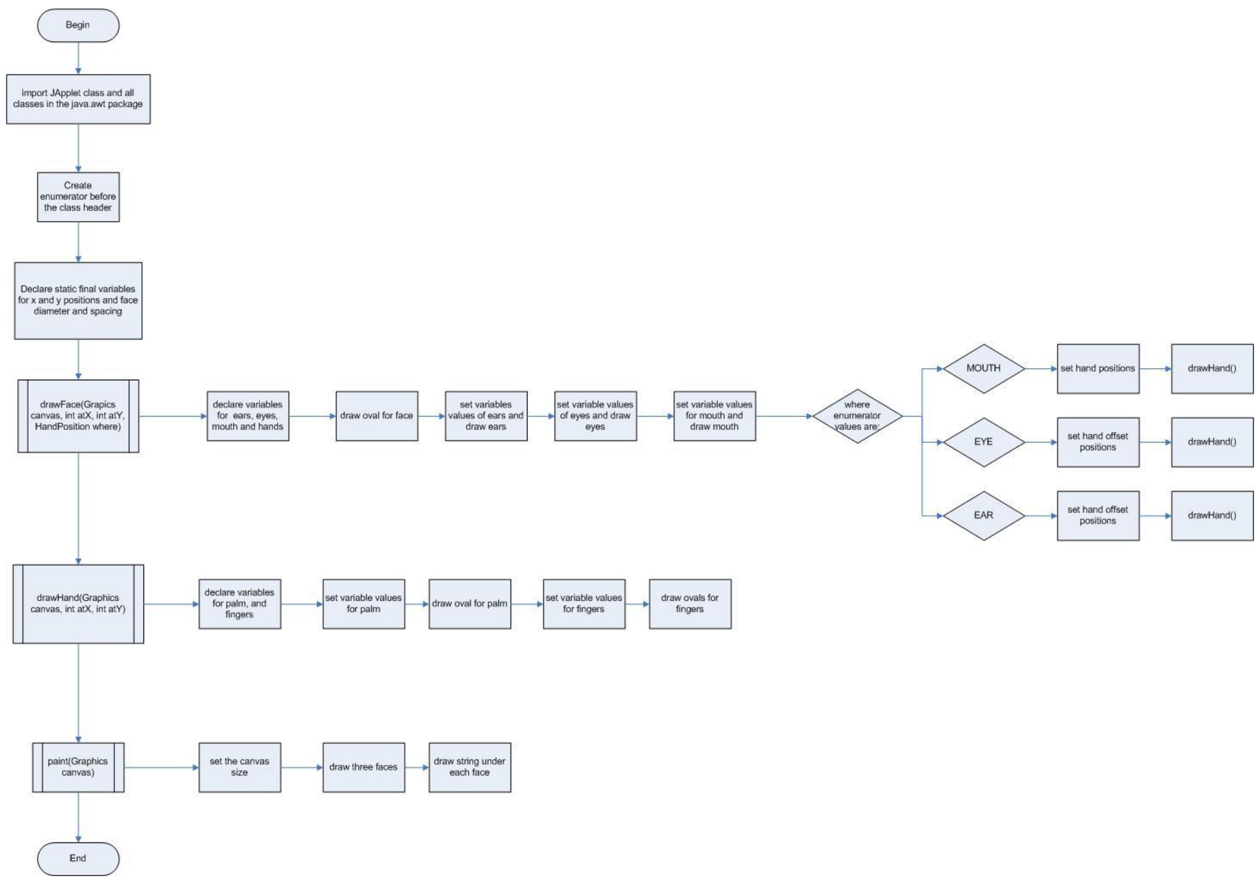
CIT 149: Java I
Chapter 5 Lab 3

In this program we will create programming project # 14 on page 367. This applet will use an enumerated data type, enumeration, to draw the Ear, Eye and Mouth. Enumerations were discussed in chapter 3. This is the first time we will actually use one. An Enumerated data type gives a list of values variables can have. For a review on enumerations reread pages 179-180. The values are often used in switch statements. Enumerators are often declared before the class header and when the class is compiled a separate class file for the enumerator will be created. The enumerator could also have been placed in a separate file with a .java extension. If the enumerator is created in a separate file, the enumerator need not be compiled. The main program that references the enumerator will automatically create a class file for the enumerator as well. The following flowchart is a quick rundown on what the program needs to do.


Begin
import JApplet class and all classes in the java.awt package
Create enumerator before the class header
Declare static final variables for x and y positions and face diameter and spacing
drawFace(Grapics canvas, int atX, int atY, HandPosition where)
declare variables for ears, eyes, mouth and hands
draw oval for face
set variables values of ears and draw ears
set variable values of eyes and draw eyes
set variable values for mouth and draw mouth
where enumerator values are;
MOUTH
set hand positions
drawHand()
EYE
set hand offset positions
drawHand()
EAR
set hand offset positions
drawHand()
drawHand(Graphics canvas, int atX, int atY)
declare variables for palm, and fingers
set variable values for palm
draw oval for palm
set variable values for fingers
draw ovals for fingers
paint(Graphics canvas)
set the canvas size
draw three faces
draw string under each face
End

1. Open a new document in Textpad and save the program as MonkeyFaces.java
2. Type the code that will import the JApplet class, and all classes in the awt package. Review the chapter if you are unsure on this.
3. Type the following code that will create an enumerator named HandPosition:

   ```
   enum HandPosition{EAR, EYE, MOUTH};
   ```

   These values will represent the possible values in a switch statement. This code could easily have been typed in a separate document with the name of HandPosition.java. Since this is the only program to use this enumerator we can simply add it to the beginning of the program, outside the class itself.
4. Type the class header and opening brace as follows:

   ```
   public class MonkeyFaces extends JApplet
   {
   ```

   Including extends allows the program to inherit all attributes from the JApplet class.
5. Within the class we will first create several variables with default values that cannot be changed. Including the words static final ensures that the variable values are static. Type:

   ```
   static final int XBASE = 100;
   static final int YBASE = 100;
   static final int FACE_DIAMETER = 75;
   static final int FACE_SPACING = 150;
   ```

   Often the access modifier of "public" is included at the beginning of each line for static final variables. If omitted the program assumes they are public.

6. Within a separate method of drawFace() we will create the ears, eyes, and mouth. Start by typing the method header and opening brace:

   ```
   public static void drawFace(Graphics canvas, int atX, int atY, HandPosition where)
   {
   ```

   When invoked, this method will receive the value of a Graphics object, two integers and the HandPosition enumerator.

7. Next we will declare several variables that will be required. Type:

   ```
   int earWidth, earHeight,earOffset, eyeWidth, eyeHeight, eyeOffsetX, eyeOffsetY,
   mouthWidth, mouthHeight, mouthOffsetX, mouthOffsetY, handOffsetX, handOffsetY;
   ```

   Seeing the name for each variable indicates what the variable values will be used for.

8. Let's draw an oval to represent the monkey's face by typing:

   ```
   canvas.drawOval(atX-FACE_DIAMETER/2, atY-FACE_DIAMETER/2,
   FACE_DIAMETER, FACE_DIAMETER);
   ```

   The drawOval() method creates a hollow oval. We have to set the x, y, and size of the oval. Values for these are received through the method's argument and the values declared at the beginning of the program. Notice that the x and y positions subtract the FACE_DIAMETER/2 from the integers passed to the method?

9. Next we'll draw the ears by typing:

   ```
   //draw the ears
   earWidth = (int)  (FACE_DIAMETER * 0.3);
   earHeight = (int) (FACE_DIAMETER * 0.5);
   earOffset = (int) (FACE_DIAMETER * 0.06);
   canvas.drawOval(atX-FACE_DIAMETER/2-earWidth+earOffset, atY-
      FACE_DIAMETER/2, earWidth, earHeight);
   canvas.drawOval(atX+FACE_DIAMETER/2-earOffset, atY-FACE_DIAMETER/2,
      earWidth, earHeight);
   ```

   variables for the ears receive values based on the FACE_DIAMETER. Since the result of the multiplication could possibly result in decimals places the result is cast to an integer. Then oval are drawn for both ears.

10. The eyes are similar to the ears. Type:

```
//draw the eyes
eyeWidth = (int)  (FACE_DIAMETER * 0.15);
eyeHeight = (int) (FACE_DIAMETER * 0.25);
eyeOffsetX = (int) (FACE_DIAMETER * 0.40);
eyeOffsetY = (int) (FACE_DIAMETER * 0.20);
canvas.fillOval(atX-FACE_DIAMETER/2-eyeWidth+eyeOffsetX, atY-
FACE_DIAMETER/2+eyeOffsetY, eyeWidth, eyeHeight);
canvas.fillOval(atX+FACE_DIAMETER/2-eyeOffsetX, atY-
FACE_DIAMETER/2+eyeOffsetY, eyeWidth, eyeHeight);
```

    The main difference between the code for the ears and eyes is that the fillOval() is used instead of drawOval() since the eyes will be solid.

11. Next for the mouth type:

```
//draw mouth
mouthWidth = (int)  (FACE_DIAMETER * 0.70);
mouthHeight = (int) (FACE_DIAMETER * 0.60);
mouthOffsetX = (int) (FACE_DIAMETER * 0.05);
mouthOffsetY = (int) (FACE_DIAMETER * 0.25);
canvas.drawArc(atX-mouthWidth/2+mouthOffsetX, atY-
FACE_DIAMETER/2+mouthOffsetY, mouthWidth, mouthHeight, 0, -180);
```

    The only difference here is the drawArc() method is used to draw the smiling face. A -180 is used to cause the arc to point upwards by 180 degrees. If a 180 was used you would have a frown instead.

12. A switch statement is used in placing the hands over the mouth, eyes, or ears. This is a typical speak no evil, see no evil and hear no evil. Type:

```
switch(where)
{

case MOUTH:
    handOffsetX = (int) (mouthWidth*0.3);
    handOffsetY = (int) (-FACE_DIAMETER/2+mouthOffsetY+mouthHeight*0.9);
    drawHand(canvas, atX-handOffsetX, atY+handOffsetY);
    drawHand(canvas, atX+handOffsetX, atY+handOffsetY);
break;

case EYE:
    handOffsetX = (int) (eyeOffsetX-eyeWidth);
    handOffsetY = (int) (-FACE_DIAMETER/2+eyeOffsetY+eyeHeight*0.6);
    drawHand(canvas, atX-handOffsetX, atY+handOffsetY);
    drawHand(canvas, atX+handOffsetX, atY+handOffsetY);
break;

case EAR:
    handOffsetX = (int) (-FACE_DIAMETER/2-earWidth/2);
    handOffsetY = (int) (-FACE_DIAMETER/2+earHeight*0.75);
    drawHand(canvas, atX-handOffsetX, atY+handOffsetY);
    drawHand(canvas, atX+handOffsetX, atY+handOffsetY);
break;

}
```

With each case the offset of the hand is determined and the values are passed to the drawHand() method.

13. Close the drawFace() method.

14. Our next method is used to draw the hands. It receives its argument from the drawFace() method, previously created. Type:

```
public static void drawHand(Graphics canvas, int atX, int atY)
{
    int palmWidth, palmHeight, fingerWidth, fingerHeight,fingerOffsetX, fingerOffsetY;

     palmWidth = (int)  (FACE_DIAMETER * 0.4);
     palmHeight = (int) (FACE_DIAMETER * 0.2);

    canvas.fillOval(atX-palmWidth/2, atY-palmHeight/2, palmWidth, palmHeight);

    //draw the fingers
    fingerWidth = (int)  (palmWidth * 0.2);
    fingerHeight = (int) (palmHeight * 1.2);
    fingerOffsetX = (int) (fingerWidth * 1.8);
    fingerOffsetY = (int) (fingerHeight * 0.8);

    canvas.fillOval(atX-fingerWidth/2-fingerOffsetX, atY-palmHeight/2-fingerOffsetY,
        fingerWidth, fingerHeight);
    canvas.fillOval(atX-fingerWidth/2, atY-palmHeight/2-fingerOffsetY, fingerWidth,
        fingerHeight);
canvas.fillOval(atX-fingerWidth/2+fingerOffsetX, atY-palmHeight/2-fingerOffsetY,
        fingerWidth, fingerHeight);
}
```

Here the palms and fingers are drawn. The actual painting on the screen will be handled by the paint() method.

The palm's width and height are set then drawn as solid ovals; the fingers size and offset from the palm are determined; each finger is drawn as solid ovals.

15. The last method is the paint() method which is used to paint onto the screen. Type:

```
public void paint(Graphics canvas)
{

    setSize(500, 300);
   drawFace(canvas, XBASE, YBASE, HandPosition.EAR);

    canvas.drawString("Hear no evil", XBASE-FACE_DIAMETER/2, YBASE +
       FACE_DIAMETER);

    drawFace(canvas, XBASE+FACE_SPACING, YBASE, HandPosition.EYE);

    canvas.drawString("See no evil", XBASE+FACE_SPACING-FACE_DIAMETER/2,
        YBASE + FACE_DIAMETER);

    drawFace(canvas, XBASE+2*FACE_SPACING, YBASE, HandPosition.MOUTH);

    canvas.drawString("Speak no evil", XBASE+2*FACE_SPACING-
        FACE_DIAMETER/2, YBASE + FACE_DIAMETER);
}
```

- the setSize() method sets the canvas size.
- the drawFace() method is invoked to draw a face on the screen.
- the drawString() method is used to draw text onto the screen.
- the above is repeated twice more for two additional monkey faces.

16. Close the class.
17. Compile the program and fix any errors if necessary.
18. Create an html document with the width and height set to the same dimensions as the argument in the referral to the setSize() method above.
19. Run your program to make sure your monkeys look like the figure on page 336.
20. Compress the following files into a single zip or rar file. You will have more files to compress than usual.

MonkeyFaces.java
MonkeyFaces.class
MonkeyFaces$1.class
HandPosition.class
HTML document

21.