

CS 141 Graphics Library (Simplegraphics)

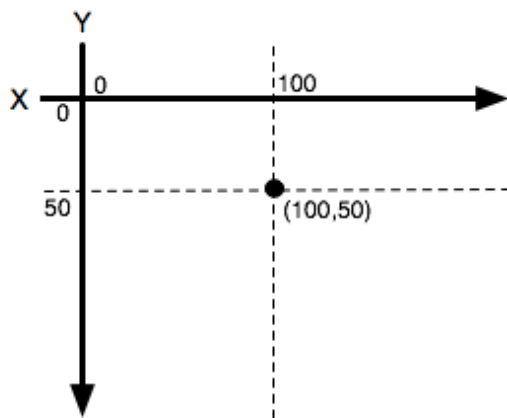
The graphics library uses the concept of a “canvas,” which you can think of as a dedicated window on the screen that can be drawn on. To create a canvas, you call the `open_canvas()` function, then you can call any drawing-related function. You should always call `close_canvas()` or `close_canvas_after_click()` at the end of your program.

To use the library, make sure both the files “simplegraphics.py” and “cs1graphics.py” are in the same directory as your program. Make sure your program has the following line at the top:

```
from simplegraphics import *
```

This line “imports” all of the simplegraphics functions into your Python program so you can use them in your code.

The canvas is based on a grid of pixels, set up in a coordinate system with $(0, 0)$ at the top-left corner. So the x-coordinate increases left to right (as you’d expect), but the y-coordinate increases from top to bottom, unlike in mathematical graphs, where the y-coordinate usually increases bottom to top.



Functions in the library

Opening and closing the canvas:

- `open_canvas(width, height)`: Creates a new canvas in a new window of the given width and height in pixels (as integers). There can only be one canvas open at a time.
- `close_canvas()`: Immediately closes the canvas window. Everything drawn is lost.
- `close_canvas_after_click()`: Waits for a mouse click anywhere in the canvas window, then closes the window. The program temporarily pauses while waiting for the click.

Changing the pen color: Think of the canvas as having a colored pen associated with it that is used to draw on the canvas. You can change the pen's color and the thickness of the lines it draws.

- `set_color(color)`: Sets the color of the drawing pen to the color specified as a **string**. Examples: "red", "blue", "green", etc.
- `set_line_thickness(thickness)`: Sets the pen thickness (width of lines drawn) to the given thickness value in pixels (as an integer).

Drawing shapes, lines, and text:

- `draw_circle(centerx, centery, radius)`: Draws a circle of the specified radius with the center at the point (centerx, centery).
- `draw_filled_circle(centerx, centery, radius)`: Same as above, but the circle is filled in with the current pen color.
- `draw_oval(centerx, centery, radiusx, radiusy)`: Draws an oval centered at the point (centerx, centery), with a radius in the x direction of radiusx, and in the y direction as radiusy.
- `draw_filled_oval(centerx, centery, radiusx, radiusy)`: Same as above, but the oval is filled in with the current pen color.
- `draw_line(x1, y1, x2, y2)`: Draws a line from the point (x1, y1) to (x2, y2).
- `draw_rect(x, y, width, height)`: Draws a rectangle with the upper-left corner at (x, y) and the width and height as specified.
- `draw_polygon(x1, y1, x2, y2, ...)`: Draws a polygon on the canvas. The points of the polygon are (x,y) pairs specified as one big list. E.g.: `draw_polygon(10, 10, 20, 20, 30, 40)` draws a polygon bounded by the line segments between the points (10, 10), (20, 20), (30, 40), and (10, 10).
- `draw_filled_polygon(x1, y1, x2, y2, ...)`: Same as above, but the polygon is filled with the current pen color.
- `draw_polyline(x1, y1, x2, y2, ...)`: Similar to drawing a polygon, except no line segment is drawn back to the starting point from the final point.
- `draw_string(message, x, y, textSize)`: Draws the message at the location (x, y) in the text size specified in points. (locx, locy) will be where the midpoint of the message is placed.

Mouse clicks:

- `wait_for_click()`: Pauses the program until the mouse is clicked in the canvas window. Then the program continues as soon as the user clicks the mouse.
- `get_last_click_x()`: Returns the x coordinate of the location on the canvas of the last mouse click. **Only can be used after wait_for_click()!**
- `get_last_click_y()`: Same as previous function, but returns the y coordinate.

Other functions:

- `set_background_color(color)`: Sets the background color of the canvas to the color specified as a **string**. Can be set multiple times and will be redrawn every time called.
- `clear_canvas()`: Removes all shapes and text from the canvas.