08-POO-heranca_multipla

March 3, 2020

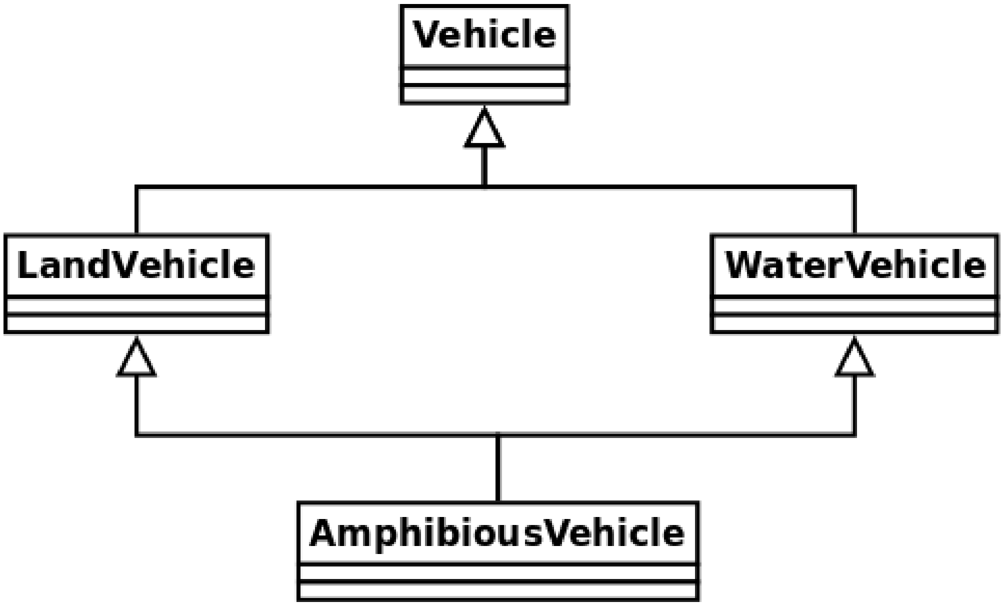
1 Programação Orientada aos Objetos (POO) - parte VIII

Pedro Cardoso
(ISE/UAlg - pcardoso@ualg.pt)

A herança múltipla ocorre sempre que uma subclasse possui duas ou mais superclasses imediatas, ou seja, é "filha" de mais de uma classe.

```
In [1]: class Vehicle:
            def __init__(self, owner, brand):
                self.owner = owner
                self.brand = brand

            def vehicle_info(self):
                raise NotImplementedError("vehicle_info: não implementado")

            @property
            def owner(self):
                return self.__owner

            @owner.setter
            def owner(self, owner):
                self.__owner = owner

            @property
            def brand(self):
                return self.__brand

            @brand.setter
            def brand(self, brand):
                self.__brand = brand
```

```
In [2]: class LandVehicle(Vehicle):
            def __init__(self, owner, brand, land_velocity):
                print(super(LandVehicle, self))
                Vehicle.__init__(self, owner, brand)
                self.land_velocity = land_velocity
```



alt text

```
            @property
            def land_velocity(self):
                return self.__land_velocity

            @land_velocity.setter
            def land_velocity(self, lv):
                self.__land_velocity = lv

In [3]: class WaterVehicle(Vehicle):
            def __init__(self, owner, brand, water_velocity):
                print(super(WaterVehicle, self))
                Vehicle.__init__(self, owner, brand)
                self.water_velocity = water_velocity

            @property
            def water_velocity(self):
                return self.__water_velocity

            @water_velocity.setter
            def water_velocity(self, wv):
                self.__water_velocity = wv
```

e a classe `AmphibiousVehicle` deriva das anteriores

```
In [4]: class AmphibiousVehicle(LandVehicle, WaterVehicle):
            def __init__(self, owner, brand, land_velocity, water_velocity):
                LandVehicle.__init__(self, owner, brand, land_velocity)
                WaterVehicle.__init__(self, owner, brand, water_velocity)
```

```
        def print_info(self):
            print(f'Isto é um veiculo anfíbio cujo dono é {self.owner} da marca {self.branc
```

```
In [5]: a = AmphibiousVehicle('Margarida', 'rinspeed splash', 199, 38)
        a.print_info()
```

```
<super: <class 'LandVehicle'>, <AmphibiousVehicle object>>
<super: <class 'WaterVehicle'>, <AmphibiousVehicle object>>
Isto é um veiculo anfíbio cujo dono é Margarida da marca rinspeed splash e anda a 199 km/h em t
```

A ordem de resolução dos métodos (Method Resolution Order - mro) pode ser vista no atributo
__mro__

```
In [6]: AmphibiousVehicle.__mro__
```

```
Out[6]: (__main__.AmphibiousVehicle,
         __main__.LandVehicle,
         __main__.WaterVehicle,
         __main__.Vehicle,
         object)
```