# CIS 162 Project 1

# Real Estate Advertisement

## Due Date

Please see the due date on Blackboard. (One week)

## Before Starting the Project

* Read zyBook chapter 1 and 3
* Know how to compile and run a Java application in BlueJ
* Read the entire project description before starting
* Track the amount of time you spend on the project

## Learning Objectives

After completing this project, you should be able to:

* *write, compile and run* a simple Java application
* *declare* variables to store integers and Strings
* *assign* values to variables
* *write* code to draw simple shapes

**Drawing Shapes**

Unfortunately, our textbook does not describe drawing well. Use this document to learn everything you need to know to draw elements.

**X-Y Coordinate System**

Computers often use a coordinate system a bit different from the regular x-y coordinate system. The x-axis increases from left to right like usual but the y-axis increase from top to bottom. Therefore, the origin (0, 0) is in the upper left corner.

(0, 0) x-axis

y-axis

**Draw a Rectangle**

Four numbers determine the location and size of a rectangle: 1) the x location of the upper left corner, 2) the y location of the upper left corner, 3) the width 4) and the height.

g.drawRect(x, y, width, height);

For example, the following statement will draw a 200 x 100 rectangle at location (40, 60).

g.drawRect(40, 60, 200, 100);

**Starting Template**

1. Read the comments within the code to learn how to draw rectangles, ovals, lines and text.
2. Copy and paste this template into a BlueJ project.
3. Compile and run the code below before making changes.

import javax.swing.\*;

import java.awt.\*;

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\* Drawing class - drawing example

\* @author -

\* @version -

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public class Drawing extends JPanel{

public static void main(String[] a) {

JFrame f = new JFrame();

f.setContentPane(new Drawing());

f.setSize(600, 400);

f.setVisible(true);

}

public void paintComponent(Graphics g){

// this statement is required

super.paintComponent(g);

// optional: paint the background color (default is white)

setBackground(Color.CYAN);

// display words

g.setColor(Color.BLACK);

g.drawString("Picture Title", 130, 20);

// draw a solid red rectangle

g.setColor(Color.RED);

g.fillRect(100, 100, 70, 50);

// draw the outline of a rectangle

g.drawRect(100, 170, 70, 50);

// draw a solid green oval

g.setColor(Color.GREEN);

g.fillOval(50, 10, 70, 50);

// draw the outline of an oval

g.drawOval(20, 50, 70, 50);

// draw a line

g.setColor(Color.BLUE);

g.drawLine(190, 50, 190, 150);

}

}

## Project Requirements

Create an application that displays an advertisement for a piece of property (house, business, condominium, lot, or land). All lines of code that you write will be within the paintComponent method.

* **Start with the template provided above.**
* **Draw a 500x300 rectangle to form the border (5 pts).**
* **Provide the title, description and features of the house or property plus any other fictional information you choose such as realtor or map (15 pts).**
* **Use at least five drawing elements such as lines, rectangles and ovals to create a diagram of the property. This can be a diagram of the house or apartment or survey of the property. (15 pts).**
* **Use at least two colors but not too many (10 pts).**
* **The advertisement layout should be attractive and well balanced (10 pts).**
* **Avoid using literal numbers as parameters (see below) (15 pts).**

## Program Style Guide (10 pts)

Elegant source code (including comments) that follows the GVSU [Java Style Guide](http://www.cis.gvsu.edu/java-coding-style-guide/).

**Using Variables as Method Parameters**

Your goal is to be able to change the location of your diagram and other portions by changing only a few lines of code. If you use literal numbers for the parameters then it makes it more difficult to change the location of the drawing later. For example, if you have drawRect(1,2,3,4), you would have to change the location parameters (1 and 2) for this method, and all the other methods used to draw your diagram.

Instead, assign values to variables for the base location of the diagram (like the upper-level corner) and use them as parameters. For example…

**Poor:**

g.drawRect(10, 10, 50, 40);

**Better:**

g.drawRect(x, y, width, height);

g.drawRect(x + 5, y + 10, width, height \* 1.5);

## Challenge Requirements (20 pts)

The following should only be attempted after **all** of the above requirements have been completed. Challenge exercises demonstrate that you have the initiative to investigate problems and identify solutions with minimal help from your instructor.

* Use multiple fonts sizes and styles. See additional information below. (10 pts)
* Add a photo of the property. See additional information below. (10 pts)

**Changing Font Sizes and Styles**

The setFont method takes a single parameter of type Font. However, a Font object must be instantiated first.

Font myFont = new Font("serif", Font.ITALIC, 20);

g.setFont(myFont);

* Generic font names include “serif”, “sanserif” and “monospaced”. More specific font names might work as well depending on the computer: “Times”, “Helvetica”, “Courier” and other names you might know.
* Font styles include Font.PLAIN, Font.ITALIC, Font.BOLD or you can combine Font.ITALIC+Font.BOLD.
* Font size, the third parameter, typically ranges from 10-48.

**Adding an Image**

You must import additional packages for the following code to compile.

import java.awt.image.\*;

import java.net.\*;

import javax.imageio.\*;

import java.io.\*;

The following code uses some advanced features not covered in the book to display a JPEG or GIF image. This assumes you have an image in the directory of your BlueJ project called “MyPhoto.jpg” but you should replace this with the name of your actual image file.

BufferedImage photo = null;

try {

File file = new File("MyPhoto.jpg");

photo = ImageIO.read(file);

} catch (IOException e){

g.drawString("Problem reading the file", 100, 100);

}

g.drawImage(photo, 10, 10, 150, 225, null);

## Late Policy

You should turn in your projects on time at the START of the class period. However, you are encouraged to complete a project even if you must turn it in late.

* The first day (-20 pts)
* Each subsequent weekday is an additional -10 pts
* Weekends and university holidays are free days.

## Turn In

A professional document **that is printed and stapled**.

* Cover page - Your project must have a cover page that includes your name, a title, and screenshot of your advertisement. (-5 pts if missing)
* Signed Pledge – The cover page must include the following signed pledge: "I pledge that this work is entirely mine, and mine alone (except for any code provided by my instructor).” In addition, provide names of any people you helped or received help from. Under no circumstances do you exchange code electronically. You are responsible for understanding and adhering to the [School of CIS Guidelines for Academic Honesty](http://www.cis.gvsu.edu/academic-honesty/).
* Time Card – The cover page must also include a brief statement of how much time you spent on the project. For example, “I spent 2 hours on this project from January 22-24 reading the book, designing a solution, writing code, fixing errors and putting together the printed document.”
* Source code - a printout of your elegant source code (with your name).
* Demo – be prepared to demo your project on a lab computer or your laptop. I will ask you to perform a variety of tasks including moving the diagram of the property by changing only two lines of your code.