1. Why is it recommended to explicitly assign values to enum members in some cases?

   By default the first element assigned by 0 and some cases we want to init it by n

   In another cases may be a gap between elements then we want to make an exception for some element which in some cases related by a value

2. What happens if you assign a value to an enum member that exceeds the underlying type's range?
   Throw an exception error: Constant value cannot be converted to a 'Grades' (use 'unchecked" syntax to override)
3. What is the purpose of the virtual keyword when used with properties?
   To use the property in a child class or a class that implements this iface
4. Why can't you override a sealed property or method?
   Because sealed tells the compiler that this override is the last one in the inheritance chain
5. What is the key difference between static and object members?
   Object member: u must create an instance to use methods in the class
   Static members: u can use static methods directly without instanting it
6. Can you overload all operators in C#? Explain why or why not.
   No, we can't overload operators like (=[assignment], . [access member], .? [ternary], new, typof, sizeof, is, as, checked, unchecked)
   why not: C# restricted these operators because it's a meaningless if we changed them. In addition, we will never ever need to change them
   - ➔ For the previous operator we treat with the RAM (either stack or the heap), so, it's constant always and no treat with some object specifically
7. When should you consider changing the underlying type of an enum?
   It's according to our case. For ex: if we have less than 2^8 values we will use byte
8. Why can't a static class have instance constructors?

   Because we access its methods without creating an instance of it

9. What are the advantages of using Enum.TryParse over direct parsing with int.Parse?
   Try parse is a type safety, error handling fnc
   And it can parse both names ("A") and numeric values unlike int.parse() which only works on numbers.

10. What is the difference between overriding Equals and == for object comparison in C# struct and class?

	Class	Struct
.Equals()	inherited from Object and compare between references only (if no override)	inherited from ValueType and performs the comparison btw valueTypes field by field
==	compare between references also (if no overload)	not defined by default. So, it must be overloaded

Why is overriding ToString beneficial when working with custom classes?

To avoid printing it only as {nameSpaces.ClassName} and print informative output

11. Can generics be constrained to specific types in C#? Provide an example.

    Yes, it can be by providing constraints like: [where T : class] to make T only for reference types

12. What are the key differences between generic methods and generic classes?

    Gen method: accept any DType as a parameter even if in a non-gen class
    Gen Class: accept any DType to create an object even if it contains a non-gen methods like any container {arr, list, queue, stack, hashset, ..}

13. Why might using a generic swap method be preferable to implementing custom methods for each type?

    To follow DRY (don't repeat ur self) principle, scalability, maintainability

14. How can overriding Equals for the Department class improve the accuracy of searches?

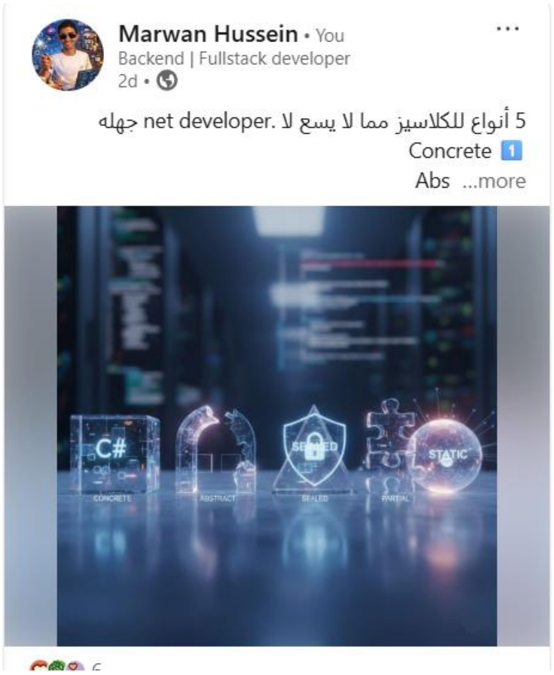
    To check the data is equals to other or not.. not only the addresses

15. Why is == not implemented by default for structs?

    To avoid ambiguity because of structs are value-type. So, .Equals() check it field by field in the stack

Part02

1. LinkedIn article about class types ?



Marwan Hussein • You
Backend | Fullstack developer
2d •

5 أنواع للكلاسيز مما لا يسع لا جهله net developer.
Concrete 1
Abs ...more

2. What we mean by Generalization concept using Generics?
   Generalization: to be DRY within different types -> avoid repeating the same logic all over different dataTypes (like stack that we implemented in part2 that accepts any data type and a constant logic).
   Generics: allow to write a code that works with different datatypes while still being safe.
3. What we mean by hierarchy design in real business?
   To set a base for collection of classes like user in any real system (e.g. all user has username, email, password) and this user may inherited into: teacher, student, admin..etc
   Each one of them have additional functions to do with different grantees.