Python Programming
Unit 04 – Lecture 02 Notes
Layout Managers, Menus, Dialogs, Message Boxes

Tofik Ali

February 14, 2026

Contents

1 Lecture Overview

After learning widgets, the next challenge is **layout**: placing widgets in a clean and consistent way. This lecture covers:

- `pack` and `grid` layout managers,
- using Frames to structure large GUIs,
- building menus, and
- dialogs and message boxes.

2 Core Concepts

2.1 pack

`pack` places widgets relative to each other (top/bottom/left/right). It is great for simple layouts.

```
tk.Label(root, text="Title").pack(pady=10)
tk.Button(root, text="OK").pack()
```

Common options:

- `side`: `"top"`, `"bottom"`, `"left"`, `"right"`
- `fill`: `"x"`, `"y"`, `"both"`
- `expand`: allow expanding in extra space
- `padx/pady`: spacing

2.2 grid

`grid` places widgets in rows and columns. It is best for forms.

```
tk.Label(root, text="Name").grid(row=0, column=0, sticky="w")
tk.Entry(root).grid(row=0, column=1, padx=6)
```

Useful options:

- `sticky`: alignment inside the cell (`"nsew"`)
- `padx/pady`: spacing
- `columnspan/rowspan`: span multiple cells

Rule: Do not mix `pack` and `grid` in the same container widget. (You can use both if they are used in different Frames.)

2.3 Frames (Containers)

Frames are used to group widgets. A common pattern:

- Header Frame (title)
- Form Frame (input fields)
- Buttons Frame (submit/cancel)

2.4 Menus

Menus provide standard navigation.

```
menubar = tk.Menu(root)
file_menu = tk.Menu(menubar, tearoff=0)
file_menu.add_command(label="Exit", command=root.destroy)
menubar.add_cascade(label="File", menu=file_menu)
root.config(menu=menubar)
```

2.5 Message Boxes and File Dialogs

Message boxes provide feedback. File dialogs allow file selection.

```
from tkinter import messagebox, filedialog

messagebox.showinfo("Saved", "Done")
path = filedialog.askopenfilename(title="Select a file")
```

3 Demo Walkthrough

File: `demo/layout_menus_dialogs_demo.py`
This demo illustrates:

- grid-based layout for a small form,
- a menu bar with File/Help,
- a message box on submit,
- and selecting a file using a file dialog.

4 Interactive Checkpoints (with Solutions)

Checkpoint 1 Solution

Question: Why not mix `pack` and `grid` in the same container?
Answer: Tkinter can raise errors and produce unpredictable layout behavior. Use one layout manager per container (Frame/root), but you can use different managers in different Frames.

Checkpoint 2 Solution

Question: Purpose of Frame?
Answer: A Frame groups widgets, improves structure, and allows using separate layout strategies inside different UI sections.

5 Practice Exercises (with Solutions)

Exercise 1: Create a Simple Form

Task: Create a GUI with two fields (Name, Age) using `grid` and a Submit button.
Solution (outline):

```
import tkinter as tk

root = tk.Tk()

tk.Label(root, text="Name").grid(row=0, column=0, sticky="w")
tk.Entry(root).grid(row=0, column=1)

tk.Label(root, text="Age").grid(row=1, column=0, sticky="w")
tk.Entry(root).grid(row=1, column=1)
```

```
tk.Button(root, text="Submit").grid(row=2, column=0, columnspan=2)
root.mainloop()
```

Exercise 2: Add an Exit Menu

Task: Add a menu item `File -> Exit`.

Solution (idea):

```
menubar = tk.Menu(root)
file_menu = tk.Menu(menubar, tearoff=0)
file_menu.add_command(label="Exit", command=root.destroy)
menubar.add_cascade(label="File", menu=file_menu)
root.config(menu=menubar)
```

6 Exit Question (with Solution)

Question: Show info message box titled `"Saved"` with message `"Done"`.

Solution:

```
from tkinter import messagebox
messagebox.showinfo("Saved", "Done")
```