
📘 Guide Tkinter : grid, largeur en pixels et redimensionnement

1. Introduction

Tkinter est le module standard de Python pour créer des interfaces graphiques.
Le gestionnaire de placement **grid** permet d'organiser les widgets en lignes et colonnes.
Ce guide explique comment :

- Fixer la largeur/hauteur en pixels.
- Gérer les espacements (padx, pady).
- Centrer ou étirer les widgets.
- Adapter automatiquement les tailles lors du redimensionnement de la fenêtre.
- Créer une mise en page symétrique et ergonomique.

---

2. Positionnement avec grid

- grid(row, column) place un widget dans une cellule.
- Les cellules s'adaptent au contenu par défaut.
- Pour contrôler la largeur :
    - grid_columnconfigure(col, minsize=px) fixe une largeur minimale.
    - Utiliser un Frame avec width et height en pixels + grid_propagate(False) pour forcer une taille.

---

3. Fixer la largeur en pixels

Les widgets (Label, Entry, Button) utilisent width en **caractères**, pas en pixels.
Pour une taille en pixels :

frame = tk.Frame(root, width=200, height=200)

frame.grid(row=0, column=0)

frame.grid_propagate(False) # Empêche le redimensionnement automatique

👉 On place ensuite un widget (Text, Label, etc.) à l'intérieur du Frame.

---

4. Espacements (padx, pady)

- padx = marge horizontale (pixels).
- pady = marge verticale (pixels).

widget.grid(row=1, column=2, padx=89, pady=100)

---

5. Centrage et étirement

- Par défaut, grid centre le widget dans sa cellule.
- Pour étirer : sticky="nsew".

widget.grid(row=1, column=1, sticky="nsew")

---

6. Redimensionnement automatique

a. Avec weight

- grid_rowconfigure et grid_columnconfigure répartissent l'espace.

fen.grid_columnconfigure(0, weight=1)

fen.grid_rowconfigure(0, weight=1)

b. Avec <Configure>

- On peut recalculer dynamiquement la taille des widgets proportionnellement à la fenêtre.

```
def resize(event):
  w, h = event.width, event.height
  frame.config(width=w//4, height=h//3)


fen.bind("<Configure>", resize)
```

7. Mise en page symétrique

Exemple d'organisation :

- Bouton 1 centré en haut.
- Deux zones de texte côte à côte au milieu.
- Bouton 2 sous la zone de gauche.
- Bouton 3 sous la zone de droite.

---

8. Exemple de code

```
import tkinter as tk

fen = tk.Tk()

fen.geometry("800x600")

# Configurer la grille

for r in range(3):

  fen.grid_rowconfigure(r, weight=1)

for c in range(5):

  fen.grid_columnconfigure(c, weight=1)

# Bouton 1 centré en haut

btn1 = tk.Button(fen, text="Bouton 1")

btn1.grid(row=0, column=2, pady=20)

# Zone de texte

frame_text1 = tk.Frame(fen, bg="lightblue")

frame_text1.grid(row=1, column=1, padx=40, pady=20, sticky="nsew")

frame_text1.grid_propagate(False)

text1 = tk.Text(frame_text1)

text1.pack(expand=True, fill="both")
```

```
# Bouton 2 en bas
btn2 = tk.Button(fen, text="Bouton 2")
btn2.grid(row=2, column=1, pady=20)

# Resize proportionnel
def resize(event):
    w, h = fen.winfo_width(), fen.winfo_height()
    new_w = max(100, w // 4)   # 1/4 largeur
    new_h = max(100, h // 3)   # 1/3 hauteur
    frame_text1.config(width=new_w, height=new_h)
fen.bind("<Configure>", resize)
fen.mainloop()
```
