Développement de jeux vidéo en Python avec Pygame

Campus du Libre 2019, 23 novembre 2019, Villeurbanne

Benoît Prieur - CC-By-SA

Les fondamentaux (1) : la fenêtre de jeu

On parle ici de jeu graphique. Ce qui implique d'avoir une **fenêtre de jeu**. Cela suppose d'évoquer les trois points suivants :

- la boucle de jeu ;
- la notion de "sprite" ;
- la gestion des collisions.

Les fondamentaux (2) : la boucle de jeu

Rapport au temps :

- obtenir les informations issues du joueur/joueuse et des périphériques (clavier, joystick, souris) ;
- mise à jour des données du système ;
- mise à jour de l'état (position, variables diverses) des éléments composants le jeu et donc de la fenêtre de jeu dans son ensemble.

Les fondamentaux (3) : les raisons de la notion de *Sprite*

Deux besoins :

- matérialisation graphique d'un personnage par exemple (apparence, position géographique au sein de la fenêtre de jeu) ;
- données relatives au personnage lui même (caractéristiques, points de vie, nom, âge etc.) ;

- Intérêt de l'approche objet.

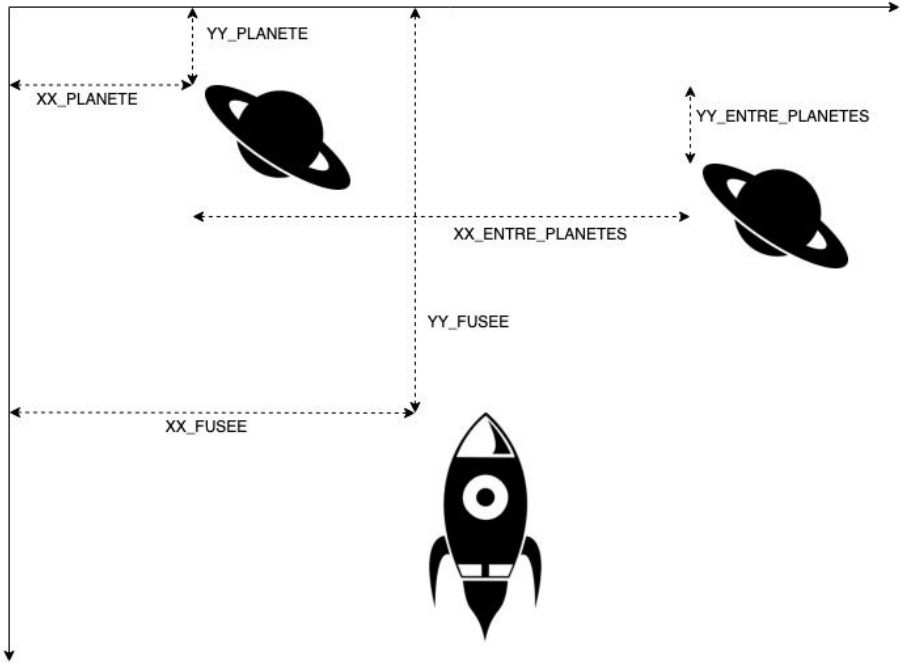
Les fondamentaux (4) : la gestion des collisions

Presque tout ce qui est constitutif d'un jeu vidéo est relatif à la gestion des collisions. C'est-à-dire à la rencontre (la collision) entre objets :

- exemple d'un projectile qui touche un personnage ;
- exemple du rebond d'un ballon sur le sol ;
- exemple de la disposition aléatoire d'objets : on gère les collisions éventuelles.

Gestion des collisions : difficultés au niveau géométrique

La résolution de la gestion des collisions peut être très complexe d'un point de vue géométrique.

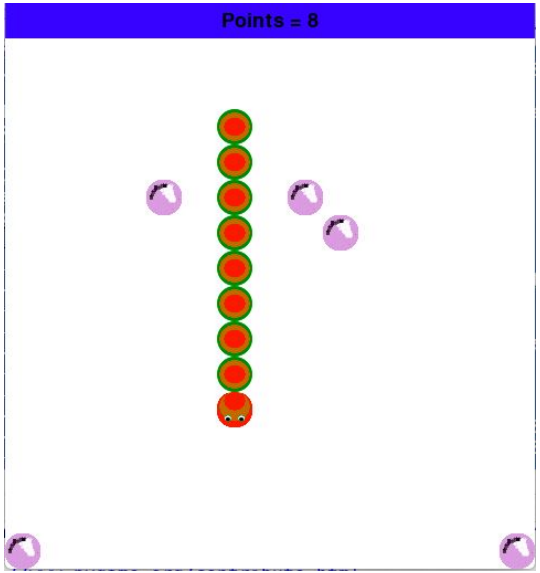


Présentation de Pygame

- Framework Python publié sous licence libre. Basé sur SDL.
- Portabilité MasOS, Windows, Linux
- Orientation jeu 2D même si la 3D est possible.
- Début des années 2000, Peter Shinners.
- *Frets on Fire* (clone de *Guitar Hero*) ou *Dangerous High School Girls in Trouble!*
- *Makers*, monde de l'éducation, etc.

Exemple du jour : le jeu du serpent

- Serpent se déplace avec les flèches du clavier.
- Quand il mange, il grandit de +1.
- Quand il mange : +1 point.
- La nourriture apparaît de manière aléatoire.
- Quand il rencontre son propre corps : perdu.
- Un espace pour le score.



Jeu du serpent : les fichiers du jeu

Disponible en ligne à cette URL :
https://github.com/benprieur/Pygame-Sprite/tree/master/Serpent

- 3 images (tête, corps, nourriture).
- 2 sons (manger, perdre).
- 1 fichier Python.

Les imports du jeu

```
import pygame, random, sys

pygame.init()

pygame.mixer.init()
```

Les constantes du programme

```
NOIR = (0, 0, 0)
BLANC = (255, 255, 255)

TUILE_TAILLE = 32
TUILE_NOMBRE = 15
SCORE_HAUTEUR = 32

LARGEUR = TUILE_TAILLE * TUILE_NOMBRE
HAUTEUR = TUILE_TAILLE * TUILE_NOMBRE + SCORE_HAUTEUR

POINT_UNITE = 1
DUREE_NOURRITURE_DISPARITION = 5500
```

Mise en place

```
pygame.init()

screen = pygame.display.set_mode([LARGEUR, HAUTEUR])
pygame.display.set_caption('Le jeu du serpent')

LISTE_SERPENT = pygame.sprite.Group()
LISTE_NOURRITURE = pygame.sprite.Group()
LISTE_GLOBALE_SPRITES = pygame.sprite.Group()

SON_NOURRITURE = pygame.mixer.Sound('NOURRITURE.wav')
SON_NOURRITURE.set_volume(1.0)

SON_PERDU = pygame.mixer.Sound('PERDU.wav')
SON_PERDU.set_volume(1.0)

_serpent = SERPENT()
LISTE_GLOBALE_SPRITES.add(_serpent)

clock = pygame.time.Clock()

print("C'est parti...")

TERMINE = False
```

Sprite et listes de sprites

LISTE_SERPENT = pygame.sprite.Group()

LISTE_NOURRITURE = pygame.sprite.Group()

LISTE_GLOBALE_SPRITES = pygame.sprite.Group()

La classe de sprite NOURRITURE

```
# CLASSE NOURRITURE
class NOURRITURE(pygame.sprite.Sprite):
  def __init__(self):
    pygame.sprite.Sprite.__init__(self)

    self.image = pygame.image.load("NOURRITURE.png").convert_alpha()

    self.rect = self.image.get_rect()
    self.rect.y = random.randint(0, TUILE_NOMBRE - 1) * TUILE_TAILLE + SCORE_HAUTEUR
    self.rect.x = random.randint(0, TUILE_NOMBRE - 1) * TUILE_TAILLE

    self.time = pygame.time.get_ticks()

  def update(self):
    if pygame.time.get_ticks() - self.time > DUREE_NOURRITURE_DISPARITION:
      self.kill()
```

La classe de sprite CORPS

```
# CLASSE CORPS
class CORPS(pygame.sprite.Sprite):
  def __init__(self, x, y):
    pygame.sprite.Sprite.__init__(self)

    self.image = pygame.image.load("CORPS.png").convert_alpha()

    self.rect = self.image.get_rect()
    self.rect.x = x
    self.rect.y = y

  def GET_X(self):
    return self.rect.x

  def GET_Y(self):
    return self.rect.y

  def set_xy(self, x, y):
    self.rect.x = x
    self.rect.y = y
```

La classe de sprite SERPENT

```
# CLASSE SERPENT
class SERPENT(pygame.sprite.Sprite):
  def __init__(self):
    pygame.sprite.Sprite.__init__(self)

    self.SON_NOURRITURE = pygame.mixer.Sound('NOURRITURE.wav')
    self.SON_NOURRITURE.set_volume(1.0)

    self.TETE = pygame.image.load("TETE.png").convert_alpha()
    self.image = self.TETE

    self.rect = self.image.get_rect()
    self.rect.y = (TUILE_NOMBRE / 2) * TUILE_TAILLE
    self.rect.x = (TUILE_NOMBRE / 2) * TUILE_TAILLE

    self.POINTS = 0
    self.DIRECTION = 'G'
    self.TERMINE = False

  def AJOUTER_NOUVEAU_CORPS(self):
    nouveau_corps = CORPS(self.rect.x, self.rect.y)
    LISTE_SERPENT.add(nouveau_corps)
    LISTE_GLOBALE_SPRITES.add(nouveau_corps)
```

SERPENT (2)

Fonction *Update*

```
def update(self):
    COORD_COURANTE_X = self.rect.x
    COORD_COURANTE_Y = self.rect.y

    if self.DIRECTION == 'G':
        self.image = pygame.transform.rotate(self.TETE, 90)
        self.rect.x -= TUILE_TAILLE
    elif self.DIRECTION == 'D':
        self.image = pygame.transform.rotate(self.TETE, -90)
        self.rect.x += TUILE_TAILLE
    elif self.DIRECTION == 'H':
        self.image = pygame.transform.rotate(self.TETE, 0)
        self.rect.y -= TUILE_TAILLE
    elif self.DIRECTION == 'B':
        self.image = pygame.transform.rotate(self.TETE, 180)
        self.rect.y += TUILE_TAILLE

    for ELT in LISTE_SERPENT:
        x = ELT.GET_X()
        y = ELT.GET_Y()
        ELT.set_xy(COORD_COURANTE_X, COORD_COURANTE_Y)
        COORD_COURANTE_X = x
        COORD_COURANTE_Y = y

    if self.rect.x >= LARGEUR:
        self.rect.x = 0
    elif self.rect.x < 0:
        self.rect.x = LARGEUR - TUILE_TAILLE
    elif self.rect.y >= HAUTEUR:
        self.rect.y = SCORE_HAUTEUR
    elif self.rect.y < SCORE_HAUTEUR:
        self.rect.y = HAUTEUR - SCORE_HAUTEUR
```

La classe de sprite SERPENT (3) : gestion des collisions

```
LISTE_COLLISION_SERPENT = pygame.sprite.spritecollide(self, LISTE_SERPENT, False)
if len(LISTE_COLLISION_SERPENT):
  print("Perdu")
  self.TERMINE = True

LISTE_COLLISION_NOURRITURE = pygame.sprite.spritecollide(self, LISTE_NOURRITURE, False)
for nourriture in LISTE_COLLISION_NOURRITURE:
  nourriture.kill()
  self.AJOUTER_NOUVEAU_CORPS()
  self.SON_NOURRITURE.play()
  self.POINTS += POINT_UNITE
```

Afficher le score

```
# AFFICHER_SCORE
def AFFICHER_SCORE():
    font = pygame.font.SysFont('Arial', TUILE_TAILLE - 5)
    background = pygame.Surface((LARGEUR, SCORE_HAUTEUR))
    background = background.convert()
    background.fill(BLANC)
    text = font.render("Points = %d" % _serpent.POINTS, 1, NOIR)
    textpos = text.get_rect(centerx=LARGEUR / 2, centery=SCORE_HAUTEUR / 2)
    background.blit(text, textpos)
    screen.blit(background, (0, 0))
```

Boucle de jeu

```
while not TERMINE:
  for event in pygame.event.get():
    if event.type == pygame.QUIT:
      TERMINE = True
    elif event.type == pygame.KEYDOWN:
      if event.key == pygame.K_LEFT and _serpent.DIRECTION != 'D':
        _serpent.DIRECTION = 'G'
        break
      elif event.key == pygame.K_RIGHT and _serpent.DIRECTION != 'G':
        _serpent.DIRECTION = 'D'
        break
      elif event.key == pygame.K_UP and _serpent.DIRECTION != 'B':
        _serpent.DIRECTION = 'H'
        break
      elif event.key == pygame.K_DOWN and _serpent.DIRECTION != 'H':
        _serpent.DIRECTION = 'B'
        break

  if random.randint(0, 18) == 0:
    _nourriture = NOURRITURE()

    LISTE_CONFLIT = pygame.sprite.spritecollide(_nourriture, LISTE_GLOBALE_SPRITES, False)
    if len(LISTE_CONFLIT) == 0:
      SON_NOURRITURE.play()
      LISTE_GLOBALE_SPRITES.add(_nourriture)
      LISTE_NOURRITURE.add(_nourriture)

  LISTE_GLOBALE_SPRITES.update()
  screen.fill(BLANC)

  LISTE_GLOBALE_SPRITES.draw(screen)
  AFFICHER_SCORE()
```

Me contacter

- Formulaire de contact sur le site de Soarthec
- Twitter : benprieur