ProfesorenVideo.com

Curso de Python

```
class  ClaseDePrueba:
    atributo_clase_1
    atributo_clase_2

    def   __init__(self):
        atributo_instancia_1 = ...
        atributo_instancia_2 = ...

    def  metodo( self ) :
        self.atributo_instancia_1 = ......
        self.atributo_instancia_2 = ......
```

Atributos de clase

Método constructor

Atributos de instancia

Métodos de instancia

ProfesorenVideo.com

Curso de Python

Tipos de MÉTODOS:

- Método de instancia: Es un método que opera en instancias individuales (objetos) de la clase, y toma el parámetro ‘**self**‘ como su primer parámetro.

- Método de clase: Es un método que opera sobre los atributos de clase y toma el parámetro ‘**cls**‘ (clase) como su primer parámetro.

- Método_estático: Es un método que no toma ni **self** ni **cls** como su primer parámetro y no opera sobre atributos de instancia ni atributos de clase.

ProfesorenVideo.com

Curso de Python

Decorador: El término decorador tiene dos acepciones, una general y otra más específica:

- General: Es la forma de definir una función que toma como parámetro a otra función y le modifica su comportamiento. (Este concepto no se abordará por ahora).

- Específica: Es la forma de definir métodos que modifican tanto los atributos de instancia como los de clase.

ProfesorenVideo.com

Curso de Python

Decorador: En la acepción específica dijimos que usamos decoradores para definir métodos que acceden tanto los atributos de instancia como los de clase.

Decoradores para atributos de instancia:

- @property : Se usa para definir un método getter que permite acceder a un atributo de instancia.

- @<nombre>.setter : Se usa para definir un método setter que permite modificar el valor de un atributo de instancia.

- @<nombre>.deleter : Se usa para definir un método deleter que permite eliminar un atributo de instancia.

ProfesorenVideo.com

Curso de Python

Los decoradores @property , @<nombre>setter , @<nombre>deleter:

Cuando se usan los decoradores anteriormente mencionados, los métodos NO deben ser invocados con los paréntesis "( )", sino que se comportan como si fueran atributos, es decir no se le colocan paréntesis.

Ejemplos:

- En vez de colocar: objeto.metodo(parametros) , se coloca: objeto.metodo = parametros

- Si queremos mostrar, en vez de: print(objeto.metodo()), hacemos: print(objeto.metodo)

ProfesorenVideo.com

Curso de Python

Decoradores @property , @<nombre>setter , @<nombre>deleter:

Un decorador define cómo se accede, cómo se configura o cómo se elimina un atributo de una clase en Python.

- Se usa el decorador @property para acceder a los atributos de una clase y ponerlos a disposición para que sean usados.

- Se usa el decorador @<nombre>.setter para modificar los atributos de una clase.

- Se usa el deorador @<nombre>.deleter para eliminar los atributos de una clase.

ProfesorenVideo.com

Curso de Python

Decorador: En la acepción específica dijimos que usamos decoradores para definir métodos que acceden tanto los atributos de instancia como los de clase.

Decoradores para atributos de clase:

- @classmethod : Se usa para definir un método de clase que opera en la clase en lugar de en las instancias de la clase. Este método puede acceder y modificar atributos de clase. Estos métodos reciben la clase como su primer argumento (cls) en lugar de la instancia de la clase (self). Se invocan utilizando la sintaxis **Clase.metodo()**.

Curso de Python

Basta de teoría, mejor programemos …

… pero después volvemos