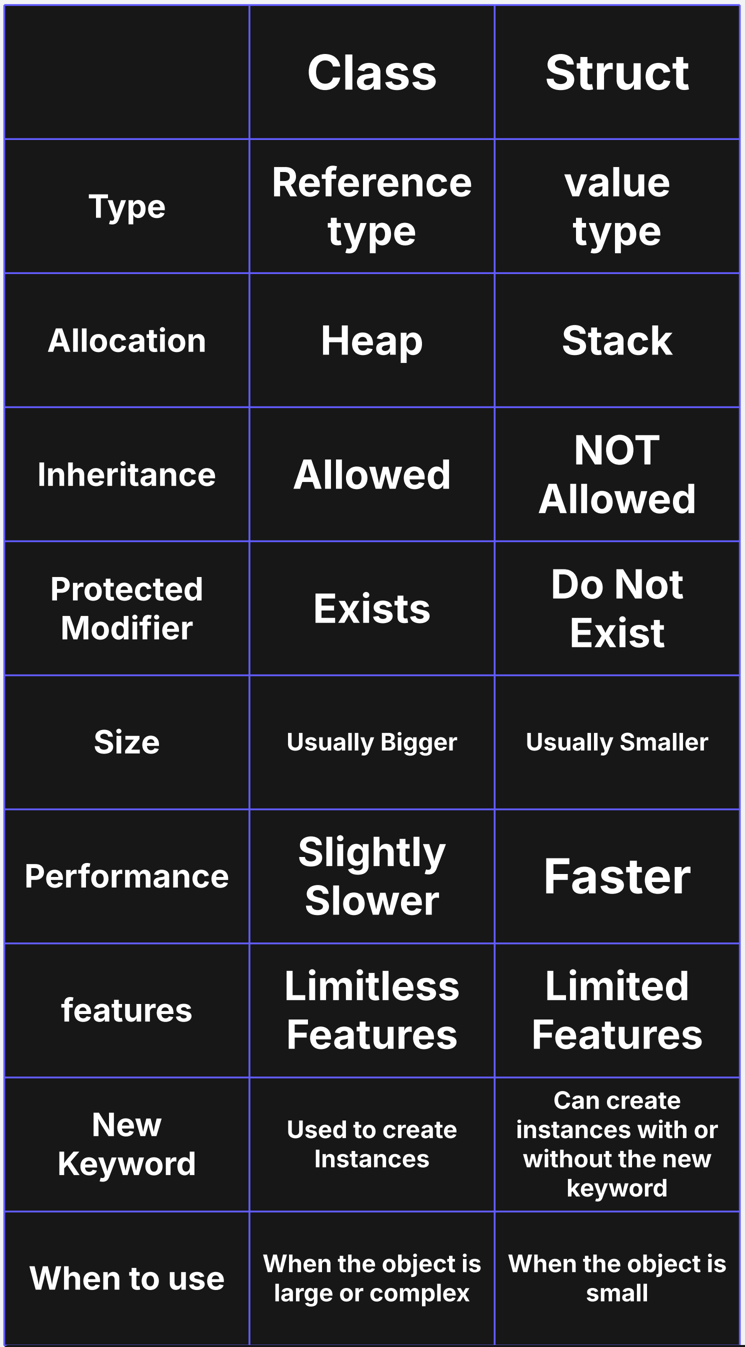
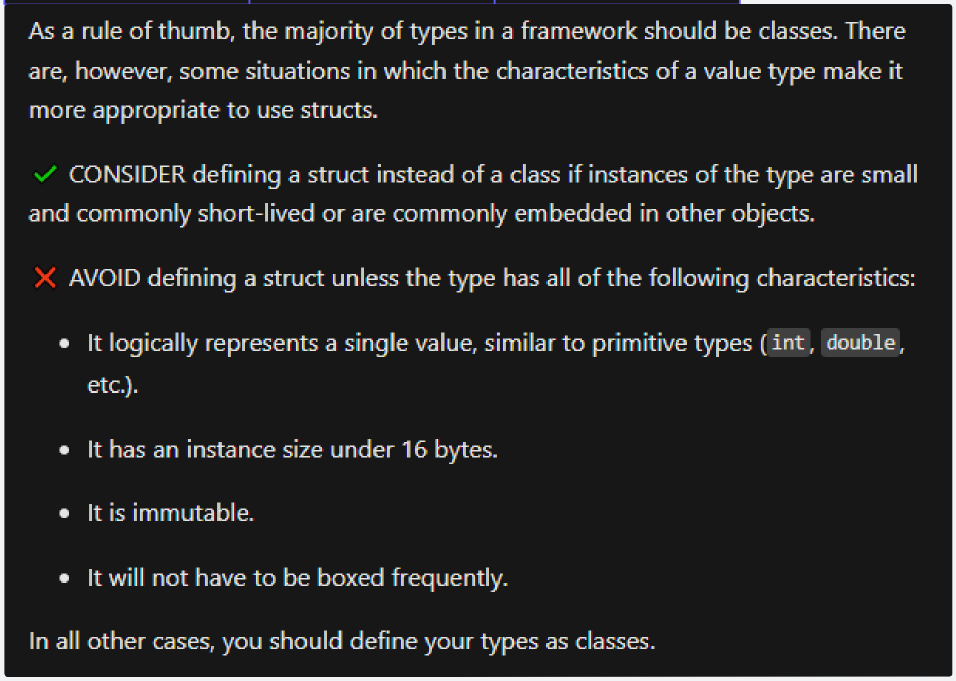


	Class	Struct
Type	Reference type	value type
Allocation	Heap	Stack
Inheritance	Allowed	NOT Allowed
Protected Modifier	Exists	Do Not Exist
Size	Usually Bigger	Usually Smaller
Performance	Slightly Slower	Faster
features	Limitless Features	Limited Features
New Keyword	Used to create Instances	Can create instances with or without the new keyword
When to use	When the object is large or complex	When the object is small



As a rule of thumb, the majority of types in a framework should be classes. There are, however, some situations in which the characteristics of a value type make it more appropriate to use structs.

✓ CONSIDER defining a struct instead of a class if instances of the type are small and commonly short-lived or are commonly embedded in other objects.

✗ AVOID defining a struct unless the type has all of the following characteristics:

- It logically represents a single value, similar to primitive types (`int`, `double`, etc.).
- It has an instance size under 16 bytes.
- It is immutable.
- It will not have to be boxed frequently.

In all other cases, you should define your types as classes.