PROJET NSI

PACMAN

Jules MULLER
Cédric GUIGNER
Thibault Rigallaud

SOMMAIRE

01 - RÈGLES

RÈGLES

01

But du jeu
PACMAN doit manger toutes les pac-gommes sans se faire toucher par les fantomes.

02

PACMAN
Afin d’éviter les fantomes, PACMAN peut utiliser des super pac-gommes afin de manger les fantomes et récupérer 200 points par fantome

03

FIN DE JEU
Si PACMAN n’arrive pas à manger toutes les pac-gommes lors de 3 tentatives car il a 3 vies alors le jeu se finit

02 - MAP

LA MAP

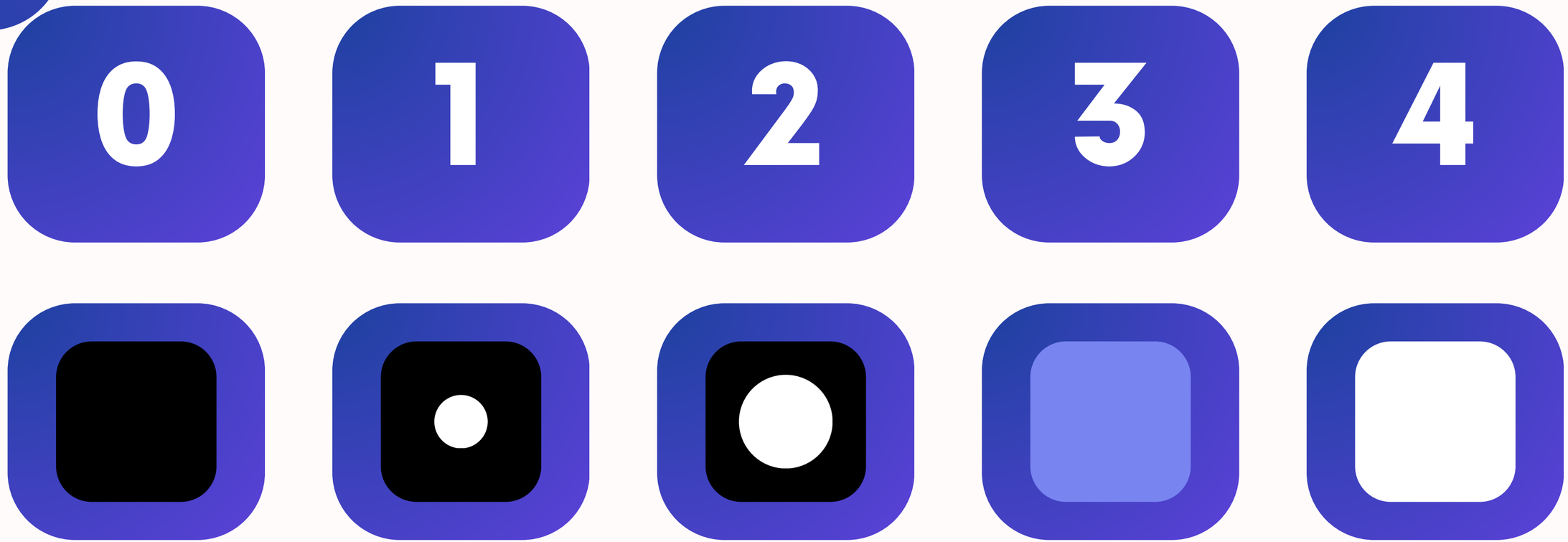
LA MAP

3	3	0	4	0	3	3
3	1	1	2	1	1	3

LA MAP

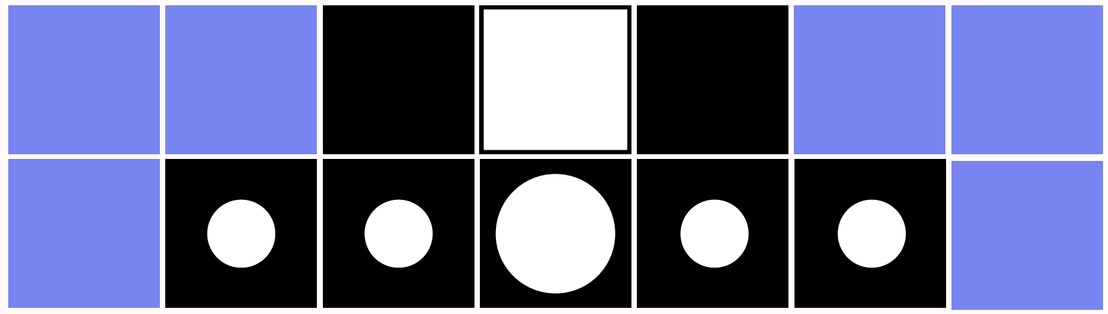
0 1 2 3 4

LA MAP



LA MAP

3	3	0	4	0	3	3
3	1	1	2	1	1	3



LA MAP

map_1.py

```
map_origin = [
  [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3],
  [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3],
  [3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 2, 3, 0, 0, 3, 1, 3, 0, 0, 0, 3, 1, 3, 3, 1, 3, 0, 0, 0, 3, 1, 3, 0, 0, 3, 2, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 1, 1, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 1, 3, 3, 3],
  [3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3],
  [3, 0, 0, 0, 0, 0, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 0, 0, 0, 0, 0, 3],
  [3, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 3],
  [3, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 3, 3, 3, 4, 4, 3, 3, 3, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 3],
  [3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3],
  [0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 3, 0, 0, 0, 0, 0, 0, 3, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0],
  [3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3],
  [3, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 3],
  [3, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 3],
  [3, 0, 0, 0, 0, 0, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 0, 0, 0, 0, 0, 3],
  [3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3],
  [3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 1, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 2, 1, 1, 3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 1, 1, 2, 3, 3],
  [3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3],
  [3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3],
  [3, 3, 1, 1, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 3, 3, 1, 1, 1, 1, 1, 3, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3, 1, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 1, 3, 3],
  [3, 3, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 3, 3, 3],
  [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3],
  [3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3, 3]
]
```

30X32

CLASS MAP

4 ATTRIBUTS DE CLASSE :

- __level__
- __map_select__
- __couleur__
- __map_origin__

2 MÉTHODES :

- dessiner_map()
- game_finish()

ATTRIBUTS PRIVÉS DONC MUTATEURS ET ACCESSEURS POUR CHAQUE ATTRIBUT

CLASS MAP

```python
import pygame
from pygame.locals import *
from Maps.map_1 import map_edited as map_1
from Maps.map_1 import map_origin as map_1_origin
pygame.init()
      You, 2 weeks ago • ClassMap, Fantome Texture, Pacma
maps = [map_1]
maps_origin = [map_1_origin]
```

→ 2 LISTES

MAPS_ORIGIN ↔ MAPS

CLASS MAP

DESSINER_MAP

```python
def dessiner_map(self, ecran, fenetre):
    y_space = ecran[0]//30 #On crée des maps en 30x32 donc on divise la largeur de l'écran par 30
    x_space = ecran[1]//32 #On crée des maps en 30x32 donc on divise la hauteur de l'écran par 32

    #On lit le tableau par liste, puis par élément. Donc par ligne puis par colonne
    for i in range(len(self.get_map_select())):
        for j in range(len(self.get_map_select()[i])):      You, 2 weeks ago • ClassMap, Fantome Texture, Pacman Texture, Ajuste…

            if self.get_map_select()[i][j] == 1:
                pygame.draw.rect(fenetre, (0, 0, 0), (j * x_space, i * y_space, 30, 30))
                pygame.draw.circle(fenetre, (255, 255, 255), (int(j * x_space + (x_space/2)), int(i * y_space + (y_space/2))), 5)#Ici, on
                calcule la position de départ x et y et on ajoute la moitié pour que la boule soit centrée

            elif self.get_map_select()[i][j] == 2:
                pygame.draw.rect(fenetre, (0, 0, 0), (j * x_space, i * y_space, 30, 30))
                pygame.draw.circle(fenetre, (255, 255, 255), (int(j * x_space + (x_space/2)), int(i * y_space + (y_space/2))), 10)

            elif self.get_map_select()[i][j] == 3:
                pygame.draw.rect(fenetre, self.get_couleur(), (j * x_space, i * y_space, 30, 30))

            elif self.get_map_select()[i][j] == 4:
                            pygame.draw.rect(fenetre, (255, 255, 255), (j * x_space, i * y_space, 30, 30))
```

CLASS MAP

GAME_FINISH

```
def game_finish(self, objet_pacman):
    if objet_pacman.get_touch() == True and objet_pacman.get_vie() < 1:
        objet_pacman.set_win(False)
        self.set_map_select(self.get_map_origin())
        return True

    for i in range(len(self.get_map_select())):
        if 1 in self.get_map_select()[i]:
            return False
    objet_pacman.set_win(True)
    self.set_map_select(self.get_map_origin())
    return True      You, 2 weeks ago • Ajout sound effect + game_finish
```

03 - PACMAN

CLASS PACMAN

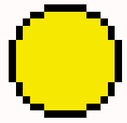
9 ATTRIBUTS DE CLASSE :

- __vie__
- __posx__
- __posy__
- __can_eat__
- __orientation__
- __score__
- __check_collision__
- __touch__
- __win__

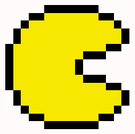
6 MÉTHODES :

- draw()
- move()
- check_collision()
- check_score()
- check_malade()
- check_eat_ghost()

2 TEXTURES



pac_man_img_neutre



pac_man_img

CLASS PACMAN

CHECK_COLLISION

[L, T, R, B]

```
num1 = ecran[1]//32
num2 = ecran[0]//30

self.set_collision(0, False)
self.set_collision(1, False)
self.set_collision(2, False)
self.set_collision(3, False)

if map[self.get_posy()//num1][self.get_posx() // num2] < 3:
    if self.get_orientation() == "right":
        if self.get_posx() < 870: #Pour éviter le "out of range"
            if map[self.get_posy()//num1][(self.get_posx() + 30) // num2] < 3:
                self.set_collision(2, True)
            else : self.__check_collision__[2] = False
        else : self.set_collision(2, True)

    elif self.get_orientation() == "left":
        if map[self.get_posy()//num1][(self.get_posx() - 15) // num2] < 3:
            self.set_collision(0, True)
        else : self.__check_collision__[0] = False

    elif self.get_orientation() == "top":
        if map[(self.get_posy() - 15)//num1][self.get_posx() // num2] < 3:
            self.set_collision(1, True)
        else : self.__check_collision__[1] = False

    elif self.get_orientation() == "bottom":
        if map[(self.get_posy() + 30)//num1][self.get_posx() // num2] < 3:
            self.set_collision(3, True)
        else : self.__check_collision__[3] = False
```

CLASS PACMAN

MOVE

```
def move(self):

    if self.get_posx() < -30:
        self.set_posx(900)
    elif self.get_posx() > 910:       You, 2 weeks ago • Mise en place du timer
        self.set_posx(-20)
    if self.get_orientation() == "left" and self.get_collision(0):
            self.min_posx(15)
    elif self.get_orientation() == "top" and self.get_collision(1):
            self.min_posy(15)
    elif self.get_orientation() == "right" and self.get_collision(2):
            self.add_posx(15)
    elif self.get_orientation() == "bottom" and self.get_collision(3):
            self.add_posy(15)
    else:
        self.add_posx(0)
        self.add_posy(0)
        self.min_posx(0)
        self.min_posy(0)
```

CLASS PACMAN

MOVE

```
def add_posx(self, valeur):
    self.set_posx(self.get_posx() + valeur)

def min_posx(self, valeur):
    self.set_posx(self.get_posx() - valeur)

def add_posy(self, valeur):
    self.set_posy(self.get_posy() + valeur)

def min_posy(self, valeur):
    self.set_posy(self.get_posy() - valeur)
```

CLASS PACMAN

03

DRAW

```python
def draw(self, surface):
    if self.get_orientation() == "None":
        surface.blit(pac_man_img_neutre, (self.get_posx(), self.get_posy()))
    elif self.get_orientation() == "right":
        surface.blit(pac_man_img, (self.get_posx(), self.get_posy()))
    elif self.get_orientation() == "top":
        surface.blit(pygame.transform.rotate(pac_man_img, 90), (self.get_posx(), self.get_posy()))
    elif self.get_orientation() == "bottom":
        surface.blit(pygame.transform.rotate(pac_man_img, -90), (self.get_posx(), self.get_posy()))
    elif self.get_orientation() == "left":
        surface.blit(pygame.transform.flip(pac_man_img, True, False), (self.get_posx(), self.get_posy()))
```

CLASS PACMAN

03

CHECK_SCORE

```
def check_score(self, ecran, map):
    num1 = ecran[1]//32
    num2 = ecran[0]//30

    if 0 < self.get_posx() < 900 : #Eviter le out of range
        if map[self.get_posy() // num1][self.get_posx() // num2] == 1:
            self.set_score(10)
            map[self.get_posy() // num1][self.get_posx() // num2] = 0
```

01

CLASS PACMAN

03

CHECK_MALADE

```
def check_malade(self, ecran, map):
    num1 = ecran[1]//32
    num2 = ecran[0]//30
    if 0 < self.get_posx() < 900 : #Eviter le out of range
        if map[self.get_posy() // num1][self.get_posx() // num2] == 2:
            self.set_can_eat(True)
            self.set_score(50)
            map[self.get_posy() // num1][self.get_posx() // num2] = 0
```

01

CLASS PACMAN

03

CHECK_EAT_GHOST

01

04 - FANTOMES

CLASS FANTOME

8 ATTRIBUTS DE CLASSE :

- __posx__
- __posy__
- __couleur__
- __respawn__
- __orientation__
- __malade__
- __finish_heal__
- __start_movement__

4 MÉTHODES :

- mvt()
- draw()
- check_collision()
- change_direction()

CLASS FANTOME

CHECK_COLLISION

```python
def check_collision(self, ecran, map):

    liste_collision = [True, True, True, True] #R, L, T, B

    if self.check_collision_right(ecran, map) == True:
        liste_collision[0] = False
    else : liste_collision[0] = True

    if self.check_collision_left(ecran, map) == True:
        liste_collision[1] = False
    else : liste_collision[1] = True

    if self.check_collision_top(ecran, map) == True:
        liste_collision[2] = False
    else : liste_collision[2] = True

    if self.check_collision_bottom(ecran, map) == True:
        liste_collision[3] = False
    else : liste_collision[3] = True

    return liste_collision
```

```python
def check_collision_right(self, ecran, map):
    num1 = ecran[1]//32
    num2 = ecran[0]//30
    if map[self.get_posy()//num1][(self.get_posx() + 30)//num2] == 3:
        return True
    return False
```

CLASS FANTOME

CHECK_COLLISION

```python
def check_collision(self, ecran, map):

    liste_collision = [True, True, True, True] #R, L, T, B

    if self.check_collision_right(ecran, map) == True:
        liste_collision[0] = False
    else : liste_collision[0] = True

    if self.check_collision_left(ecran, map) == True:
        liste_collision[1] = False
    else : liste_collision[1] = True

    if self.check_collision_top(ecran, map) == True:
        liste_collision[2] = False
    else : liste_collision[2] = True

    if self.check_collision_bottom(ecran, map) == True:
        liste_collision[3] = False
    else : liste_collision[3] = True

    return liste_collision
```

```python
def check_collision_right(self, ecran, map):
    num1 = ecran[1]//32
    num2 = ecran[0]//30
    if map[self.get_posy()//num1][(self.get_posx() + 30)//num2] == 3:
        return True
    return False
```

CLASS FANTOME

CHANGE_DIRECTION

CLASS FANTOME

MVT

```python
def mvt(self):


    if self.get_orientation() == "left":
        self.min_posx(15)        You, 5 days ago
    elif self.get_orientation() == "top":
        self.min_posy(15)
    elif self.get_orientation() == "right":
        self.add_posx(15)
    elif self.get_orientation() == "bottom":
        self.add_posy(15)
```

```python
def add_posx(self, valeur):
    self.set_posx(self.get_posx() + valeur)

def min_posx(self, valeur):
    self.set_posx(self.get_posx() - valeur)

def add_posy(self, valeur):
    self.set_posy(self.get_posy() + valeur)

def min_posy(self, valeur):
    self.set_posy(self.get_posy() - valeur)
```

CLASS FANTOME

DRAW

```
    if self.get_malade() == False :
        if self.get_couleur() == 'red':
            fenetre.blit(fantome[0], (self.get_posx(), self.get_posy()))
        elif self.get_couleur() == 'orange':
            fenetre.blit(fantome[1], (self.get_posx(), self.get_posy()))
        elif self.get_couleur() == 'blue':
            fenetre.blit(fantome[2], (self.get_posx(), self.get_posy()))
        elif self.get_couleur() == 'pink':
            fenetre.blit(fantome[3], (self.get_posx(), self.get_posy()))
    else :
        if self.get_finish_heal():
            fenetre.blit(fantome[5], (self.get_posx(), self.get_posy()))
        else : fenetre.blit(fantome[4], (self.get_posx(), self.get_posy()))

    pygame.display.flip()
```

```
fantome_1 = pygame.transform.scale(pygame.image.load(f'Texture/Fantome/red.gif'), (25, 25))
fantome_2 = pygame.transform.scale(pygame.image.load(f'Texture/Fantome/orange.gif'), (25, 25))
fantome_3 = pygame.transform.scale(pygame.image.load(f'Texture/Fantome/blue.gif'), (25, 25))
fantome_4 = pygame.transform.scale(pygame.image.load(f'Texture/Fantome/pink.gif'), (25, 25))
fantome_malade = pygame.transform.scale(pygame.image.load(f'Texture/Fantome/malade.gif'), (25, 25))
fantome_malade_finish = pygame.transform.scale(pygame.image.load(f'Texture/Fantome/malade_finish.gif'), (25, 25))

fantome = [fantome_1, fantome_2, fantome_3, fantome_4, fantome_malade, fantome_malade_finish]
```

05 - MAIN

MAIN

MAIN.PY

HOME_SCREEN.PY

END_SCREEN.PY

MAIN.PY

GAME()

Affichage visuel

```
fenetre.fill([0,0,0])
image_score = police.render("Score : " + str(pac_man.get_score()), 1, (255, 255, 255))
image_pt_de_vie = police.render("Vie : " + str(pac_man.get_vie()), 1, (255, 255, 255))
map.dessiner_map(ecran, fenetre)
fenetre.blit(image_pt_de_vie, (750, 15))
fenetre.blit(image_score, (10, 15))
pygame.display.flip()
```

MAIN.PY

GAME()

Gestion PACMAN

```
pac_man.check_collision(ecran, map.get_map_select())
pac_man.move()
pac_man.draw(fenetre)
pac_man.check_score(ecran, map.get_map_select())
pac_man.check_malade(ecran, map.get_map_select())
```

MAIN.PY

```python
#Quand pacman est touché par un fantome sans powerBall
if pac_man.get_touch():
    pacman_dying_sound.play()
    pacman_original_sound.stop()
    pac_man.set_posx(pacman_x_spawn)
    pac_man.set_posy(pacman_y_spawn)
    pac_man.set_orientation("None")
    for i in range(len(fantomes)):
        fantomes[i].set_start_movement(True)
        fantomes[i].set_posx(ecran[0]//2)
        fantomes[i].set_posy(ecran[1]//2)

    time.sleep(pacman_dying_sound.get_length())
    pac_man.set_touch(False)
    pacman_original_sound.play()


if pac_man.get_can_eat():
    pacman_eating_sound.play(-1)
    pacman_original_sound.set_volume(0.0)
    can_eat = True
    tps_zero = pygame.time.get_ticks()
    pac_man.set_can_eat(False)


#Timer pour manger
if can_eat == True and (pygame.time.get_ticks() - tps_zero) <=8000 :
    for i in range(len(fantomes)):
        fantomes[i].set_malade(True)
    if (pygame.time.get_ticks() - tps_zero) >= 5000:
        for i in range(len(fantomes)):
            fantomes[i].set_finish_heal(True)
else :
    pacman_original_sound.set_volume(1.0)
    pacman_eating_sound.stop()
    for i in range(len(fantomes)):
        fantomes[i].set_malade(False)
        can_eat = False
        fantomes[i].set_finish_heal(False)
```

GAME()

- PACMAN touché
- Manger les fantômes

MAIN.PY

GAME()

Gestion Fantômes

```
for i in range(len(fantomes)):

    pac_man.check_eat_ghost(fantomes[i], ecran, pacman_eating_ghost_sound)      You, last week • Débuggage
    if fantomes[i].get_start_movement(): #Si le fantome est mort, il revient donc à son point de respawn e
    haut
        if fantomes[i].check_collision_top(ecran, map.get_map_select()) == False:
            fantomes[i].set_orientation("top")
        else :
            if randint(0, 1) == 0:
                fantomes[i].set_orientation("right")
            else : fantomes[i].set_orientation("left")
            fantomes[i].set_start_movement(False)

    fantomes[i].change_direction(fantomes[i].check_collision(ecran, map.get_map_select()))
    fantomes[i].mvt()
    fantomes[i].draw(fenetre)
```

MAIN.PY

GAME()

Contrôles + fin de jeu

```
    # routine pour pouvoir fermer «proprement» la fenêtre Pygame

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.display.quit()
            sys.exit()
    # Bouger le PAC MAN

        if event.type == KEYDOWN:
            if event.key == K_UP:
                pac_man.set_orientation("top")
            elif event.key == K_RIGHT:
                pac_man.set_orientation("right")
            elif event.key == K_DOWN:
                pac_man.set_orientation("bottom")
            elif event.key == K_LEFT:
                pac_man.set_orientation("left")


    pygame.display.update()
    time.sleep(0.1)

if pac_man.get_win() == False:
    pygame.mixer.stop()
    pacman_dying_sound.play()
    pacman_original_sound.stop()
    pac_man.set_posx(pacman_x_spawn)
    pac_man.set_posy(pacman_y_spawn)
    pac_man.set_orientation("None")
    for i in range(len(fantomes)):
        fantomes[i].set_start_movement(True)
        fantomes[i].set_posx(ecran[0]//2)
        fantomes[i].set_posy(ecran[1]//2)

    time.sleep(pacman_dying_sound.get_length())
    pac_man.set_touch(False)
    pacman_original_sound.play()
```

HOME_SCREEN.PY

```python
title_font = pygame.font.Font("Fonts/NEW UPDATED VERSION/horizon.otf" ,60)
subtitle_font = pygame.font.Font("Fonts/NEW UPDATED VERSION/horizon.otf" ,45)
text_font = pygame.font.Font("Fonts/NEW UPDATED VERSION/horizon.otf" ,25)
    You, 5 days ago • Final Project + Fonts
play_button = pygame.image.load('Texture/Buttons/Play_Button.png')
play_button_pressed = pygame.image.load('Texture/Buttons/Play_Button_Pressed.png')


play_button_rect = play_button.get_rect()
play_button_rect.x = 330
play_button_rect.y = 480


welcome_text = title_font.render("Welcome", 3,  (255, 199, 0))
to_text = text_font.render("TO", 3, (255, 255, 255))
pacman_text = subtitle_font.render("pacman", 3, ((255, 199, 0)))
team_text = text_font.render("BY Jules - cédric - thibault", 1, (255, 255, 255))

while True:
    fenetre.fill([0, 17, 59])
    fenetre.blit(welcome_text, (225, 145))
    fenetre.blit(to_text, (430, 258))
    fenetre.blit(pacman_text, (300, 316))
    fenetre.blit(team_text, (180, 765))

    if play_button_rect.collidepoint(pygame.mouse.get_pos()):
      fenetre.blit(play_button_pressed, play_button_rect)
    else : fenetre.blit(play_button, play_button_rect)

    pygame.display.flip()
    for event in pygame.event.get():
      if event.type == pygame.QUIT:
        pygame.display.quit()
        sys.exit()

      elif event.type == pygame.MOUSEBUTTONDOWN:
        if play_button_rect.collidepoint(pygame.mouse.get_pos()):
          pygame.mixer.stop()
          return True
```

END_SCREEN.PY

```python
title_font = pygame.font.Font("Fonts/NEW UPDATED VERSION/horizon.otf" ,60)
score_font = pygame.font.Font("Fonts/NEW UPDATED VERSION/horizon.otf" ,60)
text_font = pygame.font.Font("Fonts/NEW UPDATED VERSION/horizon.otf" ,20)

end_text_win = title_font.render("Gagné", 1, (255, 199, 0))
end_text_lose = title_font.render("Perdu", 1, (255, 199, 0))
text = text_font.render("Vous avez fait", 1, (255, 255, 255))
score_text = score_font.render(str(score) + " points", 1 ,(255, 255, 255))

retry_button = pygame.image.load('Texture/Buttons/Retry_Button.png')
retry_button_pressed = pygame.image.load('Texture/Buttons/Retry_Button_pressed.png')
quit_button = pygame.image.load('Texture/Buttons/Quit_Button.png')
quit_button_pressed = pygame.image.load('Texture/Buttons/Quit_Button_pressed.png')

retry_button_rect = retry_button.get_rect()
quit_button_rect = quit_button.get_rect()


#Position du Rect Retry_Button
retry_button_rect.x = 112
retry_button_rect.y = 600
#Position du Rect Quit_Button
quit_button_rect.x = 531
quit_button_rect.y = 600
```

```python
while True:
    fenetre.fill([0,23,80])


    if win_statut:
        end_text = end_text_win
        fenetre.blit(end_text, (280, 255))
    else :
        end_text = end_text_lose
        fenetre.blit(end_text, (280, 255))
    fenetre.blit(text, (315, 380))
    fenetre.blit(score_text, (210, 440))

    if retry_button_rect.collidepoint(pygame.mouse.get_pos()):
        fenetre.blit(retry_button_pressed, retry_button_rect)
    else :  fenetre.blit(retry_button, retry_button_rect)       You, last w

    if quit_button_rect.collidepoint(pygame.mouse.get_pos()):
        fenetre.blit(quit_button_pressed, quit_button_rect)
    else : fenetre.blit(quit_button, quit_button_rect)

    pygame.display.flip()

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            pygame.display.quit()
            sys.exit()

        elif event.type == pygame.MOUSEBUTTONDOWN:
            if retry_button_rect.collidepoint(pygame.mouse.get_pos()):
                return True
            if quit_button_rect.collidepoint(pygame.mouse.get_pos()):
                pygame.display.quit()
                sys.exit()
```

END_SCREEN.PY

```
while True:
#------------ START SCREEN ------------#

    if game_statut == "Start":
        home_screen(fenetre)
        game_statut = "Game"

#------------ GAME SCREEN ------------#

    elif game_statut == "Game":
        pacman_original_sound.play(-1)

        game(can_eat)
        game_statut = "End"
        pygame.mixer.stop()

#------------ END GAME SCREEN ------------#

    elif game_statut == "End":
            end_screen(pac_man.get_win(), pac_man.get_score(), ecran, fenetre)
            game_statut = "Game"
            vie = 3
            pac_man.set_vie(vie)
```

MERCI
POUR VOTRE
ATTENTION