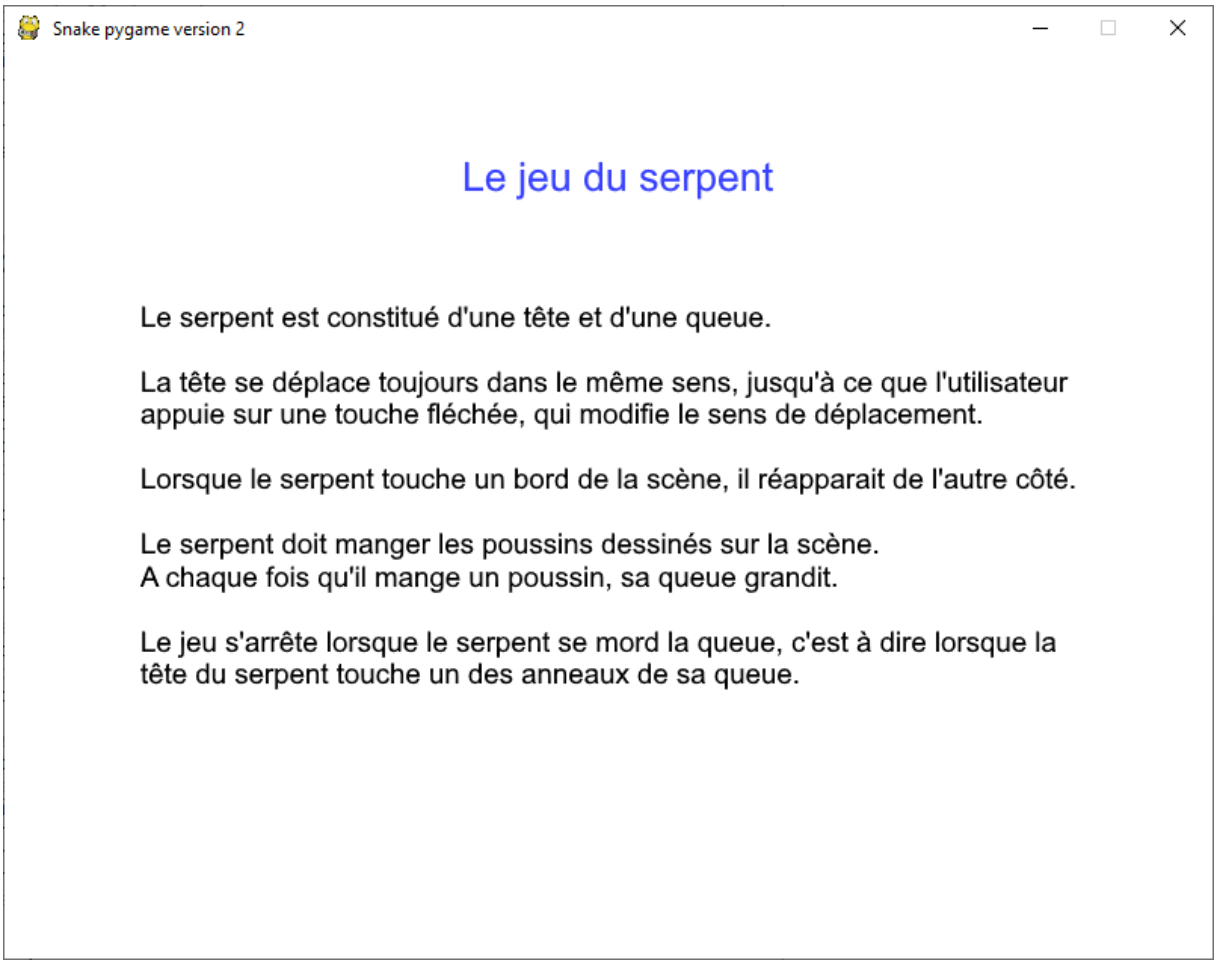
Pygame Snake

Complément du jeu Serpent

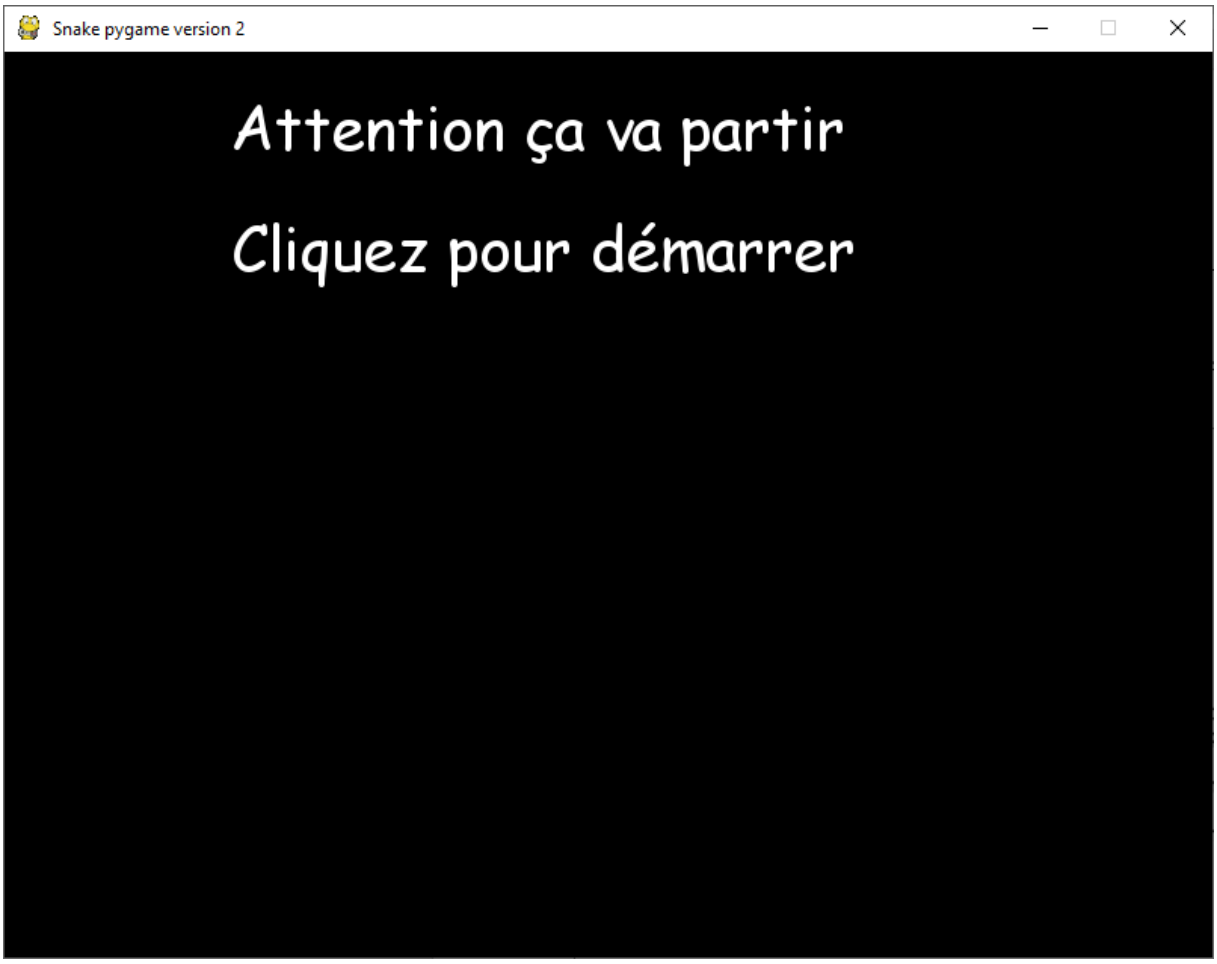
Nous allons ajouter au démarrage du jeu, des écrans affichant des instructions, et en fin de partie un écran affichant le score.

En début de partie, le programme affiche cette image, créée avec paint.net.

Elle est au dimensions de la fenêtre du jeu.



Un clic de souris fait passer au deuxième écran :



Un nouveau clic de souris démarre le jeu.

Lorsque le serpent se mord la queue, au lieu de fermer tout de suite la fenêtre, le programme affiche le score obtenu par le joueur

Analyse des ajouts et modifications à apporter au programme

Nous ajoutons en premier lieu 2 nouvelles variables

score = 0

game_over = False

Le score est incémenté de 1, à chaque poussin mangé :

```
    #La tête est t'elle entré en collision avec un poussin
      collision_liste = pygame.sprite.spritecollide(tete,lesPoussins,False)
      #Tester la liste des collisions
      for poussin in collision_liste:
         #réinitialiser les positions des poussins touchés
         poussin.reset()
```

```
            #créer de nouveaux anneaux : ici 1 seul
            #queue[-1] donne la référence du dernier anneau de la liste
            creerAnneau(2,queue[-1].rect.x,queue[-1].rect.y)
            score+=1
```

Et game_over est mis à True, lorsque le serpent s'est mordu la queue :

```
        #La tête est-elle entrée en collision avec un anneau.
        collision_liste = pygame.sprite.spritecollide(tete,lesAnneaux,False)
        if len(collision_liste)> 0:
            game_over = True #en remplacement de arret = True
```

Nouvelle structure de la boucle principale :

```
#----Boucle principale ---------
while not arret:
    #boucle des évènements
    for event in pygame.event.get():
        #Rien ne change ici.


    if not game_over:
        #On place ici toute la logique du jeu
        #logique du jeu
        #Rien ne change dans la logique du jeu en dehors de l'incrémentation
        #du score si un poussin est mangé, et la mise à True de game_over si
        #le serpent s'est mordu la queue.
        #Code traçant les objets du jeu
        #commencer par effacer l'écran
        ecran.fill(BLANC)
        #afficher tous les lutins
        tout_lutins.draw(ecran)
        #Mettre l'écran à jour
```

```
        pygame.display.flip()

    if game_over:
        #affichage du dernier écran

    #fin de la boucle principale
    #Mise à jour de l'écran à raison de 10 images/seconde
    horloge.tick(10)
```

L'affichage de texte dans pygame

Pour afficher du texte dans pygame il faut procéder en trois étapes :

1. Créer un objet detype Font qui représente la police à utiliser avec sa taille.
2. Créer une image du texte à afficher avec la fonction render(). Cette fonction crée une image du texte, invisible.
3. Afficher l'image en utilisant la méthode blit() sur la fenêtre.

```
font = pygame.font.SysFont("comicsansms", 40)
```

Ici nous créons un objet Font, chargé à partir de la police système « Comic Sans ms ». La taille de la police sera de 40.

```
texte_image= font.render("Attention ça va partir", True, BLANC)
```

Nous créons une image du texte « Attention ça va partir ». Ce texte sera blanc

Le second paramètre de la fonction render, indique si l'antialiasing est mis en œuvre ou non.

Le texte n'occupe qu'une seule ligne. On ne peut pas indiquer de retour à la ligne.

L'image créée a les dimensions nécessaires pour contenir le texte.

On peut donc obtenir ces dimentions par :

```
texte_image.get_rect()
```

```
ecran.blit(texte_image, [150, 20])
```

Affiche l'image sur l'écran aux coordonnées x = 150, y = 20

Affichage des deux premières fenêtres

Juste **avant** la boucle principale, qui elle gère la partie jeu, nous ajoutons

```
#pages d'instructions
font = pygame.font.SysFont("comicsansms", 40)
#faut'il encore afficher une page
afficher_instructions = True
#numéro de la page à afficher
instruction_page = 1
# -------- Boucle d'affichage des premières pages -----------
while not arret and afficher_instructions:
    for event in pygame.event.get():
        #On laisse la possibilité au joueur d'arrêter le jeu tout de suite
        if event.type == pygame.QUIT:
            arret = True
        #un clic de souris incrémente le numéro de la page
        if event.type == pygame.MOUSEBUTTONDOWN:
            instruction_page += 1
            #si on a déjà affichér les deux premières pages
            #il faut sortir de cette boucle, pour lancer le jeu
            if instruction_page == 3:
                afficher_instructions = False


    # Mettre l'écran en noir pour écrire en blanc dessus
    ecran.fill(NOIR)
    if instruction_page == 1:
        # Les instructions sont écrites dans un fichier png
        # Ce fichier a été créé avec Paint.net
        instruction_image =
pygame.image.load("images\instructions.png").convert_alpha()
        rectInstructions = instruction_image.get_rect()
```

```
            ecran.blit(instruction_image, rectInstructions)
        if instruction_page == 2:
            # Afficher la page 2 : texte constitué de 2 lignes
            #Première ligne
            texte_image = font.render("Attention ça va partir", True, BLANC)
            ecran.blit(texte_image, [150, 20])
            #Deuxième ligne
            texte_image = font.render("Cliquez pour démarrer", True, BLANC)
            ecran.blit(texte_image, [150, 100])

        # 60 images par seconde
        horloge.tick(60)
        # mise à jour de l'écran et retour en haut de la boucle
        pygame.display.flip()
```

Lorsqu'on sort de cette boucle, parceque **afficher_instructions** est False, nous entrons dans la boucle principale, gérant le déroulement du jeu.

Afficher le score

L'affichage du score se fait lorsque la variable **game_over = True** :

```
if game_over:
        # Mettre l'écran en noir pour écrire en blanc dessus
        ecran.fill(NOIR)
        texte_image = font.render("Game Over", True, BLANC)
        #Affichage de Game Over à peu près au centre de l'écran en x
        #et au tiers haut de l'écran en y
        text_rect = texte_image.get_rect()
        text_x = ecran.get_width() / 2 - text_rect.width / 2
        text_y = ecran.get_height() / 3 - text_rect.height / 3
        ecran.blit(texte_image, [text_x, text_y])
```

```
#Affichage d’une deuxième ligne de texte en dessous de Game Over
texte_image = font.render("Le serpent s'est mordu la queue", True, BLANC)
ecran.blit(texte_image, [50, text_y + 50])
#Affichage d’une troisième ligne avec le score
leScore = "Score : " + str(score)
texte_image = font.render(leScore, True, BLANC)
ecran.blit(texte_image, [text_x, text_y + 100])
#Mettre l’écran à jour
pygame.display.flip()
```