### Refactorización de \_actualizar\_balas()

Vamos a refactorizar \_update\_bullets() para que no haga tantas tareas diferentes. Moveremos el código para tratar las colisiones bala-alien a un método aparte:

**alien\_invasion.py**

def \_update\_bullets(self):  
 --snip--  
 # Get rid of bullets that have disappeared.  
 for bullet in self.bullets.copy():  
 if bullet.rect.bottom <= 0:  
 self.bullets.remove(bullet)  
  
 self.\_check\_bullet\_alien\_collisions()  
  
 def \_check\_bullet\_alien\_collisions(self):  
 """Respond to bullet-alien collisions."""  
 # Remove any bullets and aliens that have collided.  
 collisions = pygame.sprite.groupcollide(  
 self.bullets, self.aliens, True, True)  
  
 if not self.aliens:  
 # Destroy existing bullets and create new fleet.  
 self.bullets.empty()  
 self.\_create\_fleet()

Hemos creado un nuevo método, \_check\_bullet\_alien\_collisions(), para buscar colisiones entre balas y alienígenas y responder adecuadamente si toda la flota ha sido destruida. Hacerlo así evita que \_update\_bullets() crezca demasiado y simplifica el desarrollo posterior.

[anterior](c13_22.html)[Subtema 23 de 33: (Ver todo)](c13.html)[siguiente](c13_24.html)