Αρχικοποίηση και βιβλιοθήκες

1. import tkinter as tk
2. from PIL import Image, ImageTk
3. import pygame

1. Εισάγει τη βιβλιοθήκη Tkinter για τη δημιουργία του γραφικού περιβάλλοντος (GUI).
2. Εισάγει τις κλάσεις Image και ImageTk από τη βιβλιοθήκη Pillow για φόρτωση και επεξεργασία εικόνων.
3. Εισάγει τη βιβλιοθήκη pygame για αναπαραγωγή ήχων.

Δημιουργία της κλάσης του παιχνιδιού

```
class AdventureGame:

    def __init__(self):
```

- class AdventureGame:: Δημιουργεί μια κλάση με όνομα AdventureGame που περιέχει το παιχνίδι.
- def __init__(self):: Ορίζει τον constructor (αρχικοποιητή) της κλάσης, που εκτελείται αυτόματα όταν δημιουργείται ένα αντικείμενο της κλάσης.

Ρυθμίσεις παραθύρου και ήχου

```
self.window = tk.Tk()

self.window.title("Ο Χριστουγεννιάτικος Φάρος")

self.window.geometry("1080x1080")

self.window.configure(bg="black")
```

- self.window = tk.Tk(): Δημιουργεί το κύριο παράθυρο του παιχνιδιού.
- self.window.title("Ο Χριστουγεννιάτικος Φάρος"): Ορίζει τον τίτλο του παραθύρου.

- self.window.geometry("1080x1080"): Καθορίζει το μέγεθος του παραθύρου (1080x1080 pixels).
- self.window.configure(bg="black"): Ορίζει το χρώμα φόντου του παραθύρου σε μαύρο.

pygame.mixer.init()

- pygame.mixer.init(): Αρχικοποιεί το mixer του pygame για την αναπαραγωγή ήχων.

Ορισμός κειμένου ιστορίας

```
self.story_text = tk.StringVar()

self.story_text.set("Παραμονή Χριστουγέννων του έτους 1900.")
```

- self.story_text = tk.StringVar(): Δημιουργεί μια μεταβλητή τύπου StringVar που θα χρησιμοποιηθεί για το κείμενο της ιστορίας.
- self.story_text.set("Παραμονή Χριστουγέννων του έτους 1900."): Ορίζει το αρχικό κείμενο της ιστορίας.

Ετικέτα για την εμφάνιση του κειμένου

```
self.label = tk.Label(self.window, textvariable=self.story_text, wraplength=700, font=("Arial", 14), justify="center", bg="black", fg="white")

self.label.pack(pady=20)
```

- self.label = tk.Label(...): Δημιουργεί μια ετικέτα (Label) που εμφανίζει το κείμενο της ιστορίας:
- textvariable=self.story_text: Συνδέει την ετικέτα με το self.story_text ώστε να ενημερώνεται δυναμικά.
- wraplength=700: Ορίζει το μέγιστο πλάτος για το κείμενο (αναδίπλωση στις 700 μονάδες).
- font=("Arial", 14): Καθορίζει τη γραμματοσειρά και το μέγεθος.
- justify="center": Κεντράρει το κείμενο.

- bg="black", fg="white": Ορίζει το χρώμα φόντου (μαύρο) και το χρώμα κειμένου (λευκό).
- self.label.pack(pady=20): Τοποθετεί την ετικέτα στο παράθυρο με περιθώριο 20 pixels από πάνω και κάτω.

Ετικέτα για εικόνες

self.image_label = tk.Label(self.window, bg="black")

self.image_label.pack(pady=20)

- self.image_label = tk.Label(...): Δημιουργεί μια ετικέτα για την εμφάνιση εικόνων.
- self.image_label.pack(pady=20): Τοποθετεί την ετικέτα με περιθώριο 20 pixels.

Κουμπιά επιλογών

self.button1 = tk.Button(self.window, text="", width=30, bg="black", fg="white", highlightbackground="black")

self.button1.pack(pady=5)

- self.button1 = tk.Button(...): Δημιουργεί ένα κουμπί με:
- text="": Κενό κείμενο (θα ενημερωθεί αργότερα).
- width=30: Ορίζει το πλάτος του κουμπιού.
- bg="black", fg="white": Ορίζει το χρώμα φόντου και γραμματοσειράς.
- self.button1.pack(pady=5): Τοποθετεί το κουμπί με περιθώριο 5 pixels.

Οι ίδιες εντολές επαναλαμβάνονται για self.button2 και self.button3.

Κατάσταση παιχνιδιού και αρχικοποίηση

- self.current_state = "intro": Ορίζει την αρχική κατάσταση του παιχνιδιού.
- self.update_state("intro"): Καλεί τη συνάρτηση update_state για να φορτώσει την αρχική κατάσταση.
- self.window.mainloop(): Εκκινεί το κύριο βρόχο του Tkinter για το γραφικό περιβάλλον.

Συνάρτηση αναπαραγωγής ήχου

```
def play_sound(self, sound_file):
    try:
      pygame.mixer.music.load(sound_file)
      pygame.mixer.music.play(-1)
    except pygame.error as e:
      print(f"Error loading sound: {e}")
```

- def play_sound(self, sound_file):: Ορίζει μια συνάρτηση που φορτώνει και αναπαράγει έναν ήχο.
- pygame.mixer.music.load(sound_file): Φορτώνει το αρχείο ήχου.
- pygame.mixer.music.play(-1): Παίζει τον ήχο σε loop (επανάληψη).
- except pygame.error as e:: Αν υπάρξει σφάλμα, εμφανίζει μήνυμα σφάλματος.

Συνάρτηση διακοπής ήχου

```
  def stop_sound(self):
    pygame.mixer.music.stop()
```

- pygame.mixer.music.stop(): Σταματά την αναπαραγωγή του ήχου.

Συνάρτηση για ενημέρωση εικόνας

```
def update_image(self, image_path):
  try:
    img = Image.open(image_path)
    img = img.resize((420, 420), Image.Resampling.LANCZOS)
    photo = ImageTk.PhotoImage(img)
    self.image_label.config(image=photo)
    self.image_label.image = photo
  except Exception as e:
    print(f"Error loading image: {e}")
```

- `def update_image(self, image_path):: Ενημερώνει την εικόνα στην ετικέτα.`
- Image.open(image_path): Ανοίγει το αρχείο εικόνας.

- img.resize((420, 420), Image.Resampling.LANCZOS): Αλλάζει το μέγεθος της εικόνας.
- ImageTk.PhotoImage(img): Μετατρέπει την εικόνα σε μορφή που μπορεί να εμφανιστεί από το Tkinter.
- self.image_label.config(image=photo): Ενημερώνει την εικόνα στην ετικέτα.

Συνάρτηση ενημέρωσης κατάστασης παιχνιδιού

```
def update_state(self, state):
    self.current_state = state
    self.stop_sound()
```

- def update_state(self, state):: Ορίζει τη νέα κατάσταση του παιχνιδιού και σταματά τον προηγούμενο ήχο.

Η συνάρτηση αυτή περιέχει πολλές καταστάσεις (π.χ. intro, start, lighthouse), όπου αλλάζουν το κείμενο, η εικόνα και οι επιλογές κουμπιών.

```
self.button1.config(text="Επόμενη σελίδα", command=lambda: self.update_state("intro2"))
self.button2.config(text="", command=lambda: None)
self.button3.config(text="", command=lambda: None)
```

self.button1.config(...):

- *config είναι μια μέθοδος του Button που επιτρέπει να αλλάξουμε τις ιδιότητες (π.χ. κείμενο, εντολή) ενός κουμπιού που έχει ήδη δημιουργηθεί.*
- text="Επόμενη σελίδα": Αλλάζει το κείμενο του κουμπιού button1 ώστε να εμφανίζει "Επόμενη σελίδα".
- command=lambda: self.update_state("intro2"):
  - ❖ Ορίζει την εντολή που θα εκτελεστεί όταν ο χρήστης πατήσει το κουμπί.
  - ❖ Χρησιμοποιεί lambda για να καλέσει τη συνάρτηση self.update_state("intro2").
  - ❖ Με άλλα λόγια, όταν πατηθεί το κουμπί button1, η κατάσταση του παιχνιδιού θα αλλάξει σε "intro2".

self.button2.config(...):

- text="": Το κείμενο του κουμπιού button2 ορίζεται ως κενό (δεν εμφανίζεται τίποτα).
- command=lambda: None: Ορίζει την εντολή που θα εκτελεστεί όταν πατηθεί το κουμπί.
- lambda: None σημαίνει ότι δεν θα γίνει απολύτως τίποτα (δεν εκτελείται κάποια ενέργεια).
- Αυτό "απενεργοποιεί" ουσιαστικά το κουμπί, καθώς δεν έχει λειτουργία.

self.window.after(4000, self.window.destroy)

self.window.after(...):

- Η μέθοδος after ανήκει στο αντικείμενο παραθύρου (self.window), το οποίο είναι συνήθως ένα παράθυρο που δημιουργήθηκε με το Tkinter.
- Η after χρησιμοποιείται για να εκτελέσει μια συγκεκριμένη συνάρτηση μετά από καθυστέρηση που ορίζεται σε χιλιοστά του δευτερολέπτου.

4000:

- Ο αριθμός 4000 καθορίζει τη διάρκεια της καθυστέρησης σε χιλιοστά του δευτερολέπτου (ms).
- 4000 ms = 4 δευτερόλεπτα.
- Δηλαδή, η εντολή που ακολουθεί θα εκτελεστεί μετά από 4 δευτερόλεπτα.

self.window.destroy:

- Η destroy είναι μια μέθοδος που κλείνει/καταστρέφει το παράθυρο self.window.
- Όταν εκτελεστεί αυτή η εντολή, το παράθυρο του Tkinter κλείνει και η εφαρμογή σταματά να εμφανίζεται στην οθόνη.

if __name__ == "__main__":

AdventureGame()

- __name__ είναι μια ειδική μεταβλητή που ορίζεται αυτόματα σε κάθε αρχείο Python.
- Όταν ένα αρχείο εκτελείται ως κύριο πρόγραμμα (δηλαδή απευθείας από το χρήστη, και όχι μέσω import από άλλο αρχείο), η τιμή της μεταβλητής __name__ είναι "main".
- Αντίθετα, όταν το αρχείο εισάγεται ως module σε άλλο πρόγραμμα, η τιμή του __name__ γίνεται το όνομα του αρχείου.
- AdventureGame(): Εκτελεί τον κώδικα της κλάσης που δημιουργήθηκε στην αρχή.