4e partie

Création d'interfaces graphiques en Python avec guizero

La pire des interfaces

Apprenez les bonnes pratiques de conception en faisant tout de travers !

MAKER

Laura Sach

Laura dirige l'équipe *A Level* de la Fondation Raspberry Pi chargée des ressources pédagogiques en informatique à destination des étudiants.

@CodeBoom

MAKER

Martin O'Hanlon

Martin crée des cours, des projets et des ressources en ligne au sein de l'équipe Learning de la Fondation Raspberry Pi.

@martinohanlon

Figure 2 Sélection de lettres par listes déroulantes.

Grisé par votre nouveau savoir, vous pourriez être tenté de vous lancer tête baissée dans la création d'interfaces graphiques en mélangeant frénétiquement widgets, couleurs, polices et fonctions de votre cru. Vous apprendriez de vos erreurs, mais cela vous prendrait du temps. Alors gagnons-en ensemble en découvrant dès maintenant tout ce qu'il ne faut pas faire !

Mais que fait donc la police ?

Un mauvais contraste entre l'arrière-plan et la police rend tout texte difficilement lisible. Autrement dit il ne faut jamais utiliser deux couleurs de teintes trop proches.

Importez ces widgets en début de code :

```
from guizero import App, Text
```

Créez une fenêtre **app** et un texte :

```
app = App("it's all gone wrong")
title = Text(app, text="Some hard to read text")

app.display()
```

Jouez avec les couleurs, la police et la taille du texte (listage **worst1.py**). Nos choix ne sont *visiblement* pas les bons !

```
app = App("it's all gone wrong", bg="dark green")
title = Text(app, text="Some hard-to-read text", size="14", font="Comic Sans", color="green")
```

Pour être bien lisible, il va de soi qu'un texte doit rester affiché suffisamment longtemps à l'écran et ne pas se prendre pour un clignotant.

Tous les widgets de *guizero* peuvent être rendus invisibles (ou de nouveau visibles) avec les fonctions `hide()` et `show()`. La fonction `repeat()` permet quant à elle d'exécuter une fonction toutes les n secondes. Combinons ces trois fonctions pour faire clignoter le texte.

La fonction **flash_text()** masque le texte s'il est visible, et le montre s'il ne l'est pas :

```
def flash_text():
    if title.visible:
        title.hide()
    else:
        title.show()
```

Utilisons ensuite `repeat()` pour l'exécuter toutes les 1000 millisecondes (1 s) :

```
app.repeat(1000, flash_text)

app.display()
```

Lancez votre code (**worst2.py**) : le texte est masqué et réaffiché toutes les secondes.

Mauvais choix de widgets

La frontière entre une bonne et une mauvaise interface tient parfois à un seul widget.

Lequel utiliser pour la saisie d'une date ? Un

Figure 2

TextBox serait flexible, mais exigerait validation et formatage de la date. Trois listes déroulantes (widgets Combo) pour le jour, le mois et l'année éviteraient l'étape de validation, mais seraient d'utilisation fastidieuse.

Pour la saisie d'une date et d'une heure, pourquoi pas une glissière (**fig. 1**) ? Le widget **Slider** (code **worst3.py**) retourne un nombre compris entre 0 et 999.999.999, soit le nombre de secondes écoulées depuis le 1er janvier 1970, puis le convertit en date et en heure avec la fonction `ctime()`. Essayez !

User d'un TextBox simple ou multilignes est une façon simple de demander du texte à l'utilisateur. Mais n'est-ce pas justement *trop* simple ? N'exige-t-on pas trop de saisie de la part de l'utilisateur ? Et quid de celui qui ne voudra pas lâcher sa souris ? Et si nous lui proposions de choisir les lettres à entrer au moyen de listes déroulantes (**fig. 2**) ? Testons cette idée géniale. Nous aurons besoin de la constante de chaîne `ascii_letters` :

```
from guizero import App, Combo
from string import ascii_letters
```

La constante `ascii_letters` est une liste contenant les lettres de l'alphabet en minuscules et en majuscules. Fournissons-la au paramètre **options** d'un widget **Combo** :

```
a_letter = Combo(app, options=" " + ascii_
letters, align="left")

app.display()
```

Lancez le code (**worst4.py**) : la liste déroulante est alignée à gauche de la fenêtre et permet d'afficher une lettre suivie d'une espace. Pour que notre fanatique de souris puisse p. ex. entrer un mot de trois lettres, créons trois widgets **Combo** :

```
a_letter = Combo(app, options=" " + ascii_
letters, align="left")
b_letter = Combo(app, options=" " + ascii_
letters, align="left")
c_letter = Combo(app, options=" " + ascii_
letters, align="left")
```

Le paramètre **align="left"** les place les uns contre les autres en partant du bord gauche.

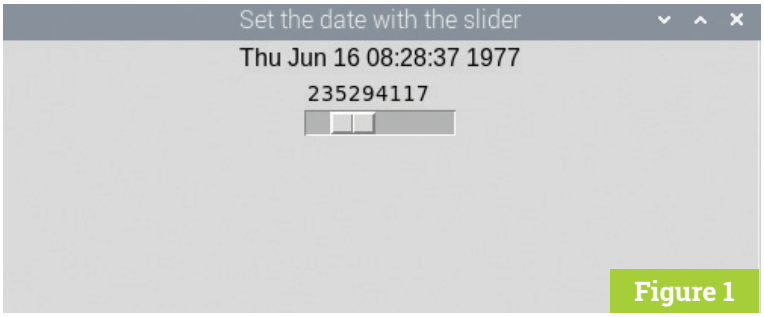
Plutôt que de définir explicitement ces widgets un par un, le code **worst5.py** utilise une boucle `for`. Une boucle `for` apporte plus de souplesse au code, car il

worst1.py

TÉLÉCHARGEZ LE CODE COMPLET :
magpi.cc/guizerocode

Langage : **Python 3**

```
001.  # Imports ---------------
002.
003.  from guizero import App, Text
004.
005.
006.  # App -------------------
007.
008.  app = App("it's all gone wrong", bg="dark green")
009.
010.  title = Text(app, text="Hard to read", size="14", font="Comic
011.  Sans", color="green")
012.
013.  app.display()
```



▲ **Figure 1** Sélection (impossible) de la date et de l'heure.

worst2.py

Langage : **Python 3**

```
001.  # Imports ---------------
002.
003.  from guizero import App, Text
004.
005.
006.  # Functions -------------
007.
008.  def flash_text():
009.      if title.visible:
010.          title.hide()
011.      else:
012.          title.show()
013.
014.
015.  # App -------------------
016.
017.  app = App("it's all gone wrong", bg="dark green")
018.
019.  title = Text(app, text="Hard to read", size="14", font="Comic
020.  Sans", color="green")
021.
022.  app.repeat(1000, flash_text)
023.
024.  app.display()
```

worst3.py

Langage : **Python 3**

```
001.  # Imports ---------------
002.
003.  from guizero import App, Slider, Text
004.  from time import ctime
005.
006.
007.  # Functions -------------
008.
009.  def update_date():
010.      the_date.value = ctime(date_slider.value)
011.
012.
013.  # App -------------------
014.
015.  app = App("Set the date with the slider")
016.  the_date = Text(app)
017.  date_slider = Slider(app, start=0, end=999999999,
018.  command=update_date)
019.
020.  app.display()
```

worst4.py

Langage : **Python 3**

```
001.  # Imports ---------------
002.  from guizero import App, Combo
003.  from string import ascii_letters
004.
005.
006.  # App -------------------
007.
008.  app = App("Enter your name")
009.
010.  a_letter = Combo(app, options=" " + ascii_letters, align="left")
011.
012.  app.display()
```

Figure 3
Une pop-up aussi inutile qu'agaçante.

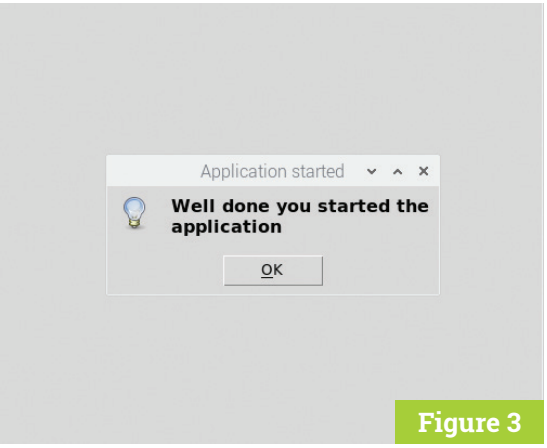


Figure 3

Widget Window

Les pop-ups permettent de questionner l'utilisateur, mais sont d'emploi limité. Pour afficher plus d'informations ou acquérir des données supplémentaires, vous pouvez créer plusieurs fenêtres à l'aide du widget **Window**.

Un objet (une fenêtre) Window s'utilise comme l'objet App et partage avec lui bon nombre de ses méthodes (fonctions).

```
from guizero import App, Window

app = App("Main window")
window = Window(app, "2nd Window")

app.display()
```

Un objet Window se masque avec **hide()** et se réaffiche avec **show()** :

```
window.show()
window.hide()
```

Pour qu'un objet Window devienne la fenêtre principale tant qu'il n'a pas été fermé, utilisez le paramètre **wait** de la méthode **show** et mettez-le sur **True** :

```
window.show(wait=True)
```

Vous trouverez plus d'informations sur l'utilisation des fenêtres multiples à la page **lawsie.github.io/guizero/multiple_windows**.

suffit de modifier la valeur de `range()` pour afficher plus ou moins de widgets.

Widgets pop-up

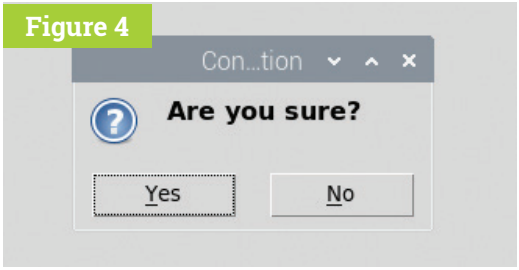
La bibliothèque guizero offre plusieurs types de pop-ups. Ces fenêtres surgissantes servent à interrompre l'utilisateur pour lui poser une question ou lui fournir une information, mais attention à leur abus !

Un bon moyen d'irriter votre prochain serait p. ex. de l'informer qu'il vient de lancer votre application (**fig. 3**) :

```
from guizero import App

app = App(title="pointless pop-ups")

app.info("Application started", "Well done
you started the application")

app.display()
```

Note de l'éditeur : cet article est paru initialement dans le magazine MagPi (le magazine officiel du Raspberry Pi).
La maison d'édition Elektor publie ce magazine en néerlandais, allemand et français (www.magpi.fr).

Figure 4



▲ **Figure 4** Yes, we are sûrs !

Le premier paramètre passé à la méthode `info` est le titre de la fenêtre, le second est le message lui-même. Les méthodes `warn` et `error` s'utilisent comme `info` et n'en diffèrent que par l'icône affichée dans la pop-up.

Certaines pop-ups servent à obtenir une information. La plus simple à cet égard est la pop-up `yesno` présentant les options de réponse `Yes` et `No`. L'option Yes retourne *True*, No retourne *False*. Cette pop-up est utile pour demander confirmation d'une action, p. ex. la suppression d'un fichier. Cela dit, ne sollicitez pas une confirmation à chaque fois que l'utilisateur appuie sur un bouton ! Pour notre exemple, importez le widget `PushButton` :

```
from guizero import App, PushButton
```

Créez une fonction demandant confirmation avec la pop-up `yesno` :

```
def are_you_sure():
    if app.yesno("Confirmation", "Are you
sure?"):
        app.info("Thanks", "Button
pressed")
    else:
        app.error("Ok", "Cancelling")
```

Ajoutez un bouton appelant `are_you_sure` lorsqu'il est pressé :

```
button = PushButton(app, command=are_you_
sure)
```

Lancez le code (**05-worlds-worst-gui.py**) et cliquez sur le bouton : une pop-up attend votre décision (**fig. 4**).

Toutes les pop-ups de *guizero* sont décrites sur **lawsie.github.io/guizero/alerts**. (VF : Hervé Moreau)

worst5.py

Langage : **Python 3**

```
001.  # Imports ---------------
002.
003.  from guizero import App, Combo
004.  from string import ascii_letters
005.
006.
007.  # App -------------------
008.
009.  app = App("Enter your name")
010.
011.  name_letters = []
012.  for count in range(10):
013.      a_letter = Combo(app, options=" " + ascii_letters,
014.  align="left")
015.      name_letters.append(a_letter)
016.
017.  app.display()
```

05-worlds-worst-gui.py

Langage : **Python 3**

```
001.  from guizero import App, PushButton
002.
003.  def are_you_sure():
004.      if app.yesno("Confirmation", "Are you sure?"):
005.          app.info("Thanks", "Button pressed")
006.      else:
007.          app.error("Ok", "Cancelling")
008.
009.  app = App(title="pointless pop-ups")
010.
011.  button = PushButton(app, command=are_you_sure)
012.
013.  app.info("Application started", "Well done you started the
014.  application")
015.
016.  app.display()
```