have other sophisticated features as well, such as executing multiple statements at a time by setting breakpoints. Another way to debug a program is to trace the output by printing out intermediate results at different points in the program. However, you can prevent logic errors by designing and coding the algorithm carefully.

1.9 Summary

After reading this chapter, you will have gained a better understanding of what is inside a computer and how programs are executed. The structure and syntax of Java programs was explained briefly. An example program called `HelloWorld` was provided, which displayed a picture on the computer screen. The `HelloWorld` program gives you only a glimpse of what you can do with Java. As we dig deeper in subsequent chapters, you will get a better appreciation of how rich the Java language is and how powerful its programs are.

Exercises

1. Give examples of programs that you can use on your computer.
2. Explain each of these terms:
   a. Algorithm
   b. Java bytecode
   c. Machine language
   d. High-level programming language
3. What does a compiler do?
4. What is the JVM, and what does the Java interpreter do?
5. How is a .class file different from a .java file?
6. What is the software development method?
7. Explain the steps for creating and running a Java program.
8. Describe the three types of errors that can be present in a program.
9. Modify the `HelloWorld` program so that it displays a different picture on the screen.
10. List some applications that use Java. (You can search the Web for information on this.)

CHAPTER 2

Introduction to Classes and Objects

CHAPTER CONTENTS

Draw an ellipse The following statements draw an ellipse called `myellipse` of the specified `width` and `height` and positioned at (x, y):

```
Ellipse2D.Float myellipse = new Ellipse2D.Float(x, y, width, height);
dk.draw(myellipse);
```

Draw a line The following statements draw a line called `myLine`:

```
Line2D.Float myLine = new Line2D.Float(x1, y1, x2, y2);
dk.draw(myLine);
```

The first two arguments (x1, y1) specify the coordinates of the starting point, and the next two arguments (x2, y2) give the coordinates of the ending point.

Draw a rounded rectangle To draw a rectangle with rounded corners called `rectRounded` having the given `width` and `height`, with its top-left corner coordinates at the point (x, y) and the specified corner width and height, use the following statement:

```
RoundRectangle2D.Float rectRounded = new RoundRectangle2D.Float(x, y,
width, height, cornerWidth, cornerHeight);
```

Color a shape To color a Java 2D graphics object called `shape1` a given color (say, red), use the following statements:

```
dk.setPaint(Color.red);
dk.fill(shape1);
```

Write text Write the string literal "`Hello there`" starting at the position (x, y) with the following statements:

```
dk.drawString("Hello there", x, y);
```

Change font To change the font to `Arial` with style `Bold` and size 12, use the following statements:

```
Font myfont = new Font("ARIAL", Font.BOLD, 12);
dk.setFont(myfont);
```

Change line thickness The following statements change the thickness of the lines used to draw shapes:

```
BasicStroke myStroke = new BasicStroke(thickness);
dk.setStroke(myStroke);
```


Draw an image The following statement draws an image from the specified file on the window:

```
dk.drawPicture("space.jpg");
```

Exercises

1. Explain briefly:
   a. What is a class?
   b. How can you create an object of a class?
   c. What is a reference variable?
   d. How is a class different from an object?
   e. What is an instance of a class?
   f. What is a method?
   g. What is a constructor?
   h. What is a package?
   i. What is the `main` method?
2. Write a program that uses the `System.out.println` method to print out your first and last names on different lines.
3. Write a program that uses the `System.out.print` method to print out the current date and time.
4. Fix the errors in the following program so that it compiles and runs correctly:

```
public class ProgramWithErrors {
   public void main(String[] args) {
         System.out.println("Program runs correctly")
      }
   }
```

5. Fix the errors in the following program so that it compiles and runs correctly:

```
public class AnotherProgramWithErrors {
   public static void main(String args) {
         System.out.println("Program runs correctly");
}
```

Graphics Problems

6. Give the statements needed to create an object of the following classes:
   a. `Rectangle2D.Float`
   b. `Ellipse2D.Float`
   c. `Line2D.Float`
   d. `QuadCurve2D.Float`
7. What do the following methods do?
   a. `draw` method of `DrawingKit`
   b. `fill` method of `DrawingKit`
   c. `drawPicture` method of `DrawingKit`
   d. `setCurve` method of `QuadCurve2D.Float`
8. Fix the order of statements in the following program so that it draws a red ellipse on the screen:

```
import java.awt.*;
import java.awt.geom.*;
import com.programwithjava.basic.DrawingKit;

public class EllipseDemo {
    public static void main(String[] args) {
      DrawingKit dk = new DrawingKit("Ellipse");
      dk.draw(rect);
      Ellipse2D.Float rect = new Ellipse2D.Float(50, 100, 200, 100);
      dk.fill(rect);
      dk.setPaint(Color.red);
    }
}
```

9. Fix the compilation errors in this program:

```
import java.awt.*;
import java.awt.geom.*;
import com.programwithjava.basic.DrawingKit;

public class RectangleDemo {
      DrawingKit dk = new DrawingKit("Ellipse");
      Rectangle2DFloat rect = new Rectangle2DFloat(50, 100, 200);
      dk.draw(rect1);
      BasicStroke stroke = new BasicStroke(22);
      dk.setStroke(stroke);
```



```
      dk.setPaint(Color = blue);
      line1 = new Line2D.Float(285, 175, 300, 155);
      dk.draw(line1);
}
```

10. Write a program to draw a rectangle inside a window with its top-left corner at (30, 50) and a width and height of 100 and 300, respectively. Color this rectangle yellow.
11. Repeat the previous problem to create the rectangle with rounded corners.
12. Write a program to draw a line inside a window joining the points (0, 0) and (500, 500) with thickness 7. Color this line blue.
13. Write a program to draw an ellipse having width 50 and height 60 that just fits in a rectangle with its top-left corner at (100, 100).
14. Write a program to draw a curve inside a window joining the points (30, 100) and (100, 300), and passing through the point (50, 150).
15. Write a program to write your name inside a window in italics using Arial font of size 15.
16. Write a program to draw a robot using the Java statements discussed in this chapter, and your imagination.
17. Repeat the previous problem to draw a vehicle of your choice.

Further Reading

For more information on Java, the interested reader can consult the references [1–5]. [1] is a set of online Java tutorials. You can find detailed information on other Java 2D classes and advanced graphics techniques in [6].

References

1. "The Java™ Tutorials." Web. <http://download.oracle.com/javase/tutorial/>.
2. Guzdial, Mark, and Barbara Ericson. *Introduction to Computing and Programming in Java: A Multimedia Approach.* Upper Saddle River, NJ: Pearson Prentice Hall, 2007. Print.

defined in the
`BasicStroke(float w)`—a constr
roke object with
the specified width `w`. It is defined in the `BasicStroke` class.
`Font(String n, int s, int r)`—a constructor that creates a font object
with the given name `n`, style `s`, and size `r`. This constructor is defined
in the `Font` class.

Java 2D `Color` Class These constructors in the `Color` class create a new color
object:

`Color(int red, int green, int blue, int alpha)`—a constructor that creates a new color using the specified values for each of its components.

`Color(int rgbvalue)`—a constructor that creates a new `Color` object using the encoded `rgbvalue`.

Java 2D `BufferedImage` Class The methods in the `BufferedImage` class retrieve useful information about an image:

`int getHeight()`—a method that returns the number of pixels in a column of the image.

`int getWidth()`—a method that returns the number of pixels in a row of the image.

`int getRGB(int x, int y)`—a method that returns the encoded color value of a pixel at location `(x, y)`.

`void setRGB(int x, int y, int rgbvalue)`—a method that sets the color of a pixel at location `(x, y)` to an encoded color value `rgbvalue`.

`DrawingKit` Class Any of these four constructors can be used to create a `DrawingKit` object:

`DrawingKit()`—a constructor that creates a window with a default width and height of 500.

`DrawingKit(String title)`—a constructor that creates a window with the specified title and a width and height of 500.

`DrawingKit(int width, int height)`—a constructor that creates a window with the specified width and height.

`DrawingKit(String title, int width, int height)`—a constructor that creates a window with the specified title, width, and height.

`DrawingKit` has the following methods:

`void drawPicture(BufferedImage picture, int x, int y)`—a method that draws a `picture` of type `BufferedImage` at coordinates `(x, y)`.

`public void drawPicture(String s)`—a method that draws the picture contained in the file with filename `s` at coordinates `(0, 0)`.

`void drawString(String s, float x, float y)`—a method that writes the string `s` at coordinates `(x, y)`.

`BufferedImage loadPicture(String f)`—a method that loads the image from file `f` into the computer's memory.

`void setFont(Font f)`—a method that sets the font to `f`.

`void setPaint(Color c)`—a method that sets the current color to `c`.

`void setStroke(BasicStroke s)`—a method that sets the line thickness to `s`.

`void draw(Shape obj)`—a method that draws the Java 2D graphics object `obj` on the window.

`void fill(Shape obj)`—a method that fills the Java 2D graphics object `obj` with the current color.

The `draw` and `fill` methods take an argument of a type that we have not discussed yet (`Shape`). Narrower types, such as `Rectangle2D.Float`, `Line2D.Float`, `Ellipse2D.Float`, and `RoundRectangle2D.Float`, are converted automatically to the wider type `Shape`. This automatic conversion of reference types is known as **upcasting**. We discuss this in Chapter 6, *Inheritance*.

Exercises

1. State whether the following statements are correct or not, and correct the errors if any occur:

   a. `Double I, j;`

   b. `float k = 200.0;`

   c. `double m = 14;`

   d. `int 123_alpha = 10;`

   e. `10 = b + c;`

   f. `int b, c, a; b + c = a;`

2. Which of the following are Java keywords?

   a. `main`

   b. `class`

c. `static`

d. `Rectangle2D`

e. `New`

3. Declare variables to hold numbers in the following ranges:
   a. from −10 to +100
   b. from 10,000 to 30,000
   c. from 101.5 to 150.5
4. Declare a variable to hold the character "c".
5. Declare a `boolean` variable initialized to `true`.
6. Explain the differences between each of the following terms:
   a. Primitive and object reference variables
   b. Literals and constants
   c. Parameters and arguments
   d. Conversion and casting
7. Explain these terms:
   a. Pixel in a digital image
   b. Primary colors
   c. RGB color model
   d. Sample
   e. Component
   f. Lossy compression
   g. Lossless compression
   h. Channel
   i. Alpha channel
   j. Bit depth
8. Give the values of variable `f` after each of the following statements (using arithmetic operators) is executed.

```
int a = 5, b = 10, f;
```

a. `f = a + b;`

b. `f = a % b;`

c. `f = a / b;`

9. Give the values of variable `f` after each of the following statements (using increment and decrement operators) is executed.

```
int a = 5, b = 10, f;
```

a. `f = ++a - b;`

b. `f = a-- - b;`

c. `f = ++a + b--;`

10. Give the values of variable `f` after each of the following statements (using special assignment operators) is executed:

```
int a = 10, b = 50, f = 5;
```

a. `f += a + 10;`

b. `f *= b / a;`

c. `f %= b + a;`

11. Find the values of the variables `fee` and `rem` after the following statements are executed:

```
double fee = 5.5;
int rem = 150;
fee *= 10.5;
rem %= 10;
```

12. Find the value of variable `g` after each of the following statements is executed. Use the mnemonic PUMAS to help you parenthesize the expression correctly:

```
int a = 1, b = 2, c = 3, d = 4, e = 5, f = 6, g;
```

a. `g = ++a * b / c - d * e % f;`

b. `g = a++ * b / (c - d) * e % f;`

c. `g = ++a * b / c - d * (e % f);`

13. Give the values of variable `f` after each of the following statements using different types of operators is executed. Use the mnemonic PUMAS to help you parenthesize the expression correctly:

```
int a = 5, b = 4, c = 3, d = 2, e = 1, f = 0;
```

a. `f += a % b++ -  (c / d) * e;`

b. `f += a % b++ - c / d * e;`

c. `f %= a % b++ - c / d * e--;`

d. `f -= a % b++ - c / d * e--;`

e. `f %= a + b - c * d + e;`

f. `f %= a + (b - c)* d + e;`

14. Identify the errors, if any, in the following statements:

```
int x = 10, y = 20; float z = 30.5f;
```

a. `int index = (float) z;`

b. `float salary = (int) (x * 1000.5f);`

c. `int value = (int) z;`

d. `boolean isRed = 10;`

e. `x = x - 100++;`

f. `y = --(x * y);`

g. `System.out.println("x =" x);`

15. Give Java statements to describe the formulas that follow:

a. Area of a circle with radius $R = \pi R^2$

b. Surface area of a sphere with radius $R = 4\pi R^2$

c. Volume of a sphere with radius $R = 4\pi R^3/3$

16. Write a program that prompts the user to enter a value for radius. Read this value from the console using an instance of class Scanner. Next, compute the surface area and volume of a sphere using the value of radius entered by the user and print them out on the consol

17. a. Write a program to calculate the value of gravity g of a planet using the following equation:

$$g = (G \times M)/R^2$$

where M is the mass of the planet, R is the radius of the planet, a the gravitational constant $G = 6.67 \times 10^{-11}\text{m}^3\text{kg}^{-1}s^{-2}$. The pr gram should prompt the user to enter the mass of the planet a its radius, and print out its gravity. Use this program to determi the value of gravity of the planet Mercury ($R = 2.43 \times 10^6$m, M 0.33×10^{24}kg) and Jupiter ($R = 71.49 \times 10^6$m, $M = 1.89$ 10^{27}kg).

b. Modify the program developed in part (a) to print out the weig of a person on that planet as well. The program reads the weight

a person on Earth, and calculates his or her weight on a different planet using the formula:

$$w_{planet} = w \times \frac{g}{g_e}$$

where w is the person's weight on Earth, g is the gravity on the planet calculated in part (a), and $g_e = 9.81\text{ms}^{-1}$ is the value of gravity on earth.

Graphics Problems

18. The following code segment draws a figure that has a circle inside a rectangle. Rewrite this code so that the position of the figure is represented by the reference point (x, y). The figure can be moved to a new position in the window by changing the values of x and y. Use variables, not literals, as arguments to the constructors.

```
DrawingKit dk = new DrawingKit();
Ellipse2D.Float e1 = new Ellipse2D.Float(100, 150, 100, 50);
dk.draw(e1);
Rectangle2D.Float r1 = new Rectangle2D.Float(100, 150, 100, 50);
dk.draw(r1);
```

19. Rewrite the following code segment so that the position of the figure is represented by the reference point (x, y). By changing the values of x and y, the figure is moved to a new position in the window. Use variables, not literals, as arguments to the constructors.

```
int x = 200; int y = 250;
Ellipse2D.Float e1 = new Ellipse2D.Float(100, 150, 100, 50);
dk.draw(e1);
Ellipse2D.Float e2 = new Ellipse2D.Float(100, 165, 50, 20);
dk.draw(e2);
Rectangle2D.Float r1 = new Rectangle2D.Float(100, 150, 100, 50);
dk.draw(r1);
```

20. Modify the code in the `StickFigure` class provided earlier in this chapter to create a different stick figure using your imagination. When the reference point (x, y) is changed, the figure should move to another position in the window. Use variables instead of literals in the constructors and methods in your program.

```
        int green = pixelColor.getGreen();
        int blue = pixelColor.getBlue();

        // multiply the red, green and blue samples by 1.75
        red = red <= 145? (int)(red * 1.75f): 255;
        green = green <= 145? (int)(green * 1.75f) : 255;
        blue = blue <= 145? (int)(blue * 1.75f) : 255;

        // update the pixel color in picture
        Color newPixelColor = new Color(red, green, blue);
        int newRgbvalue = newPixelColor.getRGB();
        picture.setRGB(x, y, newRgbvalue);
      } // end inner for loop
    } // end outer for loop

    // draw the modified picture
    dk.drawPicture(picture, 860, 50);
  }
}
```

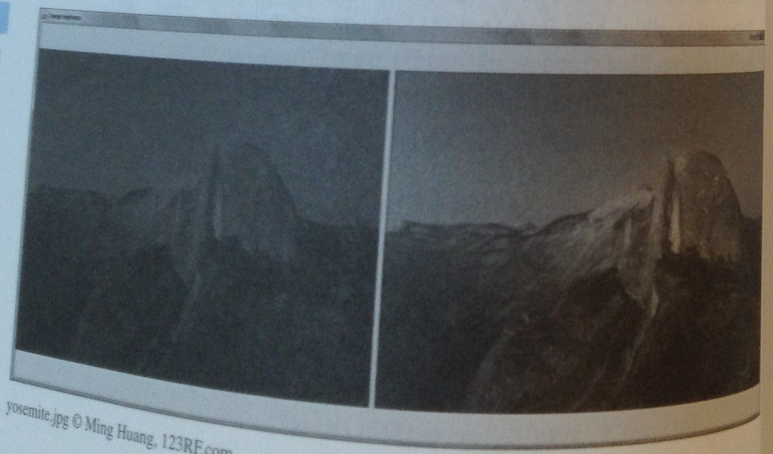
Figure 4–18 shows the result. The following statement scales the value of the red sample by a factor of 1.75 only if it is less than or equal to 145; otherwise, it sets the value to 255.

```
red = red <= 145? (int)(red * 1.75f): 255;
```

This process is repeated for the other two components. The previous statement has the same meaning as the following:

```
if (red <= 145)
  red = (int)(red * 1.75f);
else
  red = 255;
```

Figure 4–18
The image on the right is made brighter than the original on the left.



yosemite.jpg © Ming Huang, 123RF.com

If you multiply the pixels by a value in the range 0 to 1, it will darken the image. Experiment with different factors and note how the brightness varies. You can also use a different factor for each component to produce other interesting visual effects. ■

4.11 Summary

In this chapter, we discussed how you can describe conditions using `boolean` expressions and use them in conditional statements and loops. Conditional statements are used to select one of many operations depending on whether a `boolean` expression evaluates to `true` or `false`. Loops are useful when an operation must be performed multiple times. Branching statements allow you to perform certain operations selectively inside a loop or terminate the loop. We also introduced image manipulation techniques, which show you how to change the colors and brightness of a picture, and how to combine pictures together. In the next chapter, you will learn other interesting techniques to manipulate images. So keep reading!

Class Summary

`Random()`—a constructor in class `Random` in the `java.util` package that is used to create a random number generator.

`int nextInt(int n)`—a method in class `Random` that returns a pseudorandom integer number in the range 0 to `n`−1.

`int getRGB()`—a method in class `Color` that returns the encoded color value of this `Color` object.

Exercises

1. Draw a flowchart for each of the following problems. Also, find the values of `x`, `y`, and `z` after all the statements in each problem are executed, assuming the following initial values for these variables:

   ```
   int x = 100, y = 5, z = 100;
   ```

   a.
   ```
   if (x > 50) {
       y *= 5;
       z--;
   }
   x += 5;
   ```

b.
```
if (x <= 30)
      y -= 10;
else {
      y += 2;
      z *= 5;
}
y++;
```

c.
```
switch(x) {
      case 100: y +=  20;
                break;
      case 200: z *=  10;
                break;
      case 300: z +=  y;
                break;
     default: x = 0;
                break;
}
```

d.
```
while (x <= 100) {
      y -= 5;
      x += 10;
}
z += 2;
```

e.
```
for (int i = 0; i < 4; i++) {
      x += i;
      y -= i;
}
```

2. Find the syntax errors in the following code segments. Assume that the variables have been declared as integers.

a.
```
If (x = y)
   System.out.println(x)
```

b.
```
if (x == 10)
   x = x + 3.5f;
else (x < 10)
   x--;
```

c.
```
for (int l = 0, l <= 5, l++)
  sum += l;
}
```

d.
```
do {
  sum = sum + 10;
} while (x < 5)
```


3. Find the error in the following code segment:

```
int x = 50;
while (x < 100) {
   y -= 1;
   x -= 10;
}
y += 2;
```

4. Determine if there is an error in the following nested `for` loop:

```
for (int j = 0; j < 5; j++)
  for (j = 0; j < 10; j++)
    // do something here
```

5. Determine the value of the `boolean` variable `bool` in each of the following problems. Assume that the variables `a`, `b`, `c`, and `d` are declared as follows:

```
int a = 1, b = 2, c = 3, d = 4;
```

a. `bool = a > b  &&  c < d;`

b. `bool = a > d++ || c <= d;`

c. `bool = !(a--== d);`

d. `bool = (a > b && c + d * 0.5 == 5);`

6. Parenthesize the following boolean/arithmetic expressions and determine their values:

a. `4 > 3 && -7 < 6`

b. `4 + 3 * 5 - 6 / 2`

c. `11 < 13 && 17 > 23 || 5 != 6`

d. `23 % 3 * 5 + 8`

e. `!(5 < 23)`

7. Write a program to print numbers from 1 to 5000.

8. Write a program to print all numbers from 1 to 100 except those that end in 7 or are a multiple of 7. For instance, the program should not print the numbers 7, 14, 17, 21, 27, and so on.

Graphics Problems

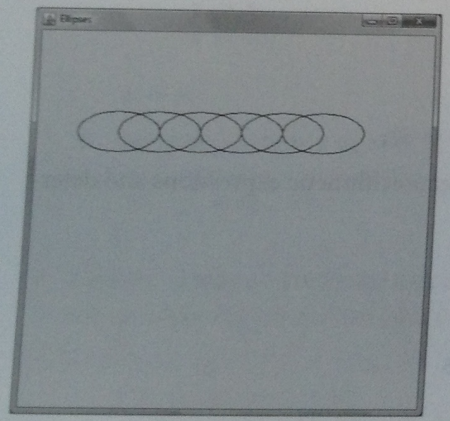
9. Write a code segment using `else if` statements to produce graphical output that depends on the value of a variable `x`. Draw a rectangle,

circle, or ellipse if x is 0, 1, or 2, respectively. For all other values of x, draw a line. You can select any position and size for each shape.

10. The following code segment should draw five squares, each with a side of length 25, at different positions in a window. Complete the code marked with "?".

```
int x = 0, y = 100;
DrawingKit dk = new DrawingKit();
while(?) {
  x = x + 1;
  Rectangle2D.Float r = new Rectangle2D.Float(x * 50, y, 25, 25);
  dk.draw(r);
}
```

11. Using a `while` loop, write a program that creates six ellipses each of width 100 and height 50, as shown in Figure 4–19. The x and y coordinates of the first ellipse are at 50 and 100, respectively.



ure 4–19
t of ellipses.

12. Repeat the previous problem using a `do-while` loop.

13. Complete the following `for` loop to create nine squares of side 20 arranged in a single column:

```
for (int x = 1; ?; ?) {
  Rectangle2D.Float r = new Rectangle2D.Float(10, x*30, 20, 20);
  dk.draw(r);
}
```


14. Complete the following statements to create five squares of side 10 arranged in a single row:

```
for (int y = 0; ?; ?) {
  Rectangle2D.Float r = new Rectangle2D.Float(?, 50, 10, 10);
  dk.draw(r);
}
```

15. Write a program to create a row of 20 shapes consisting of alternating squares and circles using a `for` loop. Every square has a side of length 20 and each circle has diameter 20. The distance from the center of a square to the center of its neighboring circle is 20. A row should start with a square.

16. Modify the preceding program so that every third shape is red, and the remaining shapes are blue.

17. Modify the program `ConcentricCircles` (see Example 6) to draw a set of circles such that they have the same center and the radii of consecutive circles differ by 10. Assume that the outermost circle has radius *R*. The program draws all of the circles until the diameter of the innermost circle becomes less than 20. The program should then print out on the console the number of circles that have been drawn and the diameter of the innermost circle. Test your program for various values of *R*. Use a `while` loop.

18. Repeat the preceding problem using a `for` loop.

19. Write a program using nested `for` loops to create a grid of 81 squares, each of side 20, as shown in Figure 4–20. The *x*- and *y*-coordinates of

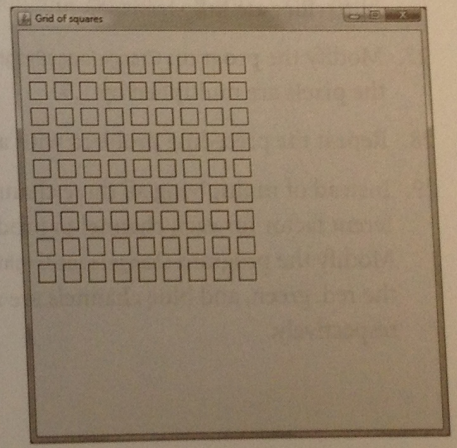
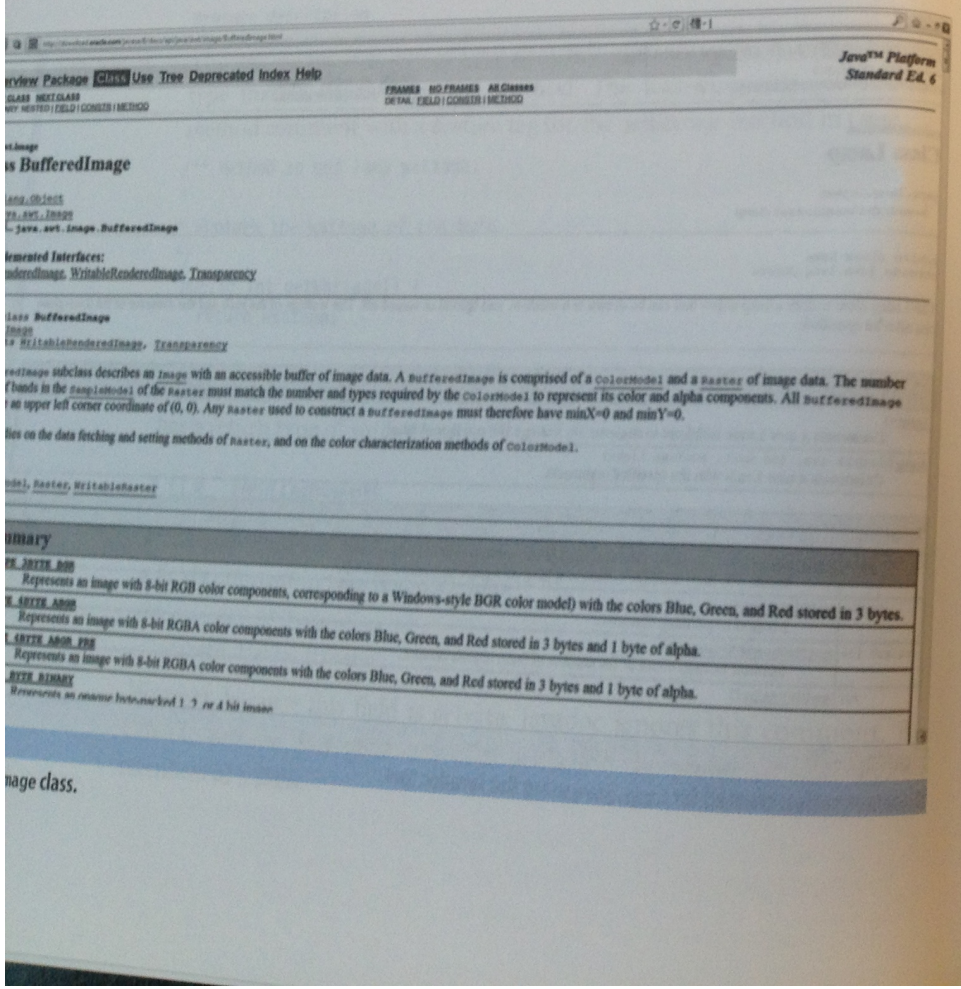


Figure 4
A grid of

5.12 The Java API

The documentation for the Java language is referred to as the Java API. It is available online and is in a format similar to the format shown in Figure 5–20. The Java API is a part of the Java Standard Edition documentation, which also provides tutorials, user guides, and other information about the language. You can either download this documentation to your computer or browse it online. (The entire API is very extensive; as a result, we have limited ourselves to describing only a small subset of it in this book. Anytime we introduce a new class, you should refer to the API.)

Figure 5–21 shows some of the fields in the `BufferedImage` class in the `java.awt.image` package. The listing for this class includes a summary as well as a detailed explanation of its fields, constructors, and methods.



nage class.

5.13 Summary

This chapter detailed how to create classes, and how to write constructors and methods. Other topics covered included the use of the keyword `this`; organizing .java files in packages, access modifiers, static variables and methods; and a brief explanation of some of the graphics classes that we have been using. Read further; in Chapter 6 you will learn how to create animations and, in Chapter 7, we will develop a board game called Crystals.

`BufferedImage` Class Summary

`BufferedImage(int width, int height, int type)`—a constructor that creates a `BufferedImage` having the specified width, height, and type. The type is a constant such as `TYPE_INT_ARGB`, which represents an image in the RGB color space with an alpha channel.

`int getHeight()`—a method that returns the number of pixels in a column of the image.

`int getWidth()`—a method that returns the number of pixels in a row of the image.

`int getRGB(int x, int y)`—a method that returns the encoded color value of a pixel at location `(x, y)`.

`void setRGB(int x, int y, int rgbvalue)`—a method that sets the color of a pixel at location `(x, y)` to an encoded color value `rgbvalue`.

`Graphics2D` Class Summary

`void draw(Shape s)`—a method that is used to draw an object `s`.

`void setPaint(Paint p)`—a method that is used to set the color to `p`.

`void fill(Shape s)`—a method that is used to color the object `s`.

`void setStroke(Stroke st)`—a method that is used to set the line thickness to `st`.

`void setFont(Font f)`—a method that is used to set the font to `f`.

`void drawString(String str, int x, int y)`—a method that is used to write the text `str` at the coordinates `(x, y)`.

Exercises

1. Explain the following terms:
   a. Method signature
   b. Local variable

c. Access modifiers
d. `return` keyword
e. Overloaded constructors
f. Compositing

2. Explain the differences between the following items:
   a. Instance field and static field
   b. Instance method and class method
   c. `public` and `private` access modifiers
   d. Default constructor and constructor without parameters
   e. Accessor methods and mutator methods
3. State whether the following statements are true or false.
   a. Constructors have a return type.
   b. The default constructor can be invoked by an instance of a class that contains a constructor with parameters.
   c. A method can return only one value.
   d. A class can be declared using any of the four access modifiers.
   e. The signature of a method contains its name, return type, and the data types of its parameters.
   f. By convention, the name of a class should begin with an uppercase letter.
   g. By convention, a package name can contain uppercase letters.
   h. A `private` constructor can be called in another class.
4. Write the states and behaviors of any object in your room. Write a class using these specifications.
5. Correct the compilation errors in this class:

```
Class HasErrors {
  int a;

  void getA(){
    return a;
  }

  int getA(float val){
    a = val;
  }
}
```


6. Correct the access modifiers used in the following program so that it compiles correctly. Consider each of these cases separately:
   a. The classes `CheckAccess` and `Test` are in the same package.
   b. The classes `CheckAccess` and `Test` are in different packages.

```
private class CheckAccess {
  private int code;

  private int getCode() {
        return code;
  }
}

public class Test {
  public static void main(String[] args) {
        CheckAccess c = new CheckAccess();
        System.out.println(c.getCode());
  }
}
```

7. Create a class called `Country` with these fields: `name`, `population`, and `capital`. Write the accessor and mutator methods for all of its fields. Make the fields `private` and the methods `public`.
8. Add comments to the class in Problem 7. Create documentation for it using Javadoc.
9. Give the output of the following program without running it first, and then run the program to verify your answer.

```
public class PredictOutput {
  int x = -5;
  static int y = -5;

  public PredictOutput() {
    x++;
    y++;
  }

  public static void main(String[] args) {
    PredictOutput p = new PredictOutput();
    for (int i = 0; i < 10; i++) {
       p = new PredictOutput();
    }
```