Статический метод класса (@staticmethod)

In [62]:
```
class Human:

    def __init__(self, name, age=0):
        self.name = name
        self.age = age

    @staticmethod
    def is_age_valid(age):
        return 0 < age < 150
```

In [63]:
```
# можно обращаться от имени класса
Human.is_age_valid(35)
```

Out[63]: True

In [67]:
```
# или от экземпляра:
human = Human("Old Bobby")
human.is_age_valid(234)
```

Out[67]: False

Вычисляемые свойства класса (property)

In [101]:
```
class Robot:

    def __init__(self, power):
        self.power = power
```

In [102]:
```
wall_e = Robot(100)
wall_e.power = 200
print(wall_e.power)
```

```
200
```

In [103]:
```
wall_e.power = -20
```

In [ ]:
```
class Robot:

    def __init__(self, power):
        self.power = power

    def set_power(self, power):
        if power < 0:
            self.power = 0
        else:
            self.power = power
```

In [109]:
```
wall_e = Robot(100)
wall_e.set_power(-20)
print(wall_e.power)
```

```
0
```

In [142]:
```
class Robot:

    def __init__(self, power):
        self._power = power

    power = property()

    @power.setter
    def power(self, value):
        if value < 0:
            self._power = 0
        else:
            self._power = value

    @power.getter
    def power(self):
        return self._power

    @power.deleter
    def power(self):
        print("make robot useless")
        del self._power
```

In [143]:
```
wall_e = Robot(100)
wall_e.power = -20
print(wall_e.power)
```

```
0
```

In [144]:
```
del wall_e.power
```

```
make robot useless
```

In [148]:
```
class Robot:
    def __init__(self, power):
        self._power = power

    @property
    def power(self):
        # здесь могут быть любые полезные вычисления
        return self._power
```

In [149]:
```
wall_e = Robot(200)
wall_e.power
```

Out[149]: 200

В этом видео:

- Узнали, что такое статический метод (@staticmethod)
- Узнали, что такое свойство класса (@property)

In [ ]: