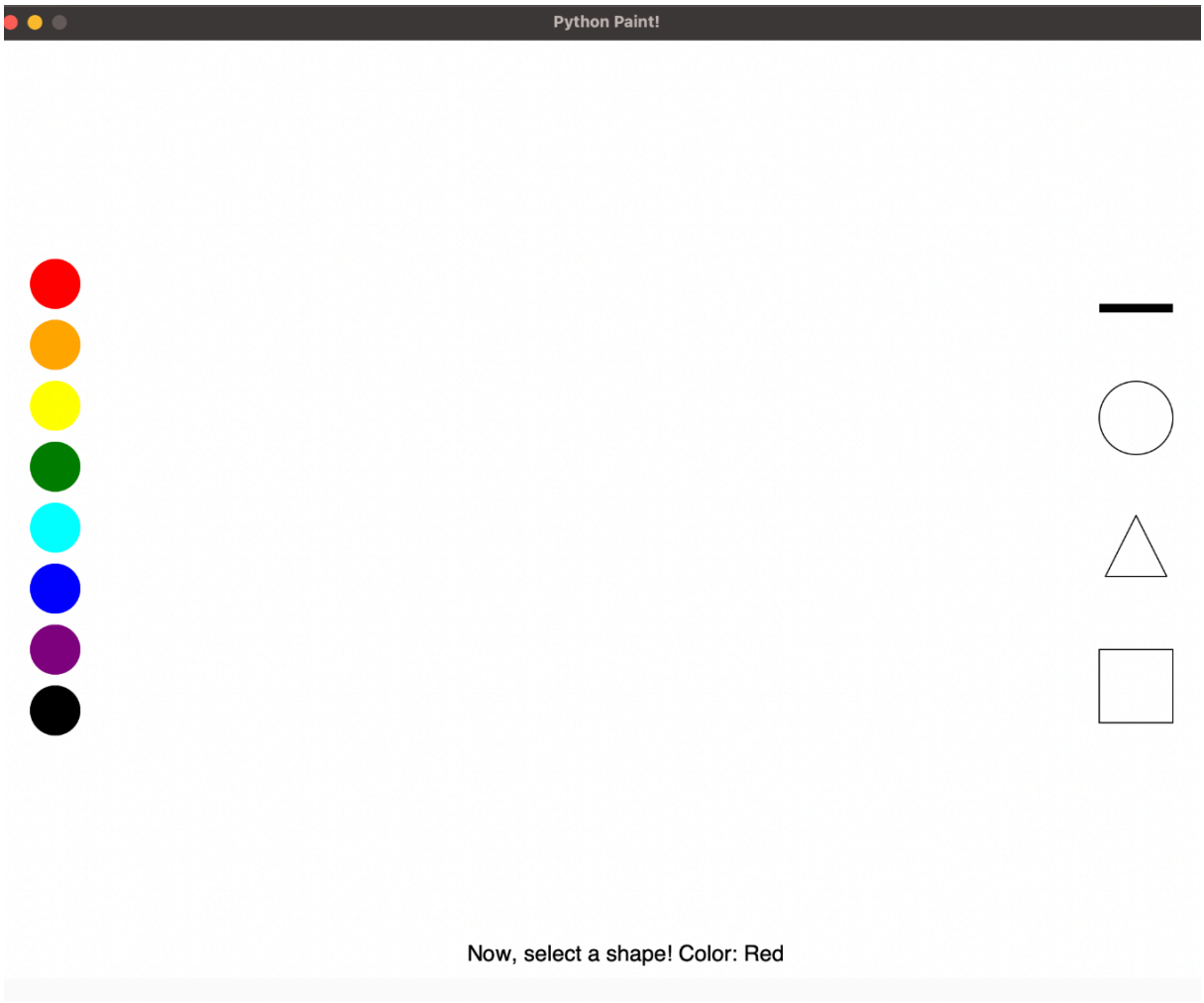
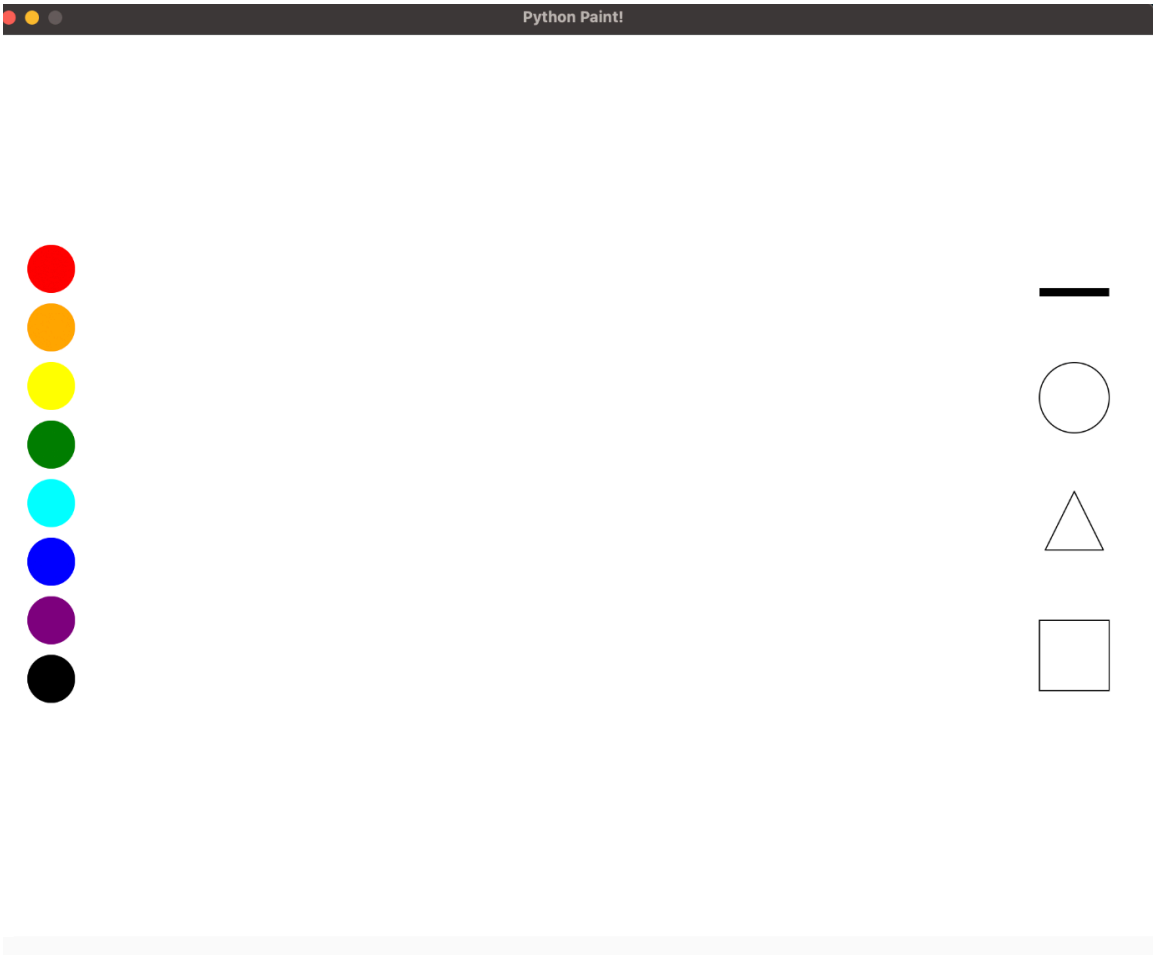
Aiden Johnson's Final Project Write Up

Welcome to Python Paint! An art software created by Aiden Johnson. Similar to Microsoft Paint, Python Paint allows the user to create original artwork utilizing eight unique colors and four different shapes. Python Paint needs to be in the same folder with Graphics.py in order to function. Once Python Paint is run, a white window will pop up titled "Python Paint!".

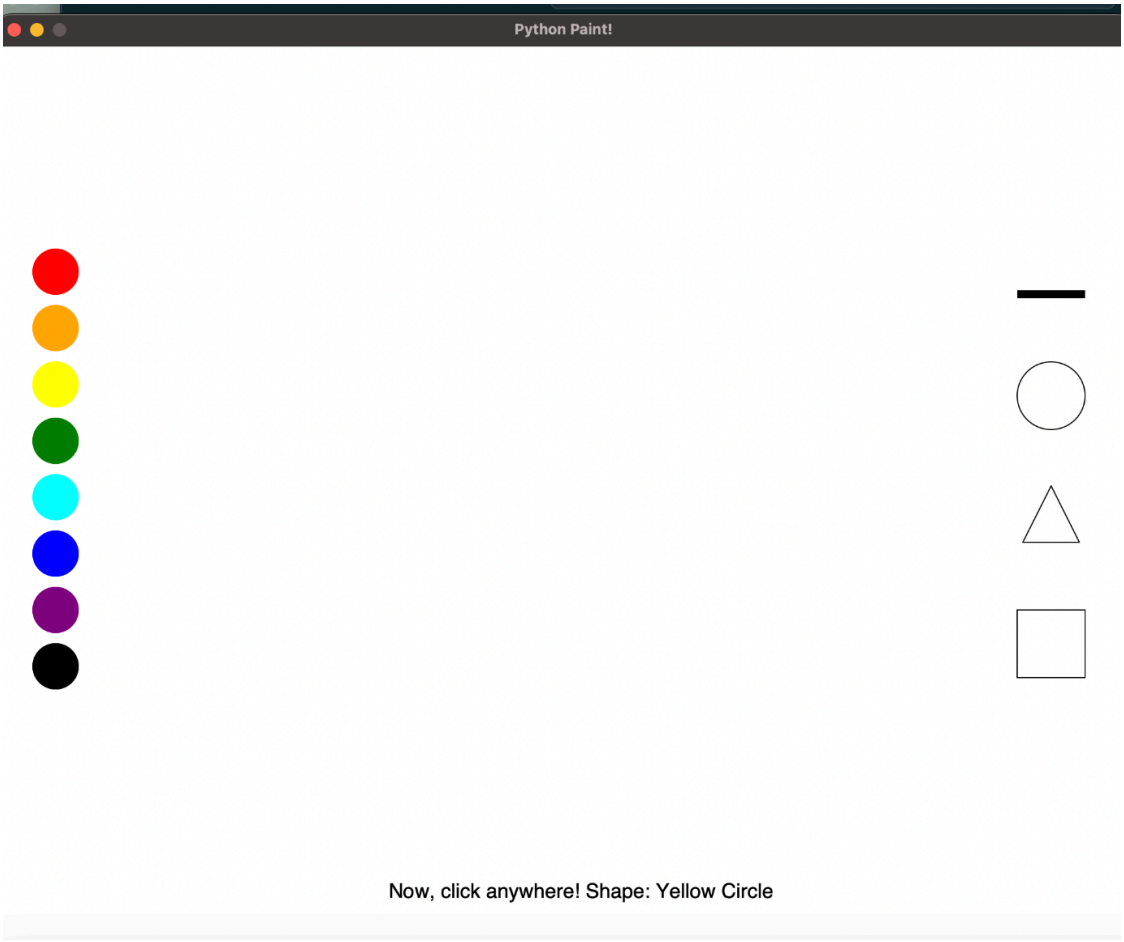
The window contains three different zones. On the left side, we have eight colors to choose from. On the right ride, we have our selection of shapes to choose from. On the bottom, at each step of the drawing process, a message pops up to remind the user on how to use the program if they get lost.



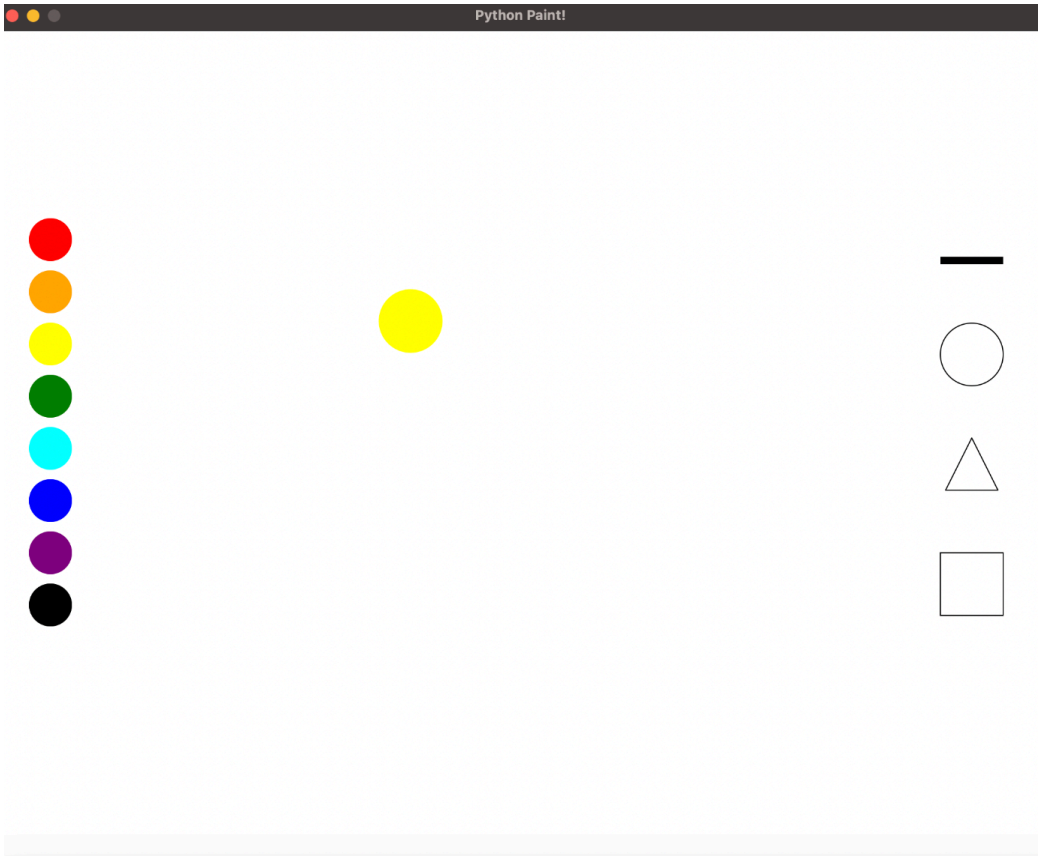
Once a color is selected, a new message at the bottom appears. This message then prompts the user to select a shape, and confirms the color they chose. If a user wants to change the color chosen before drawing it, all they need to do is click on anything but a shape, then the color will reset.

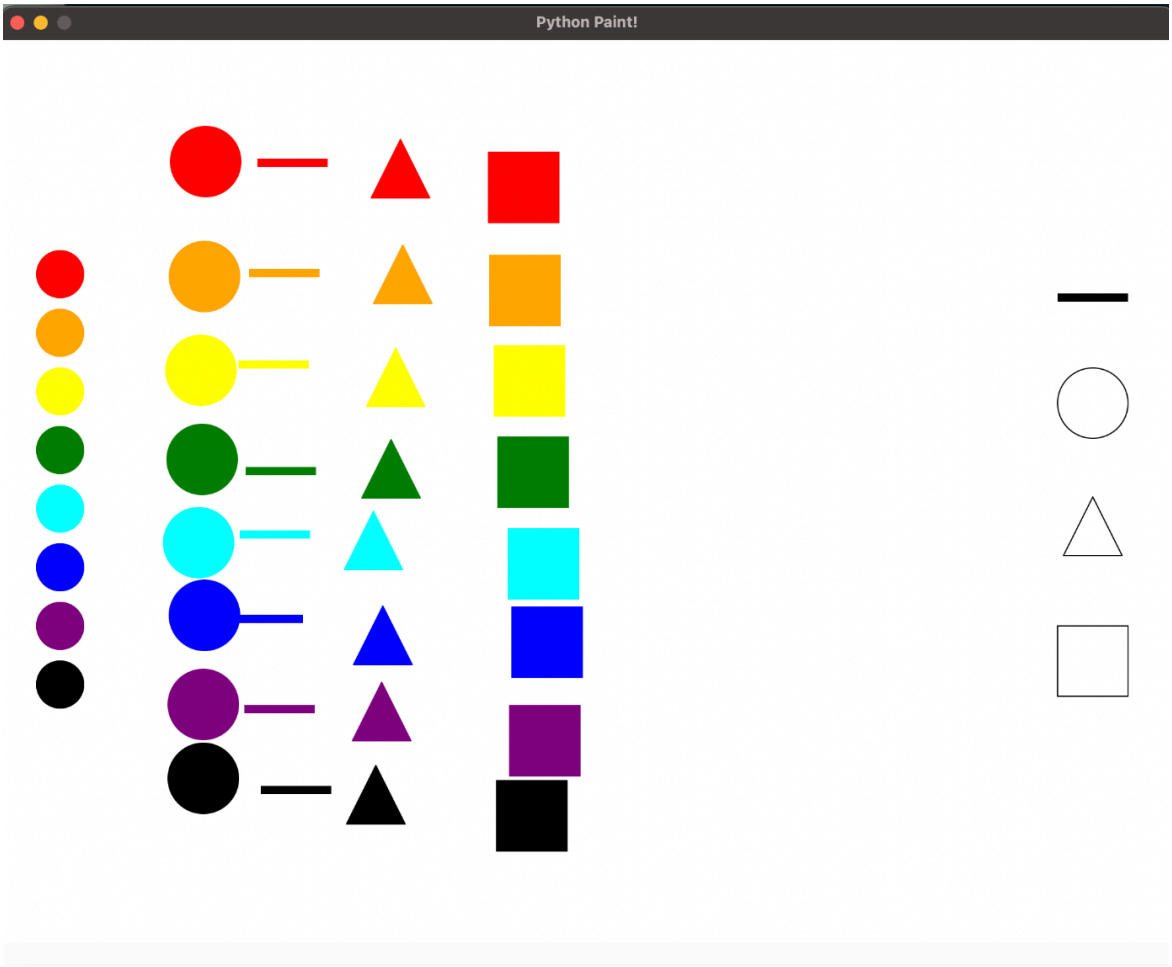


Once the user has decided their color, they can then select any of the shapes provided. Which gives another bottom message:



The message tells the user to click ***anywhere*** in the window with their point. Even if a user clicks and covers up a color or a shape, the code will still function. The shape and color is also written down in the bottom text for the user in case they forget. Below is our first shape drawn!


Python Paint!



There is no limit to how many shapes the user can draw in the window, so draw whatever you want to your heart's desire!

The order to draw follows the same pattern each time: First, click a color. Second, click a shape. Finally, click anywhere in the window to draw that colored shape.

Have fun!

Python Paint Coding Process:

First, I created the two classes I would need for this final project. The most necessary components in this code are the Color Circles and the Drawn Shapes in the window. Without the Color Circle, there would be no color selection or initiation of the drawing process, and without the Drawn Shapes, no shapes would be drawn in the window.

Two methods I would like to explicitly mention in the Color Circle class are “didClick()” and “getColor()”. The didClick method checks if the user’s click is within the radius of one of the color circles. If didClick() returns True, the click’s coordinates cycle through the circle list and select the corresponding color in the getColor() method. In class ShapeMade, For each shape, a different method is called in order to properly draw the shape. For each shape, a different method is called in order to properly draw the shape. In the draw() method, I used almost the same formulas as I did when creating the window’s shapes, but instead I had to adjust the coordinate values to be able to be used at any point instead of a fixed point like they are in the window set up.

The function getDistance() uses a standard distance formula in order to track the distance between two points. This is used in the didClick() method.

The function recordClick() records the window’s click input and then gives the user both the x and y coordinates.

The function windowSetUp() does exactly what It sounds like it does. So, a window is created and the shape templates on the right side are created as well.

The function colorCircleSetUp() takes a list of colors, then uses the Color Circle draw method to draw those shapes into the left side of the window. A circleList is also created as the for loop goes through. The circleList is returned.

The function getShape() checks the user's click when its x coordinate is within bounds of the right side's shapes. If a y coordinate fits within the zone of a specific shape, that shape is returned. If the click happens to be in the x coordinate but NOT in the y coordinate required to get a shape, then no shape is returned.

The function pythonPaintActivate() takes in the window and the circleList. First a message is displayed at the bottom of the window which helps give instructions to the user. Then, a while loop begins containing the rest of the code, which keeps the code going indefinitely. Both the selected color and the selected shape begin with no value attached to them. Then, the first recorded click is gathered, and if it is within the bounds of a color circle, a selected color is chosen. The user can click on anything they want in the window before clicking a color to initiate the next phase of the code. Then a new message appears with a new prompt. If the next click works for the getShape() function, then the final phase of the drawing process commences. Now that both selected color and selected shape have values, there is one more prompt to click anywhere and draw that colored shape. Then, the shapeMade class is called which draws the shape.

Then, all of this work goes into five lines of code in main, and main is then run.