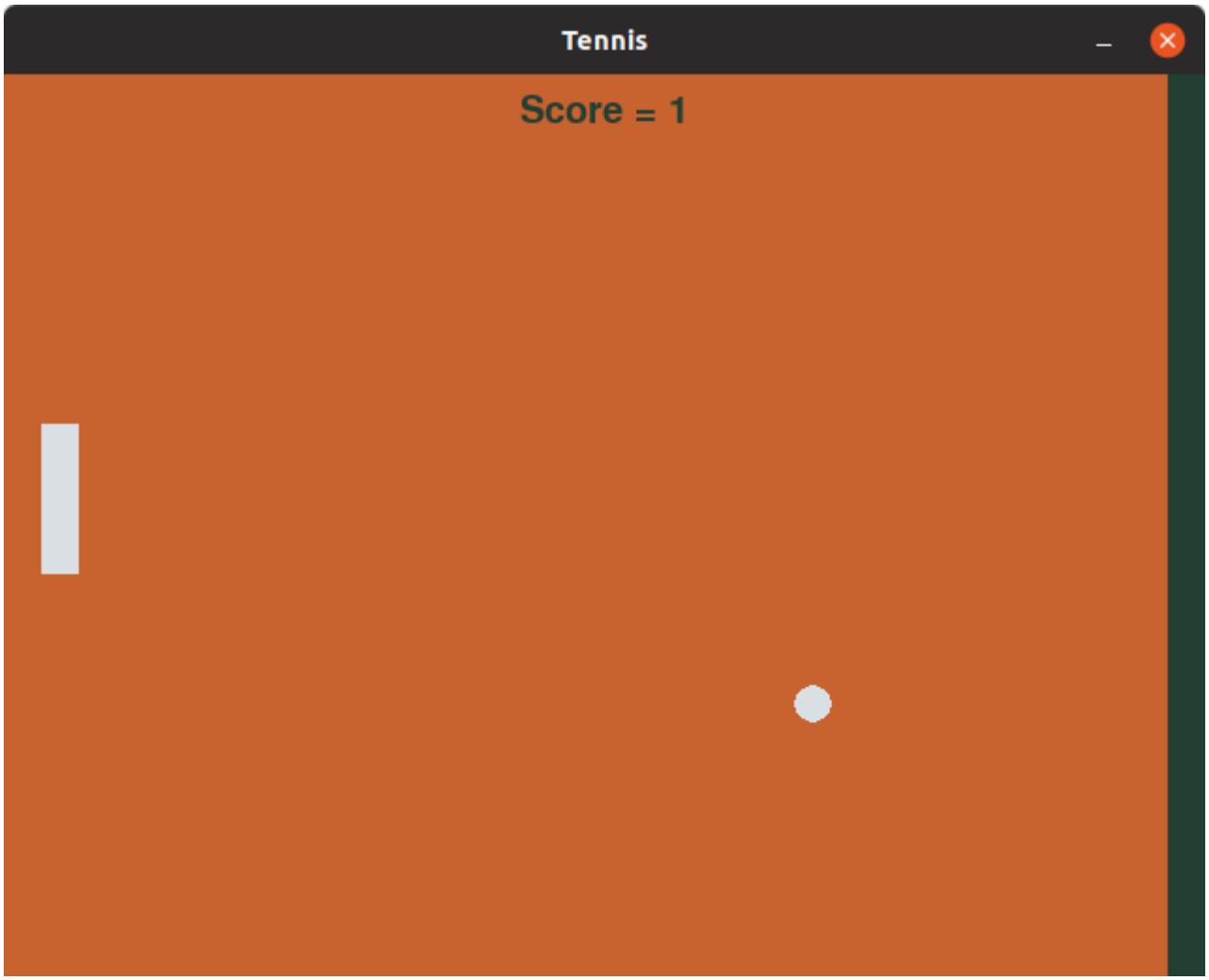
Pygame non utilise

1. Un Premier Jeu - Pong

Pong est l'un des premiers jeux vidéo d'arcade commercialisé en 1972. Il s'inspire du tennis de table. Devant l'engouement généré par le jeu, Pong est porté sur une console de salon à partir de 1975.

Le package PyGame permet de réaliser avec Python des jeux en quelques heures. Nous allons donc créer un jeu de pong rudimentaire en utilisant des classes pour gérer la balle et la raquette

Voici ce à quoi ressemblera l'interface du jeu que nous allons créer :



Nous allons mettre en oeuvre le concept objet en définissant une classe Balle et une classe Raquette pour le jeu de Pong.

1.1. Un premier exemple

Voici le code d'un premier jeu avant PONG :

Script Python

```
import pygame
from pygame.locals import *

HAUTEUR_FENETRE = 480
LARGEUR_FENETRE = 640

fenetre = pygame.display.set_mode( (LARGEUR_FENETRE, HAUTEUR_FENETRE) )
pygame.display.set_caption("Titre de la fenêtre")
pygame.font.init()
fonte = pygame.font.Font(None, 30)
```

La fenêtre s'affiche mais disparait aussitôt. il faut créer une boucle pour la garder ouverte et rafraichir l'affichage dans la fenêtre :

Script Python

```
import pygame
from pygame.locals import *

HAUTEUR_FENETRE = 480
LARGEUR_FENETRE = 640

fenetre = pygame.display.set_mode( (LARGEUR_FENETRE, HAUTEUR_FENETRE) )
pygame.display.set_caption("Titre de la fenêtre")
pygame.font.init()
fonte = pygame.font.Font(None, 30)

while True:
  pygame.display.update()
```

L'écran reste noir et rien ne s'affiche. Dans la boucle while il faut réaliser une série d'actions : afficher des images, du texte, etc.

Script Python

```
import pygame
from pygame.locals import *

HAUTEUR_FENETRE = 480
LARGEUR_FENETRE = 640

fenetre = pygame.display.set_mode( (LARGEUR_FENETRE, HAUTEUR_FENETRE) )
pygame.display.set_caption("Titre de la fenêtre")
pygame.font.init()
fonte = pygame.font.Font(None, 30)

x = LARGEUR_FENETRE // 2
y = HAUTEUR_FENETRE // 2
rayon = 10
while True:
  # dessine la balle
  pygame.draw.circle(fenetre, (255,255,255), (x, y), rayon)
  pygame.display.update()

  # attente du clic souris sur x pour fermer la fenêtre
```

```
  # ou Alt+F4 sous Linux
  for event in pygame.event.get():
    if event.type == pygame.QUIT:
      pygame.quit()
      exit()
```

Enfin il est nécessaire de déplacer les éléments dans la fenêtre :

Script Python

```
import pygame
from pygame.locals import *

# initialisation de l'écran de jeu
pygame.init()

HAUTEUR_FENETRE = 480
LARGEUR_FENETRE = 640

fenetre = pygame.display.set_mode( (LARGEUR_FENETRE, HAUTEUR_FENETRE) )
pygame.display.set_caption("Titre de la fenêtre")
pygame.font.init()
fonte = pygame.font.Font(None, 30)

# données de la balle
x = LARGEUR_FENETRE // 2
y = HAUTEUR_FENETRE // 2
rayon = 10

# vecteur de déplacement
vx = 1
vy = 1

while True:
  # temporise
  pygame.time.delay(15)

  # dessine le fond de l'écran en noir
  fenetre.fill( (0,0,0) )

  # dessine la balle
  pygame.draw.circle(fenetre, (255,255,255), (x, y), rayon)

  # nouvelle position
  x = x + vx
  y = y + vy

  if x >= LARGEUR_FENETRE or x <= 0:
    vx = -vx

  if y >= HAUTEUR_FENETRE or y <= 0:
    vy = -vy

  pygame.display.update()

del(police)
pygame.quit()
```

Exercice 1.

Améliorer la gestion du rebond car la balle sort de l'écran.

Gestion des événements Afin de gérer les évènements (appui sur une touche, clic souris, etc), il faut énumérer chacun des événements reçus et vérifier celui qui nous intéresse.

Sur l'exemple suivant on test l'appui sur la touche 'A', un clic souris ou le fait de quitter la fenêtre en cliquant sur l'icone correspondante en haut de la fenêtre.

Script Python

```
import pygame
import sys
from pygame.locals import *

HAUTEUR_FENETRE = 480
LARGEUR_FENETRE = 640

fenetre = pygame.display.set_mode( (LARGEUR_FENETRE, HAUTEUR_FENETRE) )
pygame.display.set_caption("Titre de la fenêtre")
pygame.font.init()
fonte = pygame.font.Font(None, 30)

# données de la balle
x = LARGEUR_FENETRE // 2
y = HAUTEUR_FENETRE // 2
rayon = 10
# vecteur de déplacement
vx = 1
vy = 1
while True:
  # temporise
  pygame.time.delay(15)

  # dessine le fond de l'écran en noir
  fenetre.fill( (0,0,0) )

  # dessine la balle
  pygame.draw.circle(fenetre, (255,255,255), (x, y), rayon)

  # nouvelle position
  x = x + vx
  y = y + vy

  if x >= LARGEUR_FENETRE or x <= 0:
    vx = -vx

  if y >= HAUTEUR_FENETRE or y <= 0:
    vy = -vy

  for event in pygame.event.get():
    if event.type == KEYDOWN:
      if event.key == pygame.K_a:
        vx = -vx
    if event.type == QUIT:
```

```
      pygame.quit()
      sys.exit()
    if event.type == MOUSEBUTTONDOWN:
      vy = -vy

  pygame.display.update()
```