|  |
| --- |
| Experiment No. 8 |
| Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes |
| Date of Performance: 18/03/2024 |
| Date of Submission: 25/03/2024 |

**Experiment No. 8**

**Title:** Creating GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes

**Aim:** To study and create GUI with python containing widgets such as labels, textbox, radio, checkboxes and custom dialog boxes

**Objective:** To introduce GUI, TKinter in python

**Theory:**

Python offers multiple options for developing GUI (Graphical User Interface). Out of all the GUI methods, tkinter is the most commonly used method. It is a standard Python interface to the Tk GUI toolkit shipped with Python. Python with tkinter is the fastest and easiest way to create the GUI applications. Creating a GUI using tkinter is an easy task.

To create a tkinter app:

Importing the module – tkinter

Create the main window (container)

Add any number of widgets to the main window

Apply the event Trigger on the widgets.

Importing tkinter is same as importing any other module in the Python code. Note that the name of the module in Python 2.x is ‘Tkinter’ and in Python 3.x it is ‘tkinter’.

**Program:**

import tkinter as tk

from tkinter import messagebox

def show\_custom\_dialog():

messagebox.showinfo("Custom Dialog", "This is a custom dialog box!")

def submit():

# Retrieve values from entry widgets

name = name\_entry.get()

email = email\_entry.get()

age = age\_entry.get()

gender = gender\_var.get()

# Display the retrieved values

result\_text = f"Name: {name}\nEmail: {email}\nAge: {age}\nGender: {gender}"

result\_label.config(text=result\_text)

# Create the main window

root = tk.Tk()

root.title("GUI Example")

# Create entry widgets for name, email, and age

name\_label = tk.Label(root, text="Name:")

name\_label.grid(row=0, column=0, sticky="w")

name\_entry = tk.Entry(root)

name\_entry.grid(row=0, column=1)

email\_label = tk.Label(root, text="Email:")

email\_label.grid(row=1, column=0, sticky="w")

email\_entry = tk.Entry(root)

email\_entry.grid(row=1, column=1)

age\_label = tk.Label(root, text="Age:")

age\_label.grid(row=2, column=0, sticky="w")

age\_entry = tk.Entry(root)

age\_entry.grid(row=2, column=1)

# Create radio buttons for gender

gender\_label = tk.Label(root, text="Gender:")

gender\_label.grid(row=3, column=0, sticky="w")

gender\_var = tk.StringVar()

gender\_var.set("Male")

male\_radio = tk.Radiobutton(root, text="Male", variable=gender\_var, value="Male")

male\_radio.grid(row=3, column=1, sticky="w")

female\_radio = tk.Radiobutton(root, text="Female", variable=gender\_var, value="Female")

female\_radio.grid(row=3, column=2, sticky="w")

# Create a button to submit the form

submit\_button = tk.Button(root, text="Submit", command=submit)

submit\_button.grid(row=4, column=0, columnspan=3)

# Create a label to display the submitted values

result\_label = tk.Label(root, text="")

result\_label.grid(row=5, column=0, columnspan=3)

# Create a button to show custom dialog

custom\_dialog\_button = tk.Button(root, text="Show Custom Dialog", command=show\_custom\_dialog)

custom\_dialog\_button.grid(row=6, column=0, columnspan=3)

# Run the Tkinter event loop

root.mainloop()

**Output:**



**Conclusion:**

Through Experiment No. 8, the process of creating GUI applications using Python's tkinter library was explored comprehensively. Various widgets such as labels, textboxes, radio buttons, checkboxes, and custom dialog boxes were utilized to develop an interactive form. This experiment served as an effective introduction to GUI development in Python, showcasing its versatility and ease of use. Overall, the hands-on experience provided valuable insights into the capabilities of tkinter and its practical applications in building graphical interfaces for Python programs.