```
#! /usr/bin/python3


# This is PlayerRev2.py  18 March 2024


# This program depends on a provided list of images with the following file structure;

#

# Each image is detailed in 3 lines, the first line structure contains the sub-path to the image as below

#

#  ======== /Pictures/Art/Art1/1906.99 - The Assumption of the Virgin.jpg

#  Caption                    : El Greco: 1906.99 - The Assumption of the Virgin.jpg

#  Date/Time                  : No date available

#

#  This file is produced via the windows machines with the images and picture indexing programs, and resides

#  in the USB drive subfolder "/PictureLists/"


#  The USB subfolder "/PictureLists/" also contains the files zzzDisplayTime.txt, zzzRandomFlag.txt, and

#  zzzAllBlack.jpg which are used by this program.  Prefix the file outputs is zzz so they appear at the bottom of

#  the selections in "PictureLists".


import tkinter as tk

from tkinter import *

import os

import subprocess

import time
```

```
import random


# Open the root window


root=tk.Tk()

root.geometry('1000x500+50+50') # width x height +x +y

root.title('Picture Frame Setup')


# Open the askopenfilename module - this is a frame so it requires the root window to be open

#
#################################################################################################


from tkinter.filedialog import askopenfilename


pictureList = tk.filedialog.askopenfilename(parent=root,initialdir="/media/pi/",title="Select a picture list file")

   #initialdir not required, just makes it easier

print("Full path to input file" ,pictureList)

print()


# At this point the askopenfilename should close, but the root window will remain open

# Variable "pictureList" contains the full path to the selected picture list

#
#################################################################################################
```

```
# Extract the path only from "pictureList"

justTheFile=os.path.basename(pictureList)

tempSplit=os.path.split(pictureList)

pathToPictureList=tempSplit[0]


# Set up paths to files for storing the display time and random flag data

displayTimeStore=(pathToPictureList+"/zzzDisplayTime.txt")

randomFlagStore=(pathToPictureList+"/zzzRandomFlag.txt")


# Paths to files for storing data with the selected picture file established

#
################################################################################################
#############


# Radio buttons to select image delay time


# Set up for radio buttons to look nice


frm1=Frame(root,height=10,width=200,)

frm1.grid(column="0",row="0")

frm2=Frame(root,height=10,width=200,)

frm2.grid(column="1",row="0")

frm3=Frame(root,height=10,width=200,)

frm3.grid(column="2",row="0")

frm4=Frame(root,height=10,width=200,)

frm4.grid(column="3",row="0")
```

```
frm5=Frame(root,height=10,width=200,)

frm5.grid(column="4",row="0")



# This produces an empty line at the top of the input screen, also sets the root window to 5 columns


l=tk.Label(root, bg="white", width=20, text="Select a display time",font="40")

l.grid(column="2")


def write_time_selection():

   text_file=open(displayTimeStore,"w")

   fieldContent="Display time in seconds = "+var.get()

   text_file.write(fieldContent)

   text_file.close()


var=tk.StringVar()


r1=tk.Radiobutton(root, text="2 seconds", height=2, variable=var,
value="2",command=write_time_selection)

r1.grid(column="2",sticky="W")

r2=tk.Radiobutton(root, text="5 seconds", height=2, variable=var,
value="5",command=write_time_selection)

r2.grid(column="2",sticky="W")

r3=tk.Radiobutton(root, text="10 seconds", height=2, variable=var,
value="10",command=write_time_selection)

r3.grid(column="2",sticky="W")

r3.invoke() # Default setting
```

```
r4=tk.Radiobutton(root, text="30 seconds", height=2, variable=var,
value="30",command=write_time_selection)

r4.grid(column="2",sticky="W")

r5=tk.Radiobutton(root, text="1 minute", height=2, variable=var,
value="60",command=write_time_selection)

r5.grid(column="2",sticky="W")

r6=tk.Radiobutton(root, text="2 minute", height=2, variable=var,
value="120",command=write_time_selection)

r6.grid(column="2",sticky="W")

r7=tk.Radiobutton(root, text="5 minute", height=2, variable=var,
value="300",command=write_time_selection)

r7.grid(column="2",sticky="W")

r8=tk.Radiobutton(root, text="10 minute", height=2, variable=var,
value="600",command=write_time_selection)

r8.grid(column="2",sticky="W")


button2=Button(root, text=" Done  ",fg="red",command=root.destroy)

button2.grid(column="2")


root.mainloop()


# Radio buttons to choose to randomise or not

# Root window was destroyed after display time was set, so need to re-open one


root=tk.Tk()

root.geometry('1000x500+50+50') # width x height +x +y

root.title('Randomization choice')
```

```
# Set up for radio buttons to look nice


frm1=Frame(root,height=10,width=200,)

frm1.grid(column="0",row="0")

frm2=Frame(root,height=10,width=200,)

frm2.grid(column="1",row="0")

frm3=Frame(root,height=10,width=200,)

frm3.grid(column="2",row="0")

frm4=Frame(root,height=10,width=200,)

frm4.grid(column="3",row="0")

frm5=Frame(root,height=10,width=200,)

frm5.grid(column="4",row="0")


# This produces an empty line at the top of the input screen, also sets the root window to 5 columns


l=tk.Label(root, bg="white", width=20, text="Pick one",font="40")

l.grid(column="2")


def write_randomflag_selection():

    text_file=open(randomFlagStore,"w")

    fieldContent="Random flag setting = "+var.get()

    text_file.write(fieldContent)

    text_file.close()
```

```
var=tk.StringVar()


r1=tk.Radiobutton(root, text="Randomised", height=2, variable=var,
value="True",command=write_randomflag_selection)

r1.grid(column="2",sticky="W")

r2=tk.Radiobutton(root, text="Not randomised", height=2, variable=var,
value="False",command=write_randomflag_selection)

r2.grid(column="2",sticky="W")

r2.invoke() # Default setting


# var values changed to True and False


button2=Button(root, text=" Done  ",fg="red",command=root.destroy)

button2.grid(column="2")


root.mainloop()


#
##########################################################################################
###############


# End of TkinterInput section


# Variables carried forward are;

#

#  pictureList - this is the name and path of the requested image set

#  pathToPictureList - as above, but with file name removed
```

```
#  A file containing the display time called "zzzDisplayTimeStore"

#  A file containing the random flag called "zzzRandomFlagStore"

#  Both files are in the same directory as the requested image set

#


#
#################################################################################################
################


#  Assign the value in "zzzDisplayTimeStore" to variable "pictureSpacing"

#

#  Set up path to the display time file, extract value and convert to int


displayTimeStore=(pathToPictureList+"/zzzDisplayTime.txt")

print("Display time stored in ",displayTimeStore)


with open(displayTimeStore,"r") as f:


   dataListTemp=f.readlines()

   pictureSpacing=dataListTemp[0]

   # trim off label

   pictureSpacing=pictureSpacing[26:]

   # convert remains to int

   pictureSpacing=int(pictureSpacing)

   f.close
```

```
# confirm value now available to remainder of program
print("Picture spacing = ",pictureSpacing," seconds")
print()

#  Set up path to random flag file, and extract value

randomFlagStore=(pathToPictureList+"/zzzRandomFlag.txt")
print("Random flag stored in ",randomFlagStore)

with open(randomFlagStore,"r") as f:

   dataListTemp=f.readlines()
   randomFlag=dataListTemp[0]
   # trim off label
   randomFlag=randomFlag[22:]
   f.close

# confirm value now available to remainder of program
print("RandomFlag value is = ",randomFlag)
print()

#  Set up path to "/Pictures/" directory for use later

word="/PictureLists/"
offset=pictureList.find(word)
```

```
thisUSBprefix=pictureList[:offset]
print("USB containing this picture list is ",thisUSBprefix)
print()
print("Input section completed")
print()

# ###########################################################################

# All inputs available

# Variable set now is;
#  pictureList - this is the name and path of the requested image set
#  pathToPictureList - as above, but with file name removed
#  thisUSBprefix - prefix unique to this USB
#  pictureSpacing - integer time in seconds
#  randomFlag - either True or False

# This relies on a datafile already selected (variable "pictureList"),
# with the first line structure similar to below;
# "======== /Pictures/Art/Art1/filename.jpg"
# There are 3 lines per image

# Also stored in the PictureList directory is an image file "zzzAllBlack.jpg"
# because this will be used whichever image list is selected
```

```
##### PlayerRev1.py revixed to here and fully functional ######################

################################################################################


#  set location of black background image

blackImageLocation=pathToPictureList+"/zzzAllBlack.jpg"


blackBehind=["feh","--geometry","3840x2160+1920+0","--scale-down","--image-bg","black","-
x",blackImageLocation]


backgroundImage=subprocess.Popen (blackBehind)

pictureshowB=subprocess.Popen (blackBehind)


print("Black background image set up and displayed using two sub-processes")

print()

# One black image will remain during the entire show, the other is used as an initial image

# deleted once the first real image is displayed


################################################################################



# First read the selected image list into the master file, and strip off line feeds


print()

print("Opening picture list at ",pictureList)

print()
```

```
#dataListSource=path+"/datafile.txt"
#print(dataListSource)

with open(pictureList,"r") as f:

   dataListTemp=f.readlines()
   totalLines=len(dataListTemp)
   print("Total lines read = ",totalLines)
   print("First image = ",dataListTemp[0]) # Every third item in list is an image ID
   numberOfImages=int(totalLines/3)
   print(numberOfImages, " images found")
   print()

################################################################################

# Randomization routine, create a list and randomise if required

displayOrderList = list(range(0,numberOfImages))

if randomFlag == "True": # note quoted True here
   random.shuffle(displayOrderList)
   print("A randomised displayOrderList has been created")
else:
   print("An non-randomised playOrderList has been created")
print()
```

```
#############################################################################

imageCount = 0 # count of images in the playOrderList
while imageCount <=numberOfImages:
# Determine if count is even or odd
   flag=imageCount % 2
# Mod function, will return either 0 or 1

# Get the position of the image to use
   count=displayOrderList[imageCount]*3

   imageID=dataListTemp[count]
   imageID=imageID.replace("\n","")
   # strip the first 10 characters off the imageID
   strippedImageID=imageID[9                    :]
   # add path
   displayableImageID=thisUSBprefix+strippedImageID# ************************************

   caption=dataListTemp[count+1]
   caption=caption.replace("\n","")
   strippedCaption=caption[33:]

   date=dataListTemp[count+2]
   date=date.replace("\n","")
```

```
    strippedDate=date[33:]


    # Picture information output


    print("Now showing ",strippedImageID)
    print("Caption    ",strippedCaption)
    print("Date taken ",strippedDate)
    print()


    display=["feh","--geometry","1920x1080+1920+0","--scale-down","--image-bg","black","-
x",(displayableImageID)]


    if flag==0:
        time.sleep(1)
        pictureshowA = subprocess.Popen (display)
        time.sleep(1)
        pictureshowB.kill()


        imageCount+=1
        time.sleep(pictureSpacing)
    # new image shown on top


    else:
        pictureshowB = subprocess.Popen (display)
        time.sleep(1)
        pictureshowA.kill()
```

```
        imageCount+=1
        time.sleep(pictureSpacing)


pictureshowA.kill()
pictureshowB.kill()
backgroundImage.kill()


root.mainloop()
```