```
// joi/3/shapes/Screen.java
//
//
// Copyright 2003 Bill Campbell and Ethan Bolker

/**
 * A Screen is a (width*height) grid of (character) 'pixels'
 * on which we may paint various shapes.  It can be drawn to
 * a Terminal.
 *
 * @version 3
 */

public class Screen
{
    /**
     * The character used to paint the screen's frame.
     */

    private static final char FRAMECHAR = '+';
    private static final char BLANK = ' ';
    private int width;
    private int height;
    private char[][] pixels;

    /**
     * Construct a Screen.
     *
     * @param width the number of pixels in the x direction.
     * @param height the number of pixels in the y direction.
     */

    public Screen( int width, int height )
    {
        this.width  = width;
        this.height = height;
        pixels = new char[width][height];
        clear();
    }

    /**
     * Clear the Screen, painting a blank at every pixel.
     */

    public void clear()
    {
        for (int x  = 0; x < width; x++) {
            for ( int y = 0; y < height; y++ ) {
                pixels[x][y] = BLANK;
            }
        }
    }

    /**
     * Paint a character pixel at position (x,y).
     *
```



```
     * @param c the character to be painted.
     * @param x the (horizontal) x position.
     * @param y the (vertical) y position.
     */

    public void paintAt( char c, int x, int y )
    {
        if ( 0 <= x && x < width &&
             0 <= y && y < height) {
            pixels[x][y] = c;
        }
        // Otherwise off the Screen - nothing is painted.
    }

    /**
     * How wide is this Screen?
     *
     * @return the width.
     */

    public int getWidth()
    {
        return width;
    }

    /**
     * How high is this Screen?
     *
     * @return the height.
     */

    public int getHeight()
    {
        return height;
    }

    /**
     * Draw this Screen on a Terminal.
     *
     * @param t the Terminal on which to draw this Screen.
     */

    public void draw( Terminal t )
    {
        for ( int col = -1; col < width+1 ; col++ ) {  // top edge
            t.print(FRAMECHAR);
        }
        t.println();
        for ( int row = 0; row < height; row++ ) {
            t.print(FRAMECHAR);                        // left edge
            for ( int col = 0; col < width; col++ ) {
                t.print( pixels[col][row] );
            }
            t.println( FRAMECHAR );                    // right edge
        }
        for ( int col = -1; col < width+1 ; col++ ) {  // bottom edge
```

```
113             t.print(FRAMECHAR);
114         }
115         t.println();
116     }
117 }
```