# this in constructors

It is often the case that overloaded methods are essentially the same except that one supplies default values for some of the arguments. In this case, your code will be easier to read and maintain (though perhaps *marginally* slower) if you put all your logic in the method that takes the most arguments, and simply invoke that method from all its overloaded variants that merely fill in appropriate default values.

This technique should also be used when one method needs to convert from one type to another. For instance one variant can convert a String to an int, then invoke the variant that takes the int as an argument.

This straight-forward for regular methods, but doesn't quite work for constructors because you can't simply write a method like this:

public Car(String licensePlate, double maxSpeed) {  
  
 Car(licensePlate, 0.0, maxSpeed);  
   
 }

Instead, to invoke another constructor in the same class from a constructor you use the keyword this like so:

public Car(String licensePlate, double maxSpeed) {  
  
 this(licensePlate, 0.0, maxSpeed);  
   
 }

Must this be the first line of the constructor?

For example,

public class Car {  
  
 private String licensePlate; // e.g. "New York A456 324"  
 private double speed; // kilometers per hour  
 private double maxSpeed; // kilometers per hour  
   
 // constructors  
 public Car(String licensePlate, double maxSpeed) {  
  
 this(licensePlate, 0.0, maxSpeed);  
   
 }  
  
 public Car(String licensePlate, double speed, double maxSpeed) {  
  
 this.licensePlate = licensePlate;   
 if (maxSpeed >= 0.0) {  
 this.maxSpeed = maxSpeed;  
 }  
 else {  
 maxSpeed = 0.0;  
 }  
   
 if (speed < 0.0) {  
 speed = 0.0;  
 }  
   
 if (speed <= maxSpeed) {  
 this.speed = speed;  
 }  
 else {  
 this.speed = maxSpeed;  
 }  
   
 }  
  
 // other methods...  
   
}

This approach saves several lines of code. In also means that if you later need to change the constraints or other aspects of construction of cars, you only need to modify one method rather than two. This is not only easier; it gives bugs fewer opportunities to be introduced either through inconsistent modification of multiple methods or by changing one method but not others.

[Previous](http://docs.google.com/01.html) | [Next](http://docs.google.com/03.html) | [Top](http://docs.google.com/index.html) | [Cafe au Lait](http://www.cafeaulait.org/)

Copyright 1997-1999 Elliotte Rusty Harold

[elharo@metalab.unc.edu](mailto:elharo@metalab.unc.edu)

Last Modified February 12, 1999