Documentation for Random Fruit Classifier Program

Objective

The purpose of this program is to randomly pick an image of a fruit (apple, banana, cherry, or dragon fruit) from a folder, display it on a graphical user interface (GUI), label it, and use text-to-speech (TTS) to announce the name of the fruit.

Prerequisites

Before running the program:

1. **Python Version**: Python 3.6 or higher.
2. **Libraries to Install**:
    a. `Pillow` for image processing: `pip install pillow`

    b. `pyttsx3` for text-to-speech: `pip install pyttsx3`

3. **Dataset Structure**:
    a. A folder named `dataset` containing subfolders for each fruit: `dataset/`

```
apple/
  image1.jpg
  image2.jpg
  ...
banana/
  ...
cherry/
  ...
dragon fruit/
  ...
```

Program Explanation

Imports

```
import tkinter as tk
from PIL import Image, ImageTk
import pyttsx3
import random
import os
```

- **`tkinter`**: Used to create the graphical user interface (GUI).
- **`PIL (Pillow)`**: Provides functionality to open, manipulate, and display images.
- **`pyttsx3`**: Enables text-to-speech conversion.
- **`random`**: Used for selecting random fruit and image.
- **`os`**: Allows directory traversal for accessing image files.

Initialization

```
engine = pyttsx3.init()
```

- Initializes the text-to-speech engine.
- `engine` is used later to convert text (fruit name) to speech.

```
base_dir = 'dataset'
classes = ['apple', 'banana', 'cherry', 'dragon fruit']
```

- **`base_dir`**: Specifies the directory where fruit images are stored.
- **`classes`**: A list of fruit names corresponding to the subfolder names in the dataset.

Building Image Paths Dictionary

```
image_paths = {fruit: [os.path.join(base_dir, fruit, file) for file in
os.listdir(os.path.join(base_dir, fruit))] for fruit in classes}
```

- Creates a dictionary `image_paths`:

    - Keys: Fruit names (e.g., `apple`, `banana`).
    - Values: List of all image file paths for that fruit.
- Uses `os.listdir()` to get files in each fruit folder and `os.path.join()` to form full paths.

Function: `display_random_image`

```
def display_random_image():
    fruit = random.choice(classes)
    image_path = random.choice(image_paths[fruit])
```

- Randomly selects:
    - A fruit class (`random.choice(classes)`).
    - An image from that fruit's folder (`random.choice(image_paths[fruit])`).

Load and Display the Image

```
img = Image.open(image_path).resize((300, 300))
img = ImageTk.PhotoImage(img)
image_label.config(image=img)
image_label.image = img
```

1. **`Image.open(image_path)`**: Opens the selected image.
2. **`.resize((300, 300))`**: Resizes the image to fit the GUI.
3. **`ImageTk.PhotoImage(img)`**: Converts the image into a format that can be displayed in the Tkinter GUI.
4. **`image_label.config()`**: Updates the `image_label` widget to display the selected image.

Display and Announce the Label

```
fruit_label.set(fruit.capitalize())
engine.say(f"It is {fruit}")
engine.runAndWait()
```

1. **`fruit_label.set()`**: Updates the text label to show the name of the fruit in title case (e.g., `Apple`).
2. **`engine.say()`**: Prepares the text-to-speech message.
3. **`engine.runAndWait()`**: Executes the text-to-speech conversion and plays the message.

Graphical User Interface (GUI)

```
root = tk.Tk()
root.title("Fruit Classifier")
```

- **`tk.Tk()`**: Creates the main application window.
- **`root.title()`**: Sets the title of the window.

Image Display Widget

```
image_label = tk.Label(root)
image_label.pack()
```

- **`tk.Label(root)`**: Creates a label widget to display the image.
- **`.pack()`**: Places the label in the application window.

Text Label

```
fruit_label = tk.StringVar()
label_text = tk.Label(root, textvariable=fruit_label,
font=("Helvetica", 20))
label_text.pack()
```

- **`tk.StringVar()`**: A special variable to dynamically update the text label.
- **`label_text`**: A label widget bound to `fruit_label` for displaying the fruit name.
- **`font=("Helvetica", 20)`**: Sets the font style and size for the label.

Button to Trigger Random Image Display

```
button = tk.Button(root, text="Show Random Fruit",
command=display_random_image)
```

```
button.pack()
```

- **`tk.Button()`**: Creates a button widget.
- **`command=display_random_image`**: Binds the button to the `display_random_image` function, so it gets called when the button is clicked.

Run the Application

```
root.mainloop()
```

- Starts the Tkinter event loop, which keeps the application running and responsive to user interactions.

Expected Output

1. A window opens with:
    a. An image of a random fruit.
    b. The name of the fruit displayed as text.
    c. A button labeled "Show Random Fruit."
2. Clicking the button displays a new random image with its label.
3. The fruit's name is announced via text-to-speech.

Error Handling

1. Ensure the `dataset` directory and subfolders exist.
2. Verify that each subfolder contains valid image files.
3. Install necessary libraries before running the code.

Enhancements

- You can add more fruits by updating the `classes` list and downloading corresponding images.
- Include additional features, such as fruit descriptions or nutritional information.