتغير مستقلة عن الثانية.
- لما تمرر متغير من النوع ده لدالة، بيتم نسخه، مش مشاركة المرجع.

#### مثال:

```
int a = 5
int b = a
b = 10
a تظل قيمتها 5
```

## ♦ Heap: التخزين الديناميكي ومرونة أكبر

الـ **Heap** هو منطقة في الذاكرة مخصصة لتخزين الكائنات (Objects) التي حجمها مش ثابت أو التي سيتم إنشاؤها أثناء التشغيل (Runtime).

### Reference Types في Heap:

الأنواع التي من نوع **Reference Types** (زي **class, array, string, object**) سيتم تخزينها في الـ **Heap**، بس المرجع ليها (العنوان) بيتخزن في الـ **Stack**.

وده معناه:

- لو نسخت المتغير، فأنت بتنسخ المرجع مش القيمة.
- أي تغيير على الكائن بينعكس في كل الأماكن التي بتستخدم نفس المرجع.


مثال:

```
class Person {  
    public string Name;  
}  
Person p1 = new Person();  
p1.Name = "Ahmed";  
Person p2 = p1;  
p2.Name = "Sheriff";  
p1.Name هتبقى "Sheriff" برضو
```

## 🧠 ملحوظة مهمة: Boxing & Unboxing

أحياناً بيتم تحويل **Value Type** إلى **Reference Type** بشكل تلقائي، وده اسمه **Boxing**، وبيتم فيه نسخ القيمة من الـ **Stack** إلى الـ **Heap**. والعكس اسمه **Unboxing**.

وده بيأثر على الأداء، ولازم ناخذ بالناسخ منه خصوصاً في الكود اللي بيتكرر كتير.

فهمك للفرق بين **Stack** و **Heap** هيساعدك تكتب كود أكفأ، وتفهم سلوك البرنامج، وتحل مشاكل زي الأداء أو تسريب الذاكرة (Memory Leaks). 

لو استفدت من المقال، شاركه مع أصحابك المهتمين بـ **DotNet#** أو **CSharp#** وتابعني للمزيد من المقالات التقنية ✨👤