



Module 1 Day 13

Abstract Classes

Little bits of Coolness

- Format specifiers in strings
 - Formatting using `string.Format`, `Console.WriteLine`
 - <https://docs.microsoft.com/en-us/dotnet/standard/base-types/formatting-types>
- Padding strings for alignment; justifying strings
 - Interpolated strings:
 - <https://docs.microsoft.com/en-us/dotnet/csharp/language-reference/tokens/interpolated>
- `ToString()` override

Communicating Design Intent

- Sealed
 - On a class, prevents the class from being sub-classed
 - On a method, prevents further overrides by subclasses
- Access modifiers – when to use public, protected, private
 - Easier to give than to take away

Abstract Methods and Classes

- Abstract Method
 - Superclass provides **no implementation**
 - Subclass **must** implement the method
- Abstract Class
 - A user cannot create an instance of this class
 - **Only subclasses** can be instantiated
 - Should it really be possible to create a new FarmAnimal?
 - What does a FarmAnimal look like?
 - What noises does it make?
 - Can we sell it?
 - Some (or even all) of its methods **may** have implementation
- If a class contains an abstract method, then it **must** be an abstract class

Classes, Abstract Classes, Interfaces, Oh My

Concrete Class	Abstract Class	Interface
Class Inheritance (max 1)	Class Inheritance (max 1)	Interface Inheritance (many)
General → Specialized	General → Specialized	Functionality (can do)
Implementation code (all)	Implementation code (some)	No code, just a contract
May contain public, protected and private members	May contain public, protected and private members	Public members only
Can create an instance	Cannot create an instance	Cannot create an instance
Use when there is a specialization relationship (a true is-a), and the class represents a real-world thing that can exist (e.g., a Pig)	Use when there is a specialization relationship (a true is-a), but the class represents something that doesn't make sense to exist without being further defined (e.g., a Farm Animal)	Use when there is a need to make a class "behave like" or "can do" some additional functionality; when there is not a true "is-a" relationship.