SESIÓN 19 · MÓDULO 2

Atributos de clase, `__str__` y herencia

Compartir datos entre todos los objetos de una clase, darles una representación legible y reutilizar una clase con la herencia.

Antonio Chanes Espigares · CC BY-NC-ND 4.0

Hoy

0:00	Corrección de la tarea 18.
0:15	Situación: un formulario de matrícula general y el de una beca, que es el mismo con dos casillas más.
0:25	**Predice**. El `__str__` de `Animal` llama al `habla` de cada hijo: primer contacto con el polimorfismo.
0:45	Construcción: atributos de clase, `__str__`, herencia, `super()`.
1:20	Ejercicios del capítulo (secciones 3 y 4).
1:45	Preguntas tipo examen y tarea 19.

Para empezar

Un formulario de matrícula general y el de una beca, que es el mismo con dos casillas más.

Predice

Escribe en papel qué crees que pasará. Después, lo ejecutamos.

```python
class Animal:
    patas = 4

    def habla(self):
        return "..."

    def __str__(self):
        return f"{type(self).__name__} que dice
{self.habla()}"

class Perro(Animal):
    def habla(self):
        return "guau"

class Gallina(Animal):
    patas = 2

for a in [Perro(), Gallina()]:
    print(a, a.patas)
```

Predice

Escribe en papel qué crees que pasará. Después, lo ejecutamos.

```python
class Animal:
    patas = 4

    def habla(self):
        return "..."

    def __str__(self):
        return f"{type(self).__name__} que dice
{self.habla()}"

class Perro(Animal):
    def habla(self):
        return "guau"

class Gallina(Animal):
    patas = 2

for a in [Perro(), Gallina()]:
    print(a, a.patas)
```

LO QUE SALE

`Perro que dice guau 4` ·

`Gallina que dice ... 2`

Cada objeto usa el `habla` y el `patas` más cercano en su clase.

Atributos de clase y `__str__`

Clases y objetos · 3

```
class Cuenta:
    banco = "Banco AC"                  # atributo de clase: común a todas

    def __init__(self, titular, saldo):
        self.titular = titular          # atributos de instancia: propios de cada una
        self.saldo = saldo

    def __str__(self):
        return f"{self.titular}: {self.saldo} € ({Cuenta.banco})"

a = Cuenta("Ana", 130)
print(a)
print(a.__dict__)                       # los atributos de instancia de a
…
```

Herencia

Clases y objetos · 4

```
class Cuenta:
    def __init__(self, titular, saldo):
        self.titular = titular
        self.saldo = saldo

    def ingresar(self, cantidad):
        self.saldo = self.saldo + cantidad

class CuentaAhorro(Cuenta):             # hereda de Cuenta
    def __init__(self, titular, saldo, interes):
        super().__init__(titular, saldo)    # reutiliza el __init__ de Cuenta
        self.interes = interes

…
```

Manos a la obra

Abre el cuaderno de la sesión en Colab, con los ejemplos de hoy y estos ejercicios:

1. Añade un método
2. ¿Qué imprime?
3. Un rectángulo. Escribe una clase Rectangulo con atributos base y…



Cuaderno en Colab

Tarea 19 · Figuras

1. Crea la clase `Figura` con un método `area()` que lance `NotImplementedError` y un `__str__` que escriba el nombre de…
2. Crea `Rectangulo(base, altura)` y `Circulo(radio)`, hijas de `Figura`, cada una con su `area()` y su `perimetro()`.
3. Haz una lista con dos rectángulos y dos círculos, ordénala por área con `sort` y `key`, e imprime cada figura.
4. ¿Por qué no hace falta escribir `__str__` en las hijas?

COMPROBACIÓN La figura más pequeña de la lista `[Rectangulo(3, 4), Circulo(1), Rectangulo(1, 1), Circulo(2)]` es el rectángulo de 1 × 1.

Antes de irte

Las **preguntas tipo examen** del capítulo, en directo: se corrigen al momento.



Preguntas del capítulo

LA PRÓXIMA SESIÓN

Sesión 20 · Nombres ocultos y polimorfismo