

# 1 documentation

elle est en anglais <http://kivy.org/docs/> et fourmille d'exemples.

## 2 Développer avec kivy

Il y a 3 façons d'envisager de développer une application :

- de façon classique ( hélas mal documenté)
- en utilisant la syntaxe de la programmation orientée objets (classes)
- en utilisant les fichiers d'extension kv (recommandé sur le site de kivy)

Dans les 3 cas la syntaxe est différente mais les idées sont les mêmes, on peut se contenter de la façon classique.

Pour porter une application kivy sous android voir [ce lien](#)

## 3 Principe

La fenêtre principale est un objet de type `App`.

Le programme commence par `monAppli = App()` et se termine par `monAppli.run()` qui prend en charge toutes les fonctionnalités liées à la gestion des événements (appui sur un bouton, glissé etc...)

L'interface d'une application kivy est une arborescence de composants graphiques ("widgets") imbriqués les uns dans les autres.

## 4 widgets d'agencement ("layouts")

<http://kivy.org/docs/gettingstarted/layouts.html>

Ces composants servent à choisir la disposition d'autres composants les uns par rapport aux autres. Par exemple pour disposer un bouton et une zone d'affichage de texte (qui sont aussi des widgets) horizontalement :

```
B = Button(text = " presse moi ")
L = Label(text = " c'est ici qu'il faut regarder ")
zone_boutons = BoxLayout(orientation='horizontal',
                          size_hint=(None,None),
                          size=(480,y0) )

zone_boutons.add_widget(B)
zone_boutons.add_widget(L)
```

## 5 Racine de l'arborescence des widgets

Sans surprise ce composant se nomme `root` et il faut bien sûr informer le programme de ce qu'il est :

```
MonAppli.root = BoxLayout(orientation='vertical')
racine = MonAppli.root # facultatif : donne un pseudo (alias) à ce widget
racine.add_widget(zone_boutons)
```

## 6 Dessiner sur un widget

Les composants ont un attribut `canvas` sur lequel on peut dessiner des lignes, des rectangles, des disques, etc...

<http://kivy.org/docs/api-kivy.graphics.html>

```
damier = racine.canvas # créé un alias
pion = Ellipse(pos=(5*dx,y0+3*dx), size=(dx,dx))
damier.add( pion )
```

## 7 Donner vie au pion avec le bouton

```
def action(instance) :
    if instance.text == " presse moi " :
        sens = +1
        pion.pos = ( pion.pos[0] + sens*dx , pion.pos[1] + dx )
```

```
B.bind(on_press = action)
```