Chapitre 4 : Entrées sorties plus évoluées

Audras

Lycée Saint-Just

année 2018-2019 ISN

1) Les fichiers

Définition

Pour sauvegarder des données lorsqu'on ferme l'application, il faut les enregistrer sur un support, disque ou clé usb, à l'intérieur d'un fichier.
On peut les écrire en données brut sous la forme de texte de type str ou sous la forme d'objets comme des types list ou dict.
Avant d'ouvrir ou de fermer un fichier il faut se placer dans le repertoire de son choix. Les lignes suivantes décrivent comment se placer dans le repertoire où se trouve le programme monProg.py.
import os ; Rep=__file__.split("\\") pour créer une liste à partir du nom complet de monProg.py, séparée par les \
NomFichier=Rep.pop() enlève monProg.py de la liste, il reste le nom du répertoire.
Rep="\\".join(Rep) fait une phrase avec la liste en replaçant les séparateurs et os.chdir(Rep) change de repertoire.

Propriété

width open suivit d'un bloc ouvre un fichier en lecture ou en écriture et le ferme à la sortie du bloc. 'w' pour write, 'r' pour read et 'a' pour append. L'encodind permet de traiter les caractères accentués.
Trois guillement pour du texte sur plusieurs lignes et \n pour un retour à la ligne.

```
 8 with open("data", "w",encoding='utf8') as mon_fichier:
 9         mon_fichier.write("""bonjour
10 et au revoir""")
11
12 with open("data", "a",encoding='utf8') as mon_fichier:
13         mon_fichier.write("\nà la ligne.")
14
15 with open("data", "r",encoding='utf8') as mon_fichier:
16         texte=mon_fichier.read()
17
18 print(texte)
```

Exemple

```
1  import os
2  Rep=__file__.split("\\")
3  NomFichier=Rep.pop()
4  Rep="\\".join(Rep)
5  os.chdir(Rep)
6  bd="Aldebaran,Léo,1,La catastrophe"
7  with open("bd", "w",encoding='utf8') as mesBds:
8          mesBds.write(bd)
9  bd="\nAldebaran,Léo,2,La blonde"
10 with open("bd", "a",encoding='utf8') as mesBds:
11         mesBds.write(bd)
12 with open("bd", "r",encoding='utf8') as mesBds:
13         listeBds=mesBds.read().split("\n")
14 print(listeBds)
15 for i in range(len(listeBds)):
16     Bd=listeBds[i].split(",")
17     series=Bd[0] ; num=Bd[2] ; title=Bd[3]
18     print("le titre du n°{} de {} est {}".format(num,series
19                                                  ,title))
```

Pour garder la structure des objets python il faut utiliser pickle. 'wb' et 'rb' pour write binary et read binary.
Pour modifier le contenu du fichier il faut charger l'objet, le modifier et le sauvegarder à nouveau en écrasant l'ancien.

```
20 import pickle
21 with open("dataB", "wb") as mon_fichier:
22         pickle.dump({"nom":"audras", "classe":"ISN",
23                                     "élèves":24},mon_fichier)
24 with open("dataB", "rb") as mon_fichier:
25         monDict=pickle.load(mon_fichier)
26 print(monDict)
```

Cela fonctionne avec n'importe quelle objet comme une liste ou autre.

2) Les images avec Pillow

Définition

from PIL import Image, ImageTk, ImageFilter,ImageEnhance.
`https://pillow.readthedocs.io/en/5.2.x/`
Image permet d'importer une image avec monImagePil=Image.open("nomPhoto.jpeg"), de l'afficher en dehors de Tkinter avec monImagePil.show() et offre tout un tas de fonction pour accéder aux pixels et les modifier :
ImageFilter offre des filtres prédéfinis à appliquer sur l'image.
ImageEnhance permet de travailler la luminosité, le contraste,...
Pour sauvegarder l'image, monImagePil.save("nomPhoto.jpeg")

Propriété

Pour utiliser l'image dans Tkinter partout où il y a l'option *image*=, on utilise ImageTk.PhotoImage() :
monImageTkinter=ImageTk.PhotoImage(monImagePil) ou directement =ImageTk.PhotoImage(Image.open("nomPhoto.jpeg"))
Attention : il ne faut pas effacer la variable monImageTkinter sinon l'image ne s'affiche pas, en effet les images sont chargées à l'écran lors de l'instruction mainloop().
Pour créer une image pixel par pixel en noir et blanc (transparent en fait), monImagePil=Image.new() et monImagePil.putpixel((x,y),0 ou 1) et dans Tkinter,
monImageTkinter=ImageTk.BitmapImage(monImagePil).

Exemple

Créer une matrice de 0 et de 1 de taille 200x200 aléatoire, en faire une image, l'enregistrer et l'afficher avec Tkinter.

```
1  from tkinter import *
2  from PIL import Image, ImageTk
3  import random
4  import os ; Rep=__file__.split("\\") ; NomFichier=Rep.pop()
5  Rep="\\".join(Rep) ; os.chdir(Rep)
6
7  matImage=Image.new("1",(200,200))
8  for i in range(200):
9      for j in range(200):
10         matImage.putpixel((i,j),random.randint(0,1))
11 matImage.save("matImage.png")
12
13 ihm=Tk()
14 matImageTk=ImageTk.BitmapImage(matImage)
15 Label(ihm,image=matImageTk).grid(row=0,column=0)
16 ihm.mainloop()
```

3) Les sons avec wave

Définition

wave fait partie de la bibliothèque standard, voir
`https://docs.python.org/3/library/wave.html`
Un son est habituellement la répétition d'un motif à une certaine fréquence qui donne la hauteur du son. Pour la voix, on pourra prendre f=300Hz. Les morceaux sont échantillonnés à 44100Hz donc 44100/f est le nombre d'entiers qu'il faut pour le motif de base. Ces entiers sont compris entre 0 et 255, l'absence de pression correspondant à la moitié, 127.5 arrondi à 128.
Les entiers doivent être codés en binaire, on utilise la fonction pack du module struct qui fournit le type bytes.
`https://docs.python.org/3/library/struct.html`
On utilise les fonctions spéciales objet.__add__(n) et objet.__mul__(n) sous leur forme additive et multiplicative objet+objet et objet*n qui fonctionnent avec des str, list et aussi avec des bytes.

Propriété

On écoute le résultat avec VLC et on peut l'encoder en mp3 avec audacity.

```
1  from struct import pack
2  import wave
3  import random
4  import os ; Rep=__file__.split("\\") ; NomFichier=Rep.pop()
5  Rep="\\".join(Rep) ; os.chdir(Rep)
6
7  duration=0.8
8  freq=300
9  num=int(round(44100/freq))
10 listSound=pack('',)
11 for i in range(num):
12     listSound+=pack('B',random.randint(0,255))
13 listSound*=int(round(freq*duration))
14
15 with wave.open("mySound.wav", 'wb') as mySound:
16     mySound.setparams((1, 1, 44100, 0, 'NONE', 'not compressed'))
17     mySound.writeframes(listSound)
```