Creating a Simple Calculator Application Using Python

[1] Dimas Ridho Aldiansyah*, [2]Aidil Adhan, [3] Muhammad Iqbal Hanafi, [4]M. Azmer Fahrezi, [5]Siti Kamilah

[1-5] Science and Informatics, Universitas Pertiba

*Corresponding Author:
dimas28082004@gmail.com

Abstract

In the digital era, the use of computer-based applications is increasingly growing to help with daily activities. One of the applications that is often used is a calculator. This research aims to design and build a simple calculator application using Python with the Tkinter library as a graphical interface (GUI). The methods used in this study include designing user interfaces, implementing basic mathematical operation functions, and testing applications to ensure the accuracy of calculations. The result of this research is a calculator application that can perform addition, subtraction, multiplication, and division operations with a user-friendly interface. This study contributes to learning Python programming for beginners as well as as the basis for further application development.

***Keywords**: Calculator, Python, Tkinter, GUI, Programming*

1. INTRODUCTION

Calculators are a very important tool in daily life to help with mathematical calculations. With the development of technology, calculators can now be implemented in the form of software-based applications (Lutz, 2013). Python is a popular programming language that is easy to learn, especially for beginner students in the field of computer systems engineering. One of the libraries that supports GUI-based application development in Python is Tkinter (Grayson, 2000).

In the world of programming, the utilization of GUIs in applications is essential to improve user interaction. Users prefer apps that have an attractive and easy-to-use look compared to text-based apps in the terminal. Therefore, the use of the Tkinter library in Python is the main choice in the development of this simple calculator application.

GUI-based programming provides a better user experience compared to the use of terminal-based calculators. Tkinter provides various components such as buttons, labels, and text boxes that make it easy to create interactive graphical interfaces (Martelli, 2006). In addition, Python as a versatile programming language has a simple syntax and extensive supporting libraries, making it suitable for the development of small to large-scale applications (Rossum, 2009).

The use of the Python programming language in software development has grown tremendously in recent years. Python's advantages in terms of ease of syntax, community support, and the large number of libraries available make it the main choice for creating desktop and web-based applications (Zelle, 2017). Tkinter itself has been used in various interactive applications because of its lightweight nature and compatibility with various operating systems (Hunt, 2018).

Previous research has shown that the use of Python and Tkinter in GUI-based application development can improve software development efficiency and make it easier for users to operate applications (Abdu & Bayu, 2010). Therefore, this study aims to design and develop a simple calculator application using Python, which can be used as a basic calculation tool as well as a means of learning programming (Sweigart, 2019). In addition, with the

growing need for efficient and easy-to-use applications, this research is expected to provide insights into how to develop GUI-based software by utilizing available technologies.

This research aims to design and develop a simple calculator application using Python with Tkinter as the main library in creating graphical interfaces. This study also evaluates the ease of use of the application and the effectiveness of the calculations carried out in this application.

2. METHOD

The research method used in the manufacture of this calculator application includes several stages as follows (Rossum, 2009):

User Interface Design

At this stage, the app's appearance is designed to have a simple but still attractive layout. Some of the key components in designing a user interface are:

1. Labels as pointers to the number display area
2. Entry Box to display the input number
3. Button for calculation operations such as addition, subtraction, multiplication, and division

Implementation of Calculation Function

The main functions in this application include:

1. Summing (+)
2. Reduction (-)
3. Multiplication (×)
4. Divide (÷)
5. Reset Input

The program code is written in Python using the Tkinter library to display the GUI and mathematical functions to execute the following calculations:

Testing and Evaluation

Testing is carried out in several scenarios, namely:

1. Enter positive and negative numbers to ensure the app can perform the operation correctly.
2. Test different combinations of calculations to see the accuracy of the results.
3. Ensure that your app doesn't get an error when it receives empty or invalid input.

3. RESULTS AND DISCUSSION

The result of this research is a GUI-based calculator application with the following main features:

1. Simple and easy-to-use interface – The app is designed with a minimalist interface so that users can quickly understand its functions without the need for special training (Dumas, 2003).
2. Ability to perform basic mathematical operations accurately – This calculator can handle addition, subtraction, multiplication, and division operations with accurate results and quick response to user input (Hunt, 2018).
3. Implementation using Python with Tkinter libraries for interface management – Utilizing Tkinter allows the creation of GUIs that are interactive, flexible, and can be run across a variety of operating systems without the need for many additional configurations (Wilkinson, 2011).

Interface Implementation

After the design process, the application has a simple display with the components of the operation button, input screen, as well as a button to delete input. Users can easily perform calculations by pressing a button according to the desired operation.

The program code is written in Python using the Tkinter library to display the GUI and mathematical functions to execute the following calculations:

```
from functools import partial
import tkinter as tk
class appCalculator(tk. Kindergarten):
    def __init__(self):
        tk.Tk.__init__ (self)
        self.title("Smart calculator")
        self.createButton()
        self.determinant = False

    def createButton(self):
        self.screen = kindergarten. Entry(self, width=25)
        self.screen.grid(row=0, column=0, columnspan=5)

        btn_list = [
            '1', '2', '3',
            '4', '5', '6',
            '7', '8', '9',
            '0', '+', '-',
            'C', '/', '*',
            '='
        ]
        row = 1
        column = 0
        for penacloading in btn_list:
            command = partial(self.calculate, container)
            if container == '=':
                kindergarten. Button(self, text='=', width=22, command=command).grid(row=row,
column=column, columnspan=5)
            else:
                kindergarten. Button(self, text=container, width=5,
command=command).grid(row=row, column=column)
            Column += 1
            if column > 2:
                column = 0
                line += 1
    def calculate(self, key):
        if key == '=':
            self.determinant = True
            try:
                result = eval(self.layar.get())
                self.layar.delete(0, tk. END)
                self.layar.insert(tk. END, str(result))
            except:
                self.layar.insert(tk. END, "-> Error!")
        elif key == 'C':
            self.layar.delete(0, tk. END)
        else:
            if self.determinant :
```

```
            self.layar.delete(0, tk. END)
            self.determinant = False
        self.layar.insert(tk. END, key)

call = applicationCalculator()
call.mainloop()
```

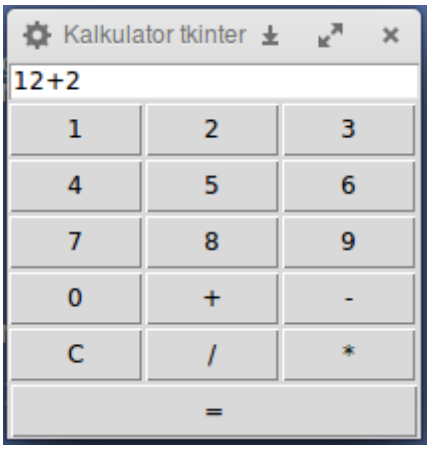

Figure 1. Simple Calculator App Display

Evaluation of Calculation Functions

Testing shows that the app can perform calculations correctly in a variety of scenarios. Here are the test results:

Table 1. Testing

Input 1	Operator	Input 2	Result
10	+	5	15
20	-	7	13
6	×	3	18
15	÷	3	5

In addition, this study provides insight for beginner students in understanding the basic concepts of GUI-based programming and the importance of input validation in the development of calculator-based applications (Guttag, 2013).

4. CONCLUSION

1. A simple calculator application was successfully developed using Python and Tkinter.
2. The app is capable of performing basic math operations with a user-friendly interface.
3. The use of Tkinter makes it easy to develop a simple but effective GUI.
4. This research can be used as a basis for the development of a calculator application with further features such as trigonometric functions or storage of calculation history.

ACKNOWLEDGMENTS

Thank you to those who have supported this research, including supervisors and fellow students who provided input during the development process.

References

Grayson, J. (2000). Python and Tkinter Programming. Manning Publications.

Hunt, J. (2018). Advanced Guide to Python 3 Programming. Springer.

Lutz, M. (2013). Learning Python. O'Reilly Media.

Rossum, G. V. (2009). The Python Language Reference Manual. Network Theory Ltd.

Zelle, J. (2017). Python Programming: An Introduction to Computer Science. Franklin, Beedle & Associates.