Programmation évènementielle en Python

Gestion des positionnements

Aurel Megnigbeto

Université d'Abomey-Calavi

École Nationale d'Économie Appliquée et de Management

4 mars 2025

Plan

Méthodes de positionnement

Il y a plusieurs méthodes pour positionner les widgets dans une fenêtre Tkinter. On peut citer :

- `pack` : Puissant mais pas trivial à comprendre
- `grid` : Puissant et plus intuitif
- `place` : Contrôle absolu sur le positionnement (équivaut à `position: absolute` en CSS)

Attention

Il est déconseillé de mélanger les méthodes de positionnement dans une même fenêtre.

Nous allons nous concentrer sur la méthode `grid` dans ce cours.

Méthodes de positionnement

Grid

Principe

- Permet de positionner les widgets dans une fenêtre en utilisant un système de coordonnées.
- Utilise un système de lignes et de colonnes pour positionner les widgets.

- Les coordonnées sont des entiers qui représentent la ligne et la colonne où le widget doit être placé.
- Les lignes et les colonnes sont numérotées à partir de 0.

Méthodes de positionnement

`Grid`

- Tous les widgets qui ont la même colonne sont alignés verticalement l'un par rapport à l'autre.
- Tous les widgets qui ont la même ligne sont alignés horizontalement l'un par rapport à l'autre.

Attention

Les widgets ne sont pas placés dans la fenêtre tant qu'on ne les a pas positionnés avec la méthode `grid`.

Exemple de positionnement avec grid

Partie 1

```
from tkinter import *
from tkinter import ttk

root = Tk()

content = ttk.Frame(root)
frame = ttk.Frame(content, borderwidth=5, relief="sunken", width=200, height=100)
namelbl = ttk.Label(content, text="Name")
name = ttk.Entry(content)

onevar = BooleanVar()
twovar = BooleanVar()
threevar = BooleanVar()
onevar.set(True)
twovar.set(False)
threevar.set(True)

content.grid(column=0, row=0, sticky=(N, S, E, W))
frame.grid(column=0, row=0, columnspan=3, rowspan=2, sticky=(N, S, E, W))
namelbl.grid(column=3, row=0, columnspan=2, sticky=(N, W), padx=5)
name.grid(column=3, row=1, columnspan=2, sticky=(N, E, W), pady=5, padx=5)

one = ttk.Checkbutton(content, text="One", variable=onevar, onvalue=True)
```

Exemple de positionnement avec grid

Partie 2

```
one = ttk.Checkbutton(content, text="One", variable=onevar, onvalue=True)
two = ttk.Checkbutton(content, text="Two", variable=twovar, onvalue=True)
three = ttk.Checkbutton(content, text="Three", variable=threevar, onvalue=True)
ok = ttk.Button(content, text="Okay")
cancel = ttk.Button(content, text="Cancel")

content.grid(column=0, row=0)
frame.grid(column=0, row=0, columnspan=3, rowspan=2)
namelbl.grid(column=3, row=0, columnspan=2)
name.grid(column=3, row=1, columnspan=2)
one.grid(column=0, row=3)
two.grid(column=1, row=3)
three.grid(column=2, row=3)
ok.grid(column=3, row=3)
cancel.grid(column=4, row=3)

root.mainloop()
```

Positionnement interne des widgets

Il se peut que le widget ait une largeur inférieure à celle de la cellule dans laquelle il est placé.
Dans ce cas, on veut pouvoir positionner le widget dans la cellule en faisant usage de l'option `sticky`.

- `sticky` permet de spécifier comment le widget doit être positionné dans la cellule.
- Les valeurs possibles sont : `N`, `S`, `E`, `W`, `NE`, `NW`, `SE`, `SW`

Positionnement interne des widgets

- `N` : Nord
- `S` : Sud
- `E` : Est
- `W` : Ouest
- `NE` : Nord-Est
- `NW` : Nord-Ouest
- `SE` : Sud-Est
- `SW` : Sud-Ouest

Gérer le redimensionnement

Les widgets peuvent être configurés pour s'adapter à la taille de la fenêtre.

- `rowconfigure` : Permet de spécifier comment les lignes doivent s'adapter à la taille de la fenêtre.
- `columnconfigure` : Permet de spécifier comment les colonnes doivent s'adapter à la taille de la fenêtre.

Les options possibles sont : `weight, minsize`

Gérer le redimensionnement

- `weight` : Permet de spécifier la priorité de redimensionnement de la ligne ou de la colonne.
- `minsize` : Permet de spécifier la taille minimale de la ligne ou de la colonne.
- `pad` : Permet de spécifier la marge intérieure de la ligne ou de la colonne.

Gérer le redimensionnement

Il est important de :

Attention

- spécifier le paramètre `sticky` pour que le widget ne soit pas centré dans la cellule.
- définir le `minsize` pour que la ligne ou la colonne ne soit pas réduite à zéro.

Gérer le redimensionnement

Exemple

```
content.grid(column=0, row=0, sticky=(N, S, E, W))
frame.grid(column=0, row=0, columnspan=3, rowspan=2, sticky=(N, S, E, W))
namelbl.grid(column=3, row=0, columnspan=2, sticky=(N, W), padx=5)
name.grid(column=3, row=1, columnspan=2, sticky=(N, E, W), pady=5, padx=5)
one.grid(column=0, row=3)
two.grid(column=1, row=3)
three.grid(column=2, row=3)
ok.grid(column=3, row=3)
cancel.grid(column=4, row=3)

root.columnconfigure(0, weight=1)
root.rowconfigure(0, weight=1)
content.columnconfigure(0, weight=3)
content.columnconfigure(1, weight=3)
content.columnconfigure(2, weight=3)
content.columnconfigure(3, weight=1)
content.columnconfigure(4, weight=1)
content.rowconfigure(1, weight=1)

root.mainloop()
```

Tp : positionnement avec grid