**Value Type & Reference Type**

**🧱 Value Type**

* **Definition**: A value type holds the actual data directly. When you assign it to another variable, a *copy* of the value is made.
* **Memory**: Typically stored in the **stack**, which is faster and scoped to the current execution context.
* **Examples**: int, float, bool, char, struct (in C#), enum

**🧠 Reference Type**

* **Definition**: A reference type holds a *reference* (or pointer) to the actual data, which is stored elsewhere in memory.
* **Memory**: The reference itself is stored in the **stack**, but the actual object/data is stored in the **heap**.
* **Examples**: class, array, string, delegate, interface

**✅ Best Definitions (Concise & Clear)**

| **Type** | **Best Definition** |
| --- | --- |
| **Value Type** | Stores the actual value directly; copying creates a new independent copy. |
| **Reference Type** | Stores a reference to the value; copying shares access to the same object. |

**🔍 Additional Notes**

* **Scope**: Value types are scoped to their block (e.g., method or loop), but reference types can persist longer if referenced elsewhere.
* **Performance**: Value types are generally faster due to stack allocation, but reference types offer more flexibility for complex data structures.