

Benoît PRIEUR



+ QUIZ

Version en ligne
OFFERTE !
pendant 1 an

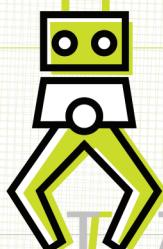
Pygame

Initiez-vous au développement
de jeux vidéo en Python

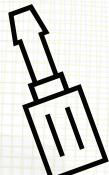
En téléchargement



Le code source
des exemples du livre



LA FABRIQUE



Les éléments à télécharger sont disponibles à l'adresse suivante :

<http://www.editions-eni.fr>

Saisissez la référence ENI de l'ouvrage **LFPYG** dans la zone de recherche et validez. Cliquez sur le titre du livre puis sur le bouton de téléchargement.

Chapitre 1 Présentation et bases du langage Python

1. Introduction	15
2. Présentation de Python	17
2.1 Le langage.....	17
2.2 Installation de Python	18
2.2.1 Installation de Python sur Windows	18
2.2.2 Installation de Python sur Mac OS	18
2.2.3 Installation de Python sur Linux	18
3. Les bases de Python	20
3.1 L'interpréteur de commande	20
3.2 Les variables	20
3.3 De l'interpréteur de commande à l'usage d'un fichier	21
3.4 Installation des modules avec pip.....	21
3.5 Les chaînes de caractères	22
3.5.1 La fonction print.....	22
3.5.2 Une chaîne de caractères est comme une collection	23
3.5.3 La taille d'une chaîne de caractères.....	23
3.5.4 Premières manipulations de chaînes de caractères.....	23
3.6 Les listes	24
3.7 Les conditions.....	25
3.8 Les boucles.....	26
3.8.1 La boucle for	26
3.8.2 La boucle while.....	26
3.8.3 Le mot-clé break.....	27
3.9 L'environnement virtuel en bref	27
3.10 Développement d'un petit jeu en ligne de commande.....	27

2 Pygame - Initiez-vous au développement de jeux vidéo en Python

Chapitre 2

Premiers pas avec Pygame

1.	La boucle de jeu	31
2.	Présentation de Pygame.....	32
3.	Installation de Pygame.....	32
4.	Les modules composant Pygame	33
5.	Réalisation d'un premier jeu graphique : Fusée et planètes.....	34
5.1	Les images utilisées	35
5.2	La fenêtre du jeu	35
5.3	La boucle du jeu	36
5.4	Le système de coordonnées Pygame	37
5.5	Les variables du jeu	38
5.5.1	Les variables liées à la fusée.....	38
5.5.2	Les variables liées aux deux planètes qui « tombent »	39
5.5.3	Les variables relatives au comptage des points	39
5.5.4	Les variables relatives aux images.....	39
5.6	Les déplacements de la fusée	41
5.7	Les déplacements des planètes	42
5.8	Les collisions	43
5.9	Le code complet.....	44

Chapitre 3

La structure d'un jeu Pygame

1.	Introduction	47
2.	Initialisation.....	47
3.	Aide en ligne de commande.....	48
4.	Affichage de la fenêtre.....	48
4.1	Le tuple size de set_mode	49
4.2	Le paramètre flags de set_mode.....	49
5.	Rappels concernant la boucle de jeu	50

6. Les surfaces Pygame	50
6.1 Définition d'une surface	50
6.2 La fenêtre de jeu, une surface particulière.....	51
6.3 Exemple de manipulation d'une surface	51
6.4 Manipulation de la surface.....	52
6.5 Surface ou copie de surface ?.....	52
6.6 Coloration de la surface	53
7. Gestion des couleurs.....	53
8. Système de coordonnées	54
9. Gestion du temps et des évènements	57
9.1 Gestion du temps dans Pygame	57
9.2 Gestion des évènements dans Pygame	58
9.2.1 La fonction pygame.event.get	58
9.2.2 La fonction pygame.event.wait	59
9.2.3 La fonction pygame.event.poll.....	59
9.3 Un exemple : le carré qui rebondit	59

Chapitre 4

Le dessin et le graphisme dans tous ses états avec Pygame

1. Introduction	63
2. Dessiner des formes avec Pygame	64
2.1 Le module pygame.draw	64
2.2 Dessiner une ligne.....	64
2.3 Dessiner une ligne brisée	65
2.4 Dessiner un rectangle	66
2.5 Dessiner un polygone.....	66
2.6 Dessiner un cercle.....	67
2.7 Dessiner une ellipse	67
2.8 Dessiner un arc de cercle.....	68
2.9 L'anti-aliasing	71

4 Pygame - Initiez-vous au développement de jeux vidéo en Python

3. Afficher et sauvegarder des images avec Pygame72
3.1 Le module pygame.image72
3.2 Charger des images avec Pygame73
3.2.1 La fonction pygame.draw.load73
3.2.2 La bonne pratique de l'appel de convert()73
3.2.3 Exemple de chargement et d'affichage d'une image74
3.3 Sauver des images avec Pygame75
4. Utiliser et manipuler du texte dans Pygame77
4.1 Le module pygame.font77
4.2 Utiliser des polices de caractères système avec Pygame77
4.3 Utiliser ses propres polices de caractères avec Pygame79
5. Concevoir un logiciel de dessin avec Pygame79
5.1 Gérer les événements de la souris ou du clavier en Pygame79
5.2 Première version du logiciel : afficher le tracé réalisé avec la souris81
5.3 Seconde version du logiciel : améliorations diverses82
5.3.1 Amélioration de l'expérience du tracé82
5.3.2 Ajout de fonctionnalités (couleur, épaisseur, etc.)83
6. Appliquer des transformations géométriques dans Pygame85
6.1 Le module pygame.transform85
6.2 Exemple d'utilisation des transformations Pygame86

Chapitre 5 L'ajout de sons dans un jeu Pygame

1. Introduction91
2. La gestion du son avec Pygame91
2.1 Les modules pygame.mixer et pygame.mixer.music92
2.1.1 Le module pygame.mixer.music (fond sonore)92
2.1.2 Le module pygame.mixer (effets sonores)92
2.2 Les fichiers son92
2.3 La notion de channel (canal) dans Pygame93
3. Exemple d'utilisation du son avec Pygame94

Chapitre 6**Les sprites avec Pygame**

1. La notion de sprite dans Pygame	97
2. La notion de group dans Pygame	98
3. Une gestion des collisions simplifiée	98
4. Quelques explications sur la programmation orientée objet	99
4.1 Le paradigme objet, les grandes lignes	99
4.2 L'héritage	100
4.3 Des mots-clés fondamentaux en Python	101
4.3.1 Le mot-clé self	101
4.3.2 Le mot-clé class	101
4.3.3 Le mot-clé def	102
4.3.4 __init__	102
4.4 L'exemple de la classe Voiture en Python	102
4.5 Ce que l'on savait déjà... sans le savoir	104
5. Le module sprite et son utilisation	104
5.1 Le contenu du module sprite	104
5.2 Création d'un sprite	105
5.2.1 Premier exemple d'utilisation	105
5.2.2 Retour sur le premier exemple	109
5.3 Gros plan sur l'attribut rect de la classe Sprite	109
5.4 La liste de sprites (group)	110
5.5 Gestion des collisions grâce aux sprites	111

Chapitre 7**Plus loin avec le module sprite, exemples appliqués**

1. Introduction	113
2. Le jeu du serpent (snake)	114
2.1 Le contexte	114
2.2 Les images utilisées	114
2.3 Les effets sonores utilisés	115
2.4 Le programme global	115
2.5 Les listes de sprites (group)	119

6 Pygame - Initiez-vous au développement de jeux vidéo en Python

2.6 Les variables globales	119
2.7 Les classes	120
2.7.1 La classe SERPENT	120
2.7.2 La classe CORPS	123
2.7.3 La classe NOURRITURE	124
2.8 Le programme lui-même	124
2.8.1 La fonction AFFICHER_SCORE	124
2.8.2 Le corps du programme	125
3. Le jeu du labyrinthe	127
3.1 Le contexte	127
3.2 Conception du labyrinthe	127
3.3 Les images utilisées	128
3.4 Le programme global	129
3.5 Les listes de sprites (group)	133
3.6 Les classes	133
3.6.1 La classe MUR	133
3.6.2 La classe OBJET	133
3.6.3 La classe Chrono	134
3.6.4 La classe PERSONNAGE	134
3.7 Le programme lui-même	135
4. Le jeu de casse-briques	136
4.1 Le contexte	136
4.2 Les images utilisées	137
4.3 Le programme global	137
4.4 Les listes de sprites (group)	140
4.5 Les constantes	141
4.6 Les classes	141
4.6.1 La classe OBJET	142
4.6.2 La classe RAQUETTE	143
4.6.3 La classe BRIQUE	144
4.6.4 La classe BALLE	144
4.7 Le programme lui-même	145
5. Le jeu de défilement : Fusée et planètes (version 2)	148
5.1 Le contexte	148
5.2 Le programme global	149
5.3 Les listes de sprites (group)	151

5.4 Les classes	151
5.4.1 La classe FUSEE.....	151
5.4.2 La classe PLANETE.....	152
5.5 Le programme lui-même.....	153

Chapitre 8

Introduction à la 3D et à la notion de moteur de jeu

1. Travailler en 3D avec Pygame.....	155
1.1 La bibliothèque 3D OpenGL.....	155
1.2 OpenGL en Python/Pygame.....	156
1.2.1 PyOpenGL.....	156
1.2.2 Les notions fondamentales : sommet et arête	156
1.2.3 PyOpenGL et Pygame	157
1.3 PyOpenGL/Pygame : l'exemple du cube	157
1.3.1 Le code global	157
1.3.2 Explication détaillée du code	159
1.4 Plus loin avec PyOpenGL/Pygame : l'exemple du cube [suite].....	162
1.4.1 Le code global	162
1.4.2 Explication détaillée du code	164
2. La notion de moteur de jeu vidéo	165
2.1 Définition.....	165
2.2 Créer son propre moteur de jeu ?.....	166
2.3 Une ébauche de moteur de jeu	166

Chapitre 9

Les principaux modules Pygame

1. Introduction	169
2. L'objet Color	170
2.1 La classe Color	170
2.2 Les constructeurs de Color	171
2.3 Les principales fonctions de la classe Color	171

8 Pygame - Initiez-vous au développement de jeux vidéo en Python

2.4 Les fonctions associées aux autres représentations de la couleur	172
2.4.1 La représentation CMY	172
2.4.2 La représentation HSV et HSL	172
2.4.3 La représentation I1I2I3	172
2.5 Les autres fonctions	172
2.5.1 La fonction normalize	172
2.5.2 La fonction correct_gamma	173
2.5.3 La fonction set_length	173
3. Le module time	173
3.1 La fonction get_ticks	173
3.2 La fonction wait	174
3.3 La fonction delay	174
3.4 La fonction set_timer	174
3.5 L'objet Clock	175
3.5.1 Création d'une instance	175
3.5.2 La fonction tick	175
3.5.3 La fonction get_time	176
3.5.4 La fonction get_fps	176
4. Le module event	177
4.1 La fonction pump	177
4.2 La fonction get	178
4.3 La fonction poll	178
4.4 La fonction wait	178
4.5 La fonction peek	179
4.6 La fonction clear	179
4.7 La fonction event_name	179
4.8 La fonction set_blocked	180
4.9 La fonction set_allowed	180
5. Le module display	180
5.1 La fonction init	180
5.2 La fonction quit	181
5.3 La fonction get_init	181
5.4 La fonction set_mode	181
5.5 La fonction flip	182
5.6 La fonction update	182
5.7 La fonction set_icon	182

5.8 La fonction <code>set_caption</code>	182
6. L'objet <code>Surface</code>	183
6.1 Le constructeur de <code>Surface</code>	183
6.2 La fonction <code>blit</code>	183
6.3 La fonction <code>blits</code>	184
6.4 Les fonctions <code>convert</code> et <code>convert_alpha</code>	185
6.5 La fonction <code>copy</code>	185
6.6 La fonction <code>fill</code>	186
6.7 La fonction <code>scroll</code>	186
6.8 La fonction <code>set_colorkey</code>	187
6.9 La fonction <code>get_colorkey</code>	187
7. Le module <code>draw</code>	188
7.1 La fonction <code>rect</code>	188
7.2 La fonction <code>polygon</code>	188
7.3 La fonction <code>circle</code>	188
7.4 La fonction <code>ellipse</code>	189
7.5 La fonction <code>arc</code>	189
7.6 La fonction <code>line</code>	190
7.7 La fonction <code>lines</code>	190
7.8 Les fonctions <code>aaline</code> et <code>aalines</code>	190
8. Le module <code>image</code>	191
8.1 La fonction <code>load</code>	191
8.2 La fonction <code>save</code>	191
8.3 Les fonctions <code>tostring</code> , <code>fromstring</code> , <code>frombuffer</code>	192
9. Le module <code>font</code>	192
9.1 La fonction <code>init</code>	192
9.2 La fonction <code>quit</code>	192
9.3 La fonction <code>get_init</code>	192
9.4 La fonction <code>get_default_font</code>	193
9.5 La fonction <code>get_fonts</code>	193
9.6 La fonction <code>match_font</code>	193
9.7 La fonction <code>SysFont</code>	194
9.8 L'objet <code>Font</code>	194
9.8.1 La fonction <code>Font</code>	194
9.8.2 La fonction <code>render</code>	194
9.8.3 La fonction <code>size</code>	195

10 Pygame - Initiez-vous au développement de jeux vidéo en Python

9.8.4 La fonction set_underline.....	195
9.8.5 La fonction get_underline	195
9.8.6 La fonction set_bold	195
9.8.7 La fonction get_bold.....	195
9.8.8 La fonction set_italic.....	196
9.8.9 La fonction get_italic	196
9.8.10 La fonction metrics.....	196
10. Le module mouse.....	196
10.1 La fonction get_pressed	196
10.2 La fonction get_pos.....	197
10.3 La fonction get_rel.....	197
10.4 La fonction set_pos.....	198
10.5 La fonction set_visible	198
10.6 La fonction get_visible.....	198
10.7 La fonction get_focused	198
10.8 La fonction set_cursor.....	198
10.9 La fonction get_cursor.....	198
11. Le module key	199
11.1 Les constantes correspondant aux touches du clavier	199
11.2 La fonction get_focused	201
11.3 La fonction pressed	201
11.4 La fonction set_repeat.....	201
11.5 La fonction get_repeat.....	202
11.6 La fonction name.....	202
12. Le module transform	202
12.1 La fonction flip	202
12.2 La fonction scale	202
12.3 La fonction rotate	203
12.4 La fonction rotozoom	203
12.5 La fonction scale2x.....	203
12.6 La fonction chop	203
12.7 La fonction laplacian	204
12.8 La fonction average_surfaces.....	205
12.9 La fonction average_color	205

13. Le module mixer.....	205
13.1 La fonction init	205
13.2 La fonction quit.....	206
13.3 La fonction get_init	206
13.4 L'objet Sound.....	206
13.4.1 Les constructeurs de Sound.....	206
13.4.2 La fonction play	206
13.4.3 La fonction stop	207
13.4.4 La fonction fadeout	207
13.4.5 La fonction set_volume.....	207
13.4.6 La fonction get_volume.....	207
13.4.7 La fonction get_num_channels.....	207
13.4.8 La fonction get_length	208
13.4.9 La fonction get_raw	208
13.5 L'objet Channel.....	209
13.5.1 Le constructeur de Channel	209
13.5.2 La fonction queue	209
13.5.3 La fonction set_volume.....	209
13.5.4 La fonction get_volume.....	209
13.5.5 Les fonctions play, stop, pause etc.	210
13.6 Les fonctions get_num_channels, set_num_channels, find_channel.....	210
14. Le module music.....	211
14.1 La fonction load	211
14.2 La fonction unload	211
14.3 La fonction play	211
14.4 La fonction rewind	211
14.5 La fonction stop	212
14.6 La fonction pause	212
14.7 La fonction unpause.....	212
14.8 La fonction fadeout	212
14.9 La fonction set_volume.....	212
14.10 La fonction get_volume.....	212
14.11 La fonction set_pos	212
14.12 La fonction get_pos	213
14.13 La fonction queue	213

12 Pygame - Initiez-vous au développement de jeux vidéo en Python

15. Le module sprite	213
15.1 La classe Sprite	213
15.1.1 La fonction update	213
15.1.2 La fonction add	213
15.1.3 La fonction remove	213
15.1.4 La fonction kill	214
15.1.5 La fonction alive	214
15.1.6 La fonction groups	214
15.2 La classe Group	214
15.2.1 La fonction sprites	214
15.2.2 La fonction copy	214
15.2.3 La fonction add	214
15.2.4 La fonction remove	215
15.2.5 La fonction has	215
15.2.6 La fonction update	215
15.2.7 La fonction draw	215
15.2.8 La fonction clear	215
15.2.9 La fonction empty	215
15.3 Les principales fonctions du module	216
15.3.1 La fonction spritecollide	216
15.3.2 La fonction collide_rect	216
15.3.3 La fonction collide_circle	216
15.3.4 La fonction collide_mask	216
15.3.5 La fonction groupcollide	216
15.3.6 La fonction spritecollideany	217

Chapitre 10 Les modules secondaires Pygame

1. Introduction	219
2. Le module cursors	220
2.1 Les curseurs prédéfinis du module	220
2.2 La fonction compile	221
2.3 La fonction load_xbm	221

3. Le module joystick	222
3.1 La fonction init	222
3.2 La fonction quit	222
3.3 La fonction get_init	222
3.4 La fonction get_count	222
3.5 La classe Joystick	223
3.5.1 La fonction Joystick	223
3.5.2 La fonction init	223
3.5.3 La fonction quit	223
3.5.4 La fonction get_init	223
3.5.5 La fonction get_id	223
3.5.6 La fonction get_name	224
3.5.7 La fonction get_numaxes	224
3.5.8 La fonction get_axis	224
3.5.9 La fonction get_numballs	224
3.5.10 La fonction get_ball	224
3.5.11 La fonction get_numbuttons	225
3.5.12 La fonction get_button	225
3.5.13 La fonction get_numhats	225
3.5.14 La fonction get_hat	225
4. Le module touch	225
4.1 La fonction get_num_devices	225
4.2 La fonction get_device	226
4.3 La fonction get_num_fingers	226
4.4 La fonction get_finger	226
5. Le module math	226
5.1 La classe Vector2 - Création de vecteur	227
5.2 La classe Vector3 - Création de vecteur	227
5.3 Les principales fonctions de Vector2 et Vector3	228
5.3.1 La fonction dot	228
5.3.2 La fonction length	228
5.3.3 La fonction normalize	228
5.3.4 La fonction reflect	228
5.3.5 La fonction distance_to	228
5.3.6 La fonction rotate	229
5.3.7 La fonction rotate_rad	229

14 Pygame - Initiez-vous au développement de jeux vidéo en Python

6. Le module surfarray	229
6.1 La fonction array2d	229
6.2 La fonction pixels_red	229
6.3 La fonction pixels_green	229
6.4 La fonction pixels_blue	230
6.5 La fonction make_surface	230
6.6 La fonction blit_array	230
7. Le module camera	230
7.1 La fonction list_cameras	231
7.2 L'objet Camera	231
7.2.1 Instanciation de Camera	231
7.2.2 La fonction start	232
7.2.3 La fonction stop	232
7.2.4 La fonction get_image	232
7.2.5 La fonction get_raw	232
7.2.6 La fonction query_image	232
7.2.7 La fonction get_size	232
7.2.8 La fonction get_controls	232
7.2.9 La fonction set_controls	233
Index	235

Chapitre 5 ▲

L'ajout de sons dans un jeu Pygame

1. Introduction

Animer un jeu vidéo, c'est d'abord mettre en place le visuel et donner le mouvement à certains objets graphiques. Mais cela consiste également à ajouter du son au jeu, par exemple un fond sonore, une musique, pour ainsi améliorer l'expérience utilisateur. Il y a dans Pygame un module pour cela. Son étude et sa mise en œuvre constituent l'essentiel du présent chapitre.

Le module qui permet la gestion des sons en Pygame s'appelle *pygame.mixer*. Il contient deux grandes notions :

- Le sous-module *music* qui gère la musique de fond. Il n'y en a qu'une à la fois.
- L'objet *Sound* de *mixer* que l'on peut instancier plusieurs fois pour s'en servir par exemple pour les effets sonores du jeu.

2. La gestion du son avec Pygame

En premier lieu, il est nécessaire d'initialiser ce module grâce à la fonction *pygame.mixer.init*. Cet appel doit être fait explicitement. Le module *mixer* inclut un certain nombre d'outils relatifs aux effets sonores grâce à la classe *Sound*. Ce module inclut également une sorte de sous-module, *music*, dévolu à la gestion du fond sonore.

92 Pygame - Initiez-vous au développement de jeux vidéo en Python

2.1 Les modules pygame.mixer et pygame.mixer.music

Remarque

Une documentation des modules *mixer* et *music* de Pygame est disponible dans le chapitre *Les principaux modules Pygame*.

2.1.1 Le module pygame.mixer.music (fond sonore)

Ce module Pygame permet de gérer le fond sonore (unique). Ci-après les principales fonctions disponibles dans *music*.

- *pygame.mixer.music.load* permet de charger un fichier de fond sonore.
- *pygame.mixer.music.play* permet de jouer/lire le fond sonore.
- *pygame.mixer.music.rewind* permet de reprendre au début le fond sonore.
- *pygame.mixer.music.stop* permet d'arrêter la lecture du fond sonore.
- *pygame.mixer.music.pause* permet de faire une pause dans la lecture.
- *pygame.mixer.music.set_volume* permet de régler le volume du fond sonore.
- *pygame.mixer.music.get_volume* permet de connaître le volume courant du fond sonore.

2.1.2 Le module pygame.mixer (effets sonores)

Pour les effets sonores, il faut préalablement créer (instancier) un objet de type *Sound* avant d'accéder à un ensemble de fonctions comparables à celles du sous-module *music*.

pygame.mixer.Sound permet de créer un nouvel objet de type *Sound*.

Une fois créé l'objet de type *Sound*, on peut utiliser une de ses fonctions. Entre autres :

- *pygame.mixer.Sound.play* permet de jouer le son.
- *pygame.mixer.Sound.stop* permet de stopper la diffusion du son.

2.2 Les fichiers son

Voici un court rappel sur les différents formats et extensions de fichiers sonores qui sont utilisables avec Pygame.

Deux formats sont particulièrement recommandés :

- le format WAV (*Waveform Audio File Format*)
- le format ouvert et libre OGG

Ils sont recommandés, car leur utilisation ne pose aucun problème, quelle que soit la plateforme. Il n'en va pas de même du format de compression audio MP3 par exemple, qui selon la documentation officielle peut induire des soucis d'utilisation sous certaines plateformes, en particulier avec la distribution Linux Debian (*« On some systems an unsupported format can crash the program, e.g. Debian Linux. Consider using OGG instead. »*).

Comment obtenir ou afficher des fichiers son pour un fond sonore ou pour des effets sonores ? On peut par exemple obtenir de la musique publiée sous licence libre sur Internet. On peut également créer ses propres fichiers son. Par exemple en les enregistrant grâce à l'enregistreur d'un smartphone ou en utilisant le logiciel libre d'enregistrement Audacity, qui permet aussi de faire du montage audio et d'exporter des sons dans le format de son choix, le format OGG en particulier.

2.3 La notion de channel (canal) dans Pygame

Pygame offre une notion supplémentaire dans la gestion du son. Il s'agit de la notion de channel (canal, en français). Un jeu possède plusieurs canaux son. Ainsi, on peut affecter tel son au canal numéro 1 et tel autre son au canal numéro 2. Il est ainsi possible de jouer simultanément des sons en activant leurs lectures sur des canaux différents comme le suggère l'exemple théorique suivant.

```
pygame.mixer.set_num_channels(2)
sound_1 = pygame.mixer.Sound("son1.ogg")
sound_2 = pygame.mixer.Sound("son2.ogg")

canal_1 = pygame.mixer.Channel(0)
canal_2 = pygame.mixer.Channel(1)

canal_1.play(sound_1)
canal_2.play(sound_2)
```

Remarque

Cette notion est largement abordée dans les sections Le module mixer et music au chapitre Les principaux modules Pygame.

3. Exemple d'utilisation du son avec Pygame

Le but de ce petit exemple est d'utiliser les deux aspects présentés précédemment. Ainsi nous allons créer une fenêtre de jeu à laquelle on associe un fond sonore (*music*) ainsi que quelques effets sonores déclenchés par l'appui de boutons du clavier.

- Le fond sonore est un sifflement diffusé en boucle.
- L'appui sur la touche o diffuse un cocorico.
- L'appui sur la touche c diffuse un bruit de corneille.
- L'appui sur la touche v diffuse une sonnette de vélo.

En appuyant sur la touche [Flèche en haut], on augmente le volume de chacun des quatre sons. En appuyant sur la touche [Flèche en bas], on diminue ce même volume.

On commence nécessairement par initialiser le module *mixer*.

```
■ pygame.mixer.init()
```

Puis on crée le fond sonore d'après un fichier son OGG qui inclut un sifflement mélodique d'une durée de quelques dizaines de secondes.

```
■ SIFFLEMENT = pygame.mixer.music.load("sifflement.ogg")
```

Ensuite, on peut jouer ce fond sonore avec la fonction *play* qui prend deux paramètres :

- Le premier permet d'indiquer combien de fois on désire boucler sur le son (ici 10 fois).
- Le second, une valeur décimale, indique en secondes le moment du morceau qui constitue le début de la diffusion.

```
■ pygame.mixer.music.play(10, 0.0)
```

Nous définissons maintenant les trois effets sonores.

```
■ COQ = pygame.mixer.Sound("coq.ogg")
CORNEILLE = pygame.mixer.Sound("corneille.ogg")
VELO = pygame.mixer.Sound("vélo.ogg")
```

Ils sont déclenchés par l'appui d'une touche du clavier dédiée.

```
■ elif event.key == pygame.K_o:
    COQ.play()
elif event.key == pygame.K_c:
    CORNEILLE.play()
elif event.key == pygame.K_v:
    VEL0.play()
```

Enfin, on gère le volume en l'augmentant ou en le diminuant, en commençant par obtenir le volume courant auquel on ajoute ou on retranche 0.1. En effet, le volume s'exprime entre 0.0 et 1.0 ; par défaut, le volume est à 1.0.

```
elif event.key == pygame.K_DOWN:
    VOLUME = pygame.mixer.music.get_volume() - 0.1

    pygame.mixer.music.set_volume(VOLUME)
    COQ.set_volume(VOLUME)
    CORNEILLE.set_volume(VOLUME)
    VELO.set_volume(VOLUME)

elif event.key == pygame.K_UP:
    VOLUME = pygame.mixer.music.get_volume() + 0.1

    pygame.mixer.music.set_volume(VOLUME)
    COQ.set_volume(VOLUME)
    CORNEILLE.set_volume(VOLUME)
    VELO.set_volume(VOLUME)
```

Le code complet du programme est le suivant :

```
import pygame, sys

pygame.init()
pygame.mixer.init()

# COULEURS
COULEUR_BLANCHE = pygame.Color(255, 255, 255)

# FENETRE DE 400 SUR 400
ECRAN = pygame.display.set_mode((400,400))
ECRAN.fill(COULEUR_BLANCHE)
pygame.display.set_caption("Chapitre 5, du son")

SUITE = True

# FOND SONORE
SIFFLEMENT = pygame.mixer.music.load("sifflement.ogg")
pygame.mixer.music.play(1, 0.0)

# EFFETS SONORES
COQ = pygame.mixer.Sound("coq.ogg")
CORNEILLE = pygame.mixer.Sound("corneille.ogg")
VELO = pygame.mixer.Sound("vélo.ogg")

# BOUCLE DE JEU
while SUITE:

    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            SUITE = False
        elif event.type == pygame.KEYDOWN:
            if event.key == pygame.K_ESCAPE:
                SUITE = False
            elif event.key == pygame.K_o:
                COQ.play()
            elif event.key == pygame.K_c:
                CORNEILLE.play()
```

96 Pygame - Initiez-vous au développement de jeux vidéo en Python

```
elif event.key == pygame.K_v:  
    VEL0.play()  
elif event.key == pygame.K_DOWN:  
    VOLUME = pygame.mixer.music.get_volume() - 0.1  
    pygame.mixer.music.set_volume(VOLUME)  
    COQ.set_volume(VOLUME)  
    CORNEILLE.set_volume(VOLUME)  
    VEL0.set_volume(VOLUME)  
elif event.key == pygame.K_UP:  
    VOLUME = pygame.mixer.music.get_volume() + 0.1  
    pygame.mixer.music.set_volume(VOLUME)  
    COQ.set_volume(VOLUME)  
    CORNEILLE.set_volume(VOLUME)  
    VEL0.set_volume(VOLUME)  
  
pygame.display.flip()
```