Classes Versus Structures

One obvious question to ask is: why does Ice provide structures as well as classes, when classes obviously can be used to model structures? The answer has to do with the cost of implementation: classes provide a number of features that are absent for structures:

- Classes support inheritance.
- Classes can be self-referential.
- Classes can have operations.
- Classes can implement interfaces.

Obviously, an implementation cost is associated with the additional features of classes, both in terms of the size of the generated code and the amount of memory and CPU cycles consumed at run time. On the other hand, structures are simple collections of values ("plain old structs") and are implemented using very efficient mechanisms. This means that, if you use structures, you can expect better performance and smaller memory footprint than if you would use classes (especially for languages with direct support for "plain old structures", such as C++ and C#). Use a class only if you need at least one of its more powerful features.

See Also

- Structures
- Classes with Operations
- Classes Implementing Interfaces