Chapter 13 Graphics

1. The y coordinate should increase and the x coordinate should remain unchanged.

2. jlblBanner.getGraphics() returns null, because jlblBanner has not been displayed. The message dialog in Listing 12.1 is to purposely delay the statement to draw the line. So the line will be drawn after the label is displayed. You have to use frame.jlblBanner.getGraphics() in the main method. This is because jlblBanner is an instance variable and it cannot be referenced directly in a static method.

3. The paintComponent() method is defined in the Component class. The Java runtime system invokes it to paint things on a Swing GUI component. This method cannot be invoked by the system or by the programmer. The system automatically invokes it whenever the viewing area changes. The programmer invokes it through invoking the repaint() method. The programmer should never directly invoke the paintComponent() method.

4. The paintComponent() method is protected, because (1) this method is always invoked by the JVM, not by a client program; (2) the client program need to override it in a subclass.

   If it is changed to public, it is OK, but not necessary, because the protected modifier is sufficient.

   It cannot be changed to private, because the visibility cannot be weakened.

   super.paintComponent(g) invokes the superclass's paintComponent method. In Line 12 in Listing 12.2, it causes the text of the label to be painted first. Before this text is displayed, the JLabel's paintComponent(g) method actually invokes super.paintComponent(g) to clear the viewing area. In Line 20 in Listing 12.3, it causes the viewing area to be cleared.

5. Yes. You should use JPanel as a canvas, because it is a plain panel unlike a label or a button.

6. See the Sections 12.6, 12.8, and 12.9.

7. See the Sections 12.6, 12.8, and 12.9.

8. You can use the setColor(Color) to set a color in the graphics context and use the setFont(Font) method to set a font in the graphics context.

9. Draw a thick line from (10, 10) to (70, 30). You must draw several lines next to each other to create the effect of one thick line.
   Answer:

```
for (int i = 0; i < 10; i++)
   g.drawLine(10, 10 + i, 70, 30 + i);
```

Draw/fill a rectangle of width 100 and height 50 with the upper-left corner at (10, 10).
Answer:
```
g.drawRect(10, 10, 100, 50);
g.fillRect(10, 10, 100, 50);
```

Draw/fill a rounded rectangle with width 100, height 200, corner horizontal diameter 40, and corner vertical diameter 20.
Answer:
```
g.drawRoundRect(10, 10, 100, 200, 40, 20);
g.fillRoundRect(10, 10, 100, 200, 40, 20);
```

Draw/fill a circle with radius 30.

Answer:
```
g.drawOval(10, 10, 60, 60);
g.fillOval(10, 10, 60, 60);
```

Draw an oval with width 50 and height 100.

Answer:
```
g.drawOval(10, 10 50, 100);
```

Draw the upper half of a circle with radius 50.

Answer:
```
g.drawArc(10, 10, 100, 100, 0, 180);
```

Draw a polygon connecting the following points: (20, 40), (30, 50), (40, 90), (90, 10), (10, 30).
Answer:
```
int x[] = {20, 30, 40, 90, 10};
int y[] = {40, 50, 90, 10, 30};
g.drawPolygon(x, y, x.length);
g.fillPolygon(x, y, x.length);
```

Draw a 3D cube like the one in Figure 8.28.
Answer:
```
public void paintComponent(Graphics g) {
  //draw the front rect
  g.drawRect(10, 60, 40, 40);

  //draw the back rect
  g.drawRect(30, 40, 40, 40);

  //connecting the corners
  g.drawLine(10, 60, 30, 40);
  g.drawLine(10, 100, 30, 80);
  g.drawLine(50, 60, 70, 40);
  g.drawLine(50, 100, 70, 80);
}
```

10. First obtain the FontMetrics for a font used in the Graphics context using g.getFontMetrics() or g.getFontMetrics(Font). You can then use the FontMetrics's getAscent(), getDescent(), getLeading(), getHeight() methods to obtain the font's ascent, descent, leading, and height. Use getWidth(String) to obtain the string width for the font.

11. When you create a MessagePanel, its paintComponent method is invoked. Since message is null, invoking g.drawString(message, xCoordinate, yCoordinate) causes a NullPointerException.

12. Two errors: (1) constructor TestDrawMessage cannot have void. (2) PaintComponent should be paintComponent.

13. `Use imageIcon.getImage().`

14. Use new ImageIcon(image).

15. The `drawImage(...)` method displays the image on the viewing area.

16. An image displayed on a label is non-stretchable, but an image displayed on a panel is stretchable.

17. The images cannot be stretched in JLabel. The images can be stretched in JPanel.