### Permitir el movimiento continuo

Cuando el jugador mantenga pulsada la tecla de flecha derecha, queremos que la nave siga moviéndose hasta que el jugador suelte la tecla. Haremos que el juego detecte un evento pygame.KEYUP para saber cuándo se suelta la tecla de la flecha derecha; luego utilizaremos los eventos KEYDOWN y KEYUP junto con una bandera llamada moving\_right para implementar el movimiento continuo.

Cuando la bandera moving\_right sea False, la nave estará inmóvil. Cuando el jugador pulse la tecla de flecha derecha, pondremos la bandera a True, y cuando el jugador suelte la tecla, volveremos a poner la bandera a False.

La clase Ship controla todos los atributos de la nave, así que le daremos un atributo llamado moving\_right y un método update() para comprobar el estado de la bandera moving\_right. El método update() cambiará la posición de la nave si la bandera se establece en True. Llamaremos a este método una vez en cada pasada por el bucle while para actualizar la posición de la nave.

Estos son los cambios en Ship:

**ship.py**

class Ship:  
 """A class to manage the ship."""  
  
 def \_\_init\_\_(self, ai\_game):  
 --snip--  
 # Start each new ship at the bottom center of the screen.  
 self.rect.midbottom = self.screen\_rect.midbottom  
  
 # Movement flag; start with a ship that's not moving.  
❶ self.moving\_right = False  
  
❷ def update(self):  
 """Update the ship's position based on the movement flag."""  
 if self.moving\_right:  
 self.rect.x += 1  
  
 def blitme(self):  
 --snip--

Añadimos un atributo self.moving\_right en el método \_\_init\_\_() y lo fijamos inicialmente en False ❶. Después añadimos update(), que mueve el barco a la derecha si la bandera es True ❷. El método update() será llamado desde fuera de la clase, por lo que no se considera un método ayudante.

Ahora tenemos que modificar \_check\_events() para que moving\_right se convierta en True cuando se pulse la tecla de flecha derecha y en False cuando se suelte la tecla:

**alien\_invasion.py**

def \_check\_events(self):  
 """Respond to keypresses and mouse events."""  
 for event in pygame.event.get():  
 --snip--  
 elif event.type == pygame.KEYDOWN:  
 if event.key == pygame.K\_RIGHT:  
❶ self.ship.moving\_right = True  
❷ elif event.type == pygame.KEYUP:  
 if event.key == pygame.K\_RIGHT:  
 self.ship.moving\_right = False

Aquí modificamos cómo responde el juego cuando el jugador pulsa la tecla de flecha derecha: en lugar de cambiar directamente la posición de la nave, nos limitamos a establecer moving\_right en True ❶. Luego añadimos un nuevo bloque elif, que responde a los eventos KEYUP ❷. Cuando el jugador suelta la tecla de flecha derecha (K\_RIGHT), establecemos moving\_right en False.

A continuación, modificamos el bucle while en run\_game() para que llame al método update() de la nave en cada pasada por el bucle:

**alien\_invasion.py**

def run\_game(self):  
 """Start the main loop for the game."""  
 while True:  
 self.\_check\_events()  
 self.ship.update()  
 self.\_update\_screen()  
 self.clock.tick(60)

La posición de la nave se actualizará después de comprobar los eventos del teclado y antes de actualizar la pantalla. Esto permite que la posición de la nave se actualice en respuesta a la entrada del jugador y garantiza que la posición actualizada se utilizará al dibujar la nave en la pantalla.

Cuando ejecutes *alien\_invasion.py* y mantengas pulsada la tecla de flecha derecha, la nave debería moverse continuamente hacia la derecha hasta que sueltes la tecla.

[anterior](c12_20.html)[Subtema 21 de 43: (Ver todo)](c12.html)[siguiente](c12_22.html)