Encapsulamiento

La característica de no acceder o modificar los valores de los atributos directamente y utilizar métodos para ello lo llamamos **encapsulamiento**.

```
>>> class Alumno():
...def __init__(self,nombre=""):
...    self.nombre=nombre
...    self.__secreto="asdasd"
...
>>> a1=Alumno("jose")
>>> a1.__secreto
Traceback (most recent call last):
  File "<stdin>", line 1, in <module>
AttributeError: 'Alumno' object has no
attribute '__secreto'
```

Las variables que comienzan por un doble guión bajo __ la podemos considerar como **atributos privados**.

Encapsulamiento

- En Python, las **propiedades (getters)** nos permiten implementar la funcionalidad exponiendo estos métodos como atributos.
- Los métodos **setters** son métodos que nos permiten modificar los atributos a través de un método.

```
class circulo():
    def __init__(self,radio):
      self.radio=radio

    @property
    def radio(self):
      print("Estoy dando el radio")
      return self.__radio

    @radio.setter
    def radio(self,radio):
      if radio>=0:
           self.__radio = radio
      else:
           raise ValueError("Radio positivo")
           self.__radio=0
```

```
>>> c1=circulo(3)
>>> c1.radio
Estoy dando el radio
3
>>> c1.radio=4
>>> c1.radio=-1
Radio debe ser positivo
>>> c1.radio
0
```