

Inicio (<https://recursospython.com/>)

Códigos de fuente (<https://www.recursospython.com/category/codigos-de-fuente/>)

Guías y manuales (<https://www.recursospython.com/category/guias-y-manuales/>)


Foro (<https://foro.recursospython.com/>)      Micro (<https://micro.recursospython.com/>)

Tutorial (<https://tutorial.recursospython.com/>)      Newsletter (<https://recursospython.com/newsletter/>)

Consultoría (<https://recursospython.com/consultoria/>)

Contacto (<https://recursospython.com/contacto/>)      Donar  (<https://recursospython.com/donar/>)

# Tetris con PyGame

junio 25, 2018 (<https://recursospython.com/codigos-de-fuente/tetris-pygame/>) by [Recursos Python](https://recursospython.com/author/admin/) (<https://recursospython.com/author/admin/>)  (<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comments>) 20 comentarios (<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comments>)

**Descarga:** [tetris.zip](https://www.recursospython.com/wp-content/uploads/2018/06/tetris.zip) (<https://www.recursospython.com/wp-content/uploads/2018/06/tetris.zip>).

En esta ocasión presentamos el código de fuente de una implementación somera del clásico juego «Tetris», usando la librería de desarrollo de videojuegos 2D PyGame. El programa tiene menos de 500 líneas, aunque admito que la tarea no fue tan sencilla como parecía *a priori*. Si bien la simpleza del juego se avista fácilmente, diseñar su lógica implica detenimiento y paciencia.

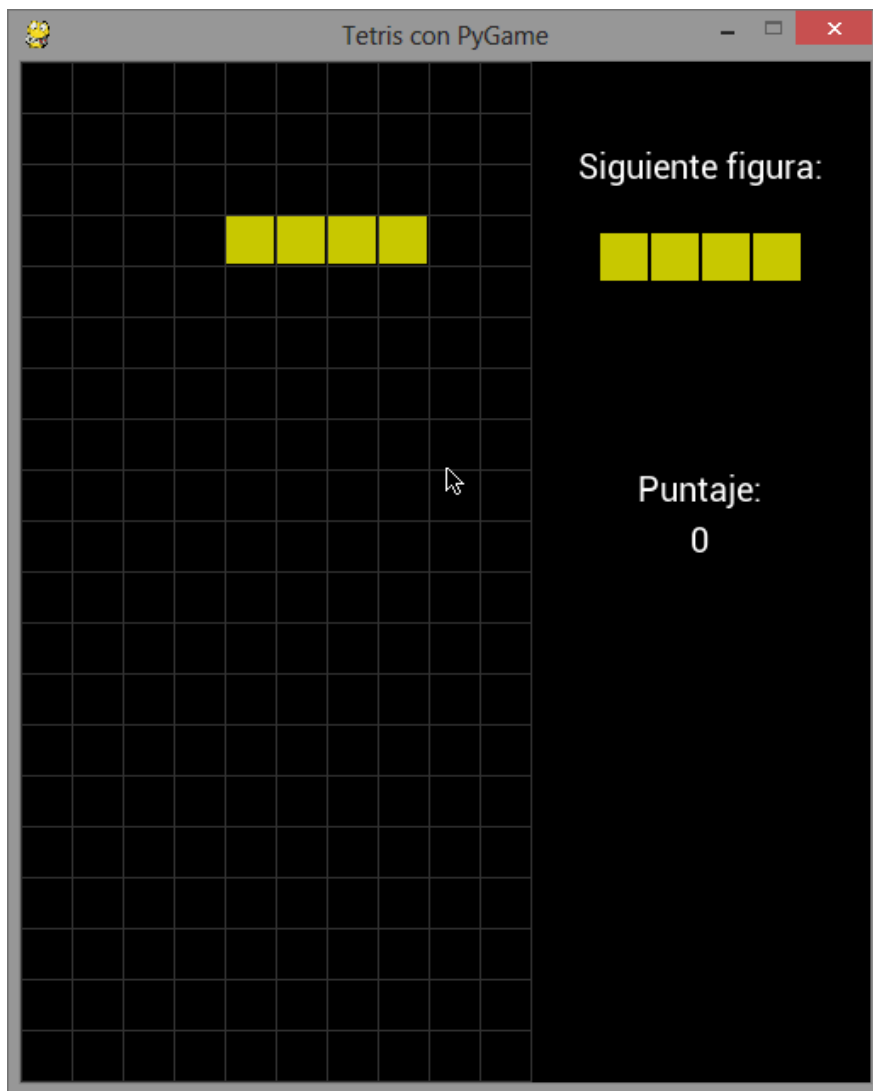


## Últimas entradas

[Reproducir inyección SQL en sqlite3 y PyMySQL](#) (<https://recursospython.com/guias-y-manuales/reproducir-inyeccion-sql-en-sqlite3-y-pymysql/>)

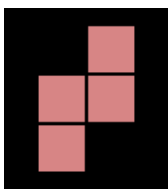
[Bloc de notas simple con Tk \(tkinter\)](#) (<https://recursospython.com/codigos-de-fuente/bloc-de-notas-simple-con-tkinter/>)

[Examinar archivo o carpeta en Tk \(tkinter\)](#) (<https://recursospython.com/guias-y-manuales/examinar-archivo-o-carpeta-en-tk-tkinter/>)



(<https://www.recurso python.com/wp-content/uploads/2018/06/tetris.gif>)

El código requiere de las librerías PyGame y NumPy (ambas se instalan fácilmente vía `pip install pygame numpy`) y la fuente Roboto (incluida en la descarga). NumPy es utilizado para representar cada uno de los bloques del juego como una matriz, aprovechando sus funciones de rotación y volteo. Por ejemplo, considérese la siguiente figura.



(<https://www.recurso python.com/wp-content/uploads/2018/06/figura.png>)

Ésta es internamente representada usando una matriz en NumPy:

```
1. np.array((
2.     (0, 1),
3.     (1, 1),
4.     (1, 0),
```



[Múltiples configuraciones \(desarrollo/producción\) en Django \(https://recursospython.com/gui-y-manuales/multiples-configuraciones-desarrollo-produccion-en-django/\)](#)

[Buscar el archivo de mayor tamaño en una ruta \(https://recursospython.com/cor-de-fuente/buscar-el-archivo-de-mayor-tamano-en-una-ruta/\)](#)

## Comentarios recientes

Recursos Python en [Generar código QR \(https://recursospython.com/gui-y-manuales/generar-codigo-qr/#comment-2586\)](#)

Joaquín en [Generar código QR \(https://recursospython.com/gui-y-manuales/generar-codigo-qr/#comment-2584\)](#)

Recursos Python en [pickle – Serialización de objetos \(https://recursospython.com/gui-y-manuales/pickle-serializacion-de-objetos/#comment-2435\)](#)

Recursos Python en [Lista desplegable \(Combobox\)](#)

```
5. ))
```

Sin más, el código es el siguiente.

```
1.  #!/usr/bin/env python
2.  # -*- coding: utf-8 -*-
3.
4.  """
5.      The classic Tetris developed using PyGame.
6.      Copyright (C) 2018 Recursos Python -
7.      recursospython.com.
8.  """
9.
10. from collections import OrderedDict
11. import random
12.
13. from pygame import Rect
14. import pygame
15. import numpy as np
16.
17. WINDOW_WIDTH, WINDOW_HEIGHT = 500, 601
18. GRID_WIDTH, GRID_HEIGHT = 300, 600
19. TILE_SIZE = 30
20.
21.
22. def remove_empty_columns(arr, _x_offset=0,
23. _keep_counting=True):
24.     """
25.     Remove empty columns from arr (i.e. those filled with
26.     zeros).
27.     The return value is (new_arr, x_offset), where x_offset
28.     is how
29.     much the x coordinate needs to be increased in order to
30.     maintain
31.     the block's original position.
32.     """
33.     for colid, col in enumerate(arr.T):
34.         if col.max() == 0:
35.             if _keep_counting:
36.                 _x_offset += 1
37.                 # Remove the current column and try again.
38.                 arr, _x_offset = remove_empty_columns(
39.                     np.delete(arr, colid, 1), _x_offset,
40.                     _keep_counting)
41.             break
42.         else:
43.             _keep_counting = False
44.     return arr, _x_offset
45.
46.
47. class BottomReached(Exception):
48.     pass
49.
50.
51. class TopReached(Exception):
52.     pass
53.
54.
55. class Block(pygame.sprite.Sprite):
56.
57.     @staticmethod
58.     def collide(block, group):
59.         """
```

[en Tcl/Tk \(tkinter\).  
\(https://recursospython.com/gui-y-manuales/lista-desplegable-combobox-en-tkinter/#comment-2434\)](https://recursospython.com/gui-y-manuales/lista-desplegable-combobox-en-tkinter/#comment-2434)

Herná en [Lista desplegable \(Combobox\) en Tcl/Tk \(tkinter\).  
\(https://recursospython.com/gui-y-manuales/lista-desplegable-combobox-en-tkinter/#comment-2424\)](https://recursospython.com/gui-y-manuales/lista-desplegable-combobox-en-tkinter/#comment-2424)

```

55.         Check if the specified block collides with some
other block
56.         in the group.
57.         """
58.         for other_block in group:
59.             # Ignore the current block which will always
collide with itself.
60.             if block == other_block:
61.                 continue
62.             if pygame.sprite.collide_mask(block,
other_block) is not None:
63.                 return True
64.             return False
65.
66.     def __init__(self):
67.         super().__init__()
68.         # Get a random color.
69.         self.color = random.choice((
70.             (200, 200, 200),
71.             (215, 133, 133),
72.             (30, 145, 255),
73.             (0, 170, 0),
74.             (180, 0, 140),
75.             (200, 200, 0)
76.         ))
77.         self.current = True
78.         self.struct = np.array(self.struct)
79.         # Initial random rotation and flip.
80.         if random.randint(0, 1):
81.             self.struct = np.rot90(self.struct)
82.         if random.randint(0, 1):
83.             # Flip in the X axis.
84.             self.struct = np.flip(self.struct, 0)
85.         self._draw()
86.
87.     def _draw(self, x=4, y=0):
88.         width = len(self.struct[0]) * TILE_SIZE
89.         height = len(self.struct) * TILE_SIZE
90.         self.image = pygame.surface.Surface([width,
height])
91.         self.image.set_colorkey((0, 0, 0))
92.         # Position and size
93.         self.rect = Rect(0, 0, width, height)
94.         self.x = x
95.         self.y = y
96.         for y, row in enumerate(self.struct):
97.             for x, col in enumerate(row):
98.                 if col:
99.                     pygame.draw.rect(
100.                         self.image,
101.                         self.color,
102.                         Rect(x*TILE_SIZE + 1, y*TILE_SIZE +
1,
103.                             TILE_SIZE - 2, TILE_SIZE - 2)
104.                     )
105.         self._create_mask()
106.
107.     def redraw(self):
108.         self._draw(self.x, self.y)
109.
110.     def _create_mask(self):
111.         """
112.         Create the mask attribute from the main surface.
113.         The mask is required to check collisions. This
should be called
114.         after the surface is created or update.
115.         """

```

```

116.         self.mask = pygame.mask.from_surface(self.image)
117.
118.     def initial_draw(self):
119.         raise NotImplementedError
120.
121.     @property
122.     def group(self):
123.         return self.groups()[0]
124.
125.     @property
126.     def x(self):
127.         return self._x
128.
129.     @x.setter
130.     def x(self, value):
131.         self._x = value
132.         self.rect.left = value*TILE_SIZE
133.
134.     @property
135.     def y(self):
136.         return self._y
137.
138.     @y.setter
139.     def y(self, value):
140.         self._y = value
141.         self.rect.top = value*TILE_SIZE
142.
143.     def move_left(self, group):
144.         self.x -= 1
145.         # Check if we reached the left margin.
146.         if self.x < 0 or Block.collide(self, group):
147.             self.x += 1
148.
149.     def move_right(self, group):
150.         self.x += 1
151.         # Check if we reached the right margin or collided
with another
152.         # block.
153.         if self.rect.right > GRID_WIDTH or
Block.collide(self, group):
154.             # Rollback.
155.             self.x -= 1
156.
157.     def move_down(self, group):
158.         self.y += 1
159.         # Check if the block reached the bottom or collided
with
160.         # another one.
161.         if self.rect.bottom > GRID_HEIGHT or
Block.collide(self, group):
162.             # Rollback to the previous position.
163.             self.y -= 1
164.             self.current = False
165.             raise BottomReached
166.
167.     def rotate(self, group):
168.         self.image = pygame.transform.rotate(self.image,
90)
169.         # Once rotated we need to update the size and
position.
170.         self.rect.width = self.image.get_width()
171.         self.rect.height = self.image.get_height()
172.         self._create_mask()
173.         # Check the new position doesn't exceed the limits
or collide
174.         # with other blocks and adjust it if necessary.
175.         while self.rect.right > GRID_WIDTH: ▲

```

```

176.         self.x -= 1
177.     while self.rect.left < 0:
178.         self.x += 1
179.     while self.rect.bottom > GRID_HEIGHT:
180.         self.y -= 1
181.     while True:
182.         if not Block.collide(self, group):
183.             break
184.         self.y -= 1
185.     self.struct = np.rot90(self.struct)
186.
187.     def update(self):
188.         if self.current:
189.             self.move_down()
190.
191.
192.     class SquareBlock(Block):
193.         struct = (
194.             (1, 1),
195.             (1, 1)
196.         )
197.
198.
199.     class TBlock(Block):
200.         struct = (
201.             (1, 1, 1),
202.             (0, 1, 0)
203.         )
204.
205.
206.     class LineBlock(Block):
207.         struct = (
208.             (1,),
209.             (1,),
210.             (1,),
211.             (1,)
212.         )
213.
214.
215.     class LBlock(Block):
216.         struct = (
217.             (1, 1),
218.             (1, 0),
219.             (1, 0),
220.         )
221.
222.
223.     class ZBlock(Block):
224.         struct = (
225.             (0, 1),
226.             (1, 1),
227.             (1, 0),
228.         )
229.
230.
231.     class BlocksGroup(pygame.sprite.OrderedUpdates):
232.
233.         @staticmethod
234.         def get_random_block():
235.             return random.choice(
236.                 (SquareBlock, TBlock, LineBlock, LBlock,
237.                  ZBlock)) ()
238.
239.     def __init__(self, *args, **kwargs):
240.         super().__init__(self, *args, **kwargs)
241.         self._reset_grid()

```

```

241.         self._ignore_next_stop = False
242.         self.score = 0
243.         self.next_block = None
244.         # Not really moving, just to initialize the
attribute.
245.         self.stop_moving_current_block()
246.         # The first block.
247.         self._create_new_block()
248.
249.     def _check_line_completion(self):
250.         """
251.         Check each line of the grid and remove the ones
that
252.         are complete.
253.         """
254.         # Start checking from the bottom.
255.         for i, row in enumerate(self.grid[::-1]):
256.             if all(row):
257.                 self.score += 5
258.                 # Get the blocks affected by the line
deletion and
259.                 # remove duplicates.
260.                 affected_blocks = list(
261.                     OrderedDict.fromkeys(self.grid[-1 -
i]))
262.
263.                 for block, y_offset in affected_blocks:
264.                     # Remove the block tiles which belong
to the
265.                     # completed line.
266.                     block.struct = np.delete(block.struct,
y_offset, 0)
267.                     if block.struct.any():
268.                         # Once removed, check if we have
empty columns
269.                         # since they need to be dropped.
270.                         block.struct, x_offset = \
271.                             remove_empty_columns(block.struct)
272.                         # Compensate the space gone with
the columns to
273.                         # keep the block's original
position.
274.                         block.x += x_offset
275.                         # Force update.
276.                         block.redraw()
277.                     else:
278.                         # If the struct is empty then the
block is gone.
279.                         self.remove(block)
280.
281.                 # Instead of checking which blocks need to
be moved
282.                 # once a line was completed, just try to
move all of
283.                 # them.
284.                 for block in self:
285.                     # Except the current block.
286.                     if block.current:
287.                         continue
288.                     # Pull down each block until it reaches
the
289.                     # bottom or collides with another
block.
290.                     while True:
291.                         try:
292.                             block.move_down(self)
293.                         except BottomReached:▲

```

```

294.                 break
295.
296.                 self.update_grid()
297.                 # Since we've updated the grid, now the i
counter
298.                 # is no longer valid, so call the function
again
299.                 # to check if there're other completed
lines in the
300.                 # new grid.
301.                 self._check_line_completion()
302.                 break
303.
304.     def _reset_grid(self):
305.         self.grid = [[0 for _ in range(10)] for _ in
range(20)]
306.
307.     def _create_new_block(self):
308.         new_block = self.next_block or
BlocksGroup.get_random_block()
309.         if Block.collide(new_block, self):
310.             raise TopReached
311.         self.add(new_block)
312.         self.next_block = BlocksGroup.get_random_block()
313.         self.update_grid()
314.         self._check_line_completion()
315.
316.     def update_grid(self):
317.         self._reset_grid()
318.         for block in self:
319.             for y_offset, row in enumerate(block.struct):
320.                 for x_offset, digit in enumerate(row):
321.                     # Prevent replacing previous blocks.
322.                     if digit == 0:
323.                         continue
324.                     rowid = block.y + y_offset
325.                     colid = block.x + x_offset
326.                     self.grid[rowid][colid] = (block,
y_offset)
327.
328.     @property
329.     def current_block(self):
330.         return self.sprites() [-1]
331.
332.     def update_current_block(self):
333.         try:
334.             self.current_block.move_down(self)
335.         except BottomReached:
336.             self.stop_moving_current_block()
337.             self._create_new_block()
338.         else:
339.             self.update_grid()
340.
341.     def move_current_block(self):
342.         # First check if there's something to move.
343.         if self._current_block_movement_heading is None:
344.             return
345.         action = {
346.             pygame.K_DOWN: self.current_block.move_down,
347.             pygame.K_LEFT: self.current_block.move_left,
348.             pygame.K_RIGHT: self.current_block.move_right
349.         }
350.         try:
351.             # Each function requires the group as the first
argument
352.             # to check any possible collision.

```





```

353.         action[self._current_block_movement_heading]
354.     (self)
355.         except BottomReached:
356.             self.stop_moving_current_block()
357.             self._create_new_block()
358.         else:
359.             self.update_grid()
360.
361.     def start_moving_current_block(self, key):
362.         if self._current_block_movement_heading is not
None:
363.             self._ignore_next_stop = True
364.             self._current_block_movement_heading = key
365.
366.     def stop_moving_current_block(self):
367.         if self._ignore_next_stop:
368.             self._ignore_next_stop = False
369.         else:
370.             self._current_block_movement_heading = None
371.
372.     def rotate_current_block(self):
373.         # Prevent SquareBlocks rotation.
374.         if not isinstance(self.current_block, SquareBlock):
375.             self.current_block.rotate(self)
376.             self.update_grid()
377.
378.     def draw_grid(background):
379.         """Draw the background grid."""
380.         grid_color = 50, 50, 50
381.         # Vertical lines.
382.         for i in range(11):
383.             x = TILE_SIZE * i
384.             pygame.draw.line(
385.                 background, grid_color, (x, 0), (x,
GRID_HEIGHT)
386.             )
387.         # Horizontal liens.
388.         for i in range(21):
389.             y = TILE_SIZE * i
390.             pygame.draw.line(
391.                 background, grid_color, (0, y), (GRID_WIDTH, y)
392.             )
393.
394.
395.     def draw_centered_surface(screen, surface, y):
396.         screen.blit(surface, (400 - surface.get_width()/2, y))
397.
398.
399.     def main():
400.         pygame.init()
401.         pygame.display.set_caption("Tetris con PyGame")
402.         screen = pygame.display.set_mode((WINDOW_WIDTH,
WINDOW_HEIGHT))
403.         run = True
404.         paused = False
405.         game_over = False
406.         # Create background.
407.         background = pygame.Surface(screen.get_size())
408.         bgcolor = (0, 0, 0)
409.         background.fill(bgcolor)
410.         # Draw the grid on top of the background.
411.         draw_grid(background)
412.         # This makes blitting faster.
413.         background = background.convert()
414.

```



```

415.         try:
416.             font = pygame.font.Font("Roboto-Regular.ttf", 20)
417.         except OSError:
418.             # If the font file is not available, the default
will be used.
419.             pass
420.         next_block_text = font.render(
421.             "Siguiente figura:", True, (255, 255, 255),
bgcolor)
422.         score_msg_text = font.render(
423.             "Puntaje:", True, (255, 255, 255), bgcolor)
424.         game_over_text = font.render(
425.             ";Juego terminado!", True, (255, 220, 0), bgcolor)
426.
427.         # Event constants.
428.         MOVEMENT_KEYS = pygame.K_LEFT, pygame.K_RIGHT,
pygame.K_DOWN
429.         EVENT_UPDATE_CURRENT_BLOCK = pygame.USEREVENT + 1
430.         EVENT_MOVE_CURRENT_BLOCK = pygame.USEREVENT + 2
431.         pygame.time.set_timer(EVENT_UPDATE_CURRENT_BLOCK, 1000)
432.         pygame.time.set_timer(EVENT_MOVE_CURRENT_BLOCK, 100)
433.
434.         blocks = BlocksGroup()
435.
436.         while run:
437.             for event in pygame.event.get():
438.                 if event.type == pygame.QUIT:
439.                     run = False
440.                     break
441.                 elif event.type == pygame.KEYUP:
442.                     if not paused and not game_over:
443.                         if event.key in MOVEMENT_KEYS:
444.                             blocks.stop_moving_current_block()
445.                             elif event.key == pygame.K_UP:
446.                                 blocks.rotate_current_block()
447.                         if event.key == pygame.K_p:
448.                             paused = not paused
449.
450.                 # Stop moving blocks if the game is over or
paused.
451.                 if game_over or paused:
452.                     continue
453.
454.                 if event.type == pygame.KEYDOWN:
455.                     if event.key in MOVEMENT_KEYS:
456.
blocks.start_moving_current_block(event.key)
457.
458.                     try:
459.                         if event.type ==
EVENT_UPDATE_CURRENT_BLOCK:
460.                             blocks.update_current_block()
461.                         elif event.type ==
EVENT_MOVE_CURRENT_BLOCK:
462.                             blocks.move_current_block()
463.                     except TopReached:
464.                         game_over = True
465.
466.                 # Draw background and grid.
467.                 screen.blit(background, (0, 0))
468.                 # Blocks.
469.                 blocks.draw(screen)
470.                 # Sidebar with misc. information.
471.                 draw_centered_surface(screen, next_block_text, 50)
472.                 draw_centered_surface(screen,
blocks.next_block.image, 100)
473.                 draw_centered_surface(screen, score_msg_text, 240)

```

```

474.         score_text = font.render(
475.             str(blocks.score), True, (255, 255, 255),
            bgcolor)
476.         draw_centered_surface(screen, score_text, 270)
477.         if game_over:
478.             draw_centered_surface(screen, game_over_text,
            360)
479.         # Update.
480.         pygame.display.flip()
481.
482.         pygame.quit()
483.
484.
485.     if __name__ == "__main__":
486.         main()

```

## Artículos relacionados

- [Animación con PyGame + Exportarla como GIF](https://recursospython.com/guias-y-manuales/animacion-pygame-exportarla-gif/)  
(<https://recursospython.com/guias-y-manuales/animacion-pygame-exportarla-gif/>)
- [Transformación del panadero con PyGame](https://recursospython.com/codigos-de-fuente/transformacion-del-panadero-pygame/)  
(<https://recursospython.com/codigos-de-fuente/transformacion-del-panadero-pygame/>)

## Donar

¿Te gusta nuestro contenido? ¡Ayúdanos a [seguir creciendo con una donación \(/donar/\)](#)!

Entrada publicada en  
Códigos de fuente (<https://recursospython.com/category/codigos-de-fuente/>) con las  
etiquetas `pygame` (<https://recursospython.com/tag/pygame/>)

◀ [«python» no se reconoce como un comando interno o externo](https://recursospython.com/guias-y-manuales/python-no-se-reconoce-como-un-comando-interno-o-externo/)  
(<https://recursospython.com/guias-y-manuales/python-no-se-reconoce-como-un-comando-interno-o-externo/>)

[Copiar objetos con el módulo estándar «copy»](https://recursospython.com/guias-y-manuales/modulo-estandar-copy/) ▶  
(<https://recursospython.com/guias-y-manuales/modulo-estandar-copy/>)

20 comentarios.



may says:



[septiembre 23, 2022 at 6:58 pm](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-1126>).

pero como se corre el juego?, para ver que funciona, hice el codigo pero al abrirlo en un buscador me da el codigo de vuelta, deberia de funcionar no?

[Responder](#)



**Recursos Python** says:

[septiembre 24, 2022 at 2:18 am](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-1127>).

Hola. Tenés que ejecutar el código con el intérprete de Python, escribiendo `python tetris.py` en la terminal (o `py tetris.py` en Windows). Igualmente sería bueno que leas primero un [tutorial de Python](https://tutorial.recursospython.com/) (<https://tutorial.recursospython.com/>) para aprender esas cosas.

Saludos

[Responder](#)



**Mario** says:

[marzo 28, 2022 at 9:17 pm](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-1057>).

ME SALE ESTOS ERRORES.

Traceback (most recent call last):

File «C:/Users/mario/OneDrive/Documentos/Juego.py», line 453, in  
main()

File «C:/Users/mario/OneDrive/Documentos/Juego.py», line 389, in main



```
next_block_text = font.render(
```

UnboundLocalError: local variable 'font' referenced before  
assignment

[Responder](#)



**Recursos Python** says:

marzo 28, 2022 at 9:24 pm

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-1058>).

Hola. Te falta el archivo `Roboto-Regular.ttf` en la carpeta donde tenés el código. Está disponible en la descarga al principio del artículo.

[Responder](#)



**Juan** says:

marzo 24, 2022 at 10:28 am

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-1055>).

Este programa no me va, se hace desde el Turtle o el normal?

[Responder](#)



**Recursos Python** says:

marzo 24, 2022 at 12:23 pm

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-1056>).

Hola. No se usa el módulo `turtle` aquí. Se usa PyGame.

Saludos



[Responder](#)

---



**Angel** says:

[mayo 29, 2021 at 3:59 pm](#)

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-923>)

Me sale un error «UnboundLocalError: local variable 'font' referenced before assignment» lo habro desde el IDLE y desde el archivo y no funciona

[Responder](#)

---



**Recursos Python** says:

[mayo 29, 2021 at 11:32 pm](#)

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-925>)

Hola, ¿tenés el archivo Roboto-Regular.ttf en el lugar desde el cual estás ejecutando el juego?

Saludos

[Responder](#)

---



**Kevin Gutierrez** says:

[agosto 23, 2021 at 11:19 pm](#)

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-960>)

Tengo el mismo problema, ¿Cómo descargo el archivo roboto-regular?

[Responder](#)

---





## Recursos Python

says:

[agosto 24, 2021 at 4:19 pm](#)

<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-961>

Viene junto con la descarga del código al principio del artículo:  
<https://www.recursospython.com/wp-content/uploads/2018/06/tetris.zip>  
(<https://www.recursospython.com/wp-content/uploads/2018/06/tetris.zip>).

Saludos

[Responder](#)



**Rafael** says:

[marzo 23, 2021 at 5:45 am](#)

<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-870>

Perfecto, es genial poder comparar trabajos con códigos como este

[Responder](#)



**sebastian cano** says:

[diciembre 4, 2019 at 2:29 am](#)

<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-607>

no me importa pygame, intento con import pygame y me aparece que no module name pygame, y ya lo tengo instalado.

ayuda por favor.





**Recursos Python** says:

[diciembre 4, 2019 at 3:01 pm](#)

<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-608>

Hola. Lo más probable es que tengas varias versiones de Python instaladas, y que pip te haya instalado PyGame en una versión diferente a la que estás usando para ejecutar el código.

[Responder](#)

---



**MIRIAM** says:

[noviembre 7, 2019 at 5:39 pm](#)

<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-596>

Como se importa la libreria de pygame, ayudaaa

[Responder](#)

---



**Recursos Python** says:

[noviembre 8, 2019 at 10:50 pm](#)

<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-597>

Primero tenés que instalarla vía `pip install pygame`.

[Responder](#)

---



**Felipe** says:



[octubre 3, 2019 at 9:38 am](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-583>)

te tomare el codigo! para hacer unas pruebas de ML.

SALUDOS! proxicamente mi post en [feedingthemachine.cl](http://feedingthemachine.cl)

[Responder](#)



**Recursos Python** says:

[octubre 4, 2019 at 9:42 am](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-584>)

Hola Felipe. Se agradece en ese caso que dejes un enlace a este post también.

Saludos

[Responder](#)



**Ruben** says:

[marzo 22, 2019 at 10:06 pm](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-535>)

Saludos, genial esta solución. Llevare el código a GitHub para hacer algunas modificaciones. Gracias

[Responder](#)



**gustavo** says:

[julio 10, 2018 at 6:25 pm](#)  
(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-431>)

fabuloso !!; para mi que estoy en los primeros pasos es material de estudio invariable. gracias por tan magnífico aporte

[Responder](#)



**Recursos Python** says:

julio 10, 2018 at 8:22 pm

(<https://recursospython.com/codigos-de-fuente/tetris-pygame/#comment-432>)

Me alegro que te haya servido Gustavo. Un saludo.

[Responder](#)

## Deja una respuesta

Comentario \*

Nombre \*

Email \*

**Publicar el comentario**

© 2013 - 2023

¡Suscríbete a nuestra newsletter! (<https://www.recursospython.com/newsletter/>)

[Políticas de Uso y Privacidad \(https://www.recursospython.com/politicas-de-uso-y-privacidad/\)](https://www.recursospython.com/politicas-de-uso-y-privacidad/)

En inglés: [Python Assets \(https://pythonassets.com/\)](https://pythonassets.com/)

