Copilot

Sure, you can center the buttons by using a grid layout for the button frame. 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(fill="both", expand=True)
        self.create_widgets()

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

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

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

        # Use grid layout to center the buttons
        self.add_button.grid(row=0, column=0)
        self.delete_button.grid(row=0, column=1)
        self.button_frame.grid_columnconfigure(0, weight=1)
        self.button_frame.grid_columnconfigure(1, weight=1)

        self.canvas = tk.Canvas(self, bg='white')
        self.canvas.pack(fill="both", expand=True)
        self.canvas.bind("<Button-1>", self.draw_or_delete_circle)
        self.canvas.bind("<Configure>", self.resize_circles)

        self.adding = False
        self.deleting = False
        self.circles = []

    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:
            id = self.canvas.create_oval(x-r, y-r, x+r, y+r, fill="blue")
            self.circles.append((id, x, y, r))
        elif self.deleting and overlapping:
            for circle in self.circles:
                if circle[0] in overlapping:
                    self.canvas.delete(circle[0])
                    self.circles.remove(circle)

    def resize_circles(self, event):
        for circle in self.circles:
            id, x, y, r = circle
            self.canvas.coords(id, x-r, y-r, x+r, y+r)
```

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

In this updated code, I’ve changed the layout manager for the `button_frame` from `pack` to `grid`. The “Add” and “Delete” buttons are placed in the same row but different columns of the grid. The `grid_columnconfigure` method is used to give equal weight to both columns, which centers the buttons in the `button_frame`. Now, the buttons should appear next to each other, centered around the middle of the canvas. I hope this helps! Let me know if you have any other questions.