

## Copilot

Absolutely, you can create a separate frame for the buttons and the canvas. Here's how you can modify the code:

```
import tkinter as tk

class Application(tk.Frame):
    def __init__(self, master=None):
        super().__init__(master)
        self.master = master
        self.pack()
        self.create_widgets()

    def create_widgets(self):
        self.button_frame = tk.Frame(self)
        self.button_frame.pack(side="top")

        self.add_button = tk.Button(self.button_frame)
        self.add_button["text"] = "Add"
        self.add_button["command"] = self.enable_add
        self.add_button.pack(side="left")

        self.delete_button = tk.Button(self.button_frame)
        self.delete_button["text"] = "Delete"
        self.delete_button["command"] = self.enable_delete
        self.delete_button.pack(side="right")

        self.canvas = tk.Canvas(self, width=400, height=400, bg='white')
        self.canvas.pack()
        self.canvas.bind("<Button-1>", self.draw_or_delete_circle)

        self.adding = False
        self.deleting = False

    def enable_add(self):
        self.adding = True
        self.deleting = False
        self.add_button.config(relief="sunken")
        self.delete_button.config(relief="raised")

    def enable_delete(self):
        self.adding = False
        self.deleting = True
        self.add_button.config(relief="raised")
        self.delete_button.config(relief="sunken")

    def draw_or_delete_circle(self, event):
        x, y = event.x, event.y
        r = 10 # radius
        overlapping = self.canvas.find_overlapping(x-r, y-r, x+r, y+r)
        if self.adding:
            self.canvas.create_oval(x-r, y-r, x+r, y+r, fill="blue")
        elif self.deleting and overlapping:
            self.canvas.delete(overlapping)

root = tk.Tk()
app = Application(master=root)
app.mainloop()
```

In this updated code, I've added a `button_frame` that contains the "Add" and "Delete" buttons. This frame is packed at the top of the application window. The canvas is packed below the `button_frame`, so the buttons appear above the canvas. The canvas has a white background to clearly delineate the area where clicks can occur. The behavior of the buttons and the canvas remains the same as in the previous version of the code.