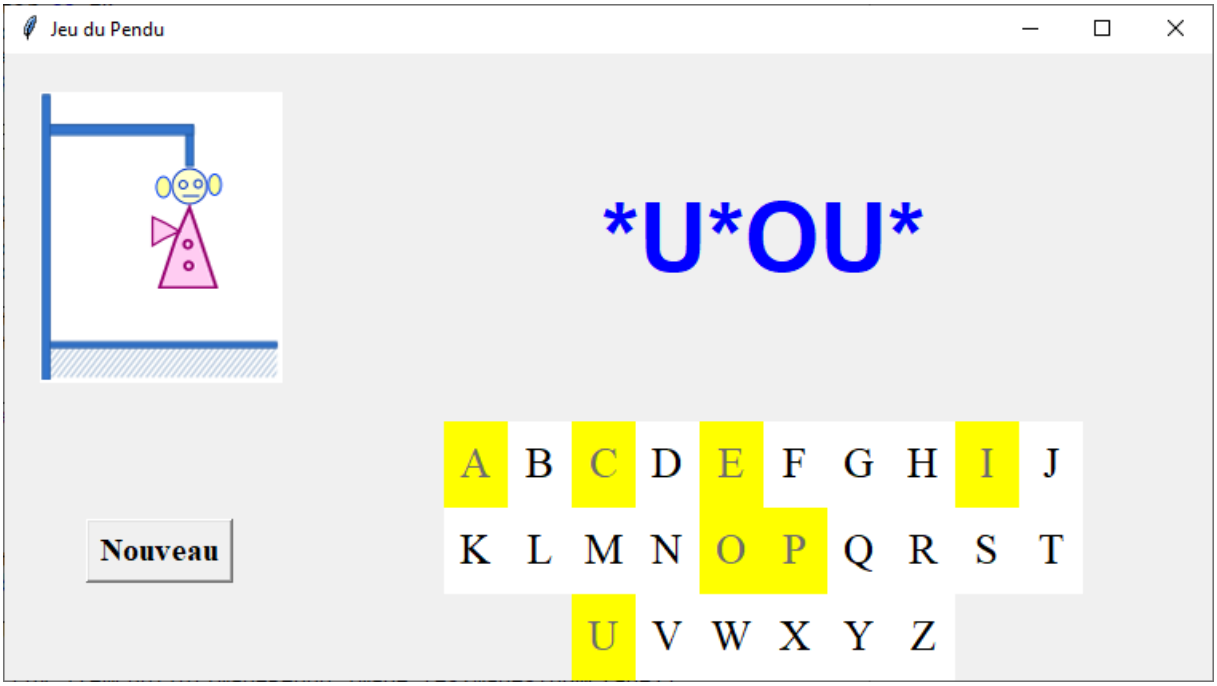
Pendu : Graphique

Le jeu



Dans ce jeu, l'ordinateur choisit un mot et le joueur doit trouver le mot en proposant différentes lettres.

Si la lettre proposée appartient au mot, l'ordinateur affiche la lettre dans le mot à tous les emplacements où elle apparaît.

Sinon, un nouveau morceau du pendu est affiché.

Lorsque le pendu est complet, le joueur a perdu et l'ordinateur affiche le mot complet.

Si toutes les lettres du mot sont trouvées, le joueur gagne.

Analyse du programme à réaliser

Nous repartons de la version console.

Les images du pendu

Ces images sont tirées de la version scratch du jeu.

Les différents costumes du lutin « pendu » ont été exportés sous forme d'images png.

Tkinter ne permet pas d'afficher directement des images au format png. Nous utiliserons deux fonctions des bibliothèques Image et ImageTk du module PIL (Python Image Library).

Une liste lesImages mémorisera les références aux 9 images lorsqu'elles seront chargées en mémoire.

Nous créerons ensuite un **Canvas**, qui affichera l'image correspondant à l'étape du jeu.

La liste des mots secrets et le choix d'un mot

Les mots secrets seront contenus dans un fichier texte.

Il faudra donc constituer la liste liste_mots à partir de ce fichier.

Dans le fichier, chaque mot est inscrit sur une ligne. Il faudra donc supprimer le saut de ligne lorsqu'on aura choisi le mot secret par la fonction **rstrip()**

secret = secret.rstrip()

Le mot en cours de progression

Au départ ce mot est une liste constituée d'autant d'étoiles que le mot secret comporte de lettres.

Pour l'afficher nous utiliserons un **label**.

Le choix d'une lettre par le joueur

Le joueur disposera d'un clavier affichant les lettres de l'alphabet.

Un clic sur une lettre provoquera l'appel de la fonction choisir_lettre

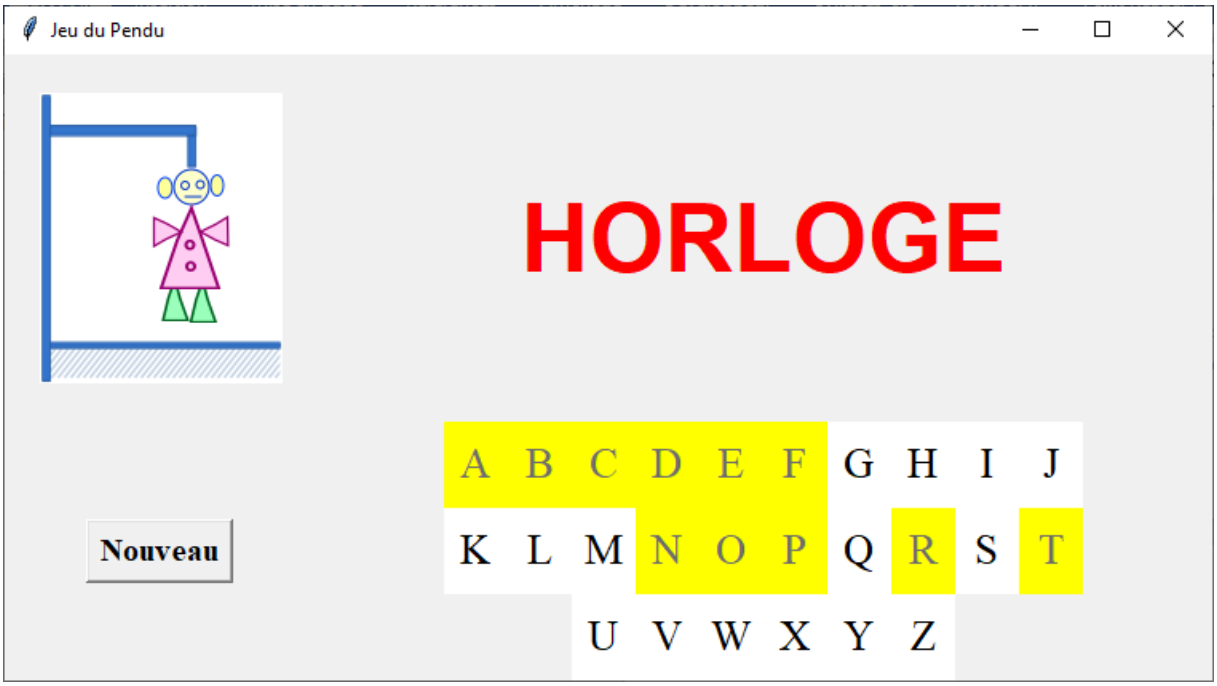
Les lettres déjà proposées

Pour empêcher que le joueur ne propose plusieurs fois la même lettre, dès qu'une lettre sera cliquée, le bouton correspondant ne sera plus « cliquable » et son fond sera mis en jaune, pour bien indiquer que la lettre a déjà été proposée.

Test de fin de partie

Lorsque le joueur gagne, le mot secret est affiché en vert dans le label affichant la liste « motProgression »

Lorsque le joueur perd, le mot secret est affiché en rouge dans le label affichant la liste « motProgression »



Une autre partie ?

Un bouton « Nouveau » permettra de lancer une nouvelle partie.

Le programme : l'interface

```
import tkinter as tk
import random as rn
import traceback
from PIL import Image, ImageTk
#----------------------------Les fonctions ------------------
Voir ci-dessous
#---------Construction de l'interface --------------------
#Les variables
largeur_image = 155
hauteur_image = 190
etape = 8
motProgression=[]
secret =""
```

```
fin = False
numEtape = 0
#lecture du fichier de mots
fichier = open("liste_mots.txt", "r")
# met tous les mots du fichier dans une liste
liste_mots = fichier.readlines()
fichier.close()
#----------
ecran=tk.Tk()
ecran.title("Jeu du Pendu")
#charger les images et les mettre dans la liste lesImages
lesImages=[]
for i in range(etape+1):
    load = Image.open("Pendu_etape%s.png"%i)
    imageP = ImageTk.PhotoImage(load)
    lesImages.append(imageP)
#Créer un canvas, créer une première image et mettre sa référence dans la variable
#ImagePendu que l'on modifira par la suite.
cnv = tk.Canvas(ecran,width=largeur_image,height=hauteur_image,highlightthickness=0)
cnv.grid(row=0,column=0,padx=20, pady=20)
ImagePendu = cnv.create_image((largeur_image/2,hauteur_image/2),image=lesImages[0])
#Créer un label affichant la progression de la recherche
textMotProgres=tk.StringVar()
lbMotProgres = tk.Label(ecran, textvariable=textMotProgres, font=('Deja Vu Sans Mono', 45, 'bold'), width=15, fg="blue")
lbMotProgres.grid(row=0, column=1)
# Rejouer
reset = tk.Button(ecran, text="Nouveau", font="Times 15 bold", command=init)
```

```
reset.grid(row=1, column=0,padx=2, pady=2)
#Création d'un clavier pour entrer les lettres proposées
lettres = tk.Frame(ecran)
lettres.grid(row=1, column=1)
#2 rangées de boutons de 10 lettres et une rangée de 6 lettres
#un clic sur un bouton déclenche l'exécussion de la fonction
#choisir_lettre
ALPHA = "ABCDEFGHIJKLMNOPQRSTUVWXYZ"
lesBoutons=[]
for i in range(2):
    for j in range(10):
        bouton = tk.Button(lettres, text=ALPHA[10 * i + j],relief = tk.FLAT,font='times 20', bg='white',width=2)
        bouton.grid(row=i, column=j)
        bouton.bind("<Button-1>", choisir_lettre)
        lesBoutons.append(bouton)
for j in range(6):
    bouton = tk.Button(lettres, text=ALPHA[20 + j], relief= tk.FLAT, font='times 20', bg='white',width=2)
    bouton.grid(row=2, column=j + 2)
    bouton.bind("<Button-1>", choisir_lettre)
    lesBoutons.append(bouton)
init()
ecran.mainloop()
ecran.quit()
```

Les fonctions

Initialisation : appelée dès que l'interface est créée et aussi par le bouton « reset ».

```
def init():
    global ImagePendu,secret,motProgression,fin,numEtape
    # prend au hasard un mot dans la liste
    secret = rn.choice(liste_mots)
    # supprime le caractère "saut à la ligne"
    secret = secret.rstrip()
    #constitue le mot en cours de progression
    motProgression = list(''.join("*"*len(secret)))
    #motifie le texte du label
    lbMotProgres["fg"] = 'blue'
    textMotProgres.set("".join(motProgression))
    #affiche l'image de départ
    cnv.itemconfig(ImagePendu,image=lesImages[0])
    #reinitialise les boutons du clavier
    for bouton in lesBoutons:
        bouton["state"] = tk.NORMAL
        bouton["bg"] = 'white'
    fin=False
    numEtape = 0
```

Le choix d'une lettre : cette fonction est appelée lors d'un clic sur un des boutons du clavier.

Nous devons déterminer quel est le bouton cliqué. Pour cela nous utilisons le paramètre event qui accompagne tous les évènements liés à un widget. Ce paramètre va nous fournir l'information attendue.

leBouton = event.widget fournit la référence du bouton qui a provoqué l'évènement.

Il suffit ensuite de modifier les attributs de ce bouton, de récupérer le texte affiché sur le bouton etc.

```
def choisir_lettre(event):
    #Si la partie est finie, on ne prend pas le clic en compte
    if fin:
        return
    #Sur quel bouton a t'on cliqué
    leBouton = event.widget
    #Récupérer la lettre affichée sur le bouton
    lettre = leBouton["text"]
    #Rendre le bouton non cliquable et changer sa couleur de fond
    leBouton["state"] = tk.DISABLED
    leBouton["bg"]= 'yellow'
    #Mettre à jour la liste motProgression et l'afficher
    maj_motProgres(lettre)
    textMotProgres.set("".join(motProgression))
    #tester cette étape
    test_etape(lettre)
```

La fonction de mise à jour de la liste motProgression est la même que celle de la version console.

Tester cette étape du jeu :

```
def test_etape(lettre):
    global numEtape, fin, ImagePendu
    #Si la lettre n'est pas dans le mot secret
    #on affiche l'image suivante du pendu
    if lettre not in secret:
        numEtape+=1
        cnv.itemconfig(ImagePendu,image=lesImages[numEtape])
```

```
#Le joueur a t'il gagné ?
 j=0
 for i in range (len(motProgression)):
    if motProgression[i]!='*':
        j+=1
 if j == len(motProgression):
    #il a gagné
    fin = True
    lbMotProgres["fg"]='green'
    return

 #Le joueur a t'il perdu?
 if numEtape==8:
    fin = True
    lbMotProgres["fg"]='red'
    textMotProgres.set(secret)
```