**GUI Programming**

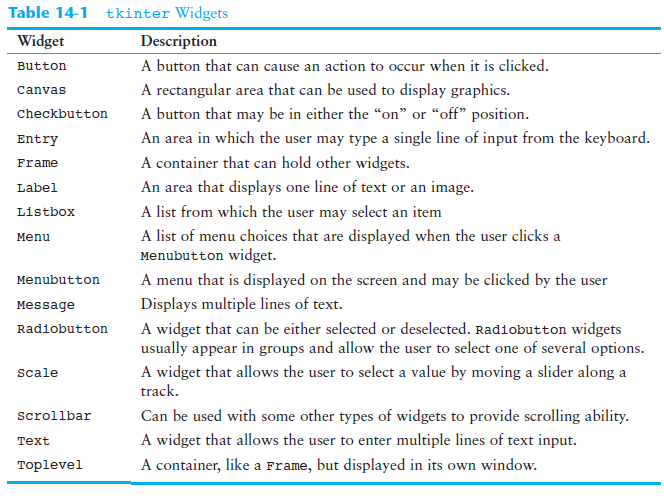
Graphical interface allows user to interact with operating system and other programs using graphical elements such as icons, buttons, and dialog boxes.

GUI programs are event-driven – GUI programs must respond to the actions of the user.

Example clicking of a button and the program respond to the events.

tkinter module is used to create simple GUI programs In Python.

tkinter provides 15 widgets so that user can interact or view with.



Callback function – also known as event handler as it handles the event that occurs when the user clicks the button.

# This program uses a GUI to get three test

# scores and display their average.

import tkinter

class TestAvg:

def \_\_init\_\_(self):

# Create the main window.

self.main\_window = tkinter.Tk()

# Create the five frames.

self.test1\_frame = tkinter.Frame(self.main\_window)

self.test2\_frame = tkinter.Frame(self.main\_window)

self.test3\_frame = tkinter.Frame(self.main\_window)

self.avg\_frame = tkinter.Frame(self.main\_window)

self.button\_frame = tkinter.Frame(self.main\_window)

# Create and pack the widgets for test 1.

self.test1\_label = tkinter.Label(self.test1\_frame, \

text='Enter the score for test 1:')

self.test1\_entry = tkinter.Entry(self.test1\_frame, \

width=10)

self.test1\_label.pack(side='left')

self.test1\_entry.pack(side='left')

# Create and pack the widgets for test 2.

self.test2\_label = tkinter.Label(self.test2\_frame, \

text='Enter the score for test 2:')

self.test2\_entry = tkinter.Entry(self.test2\_frame, \

width=10)

self.test2\_label.pack(side='left')

self.test2\_entry.pack(side='left')

# Create and pack the widgets for test 3.

self.test3\_label = tkinter.Label(self.test3\_frame, \

text='Enter the score for test 3:')

self.test3\_entry = tkinter.Entry(self.test3\_frame, \

width=10)

self.test3\_label.pack(side='left')

self.test3\_entry.pack(side='left')

# Create and pack the widgets for the average.

self.result\_label = tkinter.Label(self.avg\_frame, \

text='Average:')

self.avg = tkinter.StringVar() # To update avg\_label

self.avg\_label = tkinter.Label(self.avg\_frame, \

textvariable=self.avg)

self.result\_label.pack(side='left')

self.avg\_label.pack(side='left')

# Create and pack the button widgets.

self.calc\_button = tkinter.Button(self.button\_frame, \

text='Average', \

command=self.calc\_avg)

self.quit\_button = tkinter.Button(self.button\_frame, \

text='Quit', \

command=self.main\_window.destroy)

self.calc\_button.pack(side='left')

self.quit\_button.pack(side='left')

# Pack the frames.

self.test1\_frame.pack()

self.test2\_frame.pack()

self.test3\_frame.pack()

self.avg\_frame.pack()

self.button\_frame.pack()

# Start the main loop.

tkinter.mainloop()

# The calc\_avg method is the callback function for

# the calc\_button widget.

def calc\_avg(self):

# Get the three test scores and store them

# in variables.

self.test1 = float(self.test1\_entry.get())

self.test2 = float(self.test2\_entry.get())

self.test3 = float(self.test3\_entry.get())

# Calculate the average.

self.average = (self.test1 + self.test2 + \

self.test3) / 3.0

# Update the avg\_label widget by storing

# the value of self.average in the StringVar

# object referenced by avg.

self.avg.set(self.average)

# Create an instance of the TestAvg class.

test\_avg = TestAvg()

Graphics Using Turtle

Drawing shapes

Turtle contains methods for mving pe, pen size, lifting, and putting pen down.

**SIMPLESHAPES.py**

import turtle

turtle.pensize(3)

turtle.penup()

turtle.goto(-200, -50)

turtle.pendown()

turtle.circle(40, steps = 3) # Draw a triangle

turtle.penup()

turtle.goto(-100, -50)

turtle.pendown()

turtle.circle(40, steps = 4) # Draw a square

turtle.penup()

turtle.goto(0, -50)

turtle.pendown()

turtle.circle(40, steps = 5) # Draw a pentagon

turtle.penup()

turtle.goto(100, -50)

turtle.pendown()

turtle.circle(40, steps = 6) # Draw a hexagon

turtle.penup()

turtle.goto(200, -50)

turtle.pendown()

turtle.circle(40) # Draw a circle

turtle.done()

import turtle

turtle.pensize(3) # Set pen thickness to 3 pixels

turtle.penup() # Pull the pen up

turtle.goto(-200, -50)

turtle.pendown() # Pull the pen down

turtle.begin\_fill() # Begin to fill color in a shape

turtle.color("red")

turtle.circle(40, steps = 3) # Draw a triangle

turtle.end\_fill() # Fill the shape

turtle.penup()

turtle.goto(-100, -50)

turtle.pendown()

turtle.begin\_fill() # Begin to fill color in a shape

turtle.color("blue")

turtle.circle(40, steps = 4) # Draw a square

turtle.end\_fill() # Fill the shape

turtle.penup()

turtle.goto(0, -50)

turtle.pendown()

turtle.begin\_fill() # Begin to fill color in a shape

turtle.color("green")

turtle.circle(40, steps = 5) # Draw a pentagon

turtle.end\_fill() # Fill the shape

turtle.penup()

turtle.goto(100, -50)

turtle.pendown()

turtle.begin\_fill() # Begin to fill color in a shape

turtle.color("yellow")

turtle.circle(40, steps = 6) # Draw a hexagon

turtle.end\_fill() # Fill the shape

turtle.penup()

turtle.goto(200, -50)

turtle.pendown()

turtle.begin\_fill() # Begin to fill color in a shape

turtle.color("purple")

turtle.circle(40) # Draw a circle

turtle.end\_fill() # Fill the shape

turtle.color("green")

turtle.penup()

turtle.goto(-100, 50)

turtle.pendown()

turtle.write("Cool Colorful Shapes",

font = ("Times", 18, "bold"))

turtle.hideturtle()

turtle.done()