Empêcher le joueur de sortir de l'écran avec pygame

Lorsque tu déplaces un personnage à l'écran, il peut sortir des limites si aucune précaution n'est prise.

Objectif

À la fin de ce cours, tu seras capable de :

- **Limiter les déplacements** d'un personnage pour qu'il reste visible à l'écran dans un jeu pygame.
- Comprendre comment utiliser la **taille du personnage** pour gérer les limites de déplacement.

Situation initiale et problème à résoudre

Lorsque tu déplaces un personnage à l'écran, il peut sortir des limites si aucune précaution n'est prise. Dans ton code actuel, rien n'empêche le joueur d'aller au-delà des bords de l'écran. Nous allons voir comment fixer ce problème simplement et efficacement.

Points essentiels à maîtriser

1. Pourquoi limiter le déplacement ?

Si le personnage sort des limites, l'expérience du joueur est affectée. Le joueur peut se perdre ou ne plus être capable de revenir à l'écran. Limiter les déplacements permet une meilleure gestion du jeu et améliore l'expérience utilisateur.

2. Ce qu'il faut prendre en compte :

Pour empêcher efficacement ton personnage de sortir de l'écran, tu dois connaître :

- **La largeur et la hauteur de l'écran**

```
largeur_ecran = 800
hauteur_ecran = 600
```

- **La largeur et la hauteur du personnage** (obtenues à partir de l'image chargée) :

```
largeur_joueur = player.get_width()
hauteur_joueur = player.get_height()
```

3. Vérifier les limites de déplacement

Tu dois ajouter une vérification après chaque déplacement pour t'assurer que le personnage reste toujours à l'intérieur de l'écran :

```python
# Limite à gauche
if player_x < 0:
    player_x = 0

# Limite à droite
if player_x > largeur_ecran - largeur_joueur:
    player_x = largeur_ecran - largeur_joueur

# Limite en haut
if player_y < 0:
    player_y = 0

# Limite en bas
if player_y > hauteur_ecran - hauteur_joueur:
    player_y = hauteur_ecran - hauteur_joueur
```

On peut aussi effectuer des calculs différemment:

```python
# Limite à droite
if player_x + largeur_joueur > largeur_ecran:
    player_x = largeur_ecran - largeur_joueur

# Limite en bas
if player_y + hauteur_joueur > hauteur_ecran:
    player_y = hauteur_ecran - hauteur_joueur

```

4. Intégration complète dans ton code

Voici comment intégrer concrètement cette logique dans ton code existant :

```python
import pygame
import sys

# Initialisation
pygame.init()
largeur_ecran = 800
hauteur_ecran = 600

# Création de la fenêtre de jeu
screen = pygame.display.set_mode((largeur_ecran, hauteur_ecran))
clock = pygame.time.Clock()

# Position initiale du joueur
player_x = 100
player_y = 100
vitesse = 5

# Chargement des images
player = pygame.image.load('player.png')
background_tile = pygame.image.load('sol_paves.png')

# Dimensions du joueur
largeur_joueur = player.get_width()
```

```
hauteur_joueur = player.get_height()

# Boucle de jeu
running = True
while running:
    for event in pygame.event.get():
        if event.type == pygame.QUIT:
            running = False

    keys = pygame.key.get_pressed()

    if keys[pygame.K_LEFT] or keys[pygame.K_q]:
        player_x -= vitesse
    if keys[pygame.K_RIGHT] or keys[pygame.K_d]:
        player_x += vitesse
    if keys[pygame.K_UP] or keys[pygame.K_z]:
        player_y -= vitesse
    if keys[pygame.K_DOWN] or keys[pygame.K_s]:
        player_y += vitesse

    # Limiter les déplacements du joueur à l'écran
    player_x = max(0, min(player_x, largeur_ecran - largeur_joueur))
    player_y = max(0, min(player_y, hauteur_ecran - hauteur_joueur))

    # Mise à jour de l'écran
    for x in range(0, largeur_ecran, background_tile.get_width()):
        for y in range(0, hauteur_ecran, background_tile.get_height()):
            screen.blit(background_tile, (x, y))

    screen.blit(player, (player_x, player_y))
    pygame.display.flip()

    clock.tick(60)

pygame.quit()
sys.exit()
```

5. Bonnes pratiques à retenir

- Toujours vérifier la position du joueur après son déplacement.
- Utiliser des fonctions simples (`max` et `min`) pour clarifier ton code.
- Adapter ces limites en fonction de la taille réelle du personnage, sinon celui-ci disparaîtra partiellement ou totalement hors écran.

Synthèse à retenir :

✅ Prends en compte les dimensions du personnage pour établir des limites précises.
✅ Vérifie la position à chaque déplacement.
✅ Utilise des fonctions comme `max()` et `min()` pour simplifier ton code.

Grâce à ces notions simples, tu peux garantir que ton personnage ne quittera jamais l'écran pendant une partie !