

Coder un mini shooter



Start code

• main.py

```
import pygame, sys

def main():

    # Load
    pygame.init()
    screen = pygame.display.set_mode((800, 600))
    font = pygame.font.Font(None, 24)
    path = os.path.abspath('.') + '/'
    quit_game = False

    while not(quit_game):
        # Inputs
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                quit_game = True
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_ESCAPE:
                    quit_game = True

        # Update

        # Draw
        screen.fill((0, 0, 0))
        pygame.display.update()

if __name__ == "__main__":
    main()
```

Player

Display player

- main.py

```
import pygame, sys

# Load
...
joueur = pygame.image.load(path+'joueur.png').convert_alpha()
...
while not(quit_game):
    ...
    # Draw
    screen.fill((0, 0, 0))
    screen.blit(joueur, (0, 200))
    pygame.display.update()
```

- `pygame.image.load(...)` has one argument, which is the path to the image on your hard drive
- `path + "joueur.png"` concatenate the content of the path variable and "joueur.png"
- `convert_alpha()` function change the pixel format of the image, to give the display surface format

Move player

• main.py

```
import pygame, sys

# Load
...
key_up = False
key_down = False
joueur_x = 0
joueur_y = 200
...
while not(quit_game):
    # Inputs
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            quit_game = True

        if event.type == pygame.KEYDOWN:
            if event.key == pygame.K_ESCAPE:
                quit_game = True
            # Ajouter les touches qu'on appuie ici
            if event.key == pygame.K_UP:
                key_up = True
            if event.key == pygame.K_DOWN:
                key_down = True

        if event.type == pygame.KEYUP:
            # Ajouter les touches qu'on relache ici
            if event.key == pygame.K_UP:
                key_up = False
            if event.key == pygame.K_DOWN:
                key_down = False

    # Update
    if key_up:
        joueur_y = joueur_y - 5
    if key_down:
        joueur_y = joueur_y + 5

    # Draw
    ...
    screen.blit(joueur, (joueur_x, joueur_y))
    ...
```

Limit player move

- We want the player to stay at position 0 if position become inferior to 0
- Same if position go over screen height (- player height)

Limit player move

- main.py

```
# Load
```

```
joueur_y = 200
```

```
joueur_hauteur = 120
```

```
ecran_hauteur = 720
```

```
ecran_largeur = 1280
```

```
...
```

```
# Update
```

```
...
```

```
if joueur_y < 0:
```

```
    joueur_y = 0
```

```
if joueur_y > ekran_hauteur - joueur_hauteur:
```

```
    joueur_y = ekran_hauteur - joueur_hauteur
```

Shots

Make the player shoot

- Player will shot projectiles. We want a function to create shots, and an other to draw shots.
- Projectiles are fired when the player presses Space.
- We will store projectile's variables in a map :

```
tir = { 'x': 120, 'y': y, 'vitesse': 5, 'image': pygame.image.load(path+'tir.png').convert_alpha() }
```

- We should have a list of projectiles

Make the player shoot

• main.py

```
# Load
key_space = False
liste_tir = []

def creer_tir(y):
    tir = { 'x': 120, 'y': y, 'vitesse': 5, 'image': pygame.image.load(path+'tir.png').convert_alpha() }
    liste_tir.append(tir)

def dessiner_tir():
    for tir in liste_tir:
        screen.blit(tir['image'], (tir['x'], tir['y']))

...

# Inputs
if event.type == pygame.KEYDOWN:
    ...
    if event.key == pygame.K_SPACE:
        key_space = True

    if event.type == pygame.KEYUP:
        ...
        if event.key == pygame.K_SPACE:
            key_space = False

# Update
...
if key_space:
    creer_tir(joueur_y)

# Draw
...
dessiner_tir()

...
```

Make the shots move

- Now we want shots to move from the left to the right
- We create a function that would move each shot of the shot list

Make the shots move

- main.py

```
# Load
def deplacer_tirs():
    for tir in liste_tir:
        tir['x'] = tir['x'] + tir['vitesse']
```

```
# Update
```

```
...
```

```
deplacer_tirs()
```

```
...
```

One press one shot

- Our player shoots every frame when space bar is pressed
- We want the player to shoot only once when space bare is hit
- Use a boolean to implement this behaviour

Make the player shoot

• main.py

```
# Load
...
tir_emis = False
...
def creer_tir(y):
    nonlocal tir_emis
    tir = {'x': 120, 'y': y, 'vitesse': 5, 'image': pygame.image.load(path+'tir.png').convert_alpha()}
    liste_tir.append(tir)
    tir_emis = True
...

# Inputs
if event.type == pygame.KEYDOWN:
    ...
    if event.key == pygame.K_SPACE:
        key_space = True
        tir_emis = False

# Update
...
if key_space and not tir_emis:
    creer_tir(joueur_y + 50)
deplacer_tirs()
...
```

Erase shots to free memory

- If our code is left like that, we would manage a growing number of shots
- We want to erase shots when they get out the screen
- We cannot erase shots while we are going through them
- So we will register in a list the indexes of the shots we want to erase, then delete the shots with those indexes

Erase unused shots

• main.py

```
# Load
...
tirs_a_effacer = []
...
def deplacer_tirs():
    for index, tir in enumerate(liste_tir):
        tir['x'] = tir['x'] + tir['vitesse']
        if tir['x'] > ecran_largeur:
            tirs_a_effacer.append(index)

def effacer_tirs(tirs_a_effacer):
    for index in tirs_a_effacer:
        del liste_tir[index]
    tirs_a_effacer[:] = []

...

    # Update
    ...
    effacer_tirs(tirs_a_effacer)

...
```


Enemies

Create, draw, move and erase enemies

- You can use the shot algorithm to create enemies
- Use a counter variable to trigger enemy spawn: increment it by 1 each frame, when it exceeds 500, create an enemy
- Enemies go from right to left

Create, draw, move and erase enemies

• main.py

```
# Load
...
liste_ennemis = []
ennemis_a_effacer = []

def creer_ennemis(y):
    ennemi = {'x': ecran_largeur, 'y': y, 'vitesse': -3, 'image': pygame.image.load(path+'ennemi.png').convert_alpha()}
    liste_ennemis.append(ennemi)

def dessiner_ennemis():
    for ennemi in liste_ennemis:
        screen.blit(ennemi['image'], (ennemi['x'], ennemi['y']))

def deplacer_ennemis():
    for index, ennemi in enumerate(liste_ennemis):
        ennemi['x'] = ennemi['x'] + ennemi['vitesse']
        if ennemi['x'] < 0:
            ennemis_a_effacer.append(index)

def effacer_ennemis(ennemis_a_effacer):
    for index in ennemis_a_effacer:
        del liste_ennemis[index]
    ennemis_a_effacer[:] = []

compteur_ennemi = 0

...

# Update
...
# Ennemis
deplacer_ennemis()
effacer_ennemis(ennemis_a_effacer)
compteur_ennemi = compteur_ennemi + 1
if compteur_ennemi > 500:
    creer_ennemis(300)
    compteur_ennemi = 0

...
```

Create enemy at random position

- Import random package and use random.seed() to generate a random series of number
- random.randint(min, max) give a random integer between min and max
- Make enemies appear at a random position

Create, draw, move and erase enemies

• main.py

```
import random

...
# Load
...
random.seed()

...
# Update
...
if compteur_ennemi > 500:
    creer_ennemis(random.randint(0, ecran_hauteur - 120))
...
```

Detect player colliding with enemy

- Here is a code to detect collisions and trigger game over when there is a collision:

```
game_over = False

def collision_joueur_ennemis():
    nonlocal game_over
    for i_ennemi, ennemi in enumerate(liste_ennemis):
        x1, y1, w1, h1 = joueur_x, joueur_y, joueur_hauteur, joueur_hauteur
        x2, y2, w2, h2 = ennemi['x'], ennemi['y'], ennemi['image'].get_width(), ennemi['image'].get_height()
        if(not(x1 + w1 < x2 or x2 + w2 < x1 or y1 + h1 < y2 or y2 + h2 < y1)):
            detruire_ennemi(i_ennemi)
            game_over = True
```

- Display a game over text when game is over

Display game over

- main.py

```
import random

...

# Update
...
if not game_over:
    ...
    collision_joueur_ennemis()
else:
    if key_space:
        game_over = False

# Draw
if not game_over:
    ...
else:
    screen.blit(game_over_text, (600, 300))
```

Fixes

- We don't want the player to shoot when he or she restarts game
- We want all enemies to be erased after game restart

Fixes

• main.py

```
import random

# Load
def effacer_tous_ennemis():
    liste_ennemis[:] = []
    ennemis_a_effacer[:] = []

def collision_joueur_ennemis():
    ...
    effacer_tous_ennemis()
...
# Update
...
if not game_over:
    ...
else:
    if key_space:
        game_over = False
        tir_emis = True
```