Constructores

Contenidos

Introducción

al indice

Veíamos en la sesión anterior que cuando creamos un objeto de la clase *Coche* sin más, este objeto adquiere los atributos por defecto de la clase. Si queremos definirlo bien "de primeras" (en su instanciación o creación) para diferenciarlo de otros coches, necesitamos algo más.

Ese algo más que me permite la definición inicial, es lo que se denomina constructor de la clase.

Será un método, muy similar a una función, al que daremos unos argumentos de entrada que sustituirán a los valores por defecto y que nos permitirán diferenciar esa instancia de otras instancias de la misma clase.

¿Cómo se define un constructor? Mediante la función `__init__`, dentro de la clase.

In [1]:
```
class Coche:
    puertas = 4
    ruedas = 4

    def __init__(self, marca_coche):
        self.marca = marca_coche
```

In [2]:
```
tesla = Coche()
```

```
---------------------------------------------------------------------------
TypeError                                 Traceback (most recent call last)
Cell In[2], line 1
----> 1 tesla = Coche()

TypeError: __init__() missing 1 required positional argument: 'marca_coche'
```

In [3]:
```
tesla = Coche("Tesla")
print(tesla.marca)
```

```
Tesla
```

En la declaración del constructor hemos metido la palabra `self`. **Lo tendremos que poner siempre**. Hace referencia a la propia instancia de coche, es decir, a cuando creemos coches nuevos.

En este caso estamos diferenciando los atributos comunes de la clase *Coche*, de los atributos particulares de los coches, como por ejemplo, la marca. Por eso la marca va junto con `self`, porque no hace referencia a la clase genércia de coche, sino a cada coche que creemos.

In [4]:
```
mercedes = Coche("Mercedes")
renault = Coche("Renault")
print(mercedes.marca)
print(renault.marca)
print(mercedes.puertas)
print(renault.puertas)
```

```
Mercedes
Renault
4
4
```

Ahora ya podemos diferenciar los coches por su marca. Para acceder al atributo de la marca, lo hacemos igual que con los anteriores.

In [4]:
```
mercedes = Coche("Mercedes")
renault = Coche("Renault")
print(mercedes.marca)
print(renault.marca)
print(mercedes.puertas)
print(renault.puertas)
```

```
Mercedes
Renault
4
4
```

Ya podemos solucionar el tema de que no todos los coches tienen 4 puertas

In [5]:
```
class Coche:
    puertas = 4
    ruedas = 4

    def __init__(self, marca_coche, num_puertas):
        self.marca = marca_coche
        self.puertas =num_puertas
```

In [6]:
```
tesla = Coche("Tesla", 6)
mercedes_500 = Coche("Mercedes",8)
```

In [8]:
```
print(tesla.puertas)
```

```
6
```

In [9]:
```
class Coche:

    def __init__(self, marca_coche, num_puertas = 4, num_ruedas = 4):
        self.marca = marca_coche
        self.puertas =num_puertas
        self.ruedas = num_ruedas
```

In [10]:
```
tesla_4 = Coche("Tesla")
tesla_6r = Coche("Tesla", num_ruedas = 6)
```

In [11]:
```
print(tesla_4.puertas, tesla_6r.puertas)
print(tesla_4.ruedas, tesla_6r.ruedas)
```

```
4 4
4 6
```