## Chapter 1

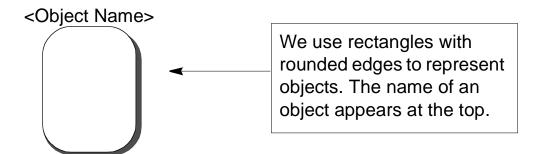
# Introduction to Object-Oriented Programming and Software Development

### **OBJECTIVES**

After you have read and studied this chapter, you should be able to

- Name the basic components of object-oriented programming.
- Differentiate classes and objects.
- Differentiate class and instance methods.
- Differentiate class and instance data values.
- Draw object diagrams using icons for classes, objects, and other components of object-oriented programming.
- Describe the significance of inheritance in object-oriented programs.
- Name and explain the stages of the software life cycle.

### FIGURE 1.1 A graphical representation of an object.



### **Example:**

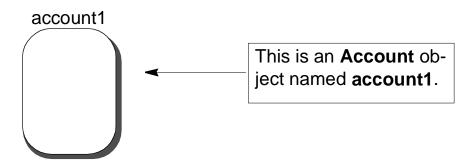
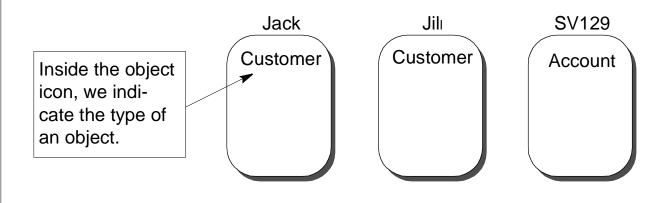


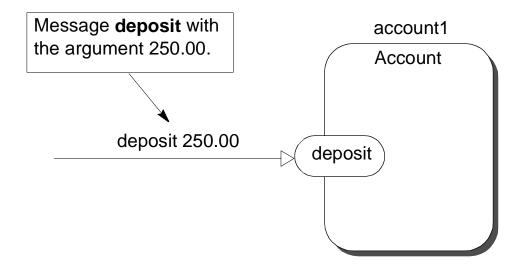
FIGURE 1.2 Two **Customer** objects with the names **Jack** and **Jill** and one **Account** object with the name **SV129**.



# FIGURE 1.3 A graphical representation of a class. Class Name> We use rectangles to represent classes. The name of a class appears at the top. Example: Account This is the Account class.

FIGURE 1.4 Two classes, Account and Customer, with two Account objects and three Customer objects. Account Customer This line shows an instance-of relationship, for example, Jack is an instance of Customer. Jack's Acct John's Acct Jack John Account Account Customer Customer Jill Customer

FIGURE 1.5 Sending the message **deposit** to an **Account** object.



Since the message name must be the same as the method name, we usually omit the message name from the arrow.

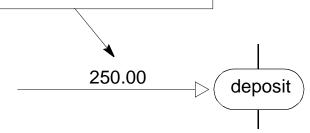


FIGURE 1.6 The result **monthly fee** is returned to the sender of the message.

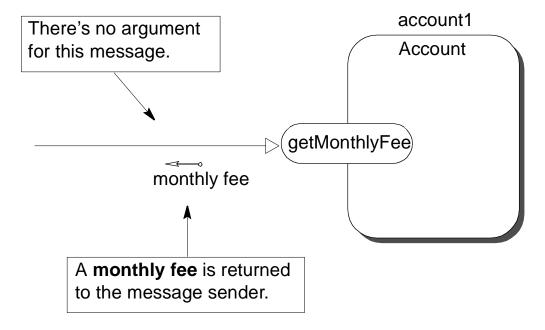
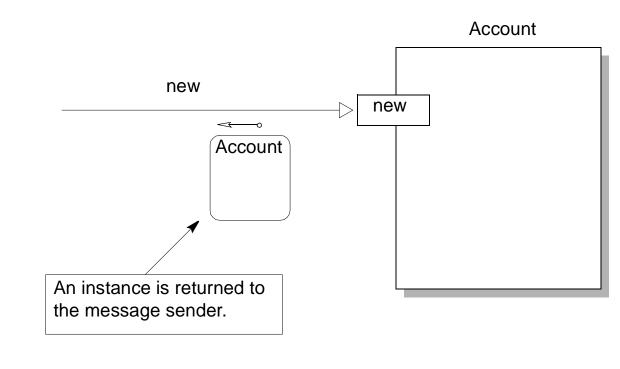


FIGURE 1.7 The newly created **Account** object is returned to the sender of a message.



Graphical representation for classes, instances, and their FIGURE 1.8 methods. <Class Name> We use squared corners for a class and class <result> The dotted line signifies that you cannot send instance messages to a class. <Object Name> We use rounded corners <Class Name> for an object and object methods. <result> Three **Account** objects possess the same data value FIGURE 1.9 current balance, but the actual dollar amounts differ. John's Acct Jill's Acct Jack's Acct current balance current balance current balance 354.00 908.55 1304.98

FIGURE 1.10 Three **Account** objects sharing information (**minimum balance** = \$100) stored as a class data value.

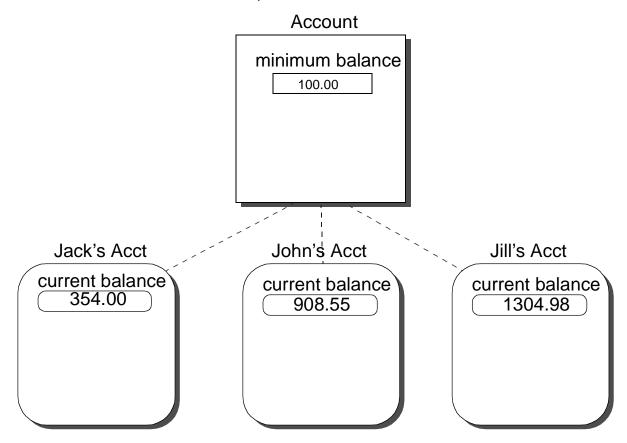


FIGURE 1.11 Three **Account** objects duplicating information (**minimum balance** = \$100) in instance data values.

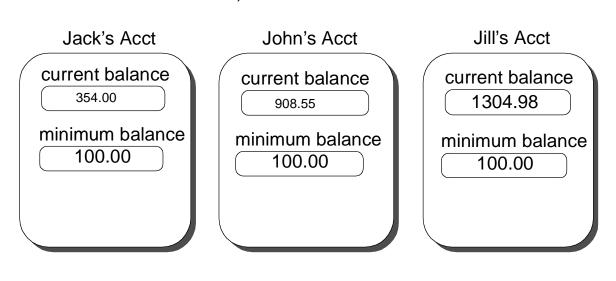


FIGURE 1.12 Graphical representations for four types of data values: class variable, class constant, instance variable, and instance constant. Account minimum balance This is a lock icon 100.00 to indicate that a This number is a prefix constant value to the account number account prefix cannot change. of all instances of the 6427 Account class; this prefix never changes. John's Acct Jack's Acct Jill's Acct current balance current balance current balance 354.00 1304.98 908.55 opening balance opening balance opening balance 50.00 100.00 246.00

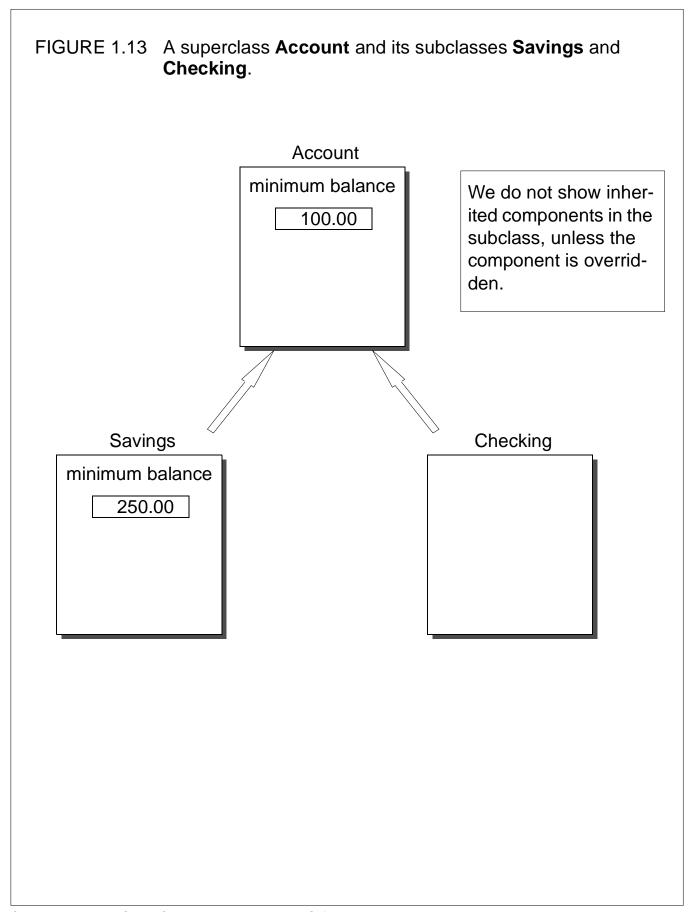


FIGURE 1.14 An example of inheritance hierarchy among different types of accounts. Account Savings Checking College Student Regular Saving Super Interest **ATM** Savers Checking Bearing

```
Program FunTime
  The program will allow you to draw a picture by
  dragging a mouse (move the mouse while holding the left mouse
  button down; hold the button on Mac). To erase the picture and
  start over, click the right mouse button (command-click on Mac).
* /
import javabook.*;
class FunTime
  public static void main(String args[])
    SketchPaddoodleBoard;
    doodleBoard = new SketchPad();
    doodleBoard.show();
}
```

This statement makes doodleBoard appear on the screen.

This statement creates a new **SketchPad** object **doodleBoard**.

FIGURE 1.15 The window that appears on the screen when the program starts running.

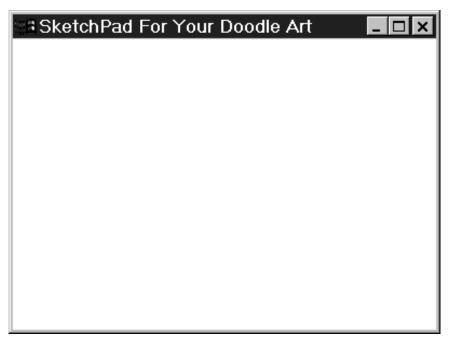


FIGURE 1.16 The same window after a picture is drawn.

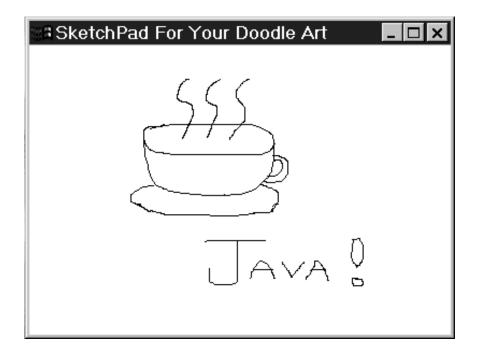


FIGURE 1.17 The object diagram for the **FunTime** program.

