tkinter Summary

import tkinter	`from tkinter import * # will include messagebox now`
import model class	`from classFileName import ClassName # will include messagebox now`
create window	`self.__windowName = tkinter.Tk()`
start listener	`mainloop()`
create static label	`self.__labelName = Label(self.__nameOfWindowOrFrameInWhichILive, text = 'label text')`
create dynamic label	`self.__labelName = Label(self.__nameOfWOFIWIL, textvariable=self.__nameOfIntVarFloatVarOrStringVar)`
create button	`self.__buttonName = Button(self.__nameOfWOFIWIL, text='text that appears on button',` `command=self.nameOfEventHandler) # Don't use parentheses when providing event handler method name`
button EH method header	`def nameOfButtonEventHandlerMethod(self)  # Can't take in any other parameters`
method that quits	`self.__nameOfWindow.destroy()  # Don't use () when in context of naming this method as event handler`
create frame	`self.__nameOfFrame = Frame(self.__nameOfWOFIWIL)`
create entry box	`self.__nameOfEntryBox = Entry(self.__nameOfWOFIWIL, width = n  # where n is integer, e.g., 10`
binding entry box to EH	`self.__nameOfEntryBox.bind('<Return>', self.nameOfEventHandler) #Note that you can't use () after method name`
entry EH method header	`def nameOfEntryBoxEventHandlerMethod(self, event) #Note that ONLY these params are provided, must have BOTH`
get value from entry box	`self.__nameOfEntryBox.get()`
clear value from entry box	`self.__nameOfEntryBox.delete(0, END)`
create string variable	`self.__nameOfStringVariable = StringVar()`
set value of string variable	`self.__nameOfStringVariable.set ('some string value')`
create int variable	`self.__nameOfIntVariable = IntVar()`
set value of int variable	`self.__nameOfIntVariable.set (n)  # where n is an integer value`
create float variable	`self.__nameOfFloatVariable = FloatVar()`
set value of float variable	`self.__nameOfFloatVariable.set (x)  # where x is a float value`
create radio button	`self.__nameOfRadioButton = Radiobutton(self.__nameOfWOFIWIL, text = 'associated text', variable =` `self.__nameOfTheIntVar4RadioGroup, value = n)  #Where n = integer value associated with THIS button`
create check button	`self.__nameOfCheckButton = Checkbutton(self.__nameOfWOFIWIL, text = 'associated text',` `variable=self.__nameOfIntVar4ThisButton)`
create model	`self.__modelName = ClassName() #Don't need classFileName. if you use from classFileName import ClassName`
packing a widget or frame	`self.__widgetName.pack()  #watch out for order!`
packing left to right	`self.__widgetName.pack(side = 'left')  #watch out for order!`
setting widget attribute	`self.__widgetName.config(attributeName = value)  #attribute and value must be valid for given widget`
invoke MessageBox info	`messagebox.showinfo('titlebar text', 'message text')`
invoke MB warning	`messagebox.showwarning('titlebar text', 'message text')`
invoke MB error	`messagebox.showerror('titlebar text', 'message text')`
invoke MB yes/no	`response = messagebox.askyesno('title', 'message text')`
invoke MB ok/cancel	`response = messagebox.askokcancel('title', 'message text')`

`self.__nameOfWOFIWIL` is short for `self.__nameOfWindowOrFrameInWhichILive`