You'll see something new.
Two things. And I call them
Thing One and Thing Two.
—Dr. Theodor Seuss Geisel

Nothing can have value without being an object of utility.
—Karl Marx

Your public servants serve you right.
—Adlai E. Stevenson

Knowing how to answer one who speaks,
To reply to one who sends a message.
—Amenemope

3

Introduction to Classes and Objects

Objectives

In this Chapter you'll learn:

- What classes, objects, methods and instance variables are.
- How to declare a class and use it to create an object.
- How to declare methods in a class to implement the class's behaviors.
- How to declare instance variables in a class to implement the class's attributes.
- How to call an object's methods to make those methods perform their tasks.
- The differences between instance variables of a class and local variables of a method.
- How to use a constructor to ensure that an object's data is initialized when the object is created.
- The differences between primitive and reference types.

Self-Review Exercises

3.1 Fill in the blanks in each of the following:

a) A house is to a blueprint as a(n) _____ is to a class.

ANS: object.

b) Each class declaration that begins with keyword _____ must be stored in a file that has exactly the same name as the class and ends with the `.java` file-name extension.

ANS: `public`.

c) Keyword _____ in a class declaration is followed immediately by the class's name.

ANS: `class`.

d) Keyword _____ requests memory from the system to store an object, then calls the corresponding class's constructor to initialize the object.

ANS: `new`.

e) Each parameter must specify both a(n) _____ and a(n) _____.

ANS: type, name.

f) By default, classes that are compiled in the same directory are considered to be in the same package, known as the _____.

ANS: default package.

g) When each object of a class maintains its own copy of an attribute, the field that represents the attribute is also known as a(n) _____.

ANS: instance variable.

h) Java provides two primitive types for storing floating-point numbers in memory: _____ and _____.

ANS: `float`, `double`.

i) Variables of type `double` represent _____ floating-point numbers.

ANS: double-precision.

j) `Scanner` method _____ returns a `double` value.

ANS: `nextDouble`.

k) Keyword `public` is an access _____.

ANS: modifier.

l) Return type _____ indicates that a method will not return a value.

ANS: `void`.

m) `Scanner` method _____ reads characters until a newline character is encountered, then returns those characters as a `String`.

ANS: `nextLine`.

n) Class `String` is in package _____.

ANS: `java.lang`.

o) A(n) _____ is not required if you always refer to a class with its fully qualified class name.

ANS: `import` declaration.

p) A(n) _____ is a number with a decimal point, such as 7.33, 0.0975 or 1000.12345.

ANS: floating-point number.

q) Variables of type `float` represent _____ floating-point numbers.

ANS: single-precision.

r) The format specifier _____ is used to output values of type `float` or `double`.

ANS: `%f`.

s) Types in Java are divided into two categories—_____ types and _____ types.

ANS: primitive, reference.

3.2 State whether each of the following is *true* or *false*. If *false*, explain why.

a) By convention, method names begin with an uppercase first letter, and all subsequent words in the name begin with a capital first letter.

ANS: False. By convention, method names begin with a lowercase first letter and all subsequent words in the name begin with a capital first letter.

b) An `import` declaration is not required when one class in a package uses another in the same package.

ANS: True.

c) Empty parentheses following a method name in a method declaration indicate that the method does not require any parameters to perform its task.

ANS: True.

d) Variables or methods declared with access modifier `private` are accessible only to methods of the class in which they are declared.

ANS: True.

e) A primitive-type variable can be used to invoke a method.

ANS: False. A primitive-type variable cannot be used to invoke a method—a reference to an object is required to invoke the object's methods.

f) Variables declared in the body of a particular method are known as instance variables and can be used in all methods of the class.

ANS: False. Such variables are called local variables and can be used only in the method in which they are declared.

g) Every method's body is delimited by left and right braces (`{` and `}`).

ANS: True.

h) Primitive-type local variables are initialized by default.

ANS: False. Primitive-type instance variables are initialized by default. Each local variable must explicitly be assigned a value.

i) Reference-type instance variables are initialized by default to the value `null`.

ANS: True.

j) Any class that contains `public static void main( String[] args )` can be used to execute an application.

ANS: True.

k) The number of arguments in the method call must match the number of parameters in the method declaration's parameter list.

ANS: True.

l) Floating-point values that appear in source code are known as floating-point literals and are type `float` by default.

ANS: False. Such literals are of type `double` by default.

3.3 What is the difference between a local variable and a field?

ANS: A local variable is declared in the body of a method and can be used only from the point at which it is declared through the end of the method declaration. A field is declared in a class, but not in the body of any of the class's methods. Also, fields are accessible to all methods of the class. (We'll see an exception to this in Chapter 8, Classes and Objects: A Deeper Look: Solutions.)

3.4 Explain the purpose of a method parameter. What is the difference between a parameter and an argument?

ANS: A parameter represents additional information that a method requires to perform its task. Each parameter required by a method is specified in the method's declaration. An argument is the actual value for a method parameter. When a method is called, the argument values are passed to the corresponding parameters of the method so that it can perform its task.

Exercises

NOTE: Solutions to the programming exercises are located in the **`ch03solutions`** *folder. Each exercise has its own folder named* **`ex03_##`** *where ## is a two-digit number representing the exercise number. For example, exercise 3.11's solution is located in the folder* **`ex03_11`**.

3.5 What is the purpose of keyword `new`? Explain what happens when you use it.

ANS: The purpose of keyword `new` is to create an object of a class. When keyword `new` is used in an application, first a new object of the class to the right of `new` is created, then the class's constructor is called to initialize the object.

3.6 What is a default constructor? How are an object's instance variables initialized if a class has only a default constructor?

ANS: A default constructor is provided by the compiler when you do not specify any constructors in the class. When a class has only the default constructor, its instance variables are initialized to their default values. Variables of types `char`, `byte`, `short`, `int`, `long`, `float` and `double` are initialized to `0`, variables of type `boolean` to `false`, and reference-type variables to `null`.

3.7 Explain the purpose of an instance variable.

ANS: A class provides an instance variable (or several instance variables) when each object of the class must maintain information separately from all other objects of the class. For example, a class called `Account` that represents a bank account provides an instance variable to represent the balance of the account. Each `Account` object maintains its own balance but does not know the balances of the bank's other accounts.

3.8 Most classes need to be imported before they can be used in an application. Why is every application allowed to use classes `System` and `String` without first importing them?

ANS: Classes `System` and `String` are both in package `java.lang`, which is implicitly imported into every Java source-code file.

3.9 Explain how a program could use class `Scanner` without importing the class.

ANS: If every use of a class's name in a program is fully qualified, there is no need to import the class. A class's fully qualified name consists of the class's package followed by the class name. For example, a program could use class `Scanner` without importing it if every use of `Scanner` in the program were specified as `java.util.Scanner`.

3.10 Explain why a class might provide a *set* method and a *get* method for an instance variable.

ANS: An instance variable is typically declared `private` in a class so that only the methods of the class in which the instance variable is declared can manipulate it. This prevents the variable from being modified accidentally by a class in another part of the program. In some cases, it may be necessary for an application to modify the `private` data. For example, the owner of a bank account should be able to deposit or withdraw funds and check the account's balance. A class's designer can provide `public` *set* and *get* methods that enable an application to specify the value for, or retrieve the value of, a particular object's `private` instance variable.