In this tutorial I'll show you how to save and retrieve our styling information with Tkinter.

```
from tkinter import *
import tkinter.filedialog
import ast


class TextEditor:

    # Quits the TkInter app when called
    @staticmethod
    def quit_app(event=None):
        root.quit()

    # ----- NEXT TUTORIAL -----

    def remake_file(self, text_area_list):
        for i in text_area_list:
            print("Key", i[0])
            print("Value", i[1])
            print("Index", i[2])

    # ----- END NEXT TUTORIAL -----

    def open_file(self, event=None):
        # Open dialog and get chosen file
        txt_file = tkinter.filedialog.askopenfilename(parent=root,
                                              initialdir='/')
        # If the file exists
        if txt_file:
            self.text_area.delete(1.0, END)

            # Holds list of tuples
            file_list = []

            # Open file and put text in the text widget
            with open(txt_file) as _file:
                # self.text_area.insert(1.0, _file.read())

                # Processes the list of tuples into a list
                file_list = list(ast.literal_eval(_file.read()))
                print(file_list)

                # Search for text in the list and put it in the right
                # index position
                for data in file_list:
                    if data[0] == "text":
                        self.text_area.insert(data[2], data[1])

                # Cycle through the list looking for tagon, but ignore sel
                i = 0
                while i < len(file_list):
                    if (file_list[i][0] == "tagon") and (file_list[i][1] != "sel"):
                        # Get the styling tag
```

```
                styling = file_list[i][1]
                # Get the index where styling begins
                start_of_style = file_list[i][2]
                # Used to get the index where styling ends
                # but set as end of file by default
                end_of_style = END
                # Make sure I'm not searching beyond the end
                # of the list
                if (i+4) < len(file_list):
                    # If not find the end index
                    end_of_style = file_list[i+4][2]
                print("Style", styling)
                print("Start", start_of_style)
                print("End", end_of_style)
                # Add styling provided along with the start
                # and ending index
                self.text_area.tag_add(styling,
                                       start_of_style,
                                       end_of_style)
            i += 1

        # Update the text widget
        root.update_idletasks()

def save_file(self, event=None):

    # Opens the save as dialog box
    file = tkinter.filedialog.asksaveasfile(mode='w')
    if file is not None:
        # Get text in the text widget and delete the last newline
        data = self.text_area.get('1.0', END + '-1c')

        # Write the text and close
        # file.write(data)

        # ----- NEXT TUTORIAL -----

        # print(str(self.text_area.dump('1.0', END)))
        # self.remake_file(self.text_area.dump('1.0', END))

        # Get list of tuples
        text_area_list = self.text_area.dump('1.0', END + '-1c')
        # Write list of tuples to file
        file.write(' '.join('("{}", "{}", "{}")'.format(x[0],
                                        x[1], x[2])
                            for x in text_area_list))

        # ----- END NEXT TUTORIAL -----

        file.close()

def make_bold(self):
    self.text_area.tag_add("bt", "sel.first", "sel.last")
```

```
def __init__(self, root):

    self.text_to_write = ""

    # Define title for the app
    root.title("Text Editor")

    # Defines the width and height of the window
    root.geometry("600x550")

    frame = Frame(root, width=600, height=550)

    # Create the scrollbar
    scrollbar = Scrollbar(frame)

    # yscrollcommand connects the scroll bar to the text
    # area
    self.text_area = Text(frame, width=600, height=550,
                          yscrollcommand=scrollbar.set,
                          padx=10, pady=10, font=("Georgia", "28"))

    # Call yview when the scrollbar is moved
    scrollbar.config(command=self.text_area.yview)

    # Put scroll bar on the right and fill in the Y direction
    scrollbar.pack(side="right", fill="y")

    # Pack on the left and fill available space
    self.text_area.pack(side="left", fill="both", expand=True)
    frame.pack()

    # ----- FILE MENU CREATION -----

    # 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", command=self.open_file)
    file_menu.add_command(label="Save", command=self.save_file)

    # 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)

    # ----- EDIT MENU CREATION -----

    edit_menu = Menu(the_menu, tearoff=0)
    edit_menu.add_command(label="Bold", command=self.make_bold)
```

```
        the_menu.add_cascade(label="Edit", menu=edit_menu)

        self.text_area.tag_config("bt", font=("Georgia", "28", "bold"))

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


root = Tk()

# Create the menu object
the_menu = Menu(root)

text_editor = TextEditor(root)

root.mainloop()
```