Réalisation du jeu Pong avec Python3, Pygame Zero et Mu

CoderDojo Seneffe

Février 2020

Pendant cette session, nous allons découvrir comment créer un jeu Pong avec Python. Pong est un des premiers jeux vidéos. Il a été créé en 1972. Si tu ne connais pas Pong, voici une vidéo du jeu : `https://www.youtube.com/watch?v=it0sf4CMDeM`.

Retrouve les traductions anglais → français dans cette marge.

Pour créer notre jeu de Pong, nous allons utiliser Pygame Zero, une bibliothèque qui facilite la création de jeu avec Python, ainsi que Mu, un environnement de programmation (IDE) pour écrire et exécuter notre programme. Pygame Zero va nous permettre de gérer facilement les images, le clavier et les sons pour nous concentrer sur la programmation du jeu.

Attention, Pygame Zero est différent (et plus simple) que Pygame. Si tu cherches des exemples sur internet, il faut bien faire la différence.

1 Installation de Mu

Si Mu n'est pas déjà installé sur ton ordinateur, tu peux le télécharger sur `https://codewith.mu/en/download`.
Il n'est pas nécessaire d'installer Pygame Zero, cela est compris lorsque tu installes Mu. Sur Windows et Mac, il n'est pas nécessaire non plus d'installer Python.

2 Hello World

1. Si tu ne l'as pas déjà fait, ouvre Mu, tu devrais voir un fichier vide. Si ce n'est pas le cas, tu peux créer un nouveau fichier en cliquant sur `New`.
2. Mu dispose de plusieurs modes en fonction de ce que tu veux faire comme type de programme. On va commencer par un exemple de Python "simple", sans Pygame Zero. Clique sur `Mode` et choisi `Python 3`.
3. Tape le code suivant dans le fichier :

   ```
   print("Hello World !")
   ```

4. Clique sur `Run` ou `Lancer` pour lancer ton programme, tu devras choisir un nom pour sauvegarder celui-ci.

Que se passe-t-il ?

Félicitations ! Tu viens d'écrire ton premier programme en Python. Tu peux modifier la ligne de code pour faire dire ce que tu veux à ton programme.

Petite astuce : pour chaque étape de cette session, utilise un fichier différent. Ça te permettra de garder une trace de ce que tu as fait. C'est pratique pour se souvenir de comment tu as fait les choses !

3 Bonjour Pygame Zero

Maintenant, jouons avec Pygame Zero. Pour cela on va changer de mode dans Mu. Clique sur `Mode` et choisi `Pygame Zero`.

Nous allons commencer par afficher la fenêtre dans laquelle le jeux se déroulera et en colorier le fond en vert.

```
WIDTH = 200
HEIGHT = 200

def draw():
    screen.fill((0, 255, 0))
```

WIDTH = largeur
HEIGHT = hauteur

draw = dessiner
screen = écran
fill = remplir

1. Clique sur `New` ou `Nouveau` pour créer un nouveau fichier.
2. Recopie le programme ci-dessus dans l'éditeur.
3. Clique sur `Play` ou `Jouer` pour exécuter le programme. Tu devras donner un nom à ton fichier si ce n'est pas déjà fait, par exemple `Pong1`.
   Une fenêtre s'est-elle ouverte ? Le fond est-il vert ?
4. Peux-tu changer la taille de la fenêtre et la couleur du fond ?

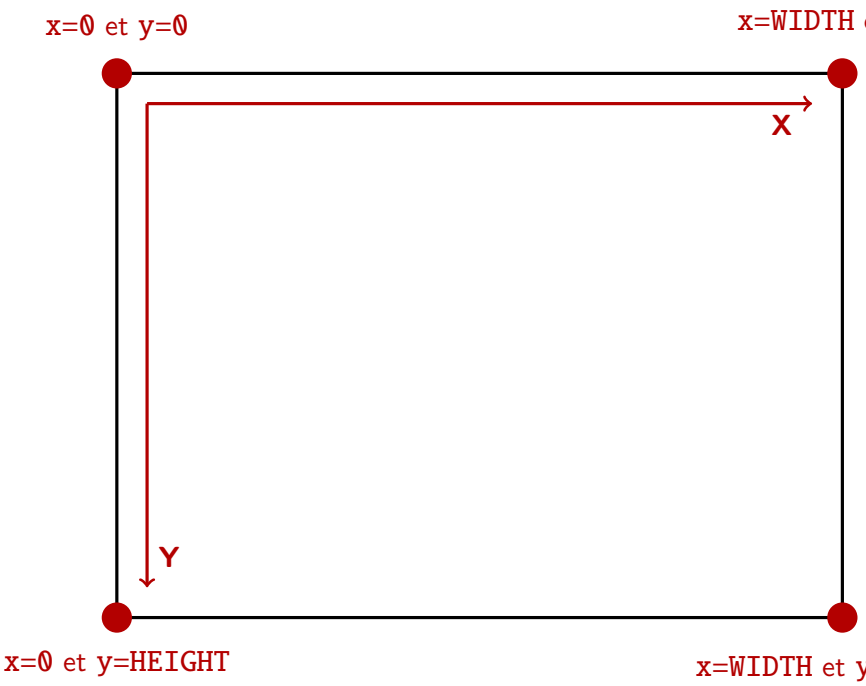
Quelques explications

Pygame Zero te simplifie la vie au maximum et a déjà programmé toute une série de choses pour toi, comme par exemple ouvrir une fenêtre pour ton jeu. Si tu ne précise rien, Pygame Zero décide de la taille de la fenêtre.
Mais il te donne la possibilité de choisir toi même la taille à travers des variables (`WIDTH` et `HEIGHT`).
En assignant une valeur à ces variables au début du programme, c'est toi qui choisi la taille de la fenêtre.

Pygame Zero sait aussi quand il est nécessaire de dessiner dans la fenêtre et quand c'est le cas, il te demande de le faire en appelant `draw()`.
Tu dois donc définir une fonction portant le nom de `draw` et y inclure le code qui va dessiner à l'écran.
C'est ce que le mot clé **`def`** indique (**`def`** pour define, qui veut dire définir en français).
Attention à ne pas oublier les parenthèses après le nom de ta fonction.
Les fonctions peuvent prendre des paramètres, qu'on défini entre les parenthèses, mais même quand il n'y a aucun paramètre, les parenthèses sont nécessaires.

4 Le positionnement des objects

Avant de faire apparaître la balle, il est important de savoir comment sont positionnés les éléments sur l'écran. La position de chaque élément est définie par des coordonnées x et y.



width = largeur

height = hauteur

L'axe horizontal (X) va de gauche à droite. Sur le bord gauche de la fenêtre, il a la valeur 0. Sur le bord droit de la fenêtre, il a la valeur `WIDTH`, qui est la largeur de la fenêtre.

L'axe vertical (Y) va de haut en bas. Sur le bord supérieur de la fenêtre, il a la valeur 0. Sur le bord inférieur de la fenêtre, il a la valeur `HEIGHT`, qui est la hauteur de la fenêtre.

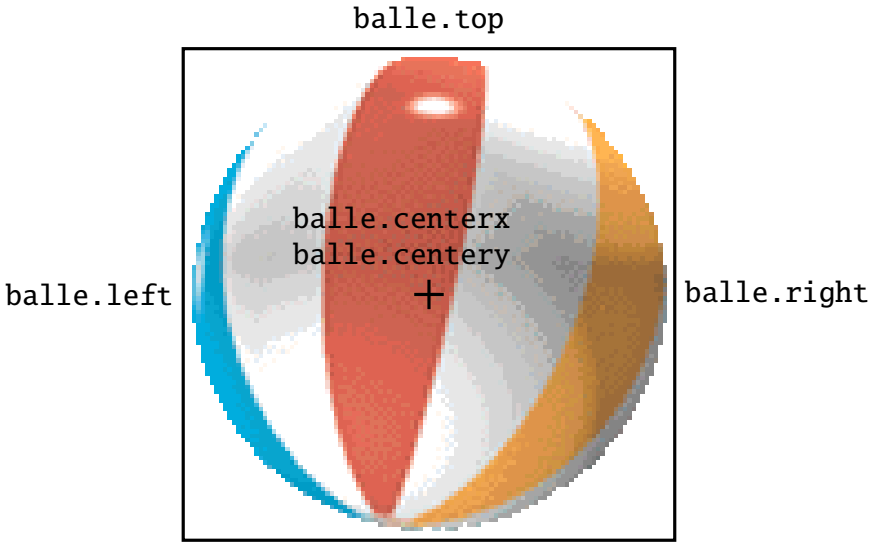
1. Double-clique sur le nom du fichier pour lui donner un nouveau nom, par exemple `Pong2`.
2. Modifie ton fichier de la façon suivante :

```
WIDTH = 800
HEIGHT = 600

balle = Actor('ball.png')
balle.center = 400, 300

def draw():
    screen.clear()
    balle.draw()
```

Actor = acteur
center = centre
clear = effacer

3. Clique sur `Play` ou `Jouer` pour exécuter le programme.
4. La balle est-elle affichée au milieu de l'image ? Peux-tu placer la balle à un autre endroit en modifiant le code ?

5 La position des bords de la balle



top = le dessus

center = centre

left = gauche
right = droite

bottom = le bas

Le bord gauche de la balle sur l'axe X est `balle.left`.

Le bord droit de la balle sur l'axe X est `balle.right`.

Le bord supérieur de la balle sur l'axe Y est `balle.top`.

Le bord inférieur de la balle sur l'axe Y est `balle.bottom`.

Le centre de la balle sur l'axe X est `balle.centerx`.

Le centre de la balle sur l'axe Y est `balle.centery`.

6 Un peu de mouvement

Il est temps de faire bouger la balle.

1. Double-clique sur le nom du fichier pour lui donner un nouveau nom, par exemple Pong3.
2. Modifie ton fichier de la façon suivante :

```
WIDTH = 800
HEIGHT = 600

balle = Actor('ball.png')
balle.center = 400, 300

vitesse_x = 10
vitesse_y = 10

def draw():
    screen.clear()
    balle.draw()

def update():
    global vitesse_x, vitesse_y
```

update = mettre à jour

```
balle.left = balle.left + vitesse_x
balle.top = balle.top + vitesse_y
```

3. Clique sur `Play` ou `Jouer` pour exécuter le programme.
4. La balle sort assez vite de la fenêtre. Comment faire pour garder la balle dans la fenêtre ? Comment détecter que la balle touche un bord ?

Par défaut, Pygame Zero cherche les images dans le folder `images`. Fais bien attention à placer tes fichiers image (par exemple `ball.png`) à cet endroit.

Quelques explications

On a vu plus haut que Pygame Zero appelait la fonction `draw()` pour te donner la possibilité de dessiner. Ici il appelle la fonction `update()` pour te donner la possibilité de mettre à jour l'état de ton jeu (pour par exemple déplacer des objets).

7 La balle rebondi sur les bords

Quand la balle sort de la fenêtre en bas, `balle.bottom` est plus grand que `HEIGHT` (la hauteur de la fenêtre). Il faut alors inverser le sens de déplacement vertical (`vitesse_y`) de la balle pour qu'elle remonte. Voici un bout de code qui fait ça.

```
if balle.bottom > HEIGHT:
    vitesse_y = -vitesse_y
```

1. Place le code ci-dessus au bon endroit et relance ton programme.
2. Fais la même chose pour les autres bords.
3. Ta balle reste-t-elle dans ta fenêtre ?

Après ces étapes, la balle se balade dans l'écran en rebondissant sur les bords. On va maintenant ajouter les raquettes pour les joueurs.

8 Ajout des raquettes des joueurs

On va simplement utiliser des rectangles rouges pour les raquettes.

Voici un exemple de code pour créer une raquette et la placer au bord de l'écran :

```
raquette = Rect(10, 10, 10, 100)

def draw():
    screen.clear()
    screen.draw.filled_rect(raquette, (255, 0, 0))
```

filled = rempli

Les paramètres de `Rect()` sont la position horizontale, la position verticale, la largeur et la hauteur.
Les paramètres de `filled_rect()` sont le rectangle à dessiner et la couleur à utiliser.

1. En t'inspirant du code ci-dessus, ajoute une raquette dans ton programme.
2. Modifie-le pour mettre une raquette de chaque côté de l'écran (choisi un nom parlant pour tes variables).
3. Exécute ton programme. Les raquettes sont-elles où tu veux les mettre ?
4. Amuse-toi à changer leurs tailles et leurs couleurs.

9 Déplacement des raquettes avec le clavier

Pygame Zero permet de savoir quelles sont les touches du clavier qui sont pressées. Ça va te permettre de déplacer les raquettes en fonction des touches sur lesquelles tu appuies.

Le code ci-dessous permet de savoir quand les flèches du haut et du bas sont pressées.

```
# Regarde si la flèche du bas est pressée
if keyboard.down:
    # Ici, il faut déplacer la raquette
# Regarde si la flèche du haut est pressée
if keyboard.up:
    # Ici, il faut déplacer la raquette
```

keyboard = clavier
down = bas

up = haut

Tu peux aussi utiliser `keyboard.w` ou `keyboard.a` pour savoir quand les touches `w` et `a` sont pressées.

1. Utilise le code ci-dessus dans ton programme pour déplacer les raquettes (indice : on déplace la raquette dans la fonction `update()`).
2. Exécute ton programme. Les raquettes se déplacent-t-elles assez vite ? Trop vite ?
3. Adapte la vitesse à ton envie.
4. Que se passe-t-il si tu laisses ton doigt sur la flèche du haut pendant longtemps ? Modifie le programme pour que les raquettes restent toujours dans l'écran.

10 Collision entre les raquettes et la balle

Il faut maintenant détecter quand ta balle touche une raquette pour la faire changer de direction.

La balle (un Actor) peut détecter si elle entre en collision avec un rectangle en utilisant la fonction `colliderect()`.

Modifie le programme pour détecter une collision entre la balle et une raquette et changer la direction de la balle.

Les coachs seront là pour t'aider à trouver une solution. N'hésite pas à poser des questions.

11 Que faire quand le joueur ne touche pas la balle ?

Pour l'instant, si le joueur ne touche pas la balle avec la raquette, elle rebondit encore sur le bord de l'écran. Ce n'est pas très intéressant comme jeu.

Modifie le programme pour que la balle revienne au centre de l'écran quand un joueur ne la touche pas avec sa raquette.

12 Pour aller plus loin

Voici quelques idées pour aller encore plus loin dans ton jeu :

1. Compte les points et affiche les en haut de l'écran.
2. Ajoute un son quand la balle touche la raquette.
3. Quand la balle revient au centre de l'écran, sa position verticale est choisie au hasard.
4. Chaque fois qu'un joueur marque 5 points, la balle accélère.

Pour cela, tu auras besoin des éléments suivants :

Afficher du texte à l'écran

Le morceau de code suivant permet d'afficher le texte 'Bonjour' à l'écran. Le deuxième paramètre représente la position du coin supérieur gauche du texte.

```
screen.draw.text('Bonjour', (0,0))
```

Convertir un nombre en texte

Les variables en Python ont un type, un nombre ce n'est pas la même chose que du texte (ou chaîne de caractères, string en anglais). Si tu as une variable qui contient un nombre et que tu veux la transformer en chaîne de caractères, utilise la fonction **`str()`** comme dans l'exemple ci-dessous.

```
nombre = 10
texte = str(nombre)
```

Jouer une note

Pygame Zero offre une fonction pour jouer une note de musique, telle que montré dans le code ci-dessous. Le premier paramètre est la fréquence de la note en Hertz (c'est sa hauteur, un plus petit nombre donne une note plus grave, un plus grand nombre une note plus aiguë), le second paramètre indique la durée de la note en seconde.

```
tone.play(100, 0.1)
```

Choisir un nombre au hasard

Python offre une fonction pour choisir un nombre entier au hasard : `randint()`, celle-ci accepte 2 paramètres indiquant l'interval dans lequel les nombres sont générés (le plus petit et le plus grand nombre possible). Attention, cette fonction se trouve dans un module séparé, qu'il faut importer dans le programme, comme le montre le code ci-dessous.

```
import random

print(random.randint(0, 10))
```