Sunday, December 30, 2018 1:58 PM

Client Class vs Object Class

Everything in Java is part of a class, which is why every Java program we've written so far is defined as has started with the keywords "public class".
So far every class you've written has included a static main method and possibly other static methods. We call these types of classes "clients". Clients are the class with the main method that you tell Java to execute.

Making Objects in a Client

Within a client class you might create objects. This is what happens when you create a "new ArrayList" or a "new Scanner". In this case "ArrayList" and "Scanner" are examples of object classes that define what these objects store and can do. When you create new instances of these objects in the client class you can fill them with data and use their methods to ask them to do things.
Here's how a client class creates three new Bicycle objects each with their own brand name, model name, list price, and wheel size:

```
public class BikeStore {
   public static void main(String[] args) {
      Bicycle bike1 = new Bicycle("Giant","OCR-1",899.99,28);
      Bicycle bike2 = new Bicycle("Specialized","Dolce",1399.99,30);
      Bicycle bike3 = new Bicycle("Schwinn","Cruiser",299.99,24);
   }
}
```

Each of these three variables are an instance of a new Bicycle object, each with its own data.
This code would be contained within a client class with a name like "BikeManager.java" that would have a main method. When you run that class it would reference a separate file like "Bicycle.java" that contains the object definition for "Bicycle" to know how to create these three variables.

Manipulating Objects in a Client

To ask an object to do things using its methods, you use the name of the specific object to "qualify" the method name. Then pass in any necessary parameters and use its return value (if any) as usual:

```
if (!bike1.isSold()) {
   bike1.setSold();
   Manual m = bike1.getManual();
}
```

These two method only refer to the inner state of the bike1 variable, but leave the bike2 and bike3 object instances untouched.

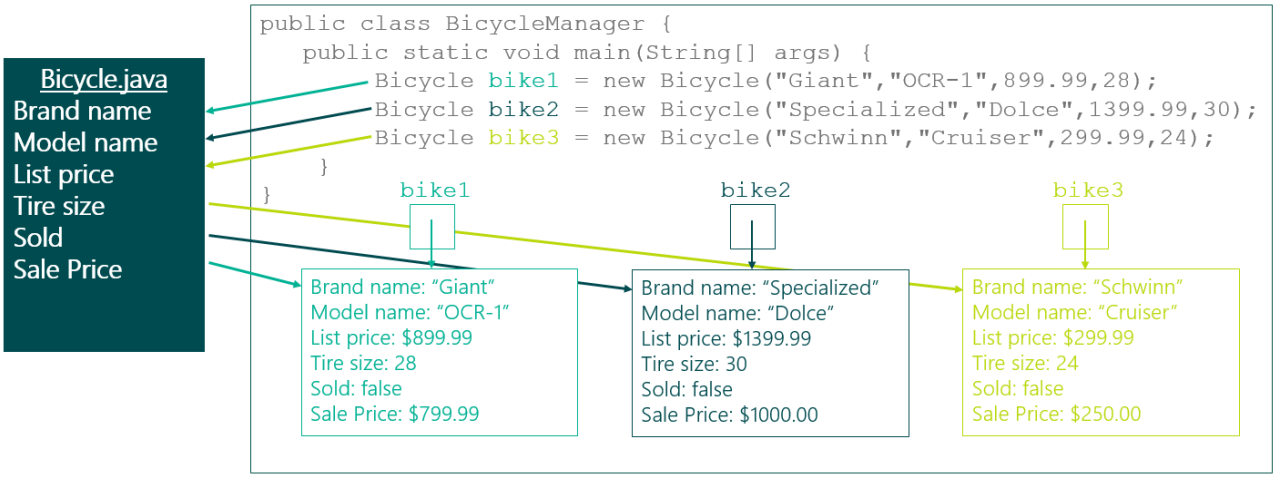
The code for these methods exist in the object definition, but they are executed in the client.

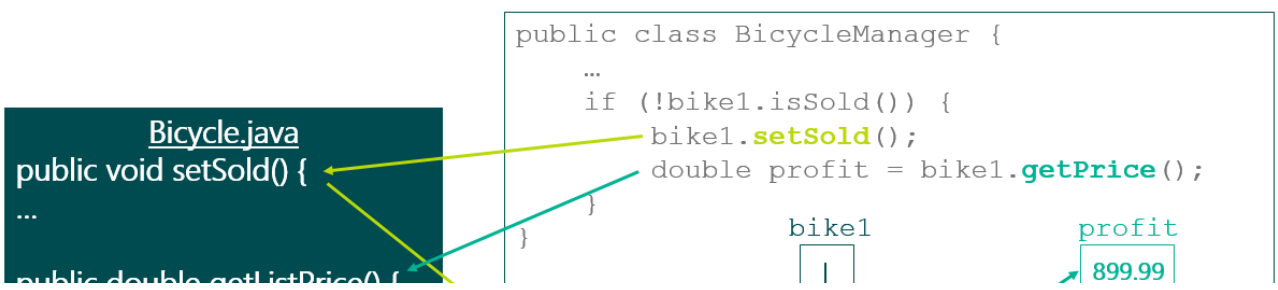
You'll probably recognize this syntax for calling methods from when we used it with String variables:

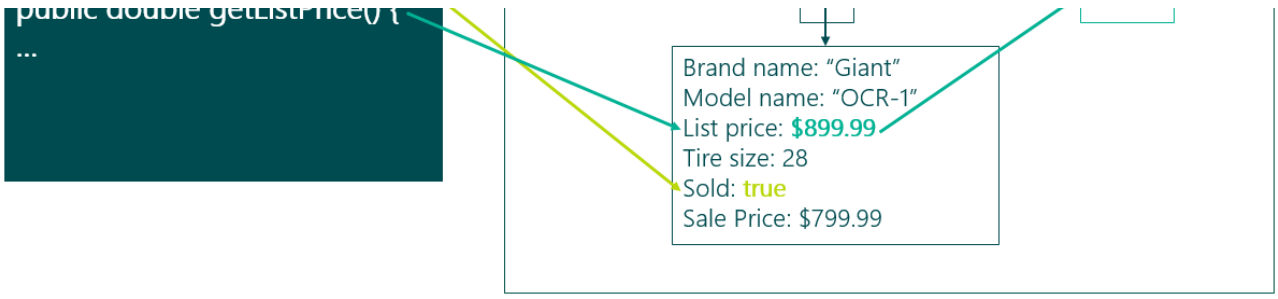
```
String name = "Theodore";
if (name.length() > 0) {
   name = name.toUpperCase();
   String nickName = name.substring(0, 4);
   int length = nickName.length();
}
```

As you can see, you use the name of the specific object variable to call the method on that instance, like so:

```
ObjectDataType variableName = new ObjectDataType();
variableName.objectMethod();
```






public double getListPrice() {
...
Brand name: “Giant”
Model name: “OCR-1”
List price: $899.99
Tire size: 28
Sold: true
Sale Price: $799.99