

Animation simple (sans événements)



L'objectif ici est de faire apparaître un personnage (une image) et de l'animer. Pour illustrer cela on va utiliser [cette image](#) que l'on va faire rebondir sur les bords de la fenêtre.

1. Création de la fenêtre et insertion de l'image

Script Python

```
1  import pygame
2  from pygame.locals import *
3
4  pygame.init()
5
6  width, height = 320, 240
7  screen = pygame.display.set_mode((width, height))
8
9  balle = pygame.image.load("basketball16.png")
10 position_balle = balle.get_rect()
11 screen.blit(balle, position_balle)
12 pygame.display.flip()
13
14 pygame.quit()
```

Explication du code

Ligne 9

On importe l'image dans une variable `balle` qui est un objet `pygame` de type `Surface`.

Ligne 10

Afin de facilement pouvoir le déplacer, on stocke **la position de cet objet** dans une variable `position_balle`, qui sera de type `rect`.

Script Python

```
>>> position_balle  
<rect(0, 0, 16, 16)>
```

où les deux premiers attributs sont les coordonnées (abscisse, ordonnée) et les deux derniers la largeur et la hauteur de l'image.

Ligne 11

On affiche cette image en la *superposant* aux éléments graphiques déjà affichés (en l'occurrence, rien...) avec la méthode `blit`.

2. Animation de l'image/personnage

Déplacement

Pour déplacer l'image/le personnage, il faut modifier ses coordonnées à chaque rafraîchissement de la fenêtre.

Déplacement absolu

On peut accéder directement aux coordonnées d'un objet de type `rect`. Par exemple:

Script Python

```
>>> position_balle.left
0
>>> position_balle.bottom
16
>>> position_balle.topleft
(0, 0)
```

Pour afficher la balle en position `(50, 100)` (pour le coin haut-gauche, qui est traditionnellement celui sur lequel on travaille), il suffit donc d'écrire:

Script Python

```
position_balle.topleft = (50, 100)
```

Déplacement relatif

On indique le déplacement (horizontal et vertical) **par rapport à la position initiale**.

Par exemple, pour décaler la balle de 10 pixels vers la droite et de 20 pixels vers le haut:

Script Python

```
1 position_balle.move(10, -20)
```

Pour notre animation, on décide donc d'un décalage relatif en abscisse et en ordonnée, qu'on stocke par exemple dans une liste.

Ici, pour déplacer d'un pixel sur chaque coordonnée:

Script Python

```
speed = [1, 1]
position_balle.move(speed)
```

Bien entendu, il faut changer ces valeurs lorsqu'on atteint le bord de la fenêtre:

- si `position_balle.right` dépasse la largeur `width`, il faut repartir vers la gauche, c'est-à-dire passer de la valeur `1` à `-1`.
- et si `position_balle.left` devient négatif, il faut repartir vers la droite, c'est-à-dire passer de la valeur `-1` à `1`.

Le code suivant va donc actualiser ce déplacement et faire rebondir notre balle de droite à gauche:

```

Script Python
1  import pygame
2  from pygame.locals import *
3
4
5  pygame.init()
6
7  width, height = 320, 240
8  screen = pygame.display.set_mode((width, height))
9
10  balle = pygame.image.load("basketball16.png")
11  position_balle = balle.get_rect()
12  screen.blit(balle, position_balle)
13  pygame.display.flip()
14
15  speed = [1, 0]
16
17  continuer = True
18  while continuer:
19      for evenement in pygame.event.get():
20          if evenement.type == QUIT:
21              continuer = False
22
23          if position_balle.left < 0 or position_balle.right > width:
24              speed[0] = -speed[0]
25
26          position_balle = position_balle.move(speed)
27
28          screen.fill([0, 0, 0])
29          screen.blit(balle, position_balle)
30          pygame.display.flip()
31
32  pygame.quit()

```

? screen.fill

Commenter la ligne 28. En quoi est-elle importante?

À vous de jouer pour terminer le programme...

Pour s'entraîner

Écrire un programme où un personnage se déplace aléatoirement dans la fenêtre.

Vous pourrez utiliser les instructions :

- `pygame.time.delay(1000)` afin de ne bouger le personnage que toutes les 1000 millisecondes (ou une autre valeur à votre guise).
- `randint(a,b)` du package `random`, qui renvoie un entier pseudo-aléatoire entre `a` et `b`.

Pour le personnage, vous pouvez utiliser un sprite (libre de droits) trouvé sur le Web, mais privilégiez le format PNG qui gère la transparence. Ou bien celui-ci:

