Documentation Pygame Zero

Classe Actor

La classe Actor est utilisée pour représenter des personnages ou des objets qui apparaissent à l'écran.

Méthode ou Attribut	Description	Exemple
Actor(image, pos)	Crée un nouvel acteur avec une image et une position initiale.	actor = Actor('alien', (50, 50))
actor.x, actor.y	Coordonnées de l'acteur sur l'écran.	actor.x = 100 actor.y = 150
actor.width, actor.height	Largeur et hauteur de l'image de l'acteur.	w = actor.width h = actor.height
actor.direction	Direction de l'acteur en degrés, à utiliser avec actor.move_in_direction(speed).	actor.direction = randint(0, 360)
actor.angle	Angle de rotation du sprite de l'acteur, en degrés.	actor.angle = 90
actor.flip_x, actor.flip_y	Tags qui définissent si le sprite de l'acteur doit être inversé horizontalement ou verticalement.	actor.flip_x = True actor.flip_y = False
actor.images	Liste d'images constituant l'animation de l'acteur. A utiliser avec actor.animate().	actor.images = ['alien1', 'alien2']
actor.to_remove	Tag qui définit si l'acteur doit être supprimé du jeu (True ou False).	actor.to_remove = True
actor.draw()	Dessine l'acteur sur l'écran.	actor.draw()
actor.animate()	Fait avancer l'animation de l'acteur.	actor.animate()
actor.collidepoint(x, y)	Retourne True si l'acteur touche le point (x, y).	if actor.collidepoint(100, 100): print('Touché !')
actor.collides_with(other_actor)	Retourne True si l'acteur touche un autre acteur.	if actor.collides_with(enemy): print('Touché !')
actor.angle_to(other_actor)	Calcule l'angle en degrés vers un autre acteur.	angle = actor.angle_to(target)
actor.distance_to(other_actor)	Calcule la distance en pixels jusqu'à un autre acteur.	dist = actor.distance_to(target)
actor.move_in_direction(speed)	Fait avancer l'acteur dans sa direction à une vitesse donnée.	actor.move_in_direction(actor.speed)

Créer un nouvel Actor en définissant une nouvelle classe

Par convention, on donne toujours une majuscule au début du nom d'une classe pour les différencier.

```
# Création de la classe
class Coin(Actor):
    def __init__(self, image, pos, **kwargs):
        super().__init__(image, pos, **kwargs)
        # Attributs de l'acteur

    def update(self):
        # Comportement de l'acteur
```

Une fois la classe définie (comme une sorte de moule), il est possible de créer les objets concrets à partir de ce moule. Ici on crée notre objet en lui donnant une image et une position.

```
coin = Coin('coin', pos)
```

L'objet « coin » hérite des méthodes et attributs de la classe Actor donnés dans le tableau et peut tous les utiliser.

Fonctions Globales

Les fonctions globales permettent de gérer des aspects comme l'affichage à l'écran, la planification de tâches et la manipulation du son.

Fonction	Description	Exemple
screen.clear()	Efface l'écran.	screen.clear()
screen.fill(color)	Remplit l'écran avec une couleur.	screen.fill('blue')
screen.blit(image, pos)	Dessine une image sur l'écran à la position spécifiée.	screen.blit('background', (0, 0))
screen.draw.text(text, pos)	Dessine un texte à l'endroit indiqué.	screen.draw.text('Vie: 100', (0, 0))
clock.schedule(callback, delay)	Planifie une fonction à exécuter après un délai en secondes.	clock.schedule(game_over, 10.0)
clock.schedule_interval(callback, delay)	Planifie une fonction à exécuter toutes les x secondes.	clock.schedule_interval(add_ennemy, 20.0)
sounds.name.play()	Joue un effet sonore spécifié.	sounds.explosion.play()
music.play(name)	Joue une musique en continu.	music.play('adventure')
music.queue(name)	Ajoute une musique à une playlist.	music.queue('adventure')
remove_actors(list_actors)	Supprime du jeu tous les acteurs de la liste dont l'attribut to_remove est True.	remove_actors(list_coins)

Gestion des Événements

Pygame Zero simplifie la gestion des événements comme les entrées clavier et souris pour faciliter le développement de jeux.

Fonction	Description	Exemple

def update():	Fonction appelée à chaque frame pour mettre à jour l'état du jeu.	def update():
		  if keyboard.left:
		  actor.x -= 2

def draw():	Fonction appelée pour redessiner l'écran.	def draw():
		  screen.clear()
		  actor.draw()

def on_mouse_down(pos, button):	Appelée lorsqu'un bouton de la souris est cliqué.	def on_mouse_down(pos, button):
		  if button == mouse.LEFT:
		  check_click(pos)

keyboard.key	Retourne True quand la touche est enfoncée.	if keyboard.a : actor.x += 1

Documentation complete

Ce document est une version simplifiée qui donne les éléments les plus couramment utilisés.

Vous pouvez trouver la documentation officielle et des exemples ici : https://pygame-zero.readthedocs.io

Ou cherchez simplement « Pygame Zero » dans un navigateur.