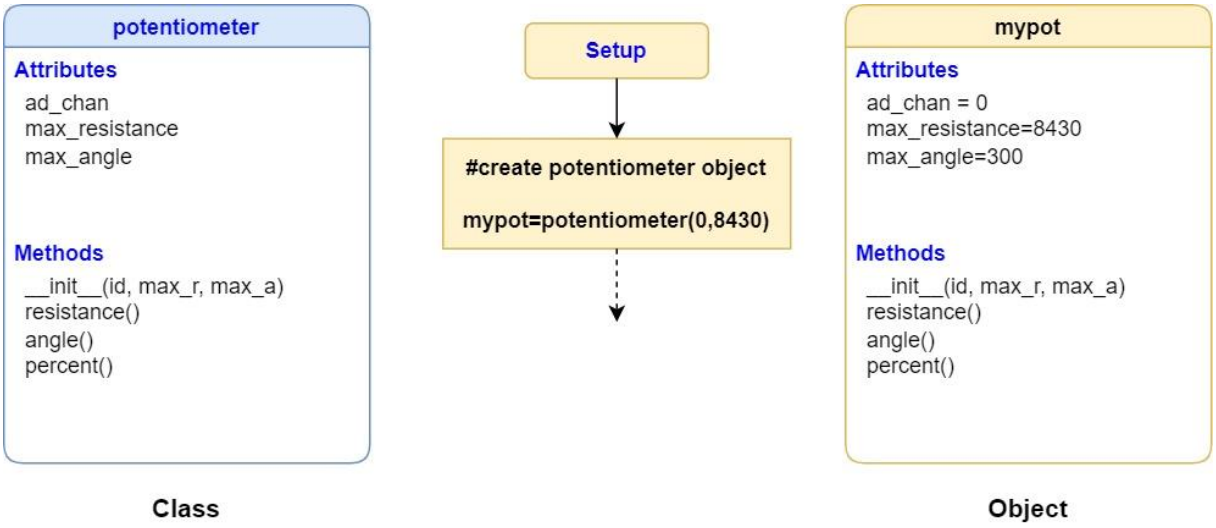
Object-oriented programming basics.

In this kind of programming, data and functions are "embedded" in a unique structure call class. In OO programming data structures are called attributes, functions are called methods.

An object is an instantiation of class with its own variables.

When an object is created from a class, a specific function called a "constructor" is called. In python the name of the constructor is __init__( )

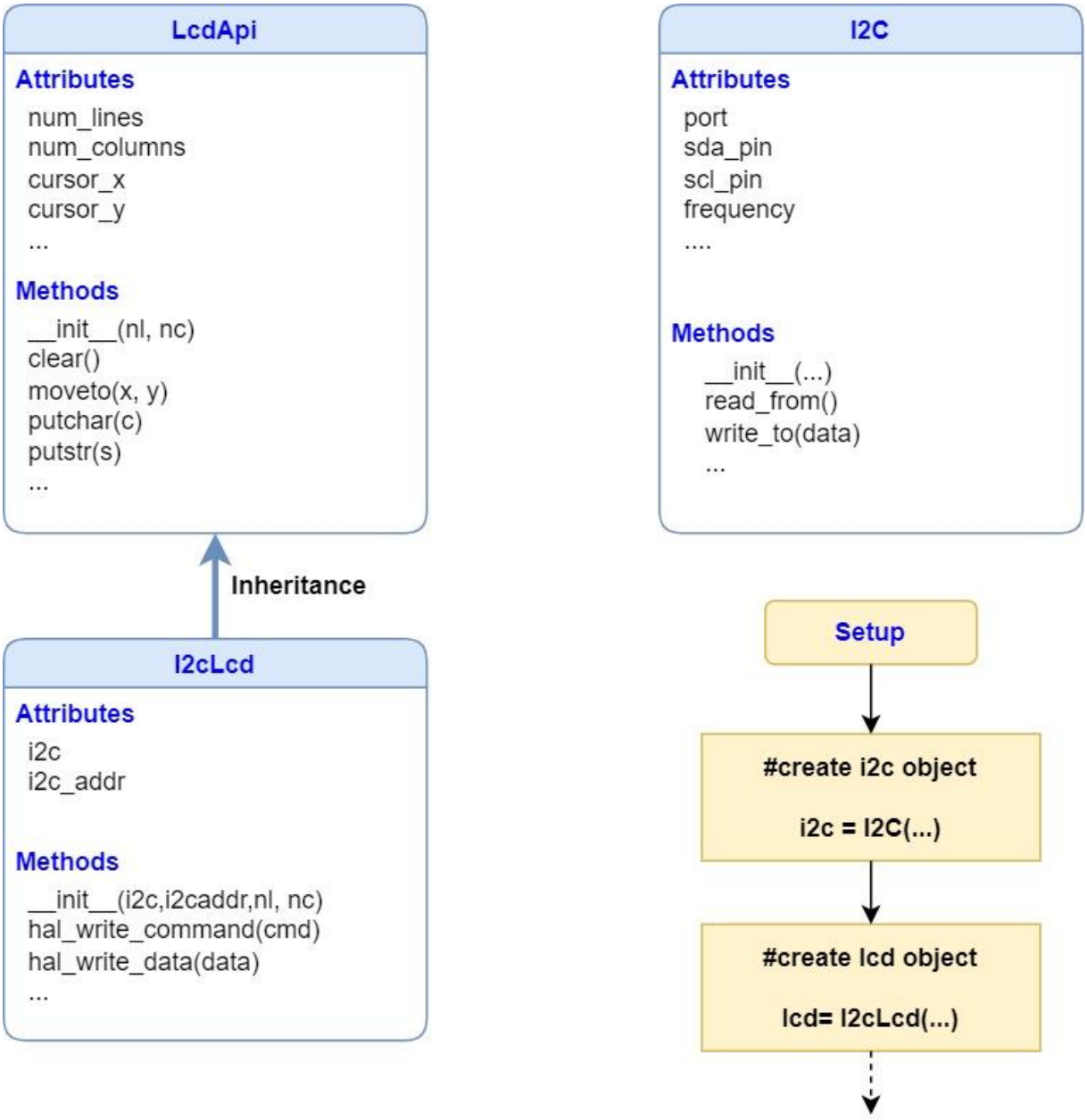
Example: a class for a potentiometer connected on one of the A/D channels of the RPi Pico.




```
class potentiometer:
    def __init__(self, id, max_r, max_a=300):
         self.ad_chan = ADC(id)
         self.max_resistance = max_r
         self.max_angle = max_a

    def resistance(self):
        return self.ad_chan.read_u16()/65535*self.max_resistance

    def angle(self):
        return self.ad_chan.read_u16()/65535*self.max_angle

    def percent(self):
        return self.ad_chan.read_u16()/65535*100

# creates a potentiometer object on A/D channel 0
# with a max resistance of 8430 Ω
mypot = potentiometer(0,8430)
```

A class can be constructed as a specialisation of another class. In this case it inherits attributes and methods from the parent class. For a complete explanation of classes in Python see: https://realpython.com/python3-object-oriented-programming/.

Classes for the I²C LCD.




```
# creates an I2C object : port = 0, sda = GP8, scl = GP9, 400 kHz clock
i2c = machine.I2C(0, sda=machine.Pin(8), scl=machine.Pin(9), freq=400000)

# creates the lcd object, passing the I2C object, the physical address 0x27
# 2 lines of 16 characters.
lcd = pico_i2c_lcd.I2cLcd(i2c, 0x27, 2, 16)
```