### La clase Coche

Escribamos una nueva clase que represente a un coche. Nuestra clase almacenará información sobre el tipo de coche con el que estamos trabajando, y tendrá un método que resuma esta información:

**car.py**

class Car:  
 """A simple attempt to represent a car."""  
  
❶ def \_\_init\_\_(self, make, model, year):  
 """Initialize attributes to describe a car."""  
 self.make = make  
 self.model = model  
 self.year = year  
  
❷ def get\_descriptive\_name(self):  
 """Return a neatly formatted descriptive name."""  
 long\_name = f"{self.year} {self.make} {self.model}"  
 return long\_name.title()  
  
❸ my\_new\_car = Car('audi', 'a4', 2024)  
print(my\_new\_car.get\_descriptive\_name())

En la clase Car, definimos el método \_\_init\_\_() con el parámetro self primero ❶, igual que hicimos con la clase Dog. También le damos otros tres parámetros: make, model, y year. El método \_\_init\_\_() toma estos parámetros y los asigna a los atributos que se asociarán a las instancias creadas a partir de esta clase. Cuando creemos una nueva instancia de Car, tendremos que especificar una marca, un modelo y un año para nuestra instancia.

Definimos un método llamado get\_descriptive\_name() ❷ que pone los datos de un coche year, make, y model en una sola cadena que describe claramente el coche. Esto nos evitará tener que imprimir individualmente el valor de cada atributo. Para trabajar con los valores de los atributos en este método, utilizamos self.make, self.model, y self.year. Fuera de la clase, creamos una instancia de la clase Car y la asignamos a la variable my\_new\_car ❸. Luego llamamos a get\_descriptive\_name() para mostrar qué clase de coche tenemos:

2024 Audi A4

Para hacer la clase más interesante, añadamos un atributo que cambie con el tiempo. Añadiremos un atributo que almacene el kilometraje total del coche.

[anterior](c09_7.html)[Subtema 8 de 34: (Ver todo)](c09.html)[siguiente](c09_9.html)