

Building Modern GUIs with tkinter and Python

Building user-friendly GUI applications with ease

Saurabh Chandrakar
Dr. Nilesh Bhaskarrao Bahadure



www.bpbonline.com

Copyright © 2023 BPB Online

All rights reserved. No part of this book may be reproduced, stored in a retrieval system, or transmitted in any form or by any means, without the prior written permission of the publisher, except in the case of brief quotations embedded in critical articles or reviews.

Every effort has been made in the preparation of this book to ensure the accuracy of the information presented. However, the information contained in this book is sold without warranty, either express or implied. Neither the author, nor BPB Online or its dealers and distributors, will be held liable for any damages caused or alleged to have been caused directly or indirectly by this book.

BPB Online has endeavored to provide trademark information about all of the companies and products mentioned in this book by the appropriate use of capitals. However, BPB Online cannot guarantee the accuracy of this information.

First published: 2023

Published by BPB Online

WeWork

119 Marylebone Road

London NW1 5PU

UK | UAE | INDIA | SINGAPORE

ISBN 978-93-55518-569

www.bpbonline.com

Dedicated to

My Parents

Dr Surendra Kumar Chandrakar

and

Smt. Bhuneshwari Chandrakar

Brother Shri Pranav Chandrakar

to my wife Priyanka Chandrakar

and to my lovely son Yathartha Chandrakar

- Saurabh Chandrakar

My Parents

Smt. Kamal B. Bahadure

and

Late Bhaskarrao M. Bahadure

to my in-laws

Smt. Saroj R. Lokhande and Shri. Ravikant A. Lokhande

and to my wife Shilpa N. Bahadure

and to beautiful daughters Nishita and Mrunmayee

And to all my beloved students.

- Dr. Nilesh Bhaskarrao Bahadure

About the Authors

- **Saurabh Chandrakar** is a Research & Development Engineer (Dy. Manager) at Bharat Heavy Electricals Limited (BHEL) Hyderabad. He is the winner of the best executive award on Operations Division by BHEL Hyderabad. Recently, he has been awarded the prestigious BHEL Excellence Award under Anusandhan category for Redundant Composite Monitoring System of Power Transformers project. He has 20 copyrights and 1 patent granted. Additionally, he has 6 patents filed. Moreover, he has published 3 books in reputed publications such as BPB New Delhi (Programming Techniques using Python), Scitech Publications Chennai (Programming Techniques using matlab) and IK International publishers (Microcontrollers and Embedded System Design). He has also launched 1 video course on BPB titled “First Time Play with Basic, Advanced Python concepts and complete guide for different python certification exams all in one umbrella.”
- **Nilesh Bhaskarrao Bahadure** received his Bachelor of Engineering degree in Electronics Engineering in 2000, his Master of Engineering degree in Digital Electronics in 2005, and the Ph.D. degree in Electronics in 2017 from KIIT Deemed to be University, Bhubaneswar, India. He is currently an Associate Professor in the Department of Computer Science and Engineering at Symbiosis Institute of Technology (SIT), Nagpur, Symbiosis International (Deemed University) (SIU), Pune, Maharashtra, India. He has more than 20 years of experience. Dr. Bahadure is a life member of IE(I), IETE, ISTE, ISCA, SESI, ISRS, and IAENG professional organizations. He has published more than 40 articles in reputed international journals and conferences, and has 5 books to his credit. He is the reviewer of many indexed journals such as IEEE Access, IET, Springer, Elsevier, Wiley and so on. His research interests are in the areas of Sensor Technology, the Internet of Things, and Biomedical Image Processing.

About the Reviewer

Dr. Prasenjeet Damodar Patil received B.E in E&TC Engineering from Sant Gadgebaba Amravati University and M. Tech. from Walchand College of Engineering Sangli, India. He did his Ph.D. degree in E&TC Engineering from Sant Gadgebaba Amravati University. He has 14+ years of teaching experience. Currently, he is working as Associate Professor at School of Computing, M.I.T A.D.T University, Pune. He has published more than 15 papers in reputed Journals. His research interest includes Computational Electromagnetics applications in Integrated Optics, IoT & Digital Image Processing.

Acknowledgements

- First and foremost, I would like to thank you all for selecting this book. It has been written with the beginner reader in mind. First of all, I take this opportunity to greet and thank my mentor Prof. Nilesh Bahadure Sir for motivating me and always communicating his expertise fully on topics related to Python. I am very thankful for being his protégé. I appreciate his belief in me, for always standing behind me and pushing me to achieve more. The phrase "Journey of Thousand Miles Begins with a Single Step" is something he always reminds me of.

Thank you to my parents, Dr. Surendra Kumar Chandrakar and Smt. Bhuneshwari Chandrakar, my brother, Shri Pranav Chandrakar, my beloved wife, Mrs. Priyanka Chandrakar, my adorable son Yathartha Chandrakar, and all of my friends have inspired me and given me confidence over the years. Last but not least, I would like to express my sincere gratitude to the staff at BPB Publications for their contributions and insights that made parts of this book possible.

- *Saurabh Chandrakar*

- It was my privilege to thank Dr. S. B. Mujumdar, Chancellor of the Symbiosis International University, Pune, and Shri. Vijay Kumar Gupta, Chairman of Beekay Industries Bhilai and BIT Trust, for his encouragement and support. I would like to thank my mentors Dr. Arun Kumar Ray, Dean, School of Electronics Engineering, KIIT Deemed to be University, Bhubaneswar, and Dr. Anupam Shukla, Director, SVNIT Surat. I would like to thank Dr. Vidya Yeravdekar, Principal Director of Symbiosis Society, and the Pro-Chancellor of Symbiosis International University, Pune, Dr. Rajani R. Gupte, Vice Chancellor of the Symbiosis International University, Pune, Dr. Ketan Kotecha, Dean, Faculty of Engineering, Symbiosis International University, Pune, and Dr. Mukesh M. Raghuwanshi, Director, SIT Nagpur, for their advice, and encouragement throughout the preparation of the book.

I would also like to thank Dr. Sanjeev Khandal, HOD, Department of Aeronautical Engineering, SGU Kolhapur, my well-wisher Dr. Prasenjeet D. Patil, Associate Professor, MIT ADT University, Pune, and my colleagues in

Symbiosis Institute of Technology Nagpur for providing valuable suggestions and lots of encouragement throughout the project.

I am thankful to Prof. Dr. N. Raju, Sr. Assistant Professor, SASTRA University, Thanjavur, Tamil Nadu, for his support, assistance during writing, and for his valuable suggestions.

I would also like to thank Dr. Ravi M. Potdar, Sr. Associate Professor, BIT Durg, and Dr. Md. Khwaja Mohiddin, Associate Professor, BIT Raipur for providing valuable suggestions and lots of encouragement throughout the project. Writing a beautiful, well-balanced, and acquainted book is not a work of one or two days or a month; it takes a lot of time and patience, as well as hours of hard work. Many thanks to my family members, parents, wife, children, and well-wishers for their kind support. Without them and their faith and support, writing this classic book would have remained just a dream. I also like to thank my students, who have always been with me, for relating problems and finding solutions too. Perfection in any work does not come in a day. It needs a lot of effort, time and hard work, and sometimes, proper guidance.

It is my privilege to thank Prof. (Dr.) Ram Dhekekar, Professor, Department of Electronics & Telecommunication Engineering, SSGMCE Shegaon, and Dr. C. G. Dethé, Director UGC Staff College Nagpur. Last, but not least, I would like to offer an extra special thanks to the staff at "BPB Publications" for their insight and contribution to polishing this book.

Most significantly, I want to thank Lord Ganesh for all of the work I was able to put into the book's preparation. I would not be as zealous as I am now if it weren't for God's amazing creation of the universe.

"For since the creation of the world God's invisible qualities - his eternal power and divine nature - have been clearly seen, being understood from what has been made, so that men are without excuse."

-Dr. Nilesh Bhaskarrao Bahadure

Preface

The purpose of this book is to introduce readers with little to no programming experience, to Python Graphical User Interface (GUI). A GUI application can be created in any programming language, say VB.Net, C#.Net etc. In this book, we shall see how to create a GUI application using Python tkinter library. We will provide the readers with the foundational knowledge and skills which is required to start writing code for creating any desktop GUI app in Python language. By mastering Python tkinter library, readers will be able to apply this technology to solve real-world problems and create various useful applications according to their needs.

The first part of the book covers basic GUI tkinter concepts followed by a touch of inbuilt variable classes for creating different tkinter GUI widgets. Then we shall see some insights of different widgets viz button, input, display, container, item and user-interactive widgets. Finally, in the later part of the book, we shall explore handling file selection and getting widget along with trace information in tkinter.

This book covers a wide range of topics, from basic definition of different widgets along with various solved examples and well explanatory code. Overall, the book provides a solid foundation for beginners to start their journey for getting trained in python GUI using tkinter library.

This book is divided into **11 chapters**. Each chapter description is listed as follows.

Chapter 1: tkinter Introduction – will cover the basic GUI example of creating a parent window along with its size maximizing by adjusting the width or height. It will also introduce about each standard attribute of python tkinter GUI, which is, dimensions, colors, fonts, cursors and so on. The concept of tkinter geometry manager with examples will be covered. Finally, we will learn how to access and set pre-defined variables sub classed from the tkinter variable class viz StringVar, IntVar, DoubleVar and BooleanVar.

Chapter 2: Inbuilt Variable Classes for Python tkinter GUI Widgets – will cover the concept of creating of a simple GUI windows app using classes and objects concepts.

Chapter 3: Getting Insights of Button Widgets in tkinter – is dedicated to the concept of dealing with one of the most commonly used GUI widgets viz tkinter Button widget. We will view the binding of events to the above widget with multiple examples and different methods, including lambda expressions. Next, we shall see the Checkbutton widget which will give the provision to the user to select more than one option. The user will also view different options to get the image in the above widget. Next, we will see how to use tkinter Radiobutton widget. The user will see different examples where exactly one of the predefined set of options will be chosen. Last but not the least, we will explore tkinter OptionMenu widget where the user views how a pop menu and button widget will be created for an individual option selection from a list of options.

Chapter 4: Getting Insights of Input Widgets in tkinter – is dedicated to cover the concept of creating a simple GUI app using tkinter Entry widget with very neat way of various options explanations, followed by different solved examples. Moreover, the validation concept in Entry widget is very neatly explained. Next, we shall see about scrollbar widget where user will look into the scrolling capability in vertical or horizontal direction with different widgets such as List Box, Entry and Text. Another one is tkinter Spinbox widget, where the range of input values will be fed to the user, out of which, the user can select the one. Next, we will be looking into how to implement a graphical slider to any Python application program by using tkinter Scale widget. Next is the concept of tkinter Text widget where the user can insert multiple text fields. Finally, we will be dealing with tkinter Combobox widget and its applications.

Chapter 5: Getting Insights of Display Widgets in tkinter – is dedicated to the concept of creating a simple GUI app using tkinter Label widget, which depicts the ways of displaying a text or image on a window form. We shall also learn about the display of prompt unedited text messages to the user with tkinter Message widget. Moreover, we will look into multiple message boxes like information, warning, error and so on, in a Python application by using tkinter MessageBox widget.

Chapter 6: Getting Insights of Container Widgets in tkinter – is dedicated to the concept of tkinter Frame widget, where the user can arrange different widgets position, can provide padding, can be used as geometry manager for other widgets and so on. We shall look into the variant of Frame widget, which is tkinter LabelFrame and is a container for complex window layouts. The user will be able to see frame features along with label display. Moreover, we shall view creating tabbed widget with the help of using tkinter Notebook widget. Here, user can

select different pages of contents by clicking on tabs. The importance of tkinter PanedWindow widget will be explored where multiple examples will be seen containing horizontal or vertical stack of child widgets. Finally, we will look into tkinter Toplevel widget where the concepts are being explained for the creation and display of top level windows.

Chapter 7: Getting Insights of Item Widgets in tkinter – is dedicated to tkinter Listbox widget where user can display different types of list of items and a number of items can be selected from the list. Different selectmode examples will be seen along with scrollbar attached to the above widget.

Chapter 8: Getting Insights of tkinter User Interactive Widgets – focuses on the user to create different menus such as pop-up, top level and pull -down menu with the help of tkinter Menu widget. The user can create different applications such as Notepad, Wordpad, any management software and so on. Moreover, we will explore drop-down menu widget which is associated with a Menu widget called tkinter Menubutton widget, which can display the choices when user clicks on the above Menubutton. User can add checkbutton or radiobutton with the help of above Menubutton. Finally, we will study the concepts of drawing different graphics like line, rectangle and so on, with the help of tkinter Canvas widget.

Chapter 9: Handling File Selection in tkinter – will handle file selection with different dialogs for opening a file, saving a file and so on, with the help of multiple examples using various Python examples.

Chapter 10: Getting Widget Information and Trace in tkinter – is dedicated to getting widget information and different trace methods viz trace_add, trace_remove, trace_info and so on, using various Python examples.

Chapter 11: UserLogin Project in tkinter GUI Library with sqlite3 Database – will cover an application created using tkinter library along with interacting with sqlite3 database.

Code Bundle and Coloured Images

Please follow the link to download the *Code Bundle* and the *Coloured Images* of the book:

<https://rebrand.ly/dq4ctt8>

The code bundle for the book is also hosted on GitHub at **<https://github.com/bpbpublications/Building-Modern-GUIs-with-tkinter-and-Python>**. In case there's an update to the code, it will be updated on the existing GitHub repository.

We have code bundles from our rich catalogue of books and videos available at **<https://github.com/bpbpublications>**. Check them out!

Errata

We take immense pride in our work at BPB Publications and follow best practices to ensure the accuracy of our content to provide with an indulging reading experience to our subscribers. Our readers are our mirrors, and we use their inputs to reflect and improve upon human errors, if any, that may have occurred during the publishing processes involved. To let us maintain the quality and help us reach out to any readers who might be having difficulties due to any unforeseen errors, please write to us at :

errata@bpbonline.com

Your support, suggestions and feedbacks are highly appreciated by the BPB Publications' Family.

Did you know that BPB offers eBook versions of every book published, with PDF and ePub files available? You can upgrade to the eBook version at www.bpbonline.com and as a print book customer, you are entitled to a discount on the eBook copy. Get in touch with us at :

business@bpbonline.com for more details.

At **www.bpbonline.com**, you can also read a collection of free technical articles, sign up for a range of free newsletters, and receive exclusive discounts and offers on BPB books and eBooks.

Piracy

If you come across any illegal copies of our works in any form on the internet, we would be grateful if you would provide us with the location address or website name. Please contact us at business@bpbonline.com with a link to the material.

If you are interested in becoming an author

If there is a topic that you have expertise in, and you are interested in either writing or contributing to a book, please visit www.bpbonline.com. We have worked with thousands of developers and tech professionals, just like you, to help them share their insights with the global tech community. You can make a general application, apply for a specific hot topic that we are recruiting an author for, or submit your own idea.

Reviews

Please leave a review. Once you have read and used this book, why not leave a review on the site that you purchased it from? Potential readers can then see and use your unbiased opinion to make purchase decisions. We at BPB can understand what you think about our products, and our authors can see your feedback on their book. Thank you!

For more information about BPB, please visit www.bpbonline.com.

Join our book's Discord space

Join the book's Discord Workspace for Latest updates, Offers, Tech happenings around the world, New Release and Sessions with the Authors:

<https://discord.bpbonline.com>



Table of Contents

1. tkinter Introduction	1
Introduction.....	1
Structure.....	1
Objectives.....	2
Introduction to tkinter.....	2
Basic Python GUI program	4
Some standard attributes of Python tkinter GUI.....	8
<i>Dimensions</i>	8
<i>borderwidth</i>	8
<i>highlightthickness</i>	8
<i>padX, padY</i>	9
<i>wrplength</i>	10
<i>height</i>	10
<i>underline</i>	10
<i>width</i>	11
Colors	11
<i>activebackground</i>	11
<i>background</i>	11
<i>activeforeground</i>	12
<i>foreground</i>	12
<i>disabledforeground</i>	13
<i>highlightbackground</i>	14
<i>selectbackground</i>	15
<i>selectforeground</i>	15
Fonts	16
<i>By creating a font object</i>	16
<i>By using tuple</i>	19
Anchors	19
<i>Placing widget position when anchor = N</i>	20

<i>Placing widget position when anchor = S</i>	21
<i>Placing widget position when anchor = E</i>	21
<i>Placing widget position when anchor = W</i>	22
<i>Placing widget position when anchor = NE</i>	22
<i>Placing widget position when anchor = NW</i>	22
<i>Placing widget position when anchor = SE</i>	23
<i>Placing widget position when anchor = SW</i>	23
<i>Placing widget position when anchor = CENTER</i>	23
Relief styles.....	24
Bitmaps.....	25
Cursors	26
Python tkinter geometry management.....	28
<i>pack()</i>	28
<i>grid()</i>	29
<i>place()</i>	32
<i>Geometry method in tkinter</i>	35
Conclusion.....	36
Points to remember	37
Questions	37
2. Inbuilt Variable Classes for Python tkinter GUI Widgets	39
Introduction.....	39
Structure.....	39
Objectives.....	40
Inbuilt variable classes.....	40
StringVar().....	40
BooleanVar().....	42
IntVar()	44
DoubleVar()	45
GUI creation using classes and objects.....	47
Conclusion.....	51
Points to remember	51
Questions	51

3. Getting Insights of Button Widgets in tkinter.....	53
Introduction.....	53
Structure.....	53
Objectives.....	54
tkinter Button Widget.....	54
<i>Events and bindings</i>	60
<i>event type</i>	60
tkinter Checkbutton widget.....	81
tkinter Radiobutton widget.....	91
tkinter OptionMenu widget.....	100
Conclusion.....	103
Points of remember	103
Questions	103
4. Getting Insights of Input Widgets in tkinter.....	105
Introduction.....	105
Structure.....	105
Objectives.....	106
tkinter Entry widget.....	106
<i>Validation in the Entry widget</i>	121
tkinter Scrollbar widget.....	126
<i>Scrollbar attached to Listbox</i>	127
<i>Scrollbar attached to Text</i>	128
<i>Scrollbar attached to Canvas</i>	130
<i>Scrollbar attached to Entry</i>	131
tkinter Spinbox widget	132
tkinter Scale widget.....	138
tkinter Text widget.....	145
tkinter Combobox Widget.....	164
Conclusion.....	173
Points to remember	173
Questions	174

5. Getting Insights of Display Widgets in tkinter.....	175
Introduction.....	175
Structure.....	175
Objectives.....	176
tkinter Label Widget.....	176
tkinter Message Widget.....	191
tkinter MessageBox Widget.....	193
<i>showinfo()</i>	194
<i>showwarning()</i>	195
<i>showerror()</i>	196
<i>askquestion()</i>	197
<i>askokcancel()</i>	198
<i>askyesno()</i>	199
<i>askretrycancel()</i>	200
Conclusion.....	201
Points of remember	201
Questions	202
6. Getting Insights of Container Widgets in tkinter.....	203
Introduction.....	203
Structure.....	204
Objectives.....	204
tkinter Frame Widget.....	204
tkinter LabelFrame Widget	209
tkinter Tabbed/Notebook Widget.....	212
tkinter PanedWindow widget.....	214
tkinter Toplevel widget.....	221
Conclusion.....	232
Points of remember	232
Questions	232
7. Getting Insights of Item Widgets in tkinter.....	235
Introduction.....	235

Structure.....	235
Objectives.....	235
tkinter Listbox widget.....	236
Conclusion.....	245
Points of remember	246
Questions	246
8. Getting Insights of tkinter User Interactive Widgets	247
Introduction.....	247
Structure.....	247
Objectives.....	248
tkinter Menu widget.....	248
tkinter Menubutton widget.....	263
tkinter Canvas widget.....	267
Conclusion.....	281
Points to remember	281
Questions	282
9. Handling File Selection in tkinter.....	283
Introduction.....	283
Structure.....	284
Objectives.....	284
Handling file selection in tkinter	284
Conclusion.....	299
Points of remember	300
Questions	300
10. Getting Widget Information and Trace in tkinter	301
Introduction.....	301
Structure.....	301
Objectives.....	302
Getting widget information	302
Trace in tkinter	308

<i>trace_add()</i>	309
<i>trace_remove()</i>	309
<i>trace_info()</i>	309
Conclusion.....	311
Points to remember	312
Questions	312
11. UserLogin Project in tkinter GUI Library with sqlite3 Database	313
Introduction.....	313
Structure.....	313
Objectives.....	313
GUI interaction with sqlite3 database	314
Displaying a GUI application	314
Conclusion.....	339
Points to remember	339
Questions.....	340
Index.....	341-345

CHAPTER 1

tkinter

Introduction

Introduction

We have learned in our previous book *Python for Everyone*, about the concepts in Python which are procedure oriented or object oriented. However, these concepts will be used as indispensable salt for our next learning, which is related to **Graphical User Interface (GUI)**. We are surrounded by GUI apps in day-to-day life. Whenever we are using our mobile phone/Desktop applications and accessing any app or software, the first thing which we look forward to is how to access these apps or software. Any app on mobile phone or a computer system consists of hardware and is controlled by an operating system. The high-level languages will be sitting on top of the operating system and Python is no exception. So, in this chapter, we will learn about tkinter library in Python.

Structure

In this chapter, we will discuss the following topics:

- Introduction to tkinter
- Basic Python GUI Program
- Some standard attributes of Python tkinter GUI

- Colors
- Fonts
- Anchors
- Relief Styles
- Bitmaps
- Cursors
- Python tkinter Geometry Manager

Objectives

By the end of this chapter, the reader will learn about creating basic GUI Python program using the tkinter library. In addition to that, we will also explore some standard attributes of Python tkinter GUI such as dimensions, colors or fonts with various options with examples. It is important to know how to position the text with a list of constants, relative to the reference point using anchor attribute. Moreover, we shall see how with the usage of relief attribute 3-D, simulated effects around the outside of the widget can be provided. We will also learn about bitmap and cursor attribute with examples. Finally, at the end of this chapter, we will learn accessing tkinter widgets using inbuilt layout geometry managers viz pack, grid and place.

Introduction to tkinter

Whenever we write any program in Python to control the hardware, Python will show the output with the help of operating system. However, if we desire to make an executable with the help of GUI, then just having the need of hardware and operating system is insufficient. Python requires some services which come from a number of resources and one such resource which is of interest to many Python programmers is Tcl/Tk. **Tcl** stands for **Tool Command Language** and it is a scripting language with its own interpreter. On the other hand, Tk is a toolkit for building GUIs. An important point to note is that Tcl/Tk is not Python and we cannot control and access the services of Tcl/Tk using Python. So, another package is introduced and is referred to as *tkinter*, and it is an intermediary between Python and Tcl/Tk. tkinter will allow us to use the services of Tcl/Tk using the syntax of Python. As Python code developers, we will not be directly concerned with the Tcl/Tk. Binding of Python to the Tk GUI toolkit will be done by tkinter. tkinter will make everything appear as an object. We can create GUI, knowing that we can regard the window as an object, a label place on window as an object and so on. Applications can be built from a view point of an object-oriented programming paradigm. All we need to make sure is that we write our code in such a way that it allows us access tkinter, as shown in the following *Figure 1.1*:



Figure 1.1: tkinter access hierarchy

So, GUI applications can be easily and quickly created when Python is combined with tkinter. Although Python offers multiple options for developing GUI such as PyQT, Kivy, Jython, WxPython, and pyGUI; tkinter is the most commonly used. In this book, we will focus only on the tkinter usage of GUI creation. Just like we import any other module, tkinter can be imported in the same way in Python code:

```
import tkinter
```

This will import the **tkinter** module.

The module name in Python 3.x is **tkinter**, whereas in Python 2.x, it is Tkinter.

More often, we can use:

```
from tkinter import *
```

Here, the '*' symbol means everything, as Python now can build Graphical User Interfaces by treating all of the widgets like Buttons, Labels, Menus and so on, as if they were objects. It is like importing **tkinter** submodule.

However, there are some important methods which the user needs to know while creating Python GUI application, and they are as follows:

- **Tk(screenName=None, baseName=None, className='Tk', useTk=1)**

tkinter offers this method in order to create a main window. For any application, a main window code can be created by using:

```
import tkinter
myroot = tkinter.Tk()
```

Here, **Tk** class is instantiated without any arguments. **myroot** is the main window object name. This method will allow blank parent window creation with close, maximize and minimize buttons on the top.

- `mainloop()`

tkinter offers this method when our application is ready to run. This method is an infinite loop used to run the application. It will wait for an event to occur and as long as the window is not closed, the event is processed.

Basic Python GUI program

Let us see a basic Python program which will create a window:

```
from tkinter import *

myroot = Tk() # creating an object of Tk class
# we should first know how to create a window if want to per-
# form graphics coding.
# but output window will not be displayed right now
myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.2*:

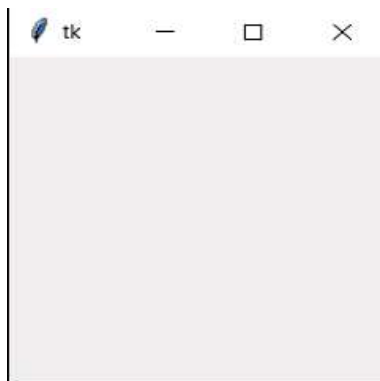
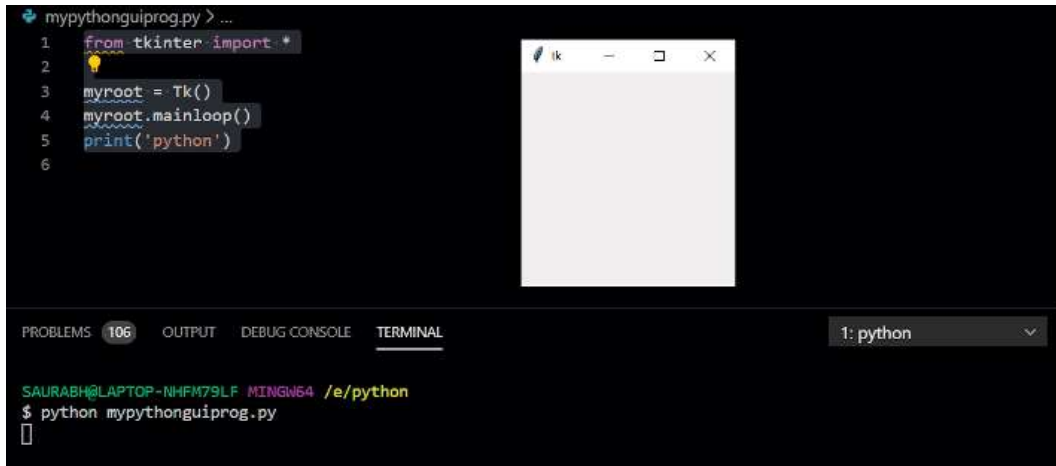


Figure 1.2: Output of Chap1_Example1.py

Note: The preceding code is covered in Program Name: `Chap1_Example1.py`

In the preceding code, we have imported tkinter submodule and created a parent widget which usually will be the main window of an application. A blank parent window is created with close, maximize and minimize buttons on the top. An infinite loop will be called to run the application, as long as the window is not closed.

The moment the window is closed, the statements after `myroot.mainloop()` will be executed, as shown in the following *Figure 1.3*:



The screenshot shows an IDE with a Python script on the left and a terminal window on the right. The script contains the following code:

```

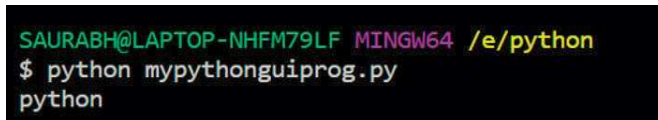
1 from tkinter import *
2
3 myroot = Tk()
4 myroot.mainloop()
5 print('python')
6

```

The terminal window shows the command `python mypythonguiprog.py` being executed, and the output `python` is displayed.

Figure 1.3: Output on running the code

The moment the window is closed by clicking on the 'X' mark, we will get the output as shown in *Figure 1.4*:



The terminal window shows the command `python mypythonguiprog.py` being executed, and the output `python` is displayed.

Figure 1.4: Execution of print statement after clicking 'X' mark

In the preceding example, we can maximize the window in both horizontal and vertical directions by using both height and width attributes. However, we can restrict the expansion of window in any direction.

Suppose we want to maximize the width only. Then, the code is as follows:

```

from tkinter import *

myroot = Tk()
myroot.resizable(width = True, height = False) # width can be maxi-
mized
myroot.mainloop()

```

Output:

The output can be seen in *Figure 1.5*:

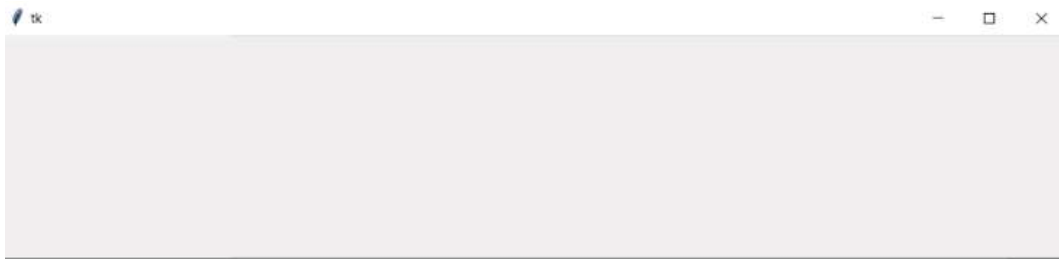


Figure 1.5: Output of Chap1_Example2.py

Note: The preceding code is covered in Program Name: Chap1_Example2.py

Suppose, we want to maximize the height only. Then the code is as follows:

```
from tkinter import *  
  
myroot = Tk()  
myroot.  
resizable(width = False, height = True) # height can be maximized  
myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.6*:

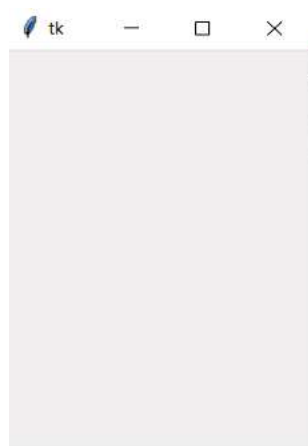


Figure 1.6: Output of Chap1_Example3.py

Note: The preceding code is covered in Program Name: Chap1_Example3.py

Now, there is a need to maximize neither height nor width. Then the following code will be used:

```
from tkinter import *

myroot = Tk()
myroot.resizable(width = False, height = False) # nei-
ther width nor height can be maximized
myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.7*:

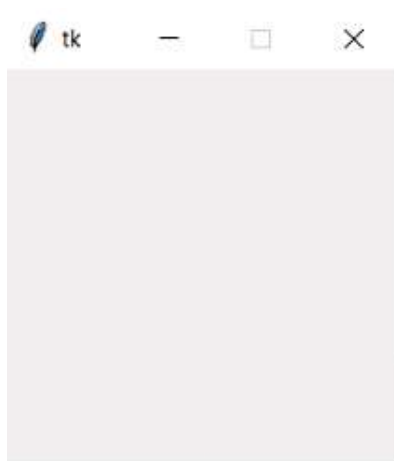


Figure 1.7: Output of Chap1_Example4.py

Note: The preceding code is covered in Program Name: Chap1_Example4.py

An important point to observe is that when we are maximizing the window in either of the directions, then the maximize button was enabled. Whereas in this case, the maximize button is disabled.

Some standard attributes of Python tkinter GUI

Now, we shall see how some of the standard attributes such as sizes, colors or fonts are specified. Just observe the standard attributes as mentioned. We shall see their usage when we will be dealing with widgets.

Dimensions

Whenever we set a dimension to an integer, it is assumed to be in pixels. A length is expressed as an integer number of pixels by tkinter. The list of common options are discussed as follows.

borderwidth

This option will give a 3-D look to the widget. It can also be represented as **bd**:

```
from tkinter import *
myroot = Tk()
myl1 = Label(myroot, text = 'Label1',bd = 8,relief = 'groove')
myl1.pack()
myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.8*:



Figure 1.8: Output of Chap1_Example5.py

Note: The preceding code is covered in Program Name: Chap1_Example5.py

highlightthickness

This option represents the width of the highlighted rectangle when the widget has focus. Refer to the following code:

```
from tkinter import *

myroot = Tk()

myb1 = Button(myroot, text = 'Without highlight thickness')
myb1.grid(row = 0, column = 1)

myb2 = Button(myroot, text = 'With highlight thickness',
              highlightthickness=10,
              )
myb2.grid(row = 1, column = 1, padx = 10, pady = 10)

myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.9*:

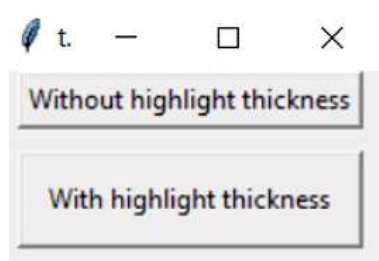


Figure 1.9: Output of Chap1_Example6.py

Note: The preceding code is covered in Program Name: Chap1_Example6.py

padX, padY

This option will provide extra space that the widget requests from its layout manager, beyond the minimum, for the display of widget contents in x and y directions. We can see in the previous example, that we have used padding in x and y direction for better look and display.

wraplength

This option will provide maximum length of line for widgets which will be performing word wrapping. Refer to the following code:

```
from tkinter import *
myroot = Tk()
myl1 = Label(myroot, text = 'Python',wraplength = 2)
myl1.pack()
myl2 = Label(myroot, text = 'awesome',wraplength = 0)
myl2.pack()
myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.10*:



Figure 1.10: Output of Chap1_Example7.py

Note: The preceding code is covered in Program Name: Chap1_Example7.py

height

This option will set the desired height of the widget as per need. It must be greater than 1.

underline

This option represents character index to underline in the widget's text. The 1st character will be 0, the 2nd character will be, 1 and so on.

width

This option will set the desired width of the widget as per need, as shown:

```
from tkinter import *
myroot = Tk()
myl1 = Label(myroot, text = 'Python',width = 20, height = 2, underline = 2, font = ('Calibri',15))
myl1.pack()
myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.11*:



Figure 1.11: Output of Chap1_Example8.py

Note: The preceding code is covered in Program Name: Chap1_Example8.py

In the above code, we have set width as 20, height as 2 and underline the 3rd character which is 't'.

Colors

Colors can be represented in tkinter using hexadecimal digits or by standard name. For example, #ff0000 is for Red color or it can be represented by using color = 'Red'. The different options available for color is are discussed as follows.

activebackground

This option will set the widget background color when it is active.

background

This option will set the widget background color and can also be represented as **bg**, as follows:

```
from tkinter import *
myroot = Tk()
myb1 = Button(myroot, activeback-
ground = "#ff0000", bg = '#00ff00', text = 'python')
myb1.pack()
myroot.mainloop()
```

Output Initially:

The output can be seen in *Figure 1.12*:



Figure 1.12: Initial Output

Output when button is clicked:

The output can be seen in *Figure 1.13*:

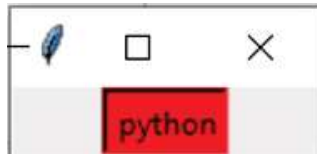


Figure 1.13: Output of Chap1_Example9.py

Note: The preceding code is covered in Program Name: Chap1_Example9.py

On running the preceding code, initially the background color of button is Green and when it is clicked, that is, when the button is active, the background color is changed to Red.

activeforeground

This option will set the widget foreground color when it is active.

foreground

This option will set the widget foreground color and can also be represented as **fg**.

```

from tkinter import *
myroot = Tk()
myb1 = Button(myroot, activefore-
ground = "#ff0000", fg = '#0000ff', text = 'python')
myb1.pack()
myroot.mainloop()

```

Output initially:

The output can be seen in *Figure 1.14*:



Figure 1.14: Initial output

Output when button is clicked:

The output can be seen in *Figure 1.15*:



Figure 1.15: Output of Chap1_Example10.py

Note: The preceding code is covered in Program Name: Chap1_Example10.py

On running the preceding code, initially the foreground color of button is Blue and when it is clicked, that is, when the button is active, the foreground color is changed to Red.

disabledforeground

This option will set the widget foreground color when it is disabled, as shown:

```

from tkinter import *
myroot = Tk()
myb1 = Button(myroot, state = 'disabled', text = 'python', dis-
abledforeground = 'Magenta')
myb1.pack()
myroot.mainloop()

```

Output:

The output can be seen in *Figure 1.16*:



Figure 1.16: Output of Chap1_Example11.py

Note: The preceding code is covered in Program Name: Chap1_Example11.py

In the preceding code, the button is disabled and the button's foreground color when it is disabled is Magenta.

highlightbackground

When the widget does not have any focus, this option will focus on the highlight color.

highlightcolor

When the widget has focus, this option will set the foreground color of the highlighted region, as shown:

```
from tkinter import *
myroot = Tk()
mye1 = Entry(myroot, bg='LightGreen', highlightthickness=10, highlightbackground="red", highlightcolor = 'Yellow')
mye1.pack(padx=5, pady=5)
mye2 = Entry(myroot)
mye2.pack()
mye2.focus()
myroot.mainloop()
```

Output when there is no focus on Entry:

The output can be seen in *Figure 1.17*:



Figure 1.17: Initial output

Output when there is focus on Entry:

The output can be seen in *Figure 1.18*:



Figure 1.18: Output of Chap1_Example12.py

Note: The preceding code is covered in Program Name: `Chap1_Example12.py`

In the preceding example, when there is no focus in Entry, the focus highlight's color is Red and when there is a focus on Entry, then the color in the focus highlight is Yellow.

selectbackground

This option will set the background color for the selected items of the widget.

selectforeground

This option will set the foreground color for the selected items of the widget, as shown:

```
from tkinter import *

myroot = Tk()
str1 = StringVar()
mye1 = Entry(myroot,selectbackground= 'Green',selectfore-
ground= 'Red', textvariable = str1)
mye1.pack()
str1.set('python')

myroot.mainloop()
```

Output:

The output can be seen in *Figure 1.19*:

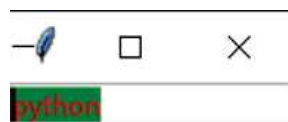


Figure 1.19: Output of Chap1_Example13.py

Note: The preceding code is covered in Program Name: Chap1_Example13.py

From the preceding code, we can observe that when the text 'python' is selected, it is highlighted in Green color in the background and Red color in the foreground.

Fonts

We can access the font in tkinter by creating a font object or by using a tuple.

By creating a font object

Suppose we have a text. We can underline the text, change the size, give some font family, overstrike it, and so on.

To create a font object, first, we need to import a **tkinter.font** module and use the **Font** class constructor:

```
from tkinter.font import Font
myobj_font = Font(options,...)
```

The options are:

- **family:** This option represents the name of the font family as a string.
- **size:** This option represents an integer in points for font height.
- **weight:** This option represents 'bold' and 'normal'.
- **slant:** This option represents 'italic' and 'unslanted'.
- **underline:** This option represents whether the text is to be underlined or not. 1 represents for underline text and 0 for normal.
- **overstrike:** This option represents whether the text is overstruck or not. 1 represents overstruck text and 0 for normal.

Refer to the following code:

```
from tkinter import * # importing module
from tkinter.font import Font
myroot = Tk() # window creation and initialize the interpreter

myfont1 = Font(family = 'Calibri',size=12, weight = 'bold', slant='italic', underline = 1, over-
strike = 1) # 1 means we require underline
myl1 = Label(myroot, text = 'Python', font = myfont1)
myl1.pack() # for displaying the label

myroot.mainloop() # display window until we press the close button
```

Output:

The output can be seen in *Figure 1.20*:



Figure 1.20: Output of Chap1_Example14.py

Note: The preceding code is covered in Program Name: Chap1_Example14.py

We can also view the font family which we can access based on our requirement, as follows:

```
from tkinter import * # importing module
from tkinter import font
myroot = Tk() # window creation and initialize the interpreter

myfont_list = list(font.families())
for loop in myfont_list:
    print(loop,end = ',')

myroot.mainloop() # display window until we press the close button
```

Output:

The output is as follows:

```
System,8514oem,Fixedsys,Terminal,Modern,Roman,Script,Courier,MS
Serif,MS Sans Serif,Small Fonts,TeamViewer15,Marlett,Arial,Arabic
Transparent,Arial Baltic,Arial CE,Arial CYR,Arial
Greek,Arial TUR,Arial Black,Bahnschrift Light,Bahnschrift
SemiLight,Bahnschrift,Bahnschrift SemiBold,Bahnschrift Light
SemiCondensed,Bahnschrift SemiLight SemiConde,Bahnschrift
SemiCondensed,Bahnschrift SemiBold SemiCondensed,Bahnschrift
Light Condensed, Bahnschrift SemiLight Condensed,Bahnschrift
Condensed,Bahnschrift SemiBold Condensed,Calibri,Calibri
Light,Cambria,Cambria Math,Candara,Candara Light, Comic Sans
MS,Consolas,Constantia,Corbel,Corbel Light,Courier New,Courier New
Baltic,Courier New CE,Courier New CYR,Courier New Greek,Courier
New TUR, Ebrima,Franklin Gothic Medium, Gabriola, Gadugi,
Georgia,Impact,Ink Free, Javanese Text,Leelawadee UI,Leelawadee UI
Semilight,Lucida Console,Lucida Sans Unicode,Malgun Gothic,@Malgun
```

```

Gothic,Malgun Gothic Semilight,@Malgun Gothic Semilight,Microsoft
Himalaya,Microsoft JhengHei,@Microsoft JhengHei,Microsoft
JhengHei UI,@Microsoft JhengHei UI,Microsoft JhengHei Light,@
Microsoft JhengHei Light,Microsoft JhengHei UI Light,@
Microsoft JhengHei UI Light,Microsoft New Tai Lue,Microsoft
PhagsPa,Microsoft Sans Serif,Microsoft Tai Le,Microsoft YaHei,@
Microsoft YaHei,Microsoft YaHei UI,@Microsoft YaHei UI,Microsoft
YaHei Light,@Microsoft YaHei Light,Microsoft YaHei UI Light,@
Microsoft YaHei UI Light,Microsoft Yi Baiti,MingLiU-ExtB,@MingLiU-
ExtB,PMingLiU-ExtB,@PMingLiU-ExtB,MingLiU_HKSCS-ExtB,@MingLiU_
HKSCS-ExtB,Mongolian Baiti,MS Gothic,@MS Gothic,MS UI Gothic,@
MS UI Gothic,MS PGothic,@MS PGothic,MV Boli,Myanmar Text,Nirmala
UI,Nirmala UI Semilight,Palatino Linotype,Segoe MDL2 Assets,Segoe
Print,Segoe Script,Segoe UI,Segoe UI Black,Segoe UI Emoji,Segoe UI
Historic,Segoe UI Light,Segoe UI Semibold,Segoe UI Semilight,Segoe
UI Symbol,SimSun,@SimSun,NSimSun,@NSimSun,SimSun-ExtB,@SimSun-
ExtB,Sitka Small,Sitka Text,Sitka Subheading,Sitka Heading,Sitka
Display,Sitka Banner,Sylfaen,Symbol,Tahoma,Times New Roman,Times New
Roman Baltic,Times New Roman CE,Times New Roman CYR,Times New Roman
Greek,Times New Roman TUR,Trebuchet MS,Verdana,Webdings,Wingdings,Yu
Gothic,@Yu Gothic,Yu Gothic UI,@Yu Gothic UI,Yu Gothic UI Semibold,@
Yu Gothic UI Semibold,Yu Gothic Light,@Yu Gothic Light,Yu Gothic
UI Light,@Yu Gothic UI Light,Yu Gothic Medium,@Yu Gothic Medium,Yu
Gothic UI Semilight,@Yu Gothic UI Semilight,HoloLens MDL2 Assets,HP
Simplified,HP Simplified Light,MT Extra,Century,Wingdings 2,Wingdings
3,Arial Unicode MS,@Arial Unicode MS,Nina,Segoe Condensed,Buxton
Sketch,Segoe Marker,SketchFlow Print,DengXian,@DengXian,Microsoft
MHei,@Microsoft MHei,Microsoft NeoGothic,@Microsoft NeoGothic,Segoe
WP Black,Segoe WP,Segoe WP Semibold,Segoe WP Light,Segoe
WP SemiLight, DigifaceWide, AcadEref, AIGDT, AmdtSymbols,
GENISO,AMGDT,BankGothic Lt

BT,BankGothic Md BT,CityBlueprint,CommercialPi BT,CommercialScript
BT,CountryBlueprint,Dutch801 Rm BT,Dutch801 XBd BT,EuroRoman,ISOC-
PEUR,ISOCTEUR,Monospac821 BT,PanRoman,Romantic,RomanS,SansSerif,Styl-
us BT,SuperFrench,Swis721 BT,Swis721 BdOul BT,Swis721 Cn BT,Swis721
BdCnOul BT,Swis721 BlkCn BT,Swis721 LtCn BT,Swis721 Ex BT,Swis721
BlkEx BT,Swis721 LtEx BT,Swis721 Blk BT,Swis721 BlkOul BT,Swis721 Lt
BT,TechnicBold,TechnicLite,Technic,UniversalMath1 BT,Vineta

CP,ISOCT2,ISOCT3,ISOCT,ItalicC,ItalicT,Italic,Monotxt,Proxy 1,Proxy
2,Proxy 3,Proxy 4,Proxy 5,Proxy 6,Proxy 7,Proxy 8,Proxy 9,RomanC,Ro-
manD,RomanT,ScriptC,Scripts,Simplex,Syastro,Symap,Symath,Symeteo,Sy-
music,Txt,

```

Note: The preceding code is covered in Program Name: Chap1_Example15.py