pygame

Par Dany Chea [1]

Introduction ¶

Pygame est un module qui offre des outils permettant de créer des jeux. Le module est lui-même subdivisé en plusieurs sous-modules, ce qui permet de ne pas appeler des modules qui seraient inutiles. On se contentera ici de présenter les bases de l'utilisation de Pygame, ainsi que le fonctionnement de certains sous-modules.

Liste des sous-modules de Pygame:

pygame.camera	*pygame.cdrom*	*pygame.cursors*	*pygame.display*
pygame.draw	*pygame.event*	*pygame.examples*	*pygame.font*
pygame.freetype	*pygame.gfxdraw*	*pygame.image*	*pygame.joystick*
pygame.key	*pygame.locals*	*pygame.mask*	*pygame.math*
pygame.midi	*pygame.mixer*	*pygame.mouse*	*pygame.pixelcopy*
pygame.scrap	*pygame.sndarray*	*pygame.sprite*	*pygame.surfarray*
pygame.test	*pygame.time*	*pygame.transform*	*pygame.version*

Indépendemment du jeu que l'on veut créer, sa réalisation passe nécessairement par cinq étapes:

1. Initialisation de pygame
2. Appel des modules nécessaires
3. L'affichage
4. Boucle infinie
5. Fermeture du programme

Initialisation

Afin de pouvoir utiliser le module Pygame, il va falloir d'abord l'importer, puis l'initier de la manière suivante:

```
>>> import pygame
>>> pygame.init()
```

pygame.locals

pygame contient une quantité de constantes prédéfinies, qui sont chargées dans le namespace pygame, lors de l'importation de celui-ci. Ainsi, lorsque l'on veut utiliser une des constantes, il faut à chaque fois l'appeler en préfixant pygame. suivi du nom de la constante. Afin d'éviter ce préfixe, on peut faire une importation des constantes depuis **`pygame.locals`**, comme ci-dessous:

```
>>> from pygame.locals import *
```

Appel des fonctions utiles

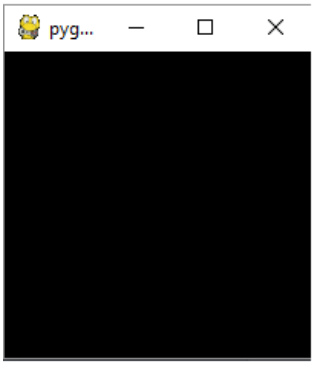
C'est dans cette partie que l'on va initialiser la fenêtre, ainsi que tout les éléments dont on aura besoin.

par exemple, pour afficher une fenêtre:

```
>>> fenetre = pygame.display.set_mode((200, 200))
```

Dans cet exemple, on appelle la fonction **`set_mode()`** du module display. Ce que l'on obtient en retour, c'est un objet de la classe Surface qui est défini par Pygame.

Résultat:



La boucle infinie

Si on se contentait d'exécuter le code ci-dessus, on n'obtiendrait le résultat que pendant une fraction de seconde, car au moment où le code a fini de s'exécuter, il se termine. Il faut donc créer une boucle infinie, de laquelle on peut sortir au moyen d'une action de l'utilisateur.

Exemple de fenêtre simple

Cet exemple provient de 2. Revision: Pygame fundamentals.

```
"""Exemple d'affichage d'une fenêtre simple."""
import pygame
from pygame.locals import QUIT

# Initialise screen
pygame.init()
screen = pygame.display.set_mode((600, 480))
pygame.display.set_caption('Basic Pygame program')

# Fill background
background = pygame.Surface(screen.get_size())
background = background.convert()
background.fill((250, 250, 250))

# Display some text
font = pygame.font.Font(None, 36)
text = font.render("Hello There", 1, (10, 10, 10))
textpos = text.get_rect()
textpos.centerx = background.get_rect().centerx
background.blit(text, textpos)

# Blit everything to the screen
screen.blit(background, (0, 0))
pygame.display.flip()

# Event loop
continuer = 1
while continuer:
    for event in pygame.event.get():
        if event.type == QUIT:
            continuer = 0
    screen.blit(background, (0, 0))
    pygame.display.flip()
```

Exemples

Son au maintien d'une touche

Cet exemple provient du tutoriel OpenClassrooms

```
"""Exemple d'utilisation des modules pygame.event et pygame.mixer."""
import pygame
from pygame.locals import K_RETURN, K_SPACE, KEYDOWN, KEYUP, QUIT, RESIZABLE

pygame.init()
fenetre = pygame.display.set_mode((200, 200), RESIZABLE)
son = pygame.mixer.Sound("./exemple/Tinquen.wav")

continuer = True
joue = False
while continuer:
    for event in pygame.event.get():
        if event.type == QUIT:
            continuer = False
        elif event.type == KEYDOWN:
            if event.key == K_SPACE:
                if not joue:
                    son.play()
                    joue = True
                else:
                    pygame.mixer.unpause()
            elif event.key == K_RETURN:
                son.stop()
                joue = False
        elif event.type == KEYUP:
            if event.key == K_SPACE:
                pygame.mixer.pause()

pygame.quit()
```

Dans cet exemple, on a utilisé deux modules: **`pygame.mixer`**, qui est utilisé pour la gestion de sons et **`pygame.event`**, qui est utilisé pour la gestion des touches au clavier.

Lors de l'appui, et du maintient de la touche espace, on joue le son qui a été chargé, ou on continue la lecture si le son était sur pause. Lorsque la touche est relâchée, le son est mis sur pause.

Affichage simple d'images

```
"""Exemple d'affichage d'une image de fond."""
import pygame
from pygame.locals import QUIT, RESIZABLE

pygame.init()
fenetre = pygame.display.set_mode((640, 480), RESIZABLE)

fond = pygame.image.load("./exemple/background.jpg").convert()
continuer = 1
while continuer:
    for event in pygame.event.get():
        if event.type == QUIT:
            continuer = 0
    fenetre.blit(fond, (0, 0))

    pygame.display.flip()
pygame.quit()
```

Dans cet exemple, on commence par charger une image. Cette image n'étant pas forcément dans un format supporté par pygame, la fonction pygame.convert est utilisée afin de s'assurer que l'image pourra s'afficher correctement.

```
>>> fond = pygame.image.load("background.jpg").convert()
```

Enfin, on peut afficher l'image en la "collant" sur la fenêtre de base, avec la fonction blit().

```
>>> fenetre.blit(fond,(0,0))
```

Il est a noter que la fonction blit prend en paramètre une Surface, donc l'objet à afficher, ainsi que la position, en sachant que la coordonnée (0,0) est en haut à gauche de la fenêtre.

Conclusion

pygame est un module très complet, qui nous permet de réaliser des jeux avec une certaine rapidité. Les quelques méthodes présentées ici ne sont qu'une infime partie des méthodes que propose pygame. Afin d'en avoir une liste exhaustive, il est recommandé d'aller consulter la documentation officielle de **pygame**,

[1] <dany.chea@he-arc.ch>