Aprende a programar con Minecraft

### **Crear varios objetos**

Puedes crear varios objetos a partir de la misma clase creando objetos con nombres diferentes utilizando el mismo constructor de clase (recuerda que *constructor* es otro nombre para el método \_\_init\_\_() ). Por ejemplo, supongamos que encontramos una segunda gata llamada Stella que ahora es amiga de Fluff. Abre *catClass.py* e introduce el siguiente código para añadir a Stella:

*catClass.py*

class Cat(object):  
def \_\_init\_\_(self, name, weight):  
self.name = name  
self.weight = weight  
  
fluff = Cat("Fluff", 4.5)  
stella = Cat("Stella", 3.9)

Ahora tenemos dos objetos gato, Pelusa y Stella. Cada uno tiene los mismos atributos, nombre y peso, pero con valores diferentes.

Añade el siguiente código a *catClass.* py para imprimir los nombres de los gatos:

*catClass.py*

print(fluff.name)  
print(stella.name)

Cuando ejecutes el archivo, obtendrás esta salida:

Fluff  
Stella

Los dos objetos gato también tienen acceso a los mismos métodos. Ambos pueden llamar a la función comer(). Añade este código a *catClass.*py:

*catClass.py*

fluff.eat("tuna")  
stella.eat("cake")

Y la salida tendrá este aspecto:

Fluff is eating tuna  
Stella is eating cake

Escribir una clase facilita mucho la creación de muchos objetos. ¡Intentemos crear varios objetos con Minecraft!

#### **Misión nº 71: Ciudad Fantasma**

¿Qué da más miedo que una casa fantasma? Exacto, dos casas fantasma. Pero tres casas fantasma darían aún más miedo. ¿Y más de tres casas fantasma? Tengo que dejar de pensar en esto, ¡o no podré dormir esta noche!

En la Misión 69[(página 267](ch12.xhtml#page_267)), creaste una clase que construye una casa que desaparece. Ahora puedes crear varios objetos utilizando la misma clase, y Python recordará cada uno de los atributos y métodos del objeto. Puedes hacer tantas casas como quieras, y puedes hacer que aparezcan y desaparezcan con facilidad.

Tu misión es crear cuatro o más objetos casa fantasma y disponerlos en un pueblo. Al cabo de cierto tiempo, haz que desaparezcan todas y reaparezcan en otro lugar del mapa, como en un verdadero pueblo fantasma.

Abre *ghostHouse.py* en IDLE: lo utilizaremos como base. Cuando hayas creado una casa en el programa *ghostHouse.* py, tu código debería tener este aspecto:

*ghostHouse.py*

ghostHouse = Building(17, 22, -54, 10, 6, 8)  
ghostHouse.build()  
  
time.sleep(30)  
  
ghostHouse.clear()

Guarda *ghostHouse*.py como un nuevo archivo llamado *ghostVillage.py*, y luego crea tres o más objetos en el archivo utilizando la clase Edificio para construir el pueblo. Para ayudarte a empezar, en [el Listado 12-4](ch12.xhtml#ch12ex4) he creado un segundo objeto llamado tienda. También he establecido las variables x, y y z para que contengan la posición actual del jugador, que encontraremos utilizando player.getTilePos(). Esto facilita la construcción de la aldea a tu alrededor.

*ghostVillage.py*

pos = mc.player.getTilePos()  
x = pos.x  
y = pos.y  
z = pos.z  
ghostHouse = Building(x, y, z, 10, 6, 8)  
shop = Building(x + 12, y, z, 8, 12, 10)  
# Create more ghost building objects here  
  
ghostHouse.build()  
shop.build()  
# Build more ghost building objects here  
  
time.sleep(30)  
  
ghostHouse.clear()  
shop.clear()

*Listado 12-4: Creación de múltiples objetos de construcción fantasma*

[La Figura 12-4](ch12.xhtml#ch12fig4) muestra mi aldea fantasma. Después de 30 segundos, los edificios fantasma desaparecen de repente.



*Figura 12-4: ¡Mira todos los edificios fantasma del pueblo fantasma!*

[anterior](ch12_5.html)[Subtema 6 de 10: (Ver todo)](ch12.html)[siguiente](ch12_7.html)