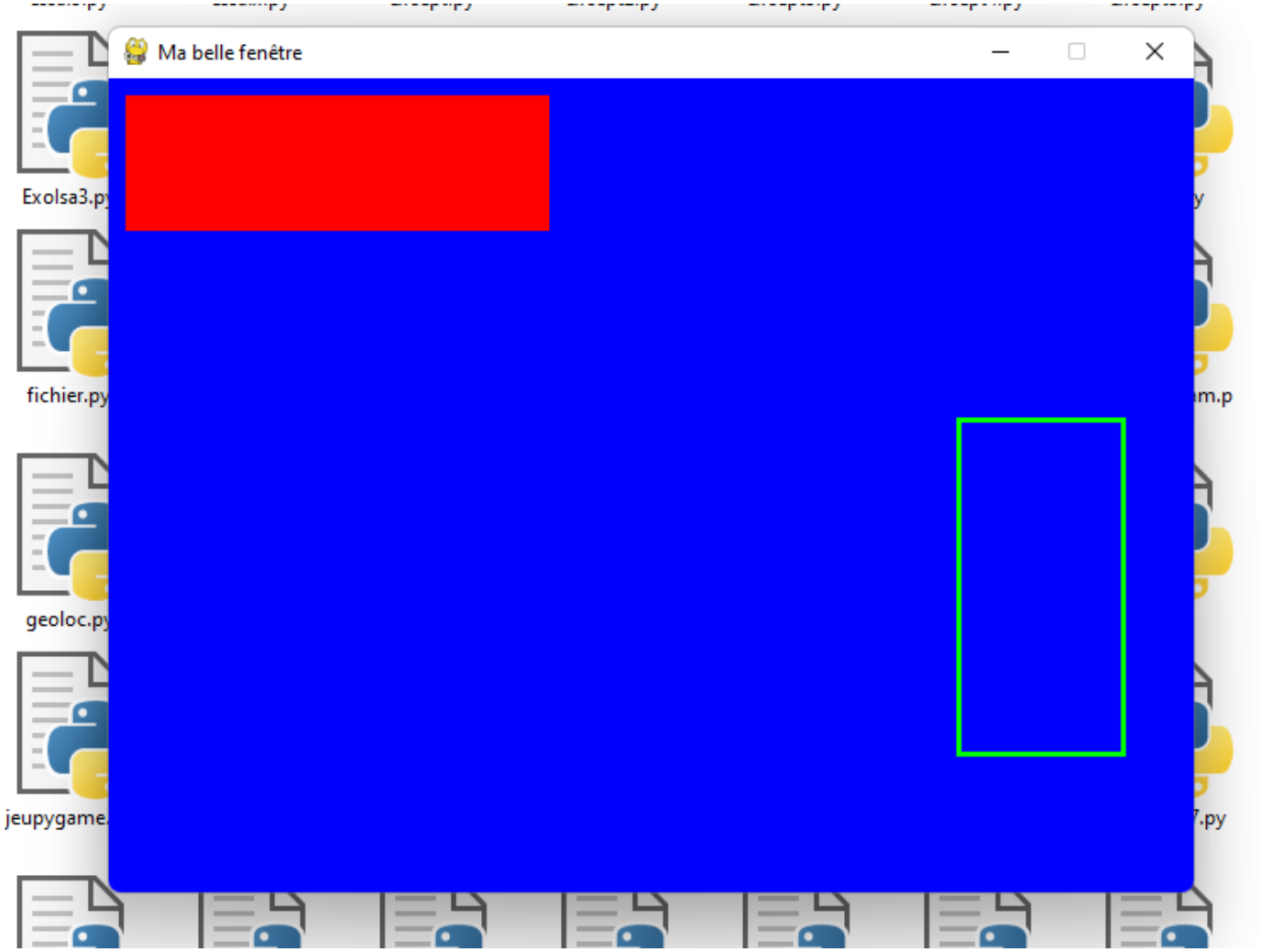
ANIMATION D'UNE FIGURE

Nous allons voir comment mettre en mouvement une figure,
par exemple un rectangle
Nous reprenons donc le code qui permet de dessiner un ou plusieurs rectangles :

```
#coding:utf-8
import pygame

pygame.init()                                   #constructeur
resolution = (640,480)
bleu    = (0,0,255)                             # creation des couleurs
rouge = (255,0,0)
vert    = (0,255, 0)
jaune  = (255,255, 0)
blanc  = (255,255,255)
pygame.display.set_caption("Ma belle fenêtre")
screen  =  pygame.display.set_mode(resolution)
screen.fill(bleu)                               #  pour colorier la fenêtre en bleu
myrect = pygame.Rect(10,10,250,80)     #constructeur de l'objet myrect
pygame.draw.rect(screen, rouge, myrect)
pygame.draw.rect(screen, vert, pygame.Rect(500,200,  100, 200),3)

pygame.display.flip()                           # pour rafraîchir la fenêtre
launched=True
while launched :                                #  boucle qui permet de laisser la surface
  for event in pygame.event.get() :                                          #affichée
    if event.type == pygame.QUIT :
      launched= False
```



Nous allons maintenant animer la fenêtre rouge et la réduire, en la faisant déplacer de façon oblique vers le bas droit de la fenêtre créée
Pour ce faire, nous importons le module time qui nous permettra d'insérer une temporisation à l'endroit voulu et nous injectons le bloc approprié à l'endroit adéquat :

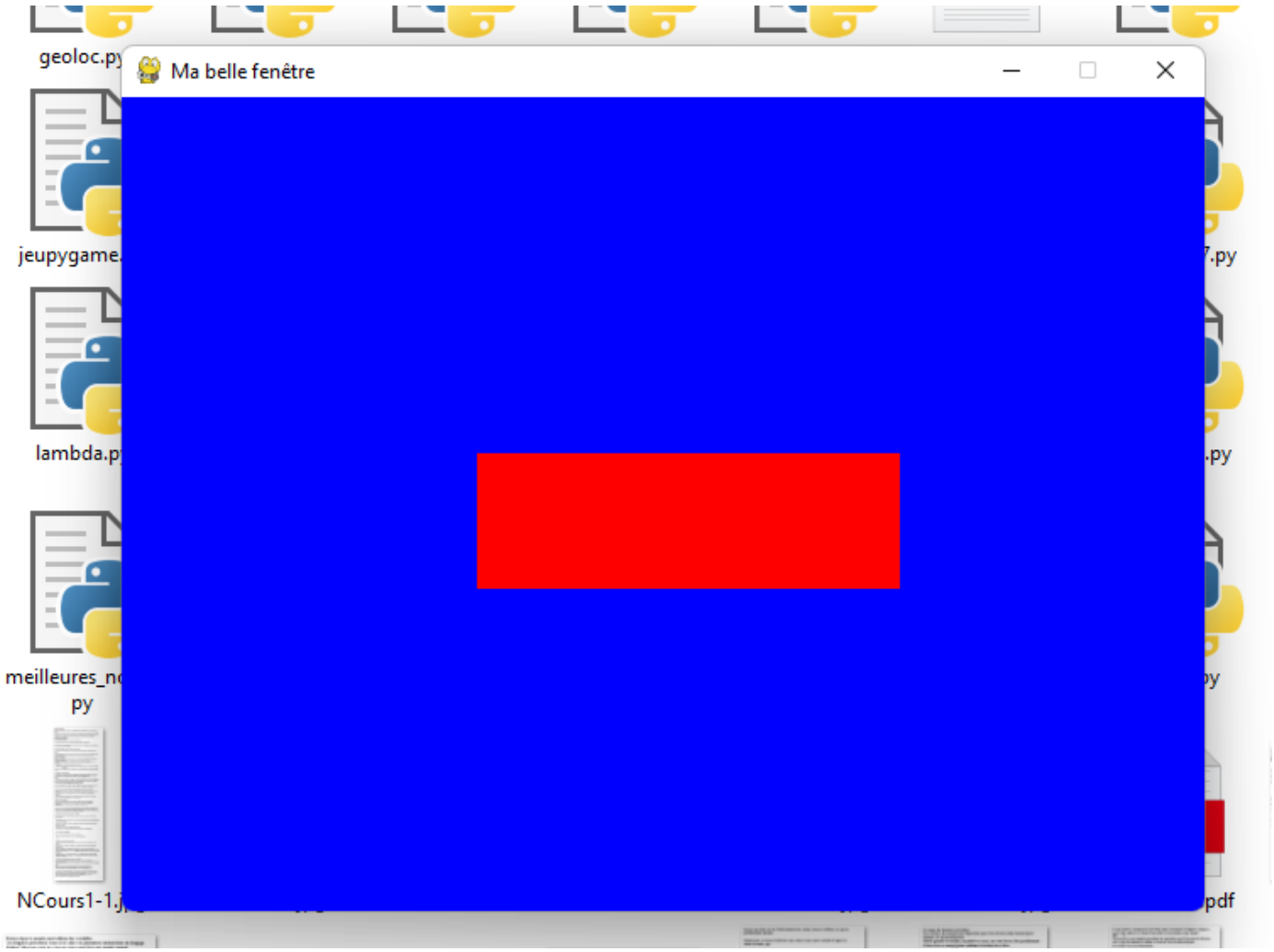
```
#coding:utf-8
import pygame
import time

pygame.init()                                   #constructeur
resolution = (640,480)
bleu    = (0,0,255)                             # creation des couleurs
rouge = (255,0,0)
black  = (0,0,0)
pygame.display.set_caption("Ma belle fenêtre")
screen  =  pygame.display.set_mode(resolution)
screen.fill(bleu)                               #  pour colorier la fenêtre en bleu
myrect = pygame.Rect(10,10,250,80)     #constructeur de l'objet myrect
pygame.draw.rect(screen, rouge, myrect)     # rafraichissement

i = 0                                                   #  bloc mouvement
while  i < 20 :                                         #boucle organisant le mouvement
    time.sleep(0.100)                                   #  un centième de seconde
    screen.fill(black)                                  #   pour effacer l'écran
    myrect.x += 10                                      # déplacement horiz de 10 pixels
    myrect.y += 10                                      # déplacement vertical de 10 pixels
    screen.fill(bleu)                                   # on réaffiche l'écran en bleu
    pygame.draw.rect(screen, rouge, myrect)       #repositionnement du rectangle
    pygame.display.flip()                               # pour rafraîchir la fenêtre
    i +=1
launched = True
while launched :                                # boucle qui permet de laisser la surface
   for event in pygame.event.get() :                                                    #affichée
      if event.type == pygame.QUIT :
         launched= False
```

En jouant sur les valeurs des paramètres de la méthode time.sleep() ou de l'incrémentation des attributs myrect.x et myrect.y, on peut rendre l'animation plus ou moins fluide.

Le rectangle rouge va se déplacer jusqu'à la position finale indiquée ici :
(le rectangle vert ne figure plus sur la surface)



On dispose de plusieurs méthodes pour l'objet Rect :

Rect.copy() **qui renvoie un autre Rect identique**
Ex : mynewrect = myrect.copy() **mais il faudra indiquer les paramètres permettant de positionner mynewrect**
Rect.move() **ou Rect.move_ip()** **change les coordonnées mais il faudra redessiner avec pygame.draw.rect(screen, rouge, mynewrect)**
Rect.inflate() **pour agrandir le rectangle ou le rétrécir si l'on passe des paramètres négatifs**
Rect.colliderect() **pour détecter d'éventuelles collisions et les gérer en y associant une action définie**

La méthode Colliderect rend False ou True selon que l'objet en mouvement rencontre un obstacle (un autre objet passé en paramètre) ou pas.
Ainsi nous pouvons créer une animation de ce type :

- **Un carré rouge par exemple (rectangle particulier) situé sur le bord gauche d'une surface, va se déplacer de la gauche vers la droite. Sur le bord droit de la surface se situera un mur , représenté par un rectangle vertical vert allongé. Lorsque le carré rencontrera le mur, la méthode colliderect() rendra True et mettra fin au mouvement.**

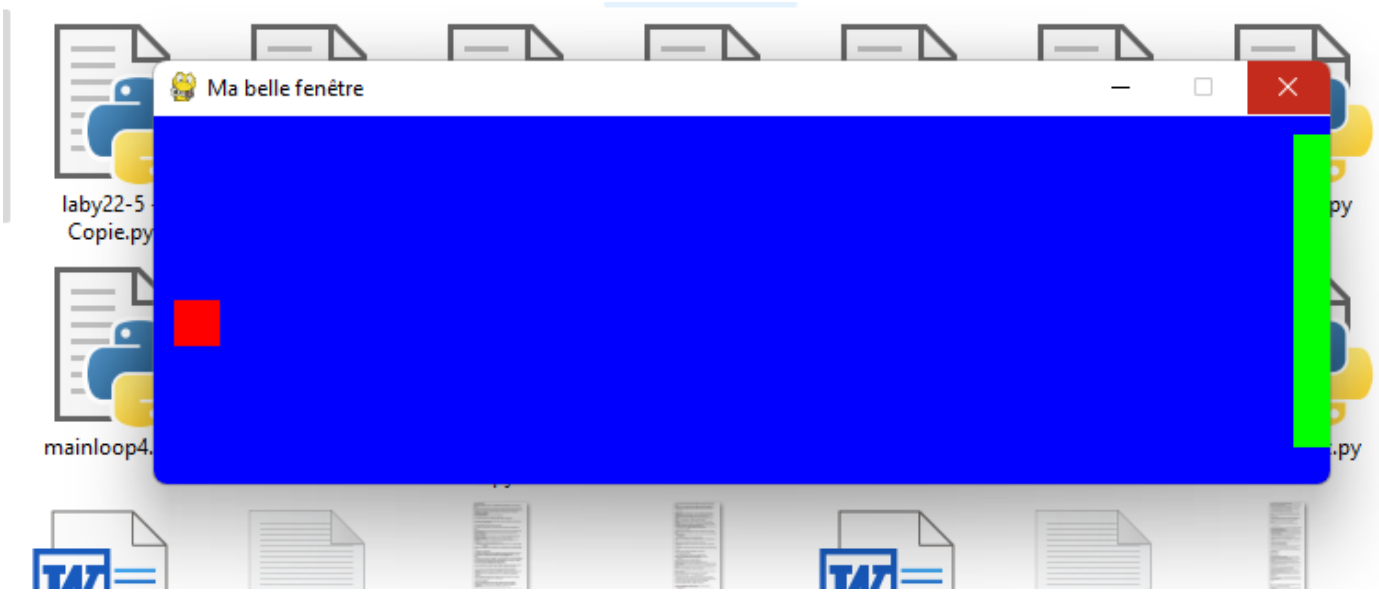
```
#coding:utf-8
import pygame
import time

pygame.init()                                   #constructeur
resolution = (640,200)
bleu    = (0,0,255)                             # creation des couleurs
rouge = (255,0,0)
vert   = (0,255,0)
pygame.display.set_caption("Ma belle fenêtre")
screen  =  pygame.display.set_mode(resolution)
screen.fill(bleu)                                #  pour colorier la surface en bleu

myrect = pygame.Rect(10,100,25,25)              #constructeur de l'objet mobile
mymur = pygame.Rect(620,10,20,170)              #constructeur du mur
pygame.draw.rect(screen, rouge, myrect)
pygame.draw.rect(screen, rouge, mymur)

launched = True
while launched :
  for event in pygame.event.get() :
    if event.type == pygame.QUIT :
      launched= False
                                                          #  bloc mouvement
  while  not myrect.colliderect(mymur)   :     # Tant que pas collision avec le mur
    time.sleep(0.010)                                    #  un centième de seconde
    screen.fill(bleu)                                     #   pour effacer l'écran
    myrect.x += 1                                     # déplacement horiz de 10 pixels
    pygame.draw.rect(screen, rouge, myrect)                    #repositionnement
    pygame.draw.rect(screen, vert, mymur)                     #repositionnement
    pygame.display.flip()                                 # pour rafraîchir la fenêtre
```

Position initiale :



Position finale (bloquée) :

A la place de while not myrect.colliderect(mymur) :
on peut évidemment écrire :
while myrect.colliderect(mymur) == False :

Si l'on enlève la condition : while not myrect.colliderect(mymur)
et qu'on la remplace par 1, le carré rouge passera sous le mur vert car il a été défini avant, et inversement