

Ping Pythong

version 1.3.1



DIVERSITY
by EPITECH

I. Introduction

Pong est le premier jeu vidéo d'arcade de sport. (Et non pas le premier jeu vidéo contrairement à ce que croient nombre d'exceptionnels personnes comme vous et moi)

Nous sommes en 1971, vous êtes un ingénieur américain, et vous venez tous juste d'accepter un poste chez Atari. N'avez jamais développé de jeu vidéo auparavant, Nolan Bushnell (le loup de Wall Street mais chez Atari) vous donne un exercice pour vous former eu développement des jeux vidéo.

L'exercice se veut être le plus simple des jeux : un point en mouvement, deux raquettes, et l'affichage du score.

Vous décidez donc de vous lancer dans la programmation sous le langage Python afin d'épater votre nouveau patron grâce à vos talents insoupçonnés, et par la suite devenir l'une des futures légendes de l'informatique.



** "Déjà vu; I've just come to this place before; higher on the street; and I know it's my time to go~*

Calling you; and the search is a mystery; standing on my feet; it's so hard when I try to be me Whooooooooaaaaa! ~"

II. Consignes

- * Vous allez devoir compléter des fonctions du jeu Pong sous Python avec la librairie Pygame (spoiler : ça va être dur section)
- * Si vous avez des questions, demandez à votre voisin de gauche, puis de droite. Coopérez !
- * En cas de blocage, et après avoir tenté la consigne précédente, n'hésitez pas à demander à un Cobra, ils sont très gentils et sont là pour vous aider.
- * Vous avez entièrement le droit d'utiliser internet comme vous le souhaitez pour vous renseigner, nous le conseillons fortement.
- * Si vous n'avez jamais fait de Python ou de programmation auparavant et que vous souhaitez apprendre quelques notions de base avant de vous lancer : demandez à un Cobra de vous assister et rendez [ici](#) pour apprendre les bases.
- * *N'hésitez pas à faire des bonus et ajouter/modifier des fonctionnalités à votre programme, la seule limite est votre imagination ! (Et 17h)*

- * Amusez-vous ! Nous ne sommes pas là pour apprendre mais pour découvrir, vous avez le droit de ne pas réussir à compléter le sujet. 😊

III. Initialisons des variables !

Tous d'abord commencez par ouvrir Visual Code dans le dossier du sujet, puis ouvrez « game.py ».

Nous possédons plusieurs variables dans notre jeu (chaque variable commence par « self. »):

`self.Score1` : le score du joueur 1

`self.Score2` : le score du joueur 2

`self.ScreenWidth` : la taille désirée de l'écran en largeur (← →)

`self.ScreenHeight` : la taille désirée de l'écran en hauteur (↑ ↓)

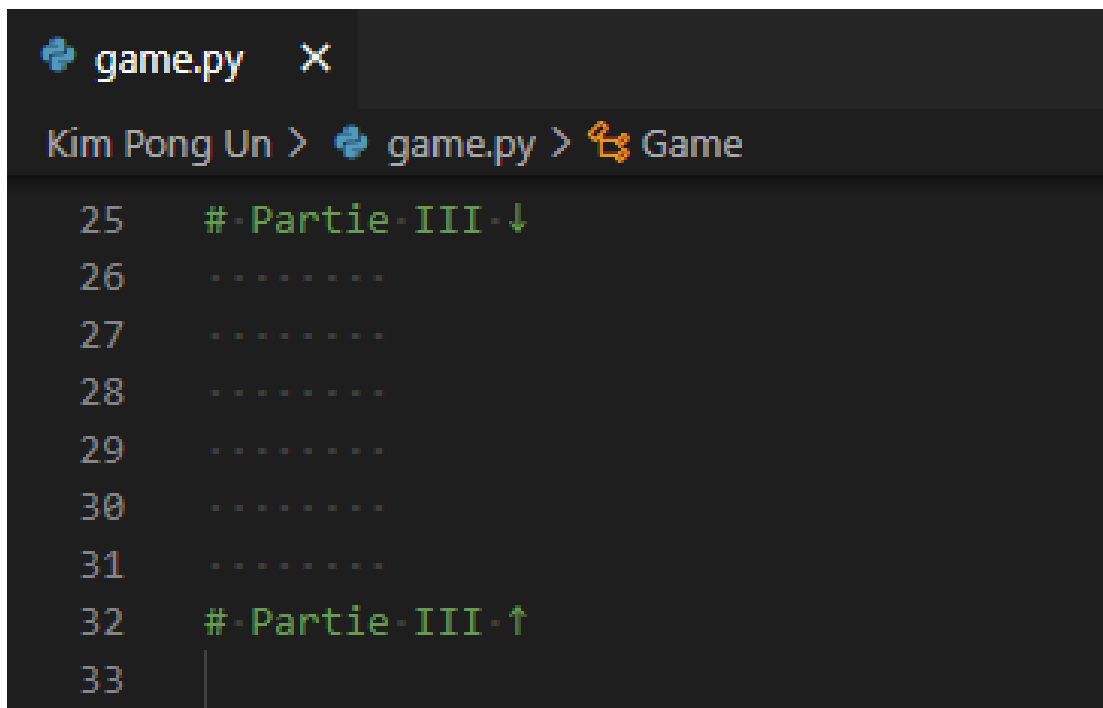
`self.filetWidth` : la taille désirée du filet en largeur

`self.filetHeight` : la taille désirée du filet en hauteur

Mettez votre instinct ainsi que votre raisonnement d'accord afin les initialiser avec les bonnes valeurs !

(Ce sont toutes des entiers numériques : int)

Dirigez-vous au même emplacement que dans la capture d'écran ci-dessous ↓



```
game.py X
Kim Pong Un > game.py > Game
25  #-Partie-III-↓
26  .....
27  .....
28  .....
29  .....
30  .....
31  .....
32  #-Partie-III-↑
33
```

IV. Ces raquettes vont danser

Maintenant que nous avons un visuel, l'on peut enfin contempler nos deux magnifiques raquettes de tennis. Mais elles ne bougent pas d'un pouce. Corrigons ça !

```
if (eventKey[pygame.K_TOUCHEDUCLAVIER]):
```

↑ Permet de détecter l'appui de la touche du clavier « TOUCHEDUCLAVIER ». Par exemple :

```
if (eventKey[pygame.K_u]):
```

↑ Détectera si l'utilisateur a appuyé sur la touche u du clavier.

Nous vous avons fourni les raquettes sous forme de variables :

`self.Bar1` : raquette de gauche

`self.Bar2` : raquette de droite

Ces variables sont spéciales, elles possèdent des fonctions. (wallah)

Nos deux raquettes possèdent les fonctions :

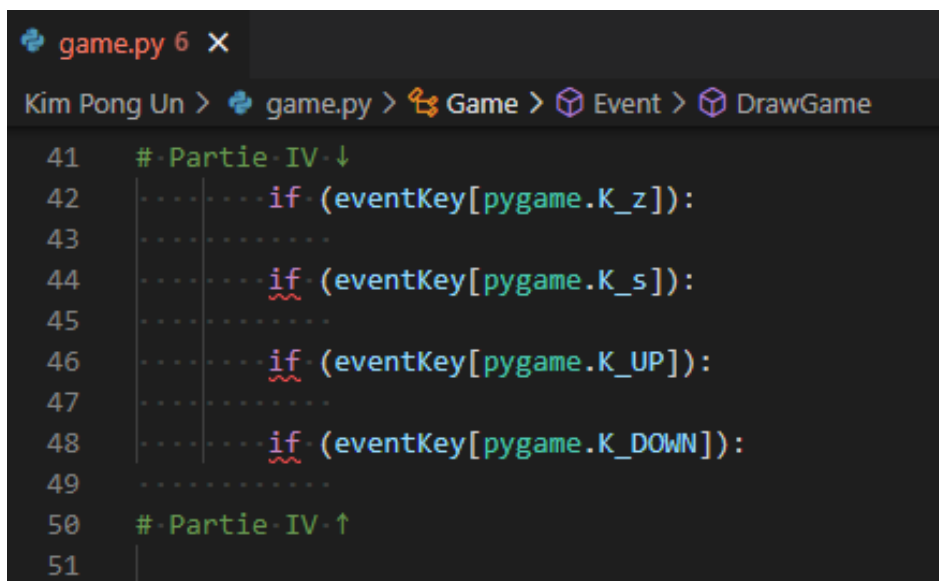
`moveUp()` : bouger vers le haut

`moveDown()` : bouger vers le bas

Petit exemple :

```
self.Bar1.moveUp() #MA RAQUETTE 1 BOUGE VERS LE HAUT
```

Codez à cet emplacement ↓



```
game.py 6 X
Kim Pong Un > game.py > Game > Event > DrawGame
41  #Partie-IV-↓
42  ..... if (eventKey[pygame.K_z]):
43  .....
44  ..... if (eventKey[pygame.K_s]):
45  .....
46  ..... if (eventKey[pygame.K_UP]):
47  .....
48  ..... if (eventKey[pygame.K_DOWN]):
49  .....
50  #Partie-IV-↑
51
```

V. Un corps à notre balle tu donneras

Les raquettes ont un mouvement que nous pouvons contrôler ! C'est magnifique ! Mais inutile. Pour que le jeu soit un véritable tennis de table, il faut que notre balle rebondisse sur nos raquettes.

Changeons de fichier, direction : « ball.py »

Notre balle se dirige grâce à deux variables :

`self.directionX` : oscille entre -1 et 1 pour donner la direction horizontale de la balle (entre -1 et 0 la balle ira vers la gauche, entre 0 et 1 la balle ira vers la droite)

`self.directionY` : oscille entre -1 et 1 pour donner la direction verticale de la balle (-1 - 0 : vers le haut ; 0 - 1 : vers le bas)

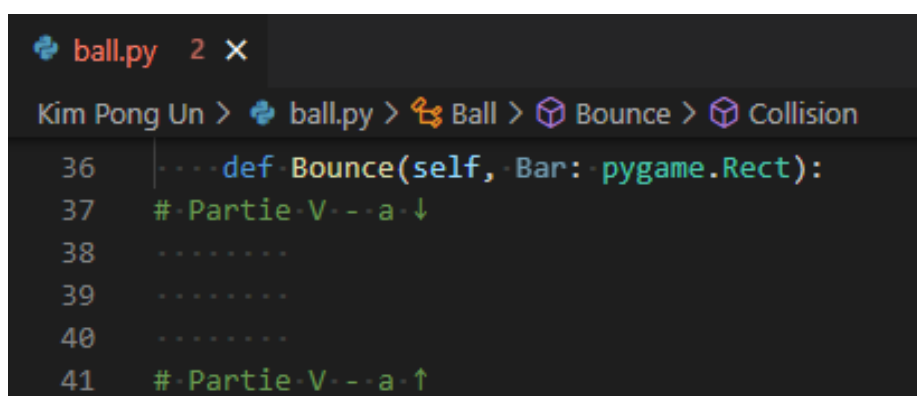
Pour résumer, si `self.directionX = 0.8` et `self.directionY = -0.2`, la balle se dirigera intensivement vers la droite et un peu vers le haut.

a. Les raquettes

Notre balle doit rebondir horizontalement sur nos raquettes, bien heureusement nos fidèles Cobra se sont déjà occupés de détecter la collision entre la balle et la raquette. 😊

Il vous reste juste à vous occuper du changement de direction horizontale de la balle

Par-là ↓



```
ball.py 2 X
Kim Pong Un > ball.py > Ball > Bounce > Collision
36     ...def Bounce(self, Bar: pygame.Rect):
37         # Partie V -- a ↓
38         .....
39         .....
40         .....
41         # Partie V -- a ↑
```

Il vous reste un peu d'espace pour augmenter la variable `self.speed` après le changement de direction si vous désirez ajouter un peu de difficulté.

`self.speed` : entier numérique qui contrôle la vitesse de la balle (est initialisé à 10).

b. Les bordures haut et bas

Maintenant que votre balle rebondit sur les raquettes, elle doit rebondir sur les bordures pour qu'elle ne s'envole pas !

Cette fois-ci vous avez à votre disposition 2 variables :

`self.rect.top` : coordonnée sur l'axe Y de votre balle, donc sa position en hauteur.

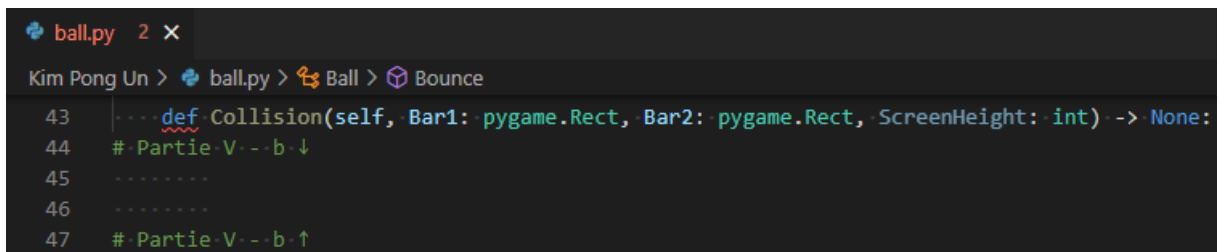
`ScreenHeight` : taille de la fenêtre du jeu en hauteur. (Attention : cette variable ne doit pas être utilisée avec « `self.` » devant !)

Indice : vous ne souhaitez pas modifier ces deux variables

Avec ces deux variables, à vous de détecter quand la balle sort de l'écran en haut ou en bas.

Lorsque cela sera fait, vous pourrez changer la direction verticale de votre balle, exactement de la même manière dont vous avez changé la direction horizontale précédemment.

À vous ! ↓



```
ball.py 2 x
Kim Pong Un > ball.py > Ball > Bounce
43     def Collision(self, Bar1: pygame.Rect, Bar2: pygame.Rect, ScreenHeight: int) -> None:
44         # Partie V -- b ↓
45         .....
46         .....
47         # Partie V -- b ↑
```

VI. Dans la vie il n'y ni perdant ni vainqueur

Le plus dur est passé, nous pouvons contrôler les raquettes, et s'en servir pour faire rebondir la balle.

Mais tous cela reste vain si nous n'avons ni moyen de savoir que l'on a marqué un point, ni d'arbitre pour les compter ces points.

Cette fois-ci, nous allons finaliser votre Ping Pythong !

a. Détecter la sortie de balle à droite et à gauche

Après avoir détecté la sortie de la balle verticalement, vous allez vous en occuper horizontalement, et pour cela vous allez devoir compléter une fonction dont nous allons nous servir juste après.

```
def outOfBorder(self, ScreenWidth: int) -> int:
```

Voilà la fonction en question ↑

Voici les variables à votre disposition :

`self.rect.left` : coordonnée sur l'axe X de votre balle, donc sa position en longueur.

`ScreenWidth` : taille de la fenêtre du jeu en largeur. (Attention : pas de « self. » comme `ScreenHeight`!)

(N'hésitez pas à revenir en arrière sur le sujet par moment pour bien vous comprendre le fonctionnement de certaines variables)

Néanmoins, vous n'allez pas faire rebondir la balle lorsque celle-ci sort de l'écran sur les côtés, vous allez faire « `return` » votre fonction `outOfBorder` une certaine valeur pour chaque cas différent cette fois.

Si la balle :

- Ne sort pas de l'écran : `return 0`
- Sort de l'écran côté joueur 1 : `return 1`
- Sort de l'écran côté joueur 2 : `return 2`

Ici ↓

```
ball.py 3 X
Kim Pong Un > ball.py > Ball > Bounce > outOfBorder > Move
60  ....def outOfBorder(self, ScreenWidth: int) -> int:
61      # Partie VI -- a ↓
62      .....
63      .....
64      .....
65      .....
66      .....
67      # Partie VI -- a ↑
```


b. Augmenter les scores

Retournons tous d'abord dans le fichier « game.py »

Grâce à votre dur labeur, vous avez codé une fonction qui vous permet de détecter la sortie de balle horizontalement.

```
self.ball.outOfBorder(self.ScreenWidth)
```

Voici la fonction en question et comment l'utiliser ↑

Cette fonction vous **return** donc des valeurs entre 0 et 2 pour chaque cas spécifique, vous allez vous en servir pour augmenter les scores des joueurs 1 et 2.

Les variables dont vous allez avoir besoin :

self.Score1 : contient le score du Joueur 1 (parce exemple 12 s'il a marqué 12 points)

self.Score2 : contient le score du Joueur 2

Mais ce serait trop facile sans une tâche minime à vous ajouter !

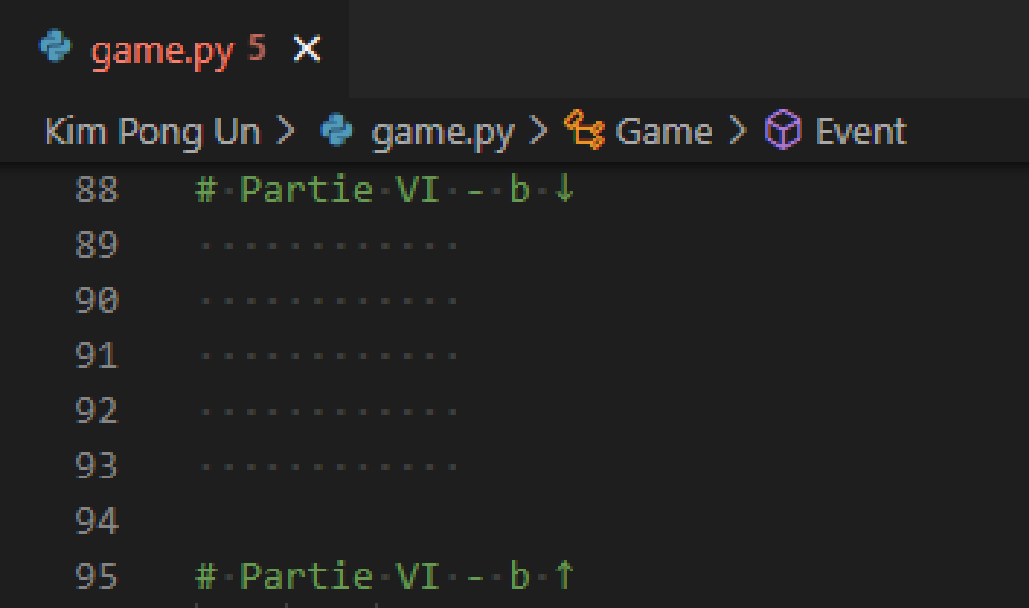
Lorsque le joueur marque un point, il faut renvoyer la balle au milieu, bien heureusement pour vous, nous avons déjà codé la fonction qui vous permettra de faire cela.

Il faudra juste vous en servir à chaque fois que vous détectez qu'un joueur marque un point :

```
self.ball.Reset(self.ScreenWidth, self.ScreenHeight)
```

Replace la balle au centre ↑

Ça se passe ici ↓



```
game.py 5 X
Kim Pong Un > game.py > Game > Event
88  # Partie VI -- b ↓
89
90
91
92
93
94
95  # Partie VI -- b ↑
```

VII. Bonus

Bien joué !

Vous êtes arrivé à la fin de ce sujet, et à l'image d'Allan Alcorn le papa de Pong, vous êtes devenu extrêmement célèbre et avez rendu votre patron très content !

Vous pouvez désormais développer vos compétences afin de réaliser plusieurs bonus comme :

- * Pimper votre Pong en modifiant les couleurs des raquettes, de la balle, etc...
- * Ajouter un peu de musique grâce à la librairie Pygame
- * Importer des images pour mettre un joli fond d'écran sur votre jeu
- * Modifier certains paramètres comme la vitesse des raquettes ou l'accélération de la balle
- * *Absolument n'importe quel autre bonus dont la seule limite est votre imagination !*

