[0499] Chapter 4 Summary

- A method's name, return type, parameters and attributes are called its "signature."
- Method arguments can be passed "by reference" or "by value." If passed by reference, the original variable can be modified by the method it was passed to. If passed by value, a copy of the variable's data is sent.
- By default, all arguments are passed "by value." However, remember that a variable-that-represents-an-object actually contains a pointer to the location on the heap where the actual object lives. So when a variable-that-represents-an-object is passed into a method, the method actually gets a ***copy of the pointer***. That copy points to the same object on the heap as the original variable and, if the object allows it, the method can use that copy of the reference to modify the object's properties and/or call its methods. And so, effectively, if you pass a variable-that-represents-an-object to a method, you are *effectively* passing it "by reference."
- If you have several methods with the same name but different parameter lists, those methods are called "overloaded" methods.
- Optional parameters have default values that are used if those parameters are not included in the method call. Optional parameters must be placed at the end of a method's parameter list.
- Enums let you create a type with a set of fixed, named values like Colors.
- If you want to store null in a type that doesn't normally take null as a value (like an int), add a question mark to the end of the type. (e.g., "int?")