```
# ---------- TKINTER PAINT APP 2 ----------

from tkinter import *
import tkinter.font
from tkinter.colorchooser import *

# Create main window
root = Tk()
root.geometry("800x600")


class PaintApp:
    text_font = StringVar()
    text_size = IntVar()
    bold_text = IntVar()
    italic_text = IntVar()
    # Stores current tool we are using
    drawing_tool = StringVar()

    # NEW STORE DRAWING SETTINGS
    stroke_size = IntVar()
    fill_color = StringVar()
    stroke_color = StringVar()

    # Tracks whether left mouse is down
    left_but = "up"

    # x and y positions for drawing with pencil
    x_pos, y_pos = None, None

    # Tracks x & y when the mouse is clicked and released
    x1_line_pt, y1_line_pt, x2_line_pt, y2_line_pt = None, None, None, None

    # Quits the TkInter app when called
    @staticmethod
    def quit_app():
        root.quit()

    def make_menu_bar(self):
        # Create the menu object
        the_menu = Menu(root)

        # ---- FILE MENU ----
        # Create a pull down menu that can't be removed
        file_menu = Menu(the_menu, tearoff=0)

        # Add items to the menu that show when clicked
        # compound allows you to add an image
        file_menu.add_command(label="Open")
        file_menu.add_command(label="Save")

        # Add a horizontal bar to group similar commands
        file_menu.add_separator()
```

```
        # Call for the function to execute when clicked
        file_menu.add_command(label="Quit", command=self.quit_app)

        # Add the pull down menu to the menu bar
        the_menu.add_cascade(label="File", menu=file_menu)

        # ---- FONT MENU ----
        font_menu = Menu(the_menu, tearoff=0)
        font_type_submenu = Menu(font_menu)
        font_type_submenu.add_radiobutton(label="Times",
                            variable=self.text_font)
        font_type_submenu.add_radiobutton(label="Courier",
                            variable=self.text_font)
        font_type_submenu.add_radiobutton(label="Ariel",
                            variable=self.text_font)
        font_menu.add_cascade(label="Font Type",
                          menu=font_type_submenu)

        font_size_submenu = Menu(font_menu)
        font_size_submenu.add_radiobutton(label="10",
                                  variable=self.text_size,
                                  value=10)
        font_size_submenu.add_radiobutton(label="15",
                                  variable=self.text_size,
                                  value=15)
        font_size_submenu.add_radiobutton(label="20",
                                  variable=self.text_size,
                                  value=20)
        font_size_submenu.add_radiobutton(label="25",
                                  variable=self.text_size,
                                  value=25)
        font_menu.add_cascade(label="Font Size",
                          menu=font_size_submenu)
        font_menu.add_checkbutton(label="Bold",
                            variable=self.bold_text,
                            onvalue=1,
                            offvalue=0)
        font_menu.add_checkbutton(label="Italic",
                            variable=self.italic_text,
                            onvalue=1,
                            offvalue=0)
        the_menu.add_cascade(label="Font", menu=font_menu)

        # ---- TOOL MENU ----
        tool_menu = Menu(the_menu, tearoff=0)
        tool_menu.add_radiobutton(label="Pencil",
                          variable=self.drawing_tool,
                          value="pencil")
        tool_menu.add_radiobutton(label="Line",
                          variable=self.drawing_tool,
                          value="line")
        tool_menu.add_radiobutton(label="Arc",
                          variable=self.drawing_tool,
```

```
                    value="arc")
        tool_menu.add_radiobutton(label="Oval",
                    variable=self.drawing_tool,
                    value="oval")
        tool_menu.add_radiobutton(label="Rectangle",
                    variable=self.drawing_tool,
                    value="rectangle")
        tool_menu.add_radiobutton(label="Text",
                    variable=self.drawing_tool,
                    value="text")
        the_menu.add_cascade(label="Tool", menu=tool_menu)

        # ---- NEW COLOR MENU ----

        color_menu = Menu(the_menu, tearoff=0)
        color_menu.add_command(label="Fill", command=self.pick_fill)
        color_menu.add_command(label="Stroke", command=self.pick_stroke)
        stroke_width_submenu = Menu(color_menu)
        stroke_width_submenu.add_radiobutton(label="2",
                                variable=self.stroke_size,
                                value=2)
        stroke_width_submenu.add_radiobutton(label="3",
                                  variable=self.stroke_size,
                                  value=3)
        stroke_width_submenu.add_radiobutton(label="4",
                                  variable=self.stroke_size,
                                  value=4)
        stroke_width_submenu.add_radiobutton(label="5",
                                  variable=self.stroke_size,
                                  value=5)
        color_menu.add_cascade(label="Stroke Size",
                    menu=stroke_width_submenu)
        the_menu.add_cascade(label="Color", menu=color_menu)

        # ---- END OF NEW COLOR MENU ----

        # Display the menu bar
        root.config(menu=the_menu)

    # ----- NEW STUFF -----

    # ---------- CATCH MOUSE UP ----------

    def left_but_down(self, event=None):
        self.left_but = "down"

        # Set x & y when mouse is clicked
        self.x1_line_pt = event.x
        self.y1_line_pt = event.y

    # ---------- CATCH MOUSE UP ----------

    def left_but_up(self, event=None):
        self.left_but = "up"
```

```
        # Reset the line
        self.x_pos = None
        self.y_pos = None

        # Set x & y when mouse is released
        self.x2_line_pt = event.x
        self.y2_line_pt = event.y

        # If mouse is released and line tool is selected
        # draw the line
        if self.drawing_tool.get() == "line":
            self.line_draw(event)
        elif self.drawing_tool.get() == "arc":
            self.arc_draw(event)
        elif self.drawing_tool.get() == "oval":
            self.oval_draw(event)
        elif self.drawing_tool.get() == "rectangle":
            self.rectangle_draw(event)
        elif self.drawing_tool.get() == "text":
            self.text_draw(event)

    # ---------- CATCH MOUSE MOVEMENT ----------

    def motion(self, event=None):
        if self.drawing_tool.get() == "pencil":
            self.pencil_draw(event)

    # ---------- DRAW PENCIL ----------

    def pencil_draw(self, event=None):
        if self.left_but == "down":

            # Make sure x and y have a value
            if self.x_pos is not None and self.y_pos is not None:
                event.widget.create_line(self.x_pos, self.y_pos, event.x, event.y, smooth=TRUE,
fill=self.stroke_color.get(), width=self.stroke_size.get())

            self.x_pos = event.x
            self.y_pos = event.y

    def line_draw(self, event=None):
        pass

    def arc_draw(self, event=None):
        pass

    def oval_draw(self, event=None):
        pass

    def rectangle_draw(self, event=None):
        pass

    def text_draw(self, event=None):
```

```
        pass

    def pick_fill(self, event=None):
        fill_color = askcolor(title='Pick Fill color')
        if None not in fill_color:
            self.fill_color.set(fill_color[1])
            print("Color ", self.fill_color.get())

    def pick_stroke(self, event=None):
        stroke_color = askcolor(title='Pick Stroke color')
        if None not in stroke_color:
            self.stroke_color.set(stroke_color[1])

    # ----- END OF NEW STUFF -----

    def __init__(self, root):
        drawing_area = Canvas(root, width=800, height=600)
        drawing_area.pack()

        self.text_font.set("Times")
        self.text_size.set(20)
        self.bold_text.set(0)
        self.italic_text.set(0)
        self.drawing_tool.set("pencil")

        # NEW COLOR DRAWING SETTINGS
        self.stroke_size.set(3)
        self.fill_color.set('#000000')
        self.stroke_color.set('#000000')

        self.make_menu_bar()

        # Set focus for catching events to the canvas
        drawing_area.focus_force()

        # NEW Assign different events to method calls
        drawing_area.bind("<Motion>", self.motion)
        drawing_area.bind("<ButtonPress-1>", self.left_but_down)
        drawing_area.bind("<ButtonRelease-1>", self.left_but_up)


paint_app = PaintApp(root)
root.mainloop()
```