

Réalisation Sujet 18 : Programmation pour les enfants : Découverte de la programmation avec des projets simples et ludiques TP 06

Salim MOUSSAOUI, Maria ZEBDA.
Enseignant de TP : Ronan Sicre
Université Paul Sabatier
Vendredi 9 mai 2025



TP Python - Animer une balle avec CodeSkulptor

Introduction

Bienvenue dans ton tout premier TP en Python avec **CodeSkulptor**, un site magique qui permet de **voir ton code prendre vie dans une vraie interface graphique** ! 

Lien vers **CodeSkulptor** : <https://py3.codeskulptor.org/index.html>

Ici, pas de "print" en noir et blanc... **on va faire bouger une balle**, la faire rebondir, et même créer un **mini-jeu avec une raquette** ! Tu n'as jamais fait de Python ? Pas grave, ce TP t'explique tout, pas à pas, de façon **fun, simple et interactive**.

Tu vas découvrir :

- ✓ Comment créer une fenêtre de jeu
- ✓ Comment dessiner une balle
- ✓ Comment la faire rebondir
- ✓ Comment contrôler une raquette avec le clavier
- ✓ Ce qu'est une **fonction** et à quoi elle sert
- ✓ Ce qu'est l'**incrément** et pourquoi c'est super utile
- ✓ Et même comment **personnaliser** ton jeu !

C'est quoi une fonction en Python ?

Une **fonction**, c'est comme une **recette** ou une **machine à instructions**. Tu lui donnes un nom, tu décris ce qu'elle fait à l'intérieur, et ensuite tu peux l'utiliser **autant de fois que tu veux** !

Exemple très simple :

```
def dire_bonjour():  
    print("Salut !")  
  
dire_bonjour()  # Affiche : Salut !  
  
dire_bonjour()  # Affiche : Salut ! (encore)
```

Ici, la fonction `dire_bonjour()` affiche un message. On peut l'appeler quand on veut !

+ C'est quoi l'incrémentation ?

L'**incrémentation**, c'est une façon rapide d'**augmenter** ou **diminuer** une **variable**.

Par exemple :

```
vitesse = 10  
vitesse += 2    # augmente de 2 → 12  
vitesse -= 4    # diminue de 4 → 8  
vitesse *= 2    # multiplie par 2 → 16  
vitesse /= 4    # divise par 4 → 2.5
```

🧠 Tous ces raccourcis sont équivalents à des écritures plus longues, comme :

```
vitesse = vitesse + 2    # augmente de 2  
vitesse = vitesse - 4    # diminue de 4  
vitesse = vitesse * 2    # multiplie par 2  
vitesse = vitesse / 4    # divise par 4
```

Partie 1 - Une balle qui rebondit toute seule !

Voici un premier petit code très simple :

```
import simplegui

# Position initiale de la balle

X = 250

Y = 200

position_balle = [X, Y]

# Vitesse de déplacement

vitesse_x = 2

vitesse_y = 3

# Taille de la balle

rayon = 20

# Fonction qui dessine la balle

def dessiner(canvas):

    global vitesse_x, vitesse_y

    canvas.draw_circle(position_balle, rayon, 5, 'red', 'white')

    position_balle[0] += vitesse_x

    position_balle[1] += vitesse_y

    # Rebonds sur les bords gauche/droit

    if position_balle[0] > 500 - rayon or position_balle[0] < rayon:

        vitesse_x = -vitesse_x

    # Rebonds sur les bords haut/bas

    if position_balle[1] > 400 - rayon or position_balle[1] < rayon:

        vitesse_y = -vitesse_y
```

```
# Créer la fenêtre de jeu


fenetre = simplegui.create_frame("Balle rebondissante", 500, 400)

fenetre.set_draw_handler(dessiner)

fenetre.start()
```


Ce que fait chaque ligne :

- `import simplegui` : importe le module graphique.
- `position_balle = [...]` : définit la position de départ de la balle.
- `vitesse_x / vitesse_y` : la vitesse du déplacement.
- `rayon` : taille de la balle.
- `def dessiner(canvas)` : une **fonction** qui va dessiner la balle et gérer ses rebonds.
- `canvas.draw_circle(...)` : dessine la balle.
- `position_balle[...] += vitesse` : fait bouger la balle.
- `if ...` : vérifie si la balle touche un bord pour la faire rebondir.
- `fenetre = ...` : crée la fenêtre graphique.
- `set_draw_handler` : indique quelle fonction utiliser pour dessiner.
- `start()` : lance l'animation !

 **Personnalise ta balle !** Change la couleur, la taille, ou la vitesse pour un résultat unique. Tu peux même faire une balle multicolore !

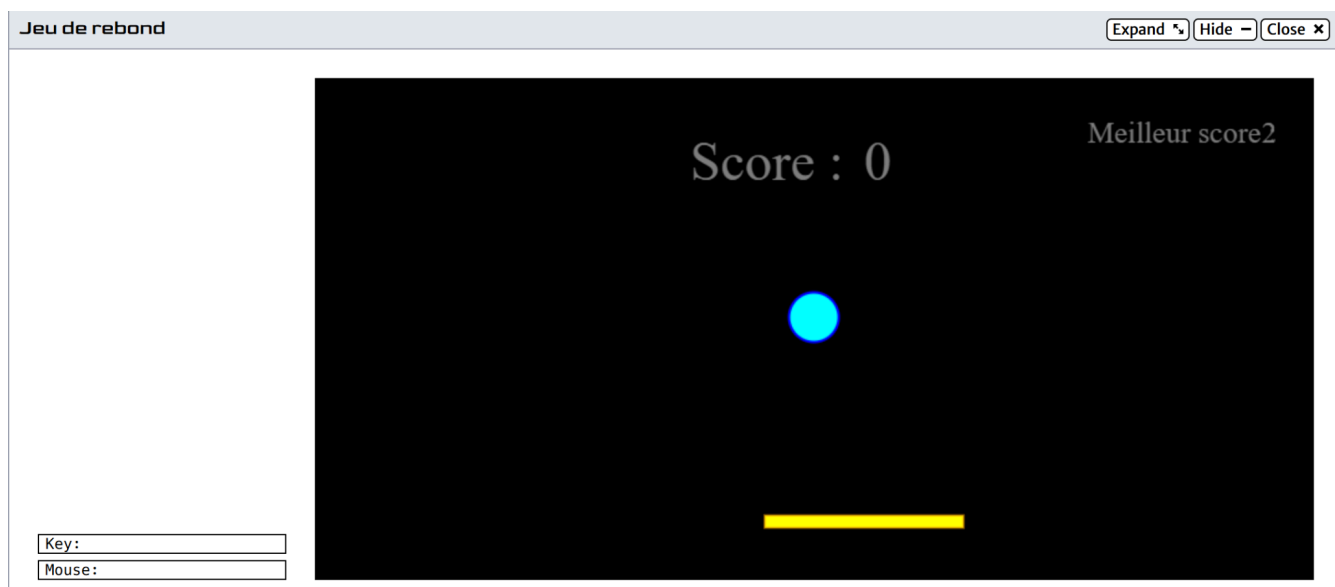


Partie 2 - Le jeu avec raquette : évite de laisser tomber la balle !

Tu te souviens du **jeu de rebond avec raquette** qu'on avait fait sur Scratch ? 
Ici, on va créer exactement **le même jeu**, mais en **Python**, avec du vrai code et une interface animée !

Voici maintenant un mini-jeu plus complet ! Tu dois empêcher la balle de tomber en bas de l'écran. Pour cela, tu contrôles une **raquette** avec les touches du clavier : ← et →.

En Python :



Sur Scratch :



Voici le code :

```
Code
1 # Implémentation d'un jeu de rebond (Bounce Game)
2
3 import simplegui
4 import random
5
6 # Initialisation des variables globales du jeu
7 LARGEUR = 800
8 HAUTEUR = 400
9 RAYON_BALLE = 20
10 score = 0
11 vitesse_raquette = 0
12 position_raquette = LARGEUR / 2
13 position_balle = [LARGEUR / 2, HAUTEUR / 2]
14
15 # La gravité tire toujours la balle vers le bas
16 gravite = 0.5
17
18 # La position de la balle est mise à jour selon la vitesse
19 vitesse_balle = [0, 0]
20
21 # Variable pour le déplacement horizontal (dérive)
22 derive = 0
23
24 # Modifier cette variable pour raccourcir la raquette et rendre le jeu plus dur !
25 largeur_raquette = 160
26
27 # Meilleur score
28 meilleur_score = 0
29
30 # Fonction pour réinitialiser la balle
31 def generer_balle():
32     global position_balle, vitesse_balle, gravite, derive, position_raquette
33     position_balle = [LARGEUR / 2, 180]
34     gravite = 0.5
35     vitesse_balle = [0, 0]
36     derive = random.choice([-0.1, -0.05, -0.07, -0.02, 0.1, 0.05, 0.07, 0.02])
37     position_raquette = 360
38
39 # Démarrer une nouvelle partie
40 def nouvelle_partie():
41     global score
42     score = 0
43     generer_balle()
44
45 # Fonction principale de dessin
46 def dessiner(canvas):
47     global score, position_raquette, vitesse_raquette, position_balle, vitesse_balle, gravite, derive, largeur_raquette, meilleur_score
48
49     # Dessine la balle
50     canvas.draw_circle(position_balle, RAYON_BALLE, 2, "Blue", "Cyan")
51
52     # Dessine la raquette
53     canvas.draw_polygon([[position_raquette, 350], [position_raquette, 360],
54                         [position_raquette + largeur_raquette, 360],
55                         [position_raquette + largeur_raquette, 350]], 1, 'Orange', 'Yellow')
56
57     # Met à jour la position de la raquette
58     position_raquette += vitesse_raquette
59
60     # Met à jour la position de la balle
61     position_balle[0] += vitesse_balle[0]
62     position_balle[1] += vitesse_balle[1]
63     vitesse_balle[0] += derive
64     vitesse_balle[1] += gravite
65
66     # Si la balle touche la raquette
67     if position_balle[1] == 330:
68         if position_raquette <= position_balle[0] <= position_raquette + largeur_raquette:
69             vitesse_balle[1] *= -1
70             vitesse_balle[1] += 0.5
71             derive = random.choice([-0.1, -0.05, -0.07, -0.02, 0.1, 0.05, 0.07, 0.02])
72             score += 1
73
74     # Rebonds sur les côtés de l'écran
75     if (position_balle[0] + derive) < 0 or (position_balle[0] + derive) > 800:
76         vitesse_balle[0] *= -1
77
78     # Si la balle touche le bas de l'écran, partie terminée
79     if position_balle[1] > 380:
80         if score > meilleur_score:
81             meilleur_score = score
82             nouvelle_partie()
83
84     # Affiche le score
85     canvas.draw_text('Score :', (300, 80), 44, 'Gray', 'serif')
86     canvas.draw_text(str(score), (440, 80), 44, 'Gray', 'serif')
87     canvas.draw_text('Meilleur score :', (620, 50), 24, 'Gray', 'serif')
88     canvas.draw_text(str(meilleur_score), (760, 50), 24, 'Gray', 'serif')
89
90 # Gérer les touches clavier
91 def touche_appuyee(touche):
92     vitesse = 12
93     global vitesse_raquette
94     if touche == simplegui.KEY_MAP["left"]:
95         vitesse_raquette -= vitesse
96     elif touche == simplegui.KEY_MAP["right"]:
97         vitesse_raquette += vitesse
98
99 def touche_relachee(touche):
100     vitesse = 12
101     global vitesse_raquette
102     if touche == simplegui.KEY_MAP["left"]:
103         vitesse_raquette += vitesse
104     elif touche == simplegui.KEY_MAP["right"]:
105         vitesse_raquette -= vitesse
106
107 # Créer la fenêtre
108 fenetre = simplegui.create_frame("Jeu de rebond", LARGEUR, HAUTEUR)
109 fenetre.set_draw_handler(dessiner)
110 fenetre.set_keydown_handler(touche_appuyee)
111 fenetre.set_keyup_handler(touche_relachee)
```



Vous trouverez le code pour le copier sur ce lien :

<https://colab.research.google.com/drive/146ty2jUFqL9jJSO-u5Xuwbq7yPgTtYBM?usp=sharing>

Copier le code et coller le sur **CodeSkulptor**

Ce que fait ce code (simplifié ligne par ligne) :

- On définit la taille du jeu, la balle, la raquette, la gravité, le score...
- `generer_balle()` place la balle au milieu avec une petite vitesse.
- `nouvelle_partie()` remet le score à 0 et appelle `generer_balle()`.
- `dessiner(canvas)` :
 - Affiche la balle et la raquette
 - Fait bouger la balle selon sa vitesse + gravité
 - Fait bouger la raquette si on appuie sur ← ou →
 - Vérifie si la balle touche la raquette → elle rebondit
 - Si elle tombe, on recommence une partie
 - On affiche le score et le meilleur score
- `touche_appuyee` et `touche_relachee` gèrent les touches ← →
- On crée la fenêtre, on lance le jeu

 **À toi de jouer !** Tu peux modifier la couleur de la balle, la taille de la raquette, le score, la gravité... Fais de ce jeu **le tien** !

Conclusion

Tu viens de coder **deux animations complètes** avec interface graphique en Python !

Grâce à ce TP, tu as appris plein de notions en programmation :

● Ce qu'est une **fonction**

✚ Ce qu'est l'**incrémentatation** et comment elle simplifie ton code

📦 Comment utiliser des **variables** pour faire bouger des objets

🎨 Comment personnaliser l'apparence d'un jeu

🔑 Et comment programmer un **vrai mini-jeu interactif** !

Tu es maintenant prêt à explorer **d'autres animations**, à créer **tes propres fonctions** et à rendre ton code **toujours plus vivant** !

Bravo, jeune programmeur Python ! 🐍💻