Topic 9
More Graphics

Based on slides by Marty Stepp and Stuart Reges
from http://www.buildingjavaprograms.com/
Clicker Question

What happens if a graphics object is used to draw a shape that exceeds the boundaries of the DrawingPanel?

DrawingPanel p3 = new DrawingPanel(100, 100);
Graphics g2 = p3.getGraphics();
g2.fillRect(50, 50, 200, 200);

A. Only the visible portion shown
B. DrawingPanel expands to show whole rectangle
C. Syntax error
D. Runtime error
E. None of A - D are correct
Animation exercise

Modify the following program to draw a "moving" car.

```java
import java.awt.*;

public class Car {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(200, 100);
        panel.setBackground(Color.LIGHT_GRAY);
        Graphics g = panel.getGraphics();
        g.setColor(Color.BLACK);
        g.fillRect(10, 30, 100, 50);
        g.setColor(Color.RED);
        g.fillOval(20, 70, 20, 20);
        g.fillOval(80, 70, 20, 20);
        g.setColor(Color.CYAN);
        g.fillRect(80, 40, 30, 20);
    }
}
```
Parameterized Drawing

- `drawTruck0` -> hard coded location and size
- `drawTruck1` -> parameterized location, hard coded size
- `drawTruck2` -> parameterized location and size
- animate the truck using the sleep method from drawing panel
Parameterized figures

- Modify the car-drawing method so that it can draw cars at different positions, as in the following image.
  - Top-left corners: (10, 30), (150, 10)
  - Increase the drawing panel's size to 260x100 to fit.
Drawing with parameters

To draw in a method, you must pass Graphics g to it.
– Otherwise, g is out of scope and cannot be used.

syntax (declaration):

```java
public static void <name> (Graphics g, <parameters>) {
    <statement(s>)
}
```

syntax (call):

```java
<name> (g, <values>);
```
import java.awt.*;

public class Car3 {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(260, 100);
        panel.setBackground(Color.LIGHT_GRAY);
        Graphics g = panel.getGraphics();
        drawCar(g, 10, 30);
        drawCar(g, 150, 10);
    }

    public static void drawCar(Graphics g, int x, int y) {
        g.setColor(Color.BLACK);
        g.fillRect(x, y, 100, 50);
        g.setColor(Color.RED);
        g.fillOval(x + 10, y + 40, 20, 20);
        g.fillOval(x + 70, y + 40, 20, 20);
        g.setColor(Color.CYAN);
        g.fillRect(x + 70, y + 10, 30, 20);
    }
}
Write a program that draws the following figure:
- drawing panel is size 200x150
- book is at (20, 35), size 100x100
- cyan background
- white "BJP" text at position (70, 55)
- stairs are (red=191, green=118, blue=73)
- each stair is 9px tall
  - 1st stair is 10px wide
  - 2nd stair is 20px wide ...
- stairs are 10px apart (1 blank pixel between)
public class Book {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(200, 150);
        panel.setBackground(Color.WHITE);
        Graphics g = panel.getGraphics();

        g.setColor(Color.CYAN); // cyan background
        g.fillRect(20, 35, 100, 100);

        g.setColor(Color.WHITE); // white "bjp" text
        g.drawString("BJP", 70, 55);

        g.setColor(new Color(191, 118, 73)); // orange "bricks"
        for (int i = 0; i < 10; i++) {
            g.fillRect(20, 35 + 10 * i, 10 + 10 * i, 9);
        }
    }
}
Multiple Java books

- Modify the Java book program so that it can draw books at different positions as shown below.
  - book top/left positions: (20, 35), (150, 70), (300, 10)
  - drawing panel's new size: 450x180
Multiple books solution

// Draws many BJP textbooks using parameters.
import java.awt.*;

public class Book2 {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(450, 180);
        panel.setBackground(Color.WHITE);
        Graphics g = panel.getGraphics();

        // draw three books at different locations
        drawBook(g, 20, 35);
        drawBook(g, 150, 70);
        drawBook(g, 300, 10);
    }
}

...
// Draws a BJP textbook at the given x/y position.
public static void drawBook(Graphics g, int x, int y) {
    g.setColor(Color.CYAN);    // cyan background
    g.fillRect(x, y, 100, 100);

    g.setColor(Color.WHITE);    // white "bjp" text
    g.drawString("BJP", x + 50, y + 20);

    g.setColor(new Color(191, 118, 73));
    for (int i = 0; i < 10; i++) {    // orange "bricks"
        g.fillRect(x, y + 10 * i, 10 * (i + 1), 9);
    }
}
Resizable Java books

- Modify the Java book program so that it can draw books at different sizes as shown below.
  - book sizes: 100x100, 60x60, 200x200
  - drawing panel's new size: 520x240
import java.awt.*;

public class Book3 {
    public static void main(String[] args) {
        DrawingPanel panel = new DrawingPanel(520, 240);
        panel.setBackground(Color.WHITE);
        Graphics g = panel.getGraphics();

        // draw three books at different locations/sizes
        drawBook(g,  20, 35, 100);
        drawBook(g, 150, 70,  60);
        drawBook(g, 300, 10, 200);
    }

    ...
}

// Draws many sized BJP textbooks using parameters.
Resizable solution, cont'd.

...
Polygon

*Objects that represent arbitrary shapes*

- Add points to a *Polygon* using its `addPoint(x, y)` method.

**Example:**

```java
DrawingPanel p = new DrawingPanel(100, 100);
Graphics g = p.getGraphics();
g.setColor(Color.GREEN);

Polygon poly = new Polygon();
poly.addPoint(10, 90);
poly.addPoint(50, 10);
poly.addPoint(90, 90);
g.fillPolygon(poly);
```
DrawingPanel methods

- `panel.save(filename);`
  Saves the image on the panel to the given file (String).

- `panel.sleep(ms);`
  Pauses the drawing for the given number of milliseconds.
Animation with \texttt{sleep}

- \texttt{DrawingPanel}'s \texttt{sleep} method pauses your program for a given number of milliseconds.
- You can use \texttt{sleep} to create simple animations.

\begin{verbatim}
DrawingPanel panel = new DrawingPanel(250, 200);
Graphics g = panel.getGraphics();

g.setColor(Color.BLUE);
for (int i = 1; i <= 10; i++) {
    g.fillOval(15 * i, 15 * i, 30, 30);
    panel.sleep(500);
}
\end{verbatim}

- Try adding \texttt{sleep} commands to loops in past exercises in this chapter and watch the panel draw itself piece by piece.
Animation exercise

- Modify the previous program to draw a "moving" animated car.