The net effect is to display two digits to the right of the decimal, including trailing zeros. Actually, only the first two are needed according to current practices, and older implementations just need the first and third. Using all three produces the same output for both implementations. See Chapter 17, "Input, Output, and Files," for the details. Meanwhile, here is the program output:

Company: NanoSmart  Shares: 20
  Share Price: $12.50  Total Worth: $250.00
Company: NanoSmart  Shares: 35
  Share Price: $18.25  Total Worth: $638.75
You can't sell more than you have! Transaction is aborted.
Company: NanoSmart  Shares: 35
  Share Price: $18.25  Total Worth: $638.75

Note that main() is just a vehicle for testing the design of the Stock class. Given that the class works as we like, we now can use the Stock class as a user-defined type in other programs. The critical point in using the new type is understanding what the member functions do; you shouldn't have to think about the implementation details. See the "The Client/Server Model" sidebar.

The Client/Server Model

OOP programmers often discuss program design in terms of a client/server model. In this conceptualization, the client is a program that uses the class. The class declaration, including the class methods, constitute the server, which is a resource available to the programs that need it. The client uses the server through the publicly defined interface only. This means the client's only responsibility, and, by extension, the client's programmer's only responsibility, is to know that interface. The server's responsibility, and, by extension, the server's designer's responsibility, is to see that the server reliably and accurately performs according to that interface. Any changes the server designer makes to the class design should be to details of implementation, not to the interface. This allows programmers to improve the client and the server independently of each other, without changes in the server having unforeseen