Interfaces graphiques

Ouverture hors programme

Ouverture hors programme. Les interfaces graphiques ne sont pas au programme de NSI, mais elles rendent les projets plus attrayants. Cette fiche présente trois bibliothèques : `tkinter`, fournie avec Python, `pygame` pour les jeux et `streamlit` pour les applications web. Les deux dernières s'installent avec `pip install pygame` et `pip install streamlit`.

Une interface graphique (*GUI*, *graphical user interface*) se programme toujours de la même façon : on crée une fenêtre et ses composants (boutons, champs de saisie, textes), on associe des fonctions aux actions de l'utilisateur, puis on lance une **boucle d'événements** qui attend ces actions.

Tkinter

Tkinter, exemple 1 : saisie et affichage

L'exemple suivant crée une fenêtre avec un texte, un champ de saisie et un bouton. La fonction `afficher` est appelée quand on valide la saisie avec la touche Entrée.

```
import tkinter as tk

def afficher(event):
    """Affiche un message de bienvenue avec le prénom saisi."""
    accueil.configure(text="Bonjour " + champ.get() + " !")

# Fenêtre principale
fenetre = tk.Tk()
fenetre.title("Ma première fenêtre")
fenetre.minsize(width=300, height=80)

# Un texte, un champ de saisie et un bouton, empilés dans l'ordre
consigne = tk.Label(fenetre, text="Entrez votre prénom puis validez :")
consigne.pack()

champ = tk.Entry(fenetre, width=30)
champ.bind("<Return>", afficher)      # touche Entrée : appel de afficher
champ.pack()

accueil = tk.Label(fenetre, text="")
accueil.pack()

bouton_quitter = tk.Button(fenetre, text="Quitter", command=fenetre.destroy)
bouton_quitter.pack()

# Boucle d'événements, interrompue quand on ferme la fenêtre
fenetre.mainloop()
```

Les composants de l'interface (*widgets*) sont placés les uns sous les autres, dans l'ordre des appels à `pack()`. La méthode `grid()` de l'exemple suivant permet de les disposer dans une grille.

Tkinter, exemple 2 : conversion Fahrenheit vers Celsius

```python
import tkinter as tk

def convertir(event=None):
    """Convertit la température saisie, en degrés Fahrenheit, en degrés Celsius."""
    fahrenheit = float(temperature.get())
    celsius = 5 / 9 * (fahrenheit - 32)
    resultat.config(text=f"{celsius:.2f} °C")

fenetre = tk.Tk()
fenetre.title("Conversion de Fahrenheit en Celsius")

consigne = tk.Label(fenetre, text="Température en degrés Fahrenheit :")
temperature = tk.Entry(fenetre)
temperature.bind("<Return>", convertir)
resultat = tk.Label(fenetre, text="")
bouton_convertir = tk.Button(fenetre, text="Convertir", command=convertir)
bouton_quitter = tk.Button(fenetre, text="Quitter", command=fenetre.destroy)

# Disposition en grille : ligne (row) et colonne (column)
consigne.grid(row=0, column=0, sticky="w", padx=5, pady=5)
temperature.grid(row=0, column=1, padx=5, pady=5)
bouton_convertir.grid(row=1, column=1, padx=5, pady=5)
resultat.grid(row=1, column=0, sticky="w", padx=5, pady=5)
bouton_quitter.grid(row=2, column=1, padx=5, pady=5)

fenetre.mainloop()
```

Le paramètre `event=None` permet d'appeler `convertir` aussi bien depuis le bouton (sans argument) que depuis la touche Entrée (avec un argument décrivant l'événement).

Tkinter, exemple 3 : image de fond

Pour afficher une photo, on utilise la bibliothèque Pillow (module `PIL`), qui s'installe avec `pip install pillow`. L'image est placée dans une étiquette (`Label`) qui occupe toute la fenêtre ; le titre et le bouton sont ensuite posés **sur** cette étiquette avec `place()`, qui les positionne en proportion de sa taille : `relx=0.5` signifie « au milieu de la largeur ». Le programme suppose qu'une image nommée `paysage.jpg` se trouve dans le même dossier.

```python
import tkinter as tk
from PIL import Image, ImageTk

fenetre = tk.Tk()
fenetre.title("Image de fond")

# Ouverture de l'image (fichier placé dans le même dossier que le programme)
image = Image.open("paysage.jpg")
image = image.resize((600, 400))
photo = ImageTk.PhotoImage(image)

# L'image occupe toute la fenêtre ; les autres widgets sont placés par-dessus
fond = tk.Label(fenetre, image=photo)
fond.pack()

titre = tk.Label(fond, text="J'adore Python !", font=("Arial", 20))
```

```
titre.place(relx=0.5, rely=0.3, anchor="center")

bouton = tk.Button(fond, text="Quitter", command=fenetre.destroy)
bouton.place(relx=0.5, rely=0.7, anchor="center")

fenetre.mainloop()
```

La variable `photo` doit rester définie tant que la fenêtre est ouverte : si l'image n'était plus stockée dans aucune variable, Python la supprimerait de la mémoire et la fenêtre resterait vide.

Pygame

Pygame dessine dans une fenêtre et gère le clavier, la souris et le temps. Un programme Pygame est une **boucle** qui, à chaque tour, lit les événements, met à jour l'état du jeu, dessine l'écran et attend le tour suivant.

Pygame, exemple 1 : déplacer un carré au clavier

À chaque tour de boucle, `pygame.key.get_pressed()` indique quelles touches sont enfoncées : tant qu'une flèche est maintenue, le carré avance de 4 pixels par image dans sa direction. Les fonctions `min` et `max` le retiennent à l'intérieur de la fenêtre.

```
import pygame

FOND = (30, 30, 60)
JAUNE = (250, 200, 40)
LARGEUR, HAUTEUR = 600, 400
COTE = 40

pygame.init()
ecran = pygame.display.set_mode((LARGEUR, HAUTEUR))
pygame.display.set_caption("Déplacer un carré au clavier")
horloge = pygame.time.Clock()

# Position du coin supérieur gauche du carré, et vitesse en pixels par image
x, y = 280, 180
vitesse = 4

fini = False
while not fini:
    # 1. Événements : seule la fermeture de la fenêtre nous intéresse ici
    for evenement in pygame.event.get():
        if evenement.type == pygame.QUIT:
            fini = True

    # 2. Mise à jour : on regarde quelles flèches sont enfoncées
    touches = pygame.key.get_pressed()
    if touches[pygame.K_LEFT]:
        x = x - vitesse
    if touches[pygame.K_RIGHT]:
        x = x + vitesse
    if touches[pygame.K_UP]:
        y = y - vitesse
    if touches[pygame.K_DOWN]:
```

```
        y = y + vitesse

    # Le carré ne doit pas sortir de la fenêtre
    x = max(0, min(x, LARGEUR - COTE))
    y = max(0, min(y, HAUTEUR - COTE))

    # 3. Dessin
    ecran.fill(FOND)
    pygame.draw.rect(ecran, JAUNE, [x, y, COTE, COTE])
    pygame.display.flip()

    # 4. Attente : 60 images par seconde au plus
    horloge.tick(60)

pygame.quit()
```

L'origine des coordonnées est le coin **supérieur** gauche de la fenêtre : `y` augmente quand on descend, c'est pourquoi la flèche vers le bas ajoute la vitesse à `y`.

Pygame, exemple 2 : attraper une cible à la souris

Une cible ronde apparaît à un endroit tiré au hasard. Un clic de souris produit un événement `MOUSEBUTTONDOWN`, dont l'attribut `pos` donne la position du clic. Si le clic tombe dans le disque, le score augmente et la cible change de place.

```
import random
import pygame

BLANC = (255, 255, 255)
ROUGE = (220, 40, 40)
NOIR = (0, 0, 0)
LARGEUR, HAUTEUR = 600, 400
RAYON = 25

pygame.init()
ecran = pygame.display.set_mode((LARGEUR, HAUTEUR))
pygame.display.set_caption("Attraper la cible")
horloge = pygame.time.Clock()
police = pygame.font.SysFont("Arial", 24)

def nouvelle_position():
    """Renvoie un centre au hasard, la cible restant entière dans la fenêtre."""
    return (random.randint(RAYON, LARGEUR - RAYON),
            random.randint(RAYON + 40, HAUTEUR - RAYON))

centre = nouvelle_position()
score = 0

fini = False
while not fini:
    for evenement in pygame.event.get():
        if evenement.type == pygame.QUIT:
            fini = True
        elif evenement.type == pygame.MOUSEBUTTONDOWN:
            # Distance entre le clic et le centre de la cible
            dx = evenement.pos[0] - centre[0]
```

```
            dy = evenement.pos[1] - centre[1]
            if dx * dx + dy * dy <= RAYON * RAYON:
                score = score + 1
                centre = nouvelle_position()

    ecran.fill(BLANC)
    pygame.draw.circle(ecran, ROUGE, centre, RAYON)
    texte = police.render(f"Score : {score}", True, NOIR)
    ecran.blit(texte, (10, 10))
    pygame.display.flip()
    horloge.tick(60)

pygame.quit()
```

Le clic est dans le disque quand sa distance au centre est au plus égale au rayon. Comparer les **carrés** de ces deux longueurs évite de calculer une racine carrée.

Pour aller plus loin, la documentation de Pygame (en anglais) décrit toutes les fonctions et propose des tutoriels.

Streamlit

Streamlit transforme un script Python en application web : chaque appel à `st.…` ajoute un composant à la page. On exécute le script avec la commande `streamlit run mon_fichier.py`, qui ouvre l'application dans le navigateur ; le script est réexécuté à chaque action de l'utilisateur.

```
import streamlit as st

st.title("Convertisseur de températures")

fahrenheit = st.slider("Température en degrés Fahrenheit", -40, 120, 68)
celsius = 5 / 9 * (fahrenheit - 32)
st.write(f"{fahrenheit} °F correspondent à {celsius:.1f} °C.")

gauche, droite = st.columns(2)
if gauche.button("Eau qui gèle"):
    droite.write("0 °C, soit 32 °F")
if gauche.button("Eau qui bout"):
    droite.write("100 °C, soit 212 °F")

with st.expander("Comment est faite la conversion ?"):
    st.write("On retire 32, puis on multiplie par 5/9.")
    st.code("celsius = 5 / 9 * (fahrenheit - 32)")
```