.NET > C#.NET > OOP

Structures

- by Harsha Vardhan

Structures

- Structure is a "type", similar to "class", which can contain fields, methods, parameterized constructors, properties and events.
- We can't create object for structure; You can create "structure instance", using which you can access structure members.
- To access structure's member:
  - Syntax: StructureInstance.MemberName
- Advantage: We can easily store and access one or two values that are of small size.

Structure - Syntax

```
struct  StructureName
{
    fields
    methods
    parameterized constructors
    properties
    events
}
```

Structures (vs) Classes

#	Structures	Classes
1	Structures "value-type data types".	Classes are "reference-type data types".
2	Structure instances (includes fields) are stored in stack. Structures doesn't require Heap.	Class instances (objects) are stored in Heap; Class reference variables are stored in stack.
3	Suitable to store small data (only one or two values)	Suitable to store large data (any no. of values)
4	Memory allocation and de-allocation is faster.	Memory allocation and de-allocation is a bit slower.
5	Structures doesn't support Parameter-less Constructor.	Classes support Parameter-less Constructor.
6	Structures doesn't support inheritance (can't be parent or child).	Classes support Inheritance.

7	The "new" keyword just initializes all fields of the "structure instance".	The "new" keyword creates a new object.
8	Structures doesn't support abstract methods and virtual methods.	Classes support abstract methods and virtual methods.
9	Structures doesn't support destructors.	Classes support destructors.
10	Structures are internally derived from "System.ValueType". System.Object → System.ValueType → Structures	Classes are internally and directly derived from "System.Object". System.Object → Structures
11	Structures doesn't support to initialize "non-static fields“, in declaration.	Classes supports to initialize "non-static fields“, in declaration.
12	Structures doesn't support "protected" and "protected internal" access modifiers.	Classes support "protected" and "protected internal" access modifiers.
13	Structure instances doesn't support to assign "null".	Class's reference variables support to assign "null".

Class (vs) Structure - Inheritance and Object

Class Type	Can Inherit from Other Classes	Can Inherit from Other Interfaces	Can be Inherited	Can be Instantiated
Normal Class	Yes	Yes	Yes	Yes
Abstract Class	Yes	Yes	Yes	No
Interface	No	Yes	Yes	No
Static Class	No	No	No	No
Structure	No	Yes	No	Yes

Class (vs) Structure - Members

Class Type	Non-Static Fields	Non-Static Methods	Non-Static Constr-uctors	Non-Static Propert-ies	Non-Static Events	Non-Static Destru-ctors	Constants	Static Fields	Static Methods	Static Constr-uctors	Static Prop-erties	Static Events	Virtual Methods	Abstract Methods	Non-Static Automatic Properties	Non-Static Indexers
Normal Class	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes
Abstract Class	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Interface	No	No	No	No	No	No	No	No	No	No	No	No	No	Yes	Yes	No
Static Class	No	No	No	No	No	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	No	No
Structure	Yes	Yes	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes	Yes	No	No	Yes	Yes

- Structure can contain non-static parameterized constructor, but it must initialize all the non-static fields of the structure.
- Structure can't have non-static parameter-less constructor.