Image Viewer App in Python using Tkinter

Prerequisites: Python GUI – tkinter, Python: Pillow

Have you ever wondered to make a Image viewer with the help of Python? Here is a solution to making the Image viewer with the help of Python. We can do this with the help of Tkinter and pillow. We will discuss the module needed and code below.

Module Needed

- **Tkinter:** Tkinter is library with the help of which we can make GUI(Graphical User Interface).

```
pip install tkinter
```

- **Pillow:** We can add photos as it is an imaging library of Python.

```
pip install pillow
```

Now let's code for it

Getting Started

- Below Code demonstrates the basic structures, button initialization, and layout of the GUI produced

Python3

```
# importing the tkinter module and PIL that
# is pillow module
from tkinter import *
from PIL import ImageTk, Image

# Calling the Tk (The initial constructor of tkinter)
root = Tk()
```

```
# We will make the title of our app as Image Viewer
root.title("Image Viewer")

# The geometry of the box which will be displayed
# on the screen
root.geometry("700x700")

# Adding the images using the pillow module which
# has a class ImageTk We can directly add the
# photos in the tkinter folder or we have to
# give a proper path for the images
image_no_1 = ImageTk.PhotoImage(Image.open("Sample.png"))
image_no_2 = ImageTk.PhotoImage(Image.open("sample.png"))
image_no_3 = ImageTk.PhotoImage(Image.open("Sample.png"))
image_no_4 = ImageTk.PhotoImage(Image.open("sample.png"))

# List of the images so that we traverse the list
List_images = [image_no_1, image_no_2, image_no_3, image_no_4]

label = Label(image=image_no_1)

# We have to show the box so this below line is needed
label.grid(row=1, column=0, columnspan=3)

# We will have three button back ,forward and exit
button_back = Button(root, text="Back", command=back,
                     state=DISABLED)

# root.quit for closing the app
button_exit = Button(root, text="Exit",
                     command=root.quit)

button_forward = Button(root, text="Forward",
                        command=lambda: forward(1))

# grid function is for placing the buttons in the frame
button_back.grid(row=5, column=0)
button_exit.grid(row=5, column=1)
button_forward.grid(row=5, column=2)

root.mainloop()
```

- **Forward function:** This function is for adding the functionality to forward button

---

Python3

```
def forward(img_no):
```

```
    # GLobal variable so that we can have
    # access and change the variable
    # whenever needed
    global label
    global button_forward
    global button_back
    global button_exit
    label.grid_forget()

    # This is for clearing the screen so that
    # our next image can pop up
    label = Label(image=List_images[img_no-1])

    # as the list starts from 0 so we are
    # subtracting one
    label.grid(row=1, column=0, columnspan=3)
    button_for = Button(root, text="forward",
                        command=lambda: forward(img_no+1))

    # img_no+1 as we want the next image to pop up
    if img_no == 4:
        button_forward = Button(root, text="Forward",
                                state=DISABLED)

    # img_no-1 as we want previous image when we click
    # back button
    button_back = Button(root, text="Back",
                         command=lambda: back(img_no-1))

    # Placing the button in new grid
    button_back.grid(row=5, column=0)
    button_exit.grid(row=5, column=1)
    button_for.grid(row=5, column=2)
```

- **Backward Function:** This function is for adding the functionality to backward button

---

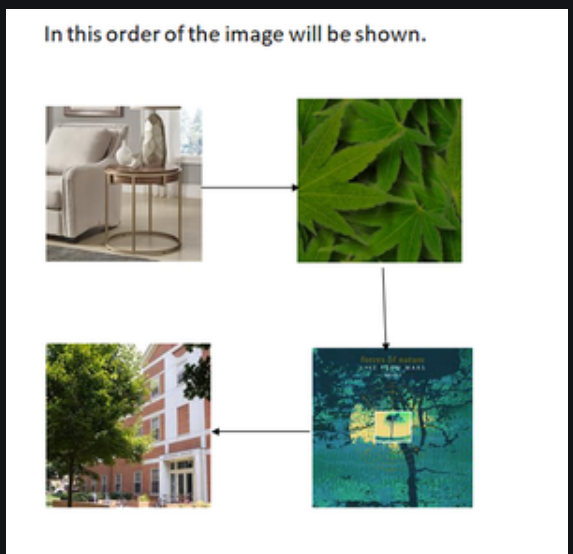
Python3

```
def back(img_no):

    # We will have global variable to access these
    # variable and change whenever needed
    global label
    global button_forward
    global button_back
    global button_exit
    label.grid_forget()

    # for clearing the image for new image to pop up
```

```
    label = Label(image=List_images[img_no - 1])
    label.grid(row=1, column=0, columnspan=3)
    button_forward = Button(root, text="forward",
                            command=lambda: forward(img_no + 1))
    button_back = Button(root, text="Back",
                         command=lambda: back(img_no - 1))
    print(img_no)

    # whenever the first image will be there we will
    # have the back button disabled
    if img_no == 1:
        button_back = Button(root, Text="Back", state=DISABLED)

    label.grid(row=1, column=0, columnspan=3)
    button_back.grid(row=5, column=0)
    button_exit.grid(row=5, column=1)
    button_for.grid(row=5, column=2)
```

Complete Code

Images Used and their order –



Order in which the images will be shown.

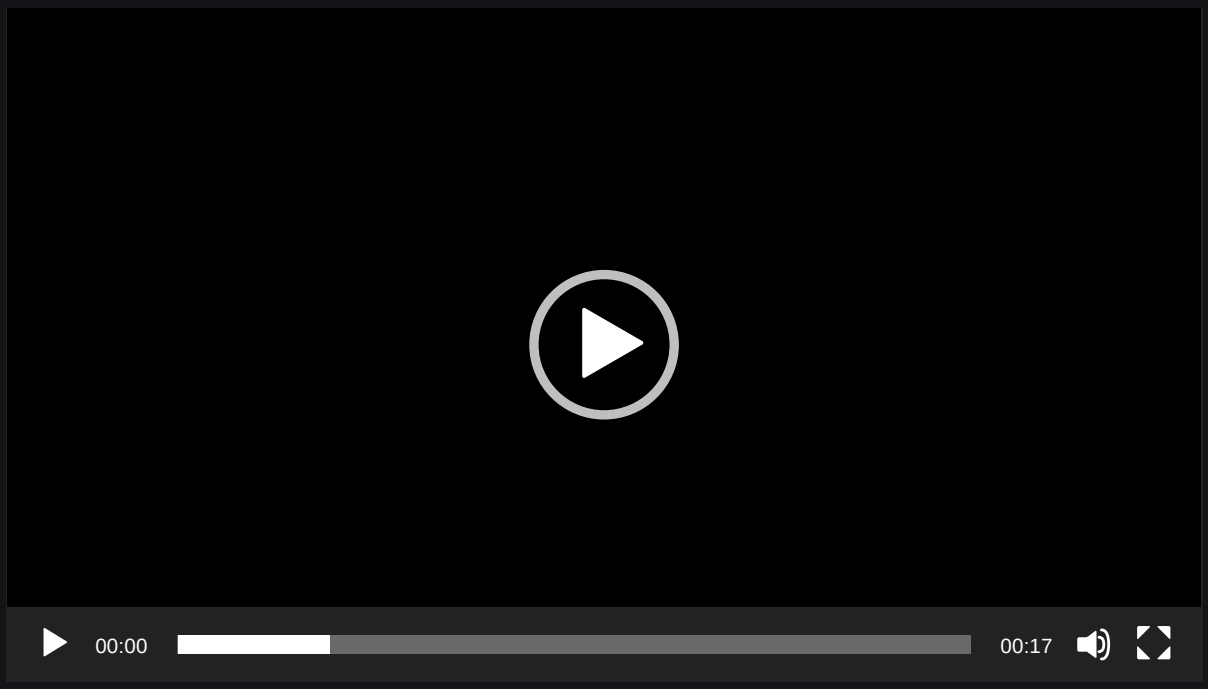
Python3

```
# importing the tkinter module and PIL
# that is pillow module
from tkinter import *
from PIL import ImageTk, Image


def forward(img_no):

    # GLobal variable so that we can have
    # access and change the variable
    # whenever needed
    global label
    global button_forward
    global button_back
    global button_exit
    label.grid_forget()

    # This is for clearing the screen so that
    # our next image can pop up
    label = Label(image=List_images[img_no-1])

    # as the list starts from 0 so we are
    # subtracting one
    label.grid(row=1, column=0, columnspan=3)
    button_for = Button(root, text="forward",
                        command=lambda: forward(img_no+1))

    # img_no+1 as we want the next image to pop up
    if img_no == 4:
        button_forward = Button(root, text="Forward",
                                state=DISABLED)

    # img_no-1 as we want previous image when we click
    # back button
    button_back = Button(root, text="Back",
                         command=lambda: back(img_no-1))

    # Placing the button in new grid
    button_back.grid(row=5, column=0)
    button_exit.grid(row=5, column=1)
    button_for.grid(row=5, column=2)


def back(img_no):

    # We will have global variable to access these
    # variable and change whenever needed
    global label
    global button_forward
```

```
    global button_back
    global button_exit
    label.grid_forget()

    # for clearing the image for new image to pop up
    label = Label(image=List_images[img_no - 1])
    label.grid(row=1, column=0, columnspan=3)
    button_forward = Button(root, text="forward",
                            command=lambda: forward(img_no + 1))
    button_back = Button(root, text="Back",
                         command=lambda: back(img_no - 1))
    print(img_no)

    # whenever the first image will be there we will
    # have the back button disabled
    if img_no == 1:
        button_back = Button(root, Text="Back", state=DISABLED)

    label.grid(row=1, column=0, columnspan=3)
    button_back.grid(row=5, column=0)
    button_exit.grid(row=5, column=1)
    button_for.grid(row=5, column=2)


# Calling the Tk (The initial constructor of tkinter)
root = Tk()

# We will make the title of our app as Image Viewer
root.title("Image Viewer")

# The geometry of the box which will be displayed
# on the screen
root.geometry("700x700")

# Adding the images using the pillow module which
# has a class ImageTk We can directly add the
# photos in the tkinter folder or we have to
# give a proper path for the images
image_no_1 = ImageTk.PhotoImage(Image.open("Sample.png"))
image_no_2 = ImageTk.PhotoImage(Image.open("sample.png"))
image_no_3 = ImageTk.PhotoImage(Image.open("Sample.png"))
image_no_4 = ImageTk.PhotoImage(Image.open("sample.png"))

# List of the images so that we traverse the list
List_images = [image_no_1, image_no_2, image_no_3, image_no_4]

label = Label(image=image_no_1)

# We have to show the box so this below line is needed
label.grid(row=1, column=0, columnspan=3)

# We will have three button back ,forward and exit
button_back = Button(root, text="Back", command=back,
```

```
                    state=DISABLED)

# root.quit for closing the app
button_exit = Button(root, text="Exit",
                     command=root.quit)

button_forward = Button(root, text="Forward",
                        command=lambda: forward(1))

# grid function is for placing the buttons in the frame
button_back.grid(row=5, column=0)
button_exit.grid(row=5, column=1)
button_forward.grid(row=5, column=2)

root.mainloop()
```

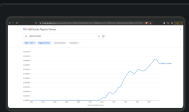
Output:

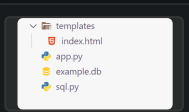


Last Updated : 13 Jul, 2022

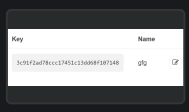
1

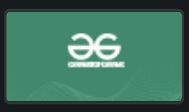
Similar Reads

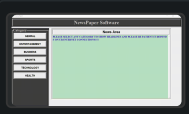

Scrape Google Ngram Viewer using Python



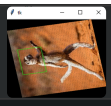
How to Use SQLite Viewer Flask


Weather App in Python using Tkinter module


Facts App Using Tkinter module


A simple News app with Tkinter and Newsapi

Python | Add image on a Tkinter button



How to resize Image in Python - Tkinter?

Build An Image Swiper App For KivyMD in Python

Color game using Tkinter in Python

Python | Message Encode-Decode using Tkinter

Related Tutorials

OpenAI Python API - Complete Guide

Pandas AI: The Generative AI Python Library

Python for Kids - Fun Tutorial to Learn Python Programming

Data Analysis Tutorial

Flask Tutorial

Previous

Simple registration form using Python Tkinter

Next

How to create a COVID19 Data Representation GUI?

Article Contributed By :

abhisheksrivastaviot18

abhisheksrivastaviot18

Vote for difficulty

Easy | Normal | Medium | Hard | Expert

Improved By : simranarora5sos, simmytarika5, surindertarika1234

Article Tags : Python Tkinter-exercises, Python-projects, Python-tkinter, Python

Practice Tags : python

Improve Article | Report Issue

GeeksforGeeks
Sanchhaya Education Private Limited

A-143, 9th Floor, Sovereign Corporate Tower, Sector-136, Noida, Uttar Pradesh - 201305

feedback@geeksforgeeks.org

Company

About Us

Legal

Careers

In Media

Contact Us

Advertise with us

Campus Training Program

Explore

Job-A-Thon Hiring Challenge

Hack-A-Thon

GfG Weekly Contest

Offline Classes (Delhi/NCR)

DSA in JAVA/C++

Master System Design

Master CP

Languages

Python

Java

C++

PHP

GoLang

SQL

R Language

Android Tutorial

DSA Concepts

Data Structures

Arrays

Strings

Linked List

Algorithms

Searching

Sorting

Mathematical

Dynamic Programming

DSA Roadmaps

DSA for Beginners

Web Development

HTML

Basic DSA Coding Problems

DSA Roadmap by Sandeep Jain

DSA with JavaScript

Top 100 DSA Interview Problems

All Cheat Sheets

CSS

JavaScript

Bootstrap

ReactJS

AngularJS

NodeJS

Express.js

Lodash

Computer Science

GATE CS Notes

Operating Systems

Computer Network

Database Management System

Software Engineering

Digital Logic Design

Engineering Maths

Python

Python Programming Examples

Django Tutorial

Python Projects

Python Tkinter

OpenCV Python Tutorial

Python Interview Question

Data Science & ML

Data Science With Python

Data Science For Beginner

Machine Learning Tutorial

Maths For Machine Learning

Pandas Tutorial

NumPy Tutorial

NLP Tutorial

Deep Learning Tutorial

DevOps

Git

AWS

Docker

Kubernetes

Azure

GCP

Competitive Programming

Top DSA for CP

Top 50 Tree Problems

Top 50 Graph Problems

Top 50 Array Problems

Top 50 String Problems

Top 50 DP Problems

Top 15 Websites for CP

System Design

What is System Design

Monolithic and Distributed SD

Scalability in SD

Databases in SD

High Level Design or HLD

Low Level Design or LLD

Top SD Interview Questions

Interview Corner

Company Wise Preparation

Preparation for SDE

Experienced Interviews

Internship Interviews

Competitive Programming

Aptitude Preparation

GfG School

CBSE Notes for Class 8

CBSE Notes for Class 9

CBSE Notes for Class 10

CBSE Notes for Class 11

CBSE Notes for Class 12

English Grammar

Commerce

Accountancy

Business Studies

Economics

Management

Income Tax

Finance

Statistics for Economics

UPSC

Polity Notes

Geography Notes

History Notes

Science and Technology Notes

Economics Notes

Important Topics in Ethics

UPSC Previous Year Papers

SSC/ BANKING

SSC CGL Syllabus

SBI PO Syllabus

SBI Clerk Syllabus

IBPS PO Syllabus

IBPS Clerk Syllabus

Aptitude Questions

SSC CGL Practice Papers

Write & Earn

Write an Article

Improve an Article

Pick Topics to Write

Write Interview Experience

Internships