Complete Guide to Building a GUI Calculator App in Python with customtkinter

This guide will help you understand and build a GUI-based calculator app for PC using Python and the customtkinter library. It covers all the programming and UI concepts you need to learn step-by-step, explains the full example code, and provides a learning checklist.

Step-by-Step Learning Roadmap

1. Python Fundamentals:

- Variables, data types, functions, control flow, exception handling, OOP

2. Tkinter Basics:

- Creating windows, buttons, labels
- Using pack and grid for layouts
- Handling events and StringVar

3. CustomTkinter Library:

- Installing and using customtkinter
- Working with CTk widgets and themes

4. Expression Handling:

- Building calculator expressions as strings
- Replacing custom symbols (×, ÷, ^) for evaluation
- Using eval() safely with error handling

5. Event Binding:

- Keyboard input bindings

- Button commands

6. App Structure:

- Organizing code into a class

- Maintaining state and UI updates

7. Advanced Features:

- History view

- Theme toggle (light/dark mode)

- Special functions (square root, percentage)

Code Explanation and Guide

Your calculator app uses a class CalculatorApp inherited from ctk.CTk.

Key components:

- __init__: Initializes window, variables, and calls UI setup

- create_widgets: Builds main UI parts including display, buttons, history box

- build_buttons: Dynamically creates all calculator buttons with colors and commands

- press_key: Updates the expression string when user inputs

- evaluate: Calculates the result using Python's eval after replacing symbols

- do_square_root, do_percentage: Special operations

- toggle_history_view: Shows/hides the history panel

- flip_theme: Switches between light and dark mode and rebuilds buttons

- clear_all, delete_last: Editing the current expression

- handle_keypress: Maps keyboard keys to calculator functions

The buttons grid is created using grid geometry manager for a responsive layout.

The app maintains state with self.expression and self.history_shown.

History of calculations is stored in a CTkTextbox.

What to Learn & Practice

- Python basics and OOP: variables, functions, exceptions, classes
- Tkinter fundamentals: windows, buttons, labels, layout managers
- CustomTkinter: widgets, theming, appearance modes
- String manipulation and safe evaluation of expressions
- Event handling: button commands and keyboard bindings
- GUI app structure: class-based design, state management
- Handling UI updates dynamically
- Debugging and error handling
- Reading and understanding existing code

Practice building small projects incrementally:

1. Console calculator
2. Basic Tkinter GUI with buttons and labels
3. CustomTkinter themed window
4. Calculator with buttons and evaluation
5. Add keyboard support
6. Add history and theme toggle
7. Organize code into a class

Resources & Tips

- Automate the Boring Stuff with Python (free book)
- Real Python tutorials on OOP and Tkinter
- CustomTkinter GitHub repo and docs
- Stack Overflow for Q&A
- YouTube tutorials on Python GUI (freeCodeCamp, Tech With Tim)
- Practice small coding exercises regularly

Tip: Start small, test often, and gradually add features.

Use version control like Git to track your progress.