Tkinter

Fiche de synthèse

1 Programmation événementielle

Ouverture hors programme, fournie avec Python. Le programme crée les **widgets**, leur associe des fonctions, puis lance la **boucle d'événements** `mainloop()`, qui attend les actions de l'utilisateur (clic, touche) et appelle les fonctions associées.

```
import tkinter as tk

compteur = 0

def cliquer():
    global compteur
    compteur = compteur + 1
    texte.config(text=f"Clics : {compteur}")

fenetre = tk.Tk()
fenetre.title("Compteur")
texte = tk.Label(fenetre, text="Clics : 0")
texte.pack()
bouton = tk.Button(fenetre, text="Cliquer",
                   command=cliquer)
bouton.pack()
fenetre.mainloop()
```

— `command=cliquer` : le **nom** de la fonction, sans parenthèses. Avec `cliquer()`, elle serait appelée tout de suite, une seule fois.

— `global` pour modifier une variable du programme principal.

— `config(text=...)` modifie un widget affiché.

2 Saisie

```
def verifier(event=None):
    saisie = champ.get()       # une chaîne
    if saisie.isdigit() and int(saisie) == 56:
        message.config(text="Bravo")
    champ.delete(0, "end")     # vider

champ.bind("<Return>", verifier)
```

`get()` renvoie toujours une chaîne : la tester, puis la convertir. Une fonction liée par `bind` reçoit un argument `event`.

3 Disposition

Une méthode par conteneur : `pack()` empile, `grid(row=..., column=...)` range en grille (`sticky`, `columnspan`), `place(x=..., y=...)` positionne. Un `Frame` regroupe des widgets. Créer et placer en deux lignes : `grid()` renvoie `None`.

4 Le temps et le dessin

```
def decompter():
    global restant
    if restant > 0:
        restant = restant - 1
        fenetre.after(1000, decompter)
```

Jamais de `while` ni de `time.sleep` dans un gestionnaire : la fenêtre se figerait. Un `Canvas` dessine (`create_rectangle`, `create_oval`, `move`, `coords`) ; l'origine est en **haut à gauche**, y augmente vers le bas. Un bouton qui transmet une valeur : `command=lambda c=c: ajouter(c)`.

À retenir.

— Créer, placer, associer, puis `mainloop()` en dernier.

— `command=f` sans parenthèses ; `get()` renvoie une chaîne.

— `after(ms, f)` remplace `while` et `sleep`.