Un logiciel de dessin minimaliste en Python

Emmanuel MORAND
(http://www.emmanuelmorand.net)

11 janvier 2008

Ce document a pour objectif de permettre la découverte de la gestion des interfaces graphiques en Python au moyen de la bibliothèque *Tkinter*. Les différentes leçons consistent en l'élaboration de petits programmes graphiques et s'enchaînent de manière progressive pour parvenir au développement d'un logiciel de dessin minimaliste.

Leçon 1

L'objectif de cette leçon est de créer une première interface graphique Python.

Mots-Clés. *Tkinter, from, import, Label, Entry, Button, title, configure, text, pack, mainloop, fg, bg, font, width, height*

Une interface graphique consiste en une fenêtre dans laquelle sont placés différents composants graphiques appelés *widgets*.

Exemple

Tester le programme graphique suivant que l'on nommera prog1.py qui utilise des widgets des trois classes de base *Label* (étiquette), *Entry* (entrée) et *Button* (bouton) :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog1')
# creation d'une etiquette
etiquette=Label(fenetre)
etiquette.configure(text='Mon premier programme graphique en Python')
etiquette.pack()
# creation d'une zone de saisie
saisie=Entry(fenetre)
saisie.pack()
# creation d'un bouton
bouton=Button(fenetre)
bouton.configure(text='Cliquer')
bouton.pack()
# attente des evenements
fenetre.mainloop()
```

Ce programme utilise des *méthodes* :
- La méthode *title* qui permet de donner un titre à la fenêtre du programme.
- La méthode *configure* qui permet de configurer les différents *attributs* des widgets, ici l'attribut *text*.
- La méthode *pack* qui place le widget considéré dans la fenêtre.
- La méthode *mainloop* qui affiche la fenêtre et la met en position d'attente des *événements* clavier et souris.

Les widgets Tkinter possèdent de nombreux attributs, citons par exemple :
- *fg* (foreground) : couleur de premier plan ('red', 'blue', 'black', 'white'...)
- *bg* (background) : couleur d'arrière-plan
- *font* : police de caractère ('Arial', 'Helvetica', 'Verdana'...)
- *width* : largeur (en unités de caractère, 1 par défaut)
- *height* : hauteur

Exercice

Modifier le programme précédent en un programme prog1bis.py comprenant une étiquette dont le texte est bleu, une zone de saisie dont le texte est rouge et un bouton dont le texte est noir sur fond blanc.

Leçon 2

L'objectif de cette leçon est d'étudier le placement des widgets au moyen de la méthode *grid*.

Mots-Clés. *grid, row, column, rowspan, columnspan, padx, pady*

La méthode *grid* permet de placer les widgets sur une grille en les repérant par leurs rangée et colonne.

Exemple

Tester le programme graphique suivant que l'on nommera grid.py :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('grid')
# creation des widgets
etiquette1=Label(fenetre)
etiquette1.configure(text='Prenom :')
etiquette1.grid(row=1,column=1)
etiquette2=Label(fenetre)
etiquette2.configure(text='Nom :')
etiquette2.grid(row=2,column=1)
saisie1=Entry(fenetre)
saisie1.grid(row=1,column=2)
saisie2=Entry(fenetre)
saisie2.grid(row=2,column=2)
# attente des evenements
fenetre.mainloop()
```

La méthode *grid* permet également de placer des widgets qui s'étendent sur plusieurs lignes et/ou plusieurs colonnes de la grille à l'aide des options *rowspan* et *columnspan* qui indiquent le nombre de lignes et de colonnes occupées par le widget. Il est aussi possible de définir un espacement (mesuré en pixels) autour du widget à l'aide des options *padx* et *pady*.

Exercice

Créer un programme nommé prog2.py permettant d'afficher un pavé numérique 3×3 composé de 9 boutons représentant les chiffres de 1 à 9.

Leçon 3

L'objectif de cette leçon est de créer un premier gestionnaire d'événements.

Mots-Clés. *bind, event, <ButtonPress-1>, def, global*

Exemple

Tester le programme suivant que l'on nommera jour-nuit.py :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('jour-nuit')
# creation des boutons
boutonjour=Button(fenetre)
boutonjour.configure(text='Jour')
boutonjour.pack()
boutonnuit=Button(fenetre)
boutonnuit.configure(text='Nuit')
boutonnuit.pack()
# gestionnaire d'evenement associe au clic sur boutonjour
def clicjour(event):
    fenetre.configure(bg='yellow')
boutonjour.bind('<ButtonPress-1>',clicjour)
# gestionnaire d'evenement associe au clic sur boutonnuit
def clicnuit(event):
    fenetre.configure(bg='black')
boutonnuit.bind('<ButtonPress-1>',clicnuit)
# attente des evenements
fenetre.mainloop()
```

Ce programme utilise la méthode *bind* qui permet de lier à l'événement clic-gauche ("<ButtonPress-1>") un gestionnaire d'événement, ici clicjour et clicnuit. Notons qu'il faut impérativement placer le gestionnaire d'événement avant la méthode bind dans le code du programme.

Exercice

Réaliser un programme nommé prog3.py permettant de compter le nombre de clics sur un bouton.

Indications : la fenêtre sera composée d'une étiquette permettant d'afficher le nombre de clics et d'un bouton permettant de cliquer. On créera une variable appelée *nbclics* représentant le nombre de clics réalisés sur le bouton et il faudra prendre garde à ajouter l'instruction *global nbclics* au début du gestionnaire d'événement associé au clic sur le bouton pour signifier que la variable *nbclics* n'est pas une variable locale.

Leçon 4

L'objectif de cette leçon est d'étudier la manière de récupérer le texte d'une zone de saisie.

Mots-Clés. *textvariable, StringVar, get, set, eval*

Pour pouvoir récupérer le texte d'une zone de saisie, il est nécessaire d'utiliser l'attribut *textvariable* et de lui associer une variable Tkinter. À la différence des variables classiques, une variable Tkinter nécessite l'utilisation de méthodes pour obtenir ou définir sa valeur à savoir les méthodes *get* et *set*.

Exemple

Tester le programme suivant que l'on nommera envers.py :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('envers')
# creation d'une variable Tkinter
texte=StringVar()
# creation d'une zone de saisie
saisie=Entry(fenetre)
saisie.configure(textvariable=texte)
saisie.pack()
# creation d'un bouton
bouton=Button(fenetre)
bouton.configure(text='Inverser')
bouton.pack()
# gestionnaire d'evenement associe au clic sur le bouton
def clic(event):
    endroit=texte.get()
    envers=endroit[::-1]
    texte.set(envers)
bouton.bind('<ButtonPress-1>',clic)
# attente des evenements
fenetre.mainloop()
```

Lors du clic sur le bouton, ce programme récupère à l'aide de la méthode *get* le texte de la zone de saisie, le renverse puis le réaffecte à la zone de saisie avec la méthode *set*.
Le renversement de la chaîne de caractères est réalisé à l'aide d'une méthode de haut niveau :

```
>>> 'Life Of Brian'[::-1]
'nairB fO efiL'
```

Exercice

Créer une calculatrice, le programme correspondant sera nommé prog4.py.

Indications : la fenêtre du programme sera composée d'une zone de saisie, d'un bouton pour effectuer le calcul et d'une étiquette pour afficher le résultat. On utilisera l'instruction *eval* qui permet d'évaluer une expression :

```
>>> eval('1+2*3')
7
```

Leçon 5

L'objectif de cette leçon est d'étudier la classe de widgets *Canvas* (canevas).

Mots-Clés. *Canvas, event.x, event.y, <Motion>*

La classe de widget *Canvas* permet de créer une zone graphique dans laquelle il est possible de dessiner.

Exemple

Tester le programme suivant que l'on nommera clicpos.py qui permet d'afficher les coordonnées du curseur de la souris lors d'un clic gauche :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('clicpos')
# creation d'un canevas
canevas=Canvas(fenetre)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# creation d'une etiquette
etiquette=Label(fenetre)
etiquette.pack()
# gestionnaire d'evenement associe au clic sur le canevas
def clic(event):
    etiquette.configure(text='x='+str(event.x)+' et y='+str(event.y))
canevas.bind('<ButtonPress-1>',clic)
# attente des evenements
fenetre.mainloop()
```

Lors d'un clic gauche, ce programme affiche les coordonnées du curseur de la souris au moyen des paramètres *event.x* et *event.y*, l'unité de longueur est le pixel.

Exercice

Modifier le programme précédent en un programme prog5.py qui affiche en continu les coordonnées du curseur de la souris lors de son passage sur le canevas.

Indications : on utilisera l'événement *<Motion>* qui correspond à un mouvement du curseur de la souris.

Leçon 6

L'objectif de cette leçon est d'apprendre à dessiner dans un canevas.

Mots-Clés. *create_line, create_rectangle, create_oval, fill, outline, width, <ButtonRelease-1>*

Les méthodes *create_line*, *create_rectangle* et *create_oval* permettent de dessiner dans un canevas.

Exemple

Tester le programme suivant que l'on nommera dessin.py :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('dessin')
# creation d'un canevas
canevas=Canvas(fenetre)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# dessin dans le canevas
canevas.create_line(100,0,400,500,fill='red',width=3)
canevas.create_rectangle(400,100,300,200,outline='blue',width=2)
canevas.create_oval(100,300,200,400,outline='green',width=2)
# attente des evenements
fenetre.mainloop()
```

Les méthodes *create_line*, *create_rectangle* et *create_oval* utilisent les coordonnées des coins supérieur gauche et inférieur droit de la figure considérée. L'option *fill* permet d'indiquer la couleur de remplissage de la figure. L'option *outline* permet d'indiquer la couleur de la bordure extérieure de la figure et l'option *width* son épaisseur.

Exercice

Créer un programme nommé prog6.py qui permet de tracer sur un canevas de manière automatique la droite reliant les positions du curseur de la souris lors d'un enfoncement et d'un relâchement successifs de sa touche gauche.

Indications : on utilisera l'événement *<ButtonRelease-1>* qui correspond à un relâchement de la touche gauche de la souris, on créera des variables correspondant aux positions du curseur de la souris lors de l'enfoncement et du relâchement de sa touche gauche.

Leçon 7

L'objectif de cette leçon est de dessiner des courbes dans un canevas.

Exemple

Tester le programme suivant que l'on nommera courbe.py qui permet de tracer dans un canevas la courbe décrite par le curseur de la souris :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('courbe')
# creation d'un canevas
canevas=Canvas(fenetre)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# initialisation des variables de position
x='vide'
y='vide'
# gestionnaire d'evenement associe au mouvement sur le canevas
def mouvement(event):
    global x,y
    if(x!='vide'):
        canevas.create_line(x,y,event.x,event.y)
    x=event.x
    y=event.y
canevas.bind('<Motion>',mouvement)
# attente des evenements
fenetre.mainloop()
```

Exercice

Réaliser un programme nommé prog7.py qui permet de tracer dans un canevas la courbe décrite par le curseur de la souris lorsque son bouton gauche est enfoncé.

Indications : on créera une variable représentant l'état du bouton gauche de la souris (haut/bas) et l'on testera la valeur de cette variable avant de tracer dans le canevas.

Leçon 8

L'objectif de cette leçon est d'étudier la classe de widgets *Radiobutton* (bouton radio).

Mots-Clés. *Radiobutton, IntVar*

La classe de widgets *Radiobutton* permet de créer des cases à cocher dont les états sont mutuellement exclusifs.

Exemple

Tester le programme suivant que l'on nommera choix.py :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('choix')
# creation d'une variable Tkinter
choix=IntVar()
choix.set(1) # initialisation de la variable choix
# creation de cases a cocher
case1=Radiobutton(fenetre)
case1.configure(text='choix 1',variable=choix,value=1)
case1.pack()
case2=Radiobutton(fenetre)
case2.configure(text='choix 2',variable=choix,value=2)
case2.pack()
case3=Radiobutton(fenetre)
case3.configure(text='choix 3',variable=choix,value=3)
case3.pack()
# attente des evenements
fenetre.mainloop()
```

Exercice

Réaliser un outil sélection de couleur, le programme correspondant sera nommé prog8.py.

Indications : on utilisera des boutons radio dont les couleurs d'arrière-plan seront choisies dans la liste des 16 couleurs de base :

```
palette=['#000000','#800000','#008000','#808000','#000080','#800080','#008080','#808080',
         '#c0c0c0','#ff0000','#00ff00','#ffff00','#0000ff','#ff00ff','#00ffff','#ffffff']
```

La création des boutons radio sera effectuée au moyen d'un boucle :

```
r={}
for i in range(0,16):
    r[i]=Radiobutton(fenetre)
```

L'objet r ci-dessus s'appelle un *dictionnaire*, il permet de numéroter de 0 à 15 les 16 boutons radio créés.

Leçon 9

L'objectif de cette leçon est de réaliser un logiciel de dessin.

Mots-Clés. *Frame, LabelFrame, bd, relief, flat, groove, raised, ridge, solid, sunken*

La classes *Frame* permet de créer des cadres contenant plusieurs widgets ce qui facilite la mise en place des composants graphiques dans la fenetre.

Exemple

Tester le programme suivant que l'on nommera frame.py :

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('frame')
# creation d'un cadre
cadre=Frame(fenetre)
cadre.configure(bd=2,relief='groove')
cadre.grid(padx=15,pady=15)
# creation des widgets du cadre
saisie=Entry(cadre)
saisie.grid(row=1,column=1,padx=5,pady=5)
bouton=Button(cadre)
bouton.configure(text='bouton')
bouton.grid(row=2,column=1,padx=5,pady=5)
# attente des evenements
fenetre.mainloop()
```

L'attribut *bd* permet d'indiquer l'épaisseur de la bordure du cadre en pixels l'attribut *relief* le style de la bordure ('flat', 'groove', 'raised', 'ridge', 'solid', 'sunken'). La classe *LabelFrame* possède un attribut *text* permettant de donner un titre au cadre.

Exercice

Réaliser un logiciel de dessin, le programme correspondant sera nommé prog9.py.

Indications : le logiciel comprendra une zone de dessin, un outil de sélection de la couleur du tracé, un outil de sélection de l'épaisseur du tracé et un outil de sélection du style de tracé (droite ou courbe). On utilisera les programmes 6, 7 et 8 déjà réalisés.

Solutions des exercices

Leçon 1

prog1bis.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog1bis')
# creation d'une etiquette
etiquette=Label(fenetre)
etiquette.configure(text='Mon premier programme graphique en Python',fg='blue')
etiquette.pack()
# creation d'une zone de saisie
saisie=Entry(fenetre)
saisie.configure(fg='red')
saisie.pack()
# creation d'un bouton
bouton=Button(fenetre)
bouton.configure(text='Cliquer',bg='white')
bouton.pack()
# attente des evenements
fenetre.mainloop()
```

Leçon 2

prog2.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog2')
# creation des touches du pave numerique
bouton1=Button(fenetre)
bouton1.configure(text='1',width='3',height='2')
bouton1.grid(row=3,column=1)
bouton2=Button(fenetre)
bouton2.configure(text='2',width='3',height='2')
bouton2.grid(row=3,column=2)
bouton3=Button(fenetre)
bouton3.configure(text='3',width='3',height='2')
bouton3.grid(row=3,column=3)
bouton4=Button(fenetre)
bouton4.configure(text='4',width='3',height='2')
bouton4.grid(row=2,column=1)
bouton5=Button(fenetre)
bouton5.configure(text='5',width='3',height='2')
bouton5.grid(row=2,column=2)
bouton6=Button(fenetre)
bouton6.configure(text='6',width='3',height='2')
bouton6.grid(row=2,column=3)
bouton7=Button(fenetre)
bouton7.configure(text='7',width='3',height='2')
bouton7.grid(row=1,column=1)
bouton8=Button(fenetre)
bouton8.configure(text='8',width='3',height='2')
bouton8.grid(row=1,column=2)
bouton9=Button(fenetre)
bouton9.configure(text='9',width='3',height='2')
bouton9.grid(row=1,column=3)
# attente des evenements
fenetre.mainloop()
```

Leçon 3

prog3.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog3')
# creation d'une etiquette
etiquette=Label(fenetre)
etiquette.configure(text='nombre de clics : 0')
etiquette.pack()
# creation d'un bouton
bouton=Button(fenetre)
bouton.configure(text='Cliquer')
bouton.pack()
# creation de la variable nombre de clics
nbclics=0
# gestionnaire d'evenement associe au clic sur le bouton
def clic(event):
    global nbclics
    nbclics=nbclics+1
    etiquette.configure(text='nombre de clics : '+str(nbclics))
bouton.bind('<ButtonPress-1>',clic)
# attente des evenements
fenetre.mainloop()
```

Leçon 4

prog4.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog4')
# creation d'une variable Tkinter
calcul=StringVar()
# creation d'une zone de saisie
saisie=Entry(fenetre)
saisie.configure(textvariable=calcul)
saisie.pack()
# creation d'une etiquette
etiquette=Label(fenetre)
etiquette.pack()
# creation d'un bouton
bouton=Button(fenetre)
bouton.configure(text='Calculer')
bouton.pack()
# gestionnaire d'evenement associe au clic sur le bouton
def clic(event):
    etiquette.configure(text=str(eval(calcul.get())))
bouton.bind('<ButtonPress-1>',clic)
# attente des evenements
fenetre.mainloop()
```

Leçon 5

prog5.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog5')
# creation d'un canevas
canevas=Canvas(fenetre)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# creation d'une etiquette
etiquette=Label(fenetre)
etiquette.pack()
# gestionnaire d'evenement associe au mouvement sur le canevas
def mouvement(event):
    etiquette.configure(text='x='+str(event.x)+' et y='+str(event.y))
canevas.bind('<Motion>',mouvement)
# attente des evenements
fenetre.mainloop()
```

Leçon 6

prog6.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog6')
# creation d'un canevas
canevas=Canvas(fenetre)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# initialisation des variables de position
x1='vide'
y1='vide'
x2='vide'
y2='vide'
# gestionnaire d'evenement associe au clic sur le canevas
def clic(event):
    global x1,y1
    x1=event.x
    y1=event.y
canevas.bind('<ButtonPress-1>',clic)
# gestionnaire d'evenement associe au declic sur le canevas
def declic(event):
    global x1,y1,x2,y2
    x2=event.x
    y2=event.y
    canevas.create_line(x1,y1,x2,y2)
canevas.bind('<ButtonRelease-1>',declic)
# attente des evenements
fenetre.mainloop()
```

Leçon 7

prog7.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog7')
# creation d'un canevas
canevas=Canvas(fenetre)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# initialisation des variables de position
x='vide'
y='vide'
# initialisation de l'etat du bouton gauche de la souris
etatboutonsouris='haut'
# gestionnaire d'evenement associe au clic sur le canevas
def clic(event):
    global etatboutonsouris,x,y
    etatboutonsouris='bas'
    x=event.x
    y=event.y
canevas.bind('<ButtonPress-1>',clic)
# gestionnaire d'evenement associe au declic sur le canevas
def declic(event):
    global etatboutonsouris
    etatboutonsouris='haut'
canevas.bind('<ButtonRelease-1>',declic)
# gestionnaire d'evenement associe au mouvement sur le canevas
def mouvement(event):
    global etatboutonsouris,x,y
    if (etatboutonsouris=='bas'):
        canevas.create_line(x,y,event.x,event.y)
    x=event.x
    y=event.y
canevas.bind('<Motion>',mouvement)
# attente des evenements
fenetre.mainloop()
```

Leçon 8

prog8.py

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('prog8')
# creation d'une variable Tkinter
couleur=IntVar()
couleur.set(0) # initialisation de la variable couleur
# creation de la palette de couleurs
palette=['#000000','#800000','#008000','#808000','#000080','#800080','#008080','#808080',
         '#c0c0c0','#ff0000','#00ff00','#ffff00','#0000ff','#ff00ff','#00ffff','#ffffff']
# creation des boutons radio
r={}
for i in range(0,16):
    r[i]=Radiobutton(fenetre)
    r[i].configure(variable=couleur,value=i,bg=palette[i])
    r[i].pack()
# attente des evenements
fenetre.mainloop()
```

Leçon 9

prog9.py (première partie)

```
# importation de la bibliotheque graphique Tkinter
from Tkinter import*
# creation d'une fenetre
fenetre=Tk()
fenetre.title('Logiciel de dessin')
# creation des differents cadres
zonedessin=LabelFrame(fenetre)
zonedessin.configure(text='zone de dessin',bd=2,relief='groove')
zonedessin.grid(row=1,rowspan=3,column=1,padx=10,pady=10)
selectioncouleur=LabelFrame(fenetre)
selectioncouleur.configure(text='selection de la couleur du trait',bd=2,relief='groove')
selectioncouleur.grid(row=1,column=2,padx=10,pady=10)
selectionepaisseur=LabelFrame(fenetre)
selectionepaisseur.configure(text='selection de l\'epaisseur du trait',bd=2,relief='groove')
selectionepaisseur.grid(row=2,column=2,padx=10,pady=10)
selectionstyle=LabelFrame(fenetre)
selectionstyle.configure(text='selection du style de trait')
selectionstyle.grid(row=3,column=2,padx=10,pady=10)
# creation des widgets du cadre zonedessin
canevas=Canvas(zonedessin)
canevas.configure(width=500,height=500,bg='white')
canevas.pack()
# creation des widgets du cadre selectioncouleur
couleur=IntVar()
couleur.set(0)
palette=['#000000','#800000','#008000','#808000','#000080','#800080','#008080','#808080',
         '#c0c0c0','#ff0000','#00ff00','#ffff00','#0000ff','#ff00ff','#00ffff','#ffffff']
rcouleur={}
for i in range(0,16):
    rcouleur[i]=Radiobutton(selectioncouleur)
    rcouleur[i].configure(variable=couleur,value=i,bg=palette[i])
    rcouleur[i].grid(row=1,column=i)
# creation des widgets du cadre selectionepaisseur
epaisseur=IntVar()
epaisseur.set(1)
repaisseur={}
for i in range(1,7):
    repaisseur[i]=Radiobutton(selectionepaisseur)
    repaisseur[i].configure(variable=epaisseur,value=i,text=str(i))
    repaisseur[i].grid(row=1,column=i)
# creation des widgets du cadre selectionstyle
style=StringVar()
style.set('droite')
rstyle1=Radiobutton(selectionstyle)
rstyle1.configure(variable=style,value='droite',text='droite')
rstyle1.grid(row=1,column=1)
rstyle2=Radiobutton(selectionstyle)
rstyle2.configure(variable=style,value='courbe',text='courbe')
rstyle2.grid(row=1,column=2)
```

prog9.py (deuxième partie)

```
# initialisation des variables de position
x='vide'
y='vide'
x1='vide'
y1='vide'
x2='vide'
y2='vide'
# initialisation de l'etat du bouton gauche de la souris
etatboutonsouris='haut'
# gestionnaire d'evenement associe au clic sur le canevas
def clic(event):
    global etatboutonsouris,x1,y1
    etatboutonsouris='bas'
    x1=event.x
    y1=event.y
canevas.bind('<ButtonPress-1>',clic)
# gestionnaire d'evenement associe au declic sur le canevas
def declic(event):
    global etatboutonsouris,x1,y1,x2,y2
    etatboutonsouris='haut'
    x2=event.x
    y2=event.y
    if (style.get()=='droite'):
        canevas.create_line(x1,y1,x2,y2,fill=palette[couleur.get()],width=epaisseur.get())
canevas.bind('<ButtonRelease-1>',declic)
# gestionnaire d'evenement associe au mouvement sur le canevas
def mouvement(event):
    global etatboutonsouris,x,y
    if (style.get()=='courbe' and etatboutonsouris=='bas'):
        canevas.create_line(x,y,event.x,event.y,fill=palette[couleur.get()],width=epaisseur.get())
    x=event.x
    y=event.y
canevas.bind('<Motion>',mouvement)
# attente des evenements
fenetre.mainloop()
```

Index