**Staticmethod dan Classmethod**

Dib episode sebelumnya kita sudah membuat encapsulasi untuk object dan sekarang kita akan mencoba membuat encapsulasi untuk class. Berikut cara membuatnya :

Kali ini kita akan mencoba menghitung berapa kali kita menggunakan class hero. Berikut kode yang saya gunakan :

class Hero:

*# private class variabel*

    \_\_jumlah = 0

    def \_\_init\_\_(self, name) -> None:

*self*.name = name

        Hero.\_\_jumlah += 1

*# method ini hanya berlaku untuk object*

    def getJumlah(self):

        return Hero.\_\_jumlah

*# method ini tidak berlaku untuk object karena tidak ada selfnya*

*# tapi berlaku untuk class*

    def getJumlah1():

        return Hero.\_\_jumlah

*# method static ( decorator ) menempel ke object dan class*

*# ( agar bisa berlaku kedua method di atas )*

*# kita menggunakan ini untuk menandakan kalau ini adalah method static*

    @staticmethod

    def getJumlah2():

        return Hero.\_\_jumlah

    @classmethod

    def getJumlah3(cls):

        return *cls*.\_\_jumlah

sniper = Hero("sniper")

print(Hero.getJumlah2())

rikimaru = Hero("rikimaru")

print(sniper.getJumlah2())

drowranger = Hero("droranger")

print(Hero.getJumlah3())

**penjelasan :**

\_\_jumlah = 0

Adalah private variable yang tidak bisa di akses dari luar

    def getJumlah(self):

        return Hero.\_\_jumlah

Kita mencoba menghitung berapa jumlah object yang telah kita buat. Kita bisa memanggilnya dengan cara berikut :

sniper = Hero("sniper")

print(sniper.getJumlah())

akan error jika anda memanggil class nya seperti contoh berikut :

print(Hero.getJumlah())

berikut method yang hanya berlaku untuk class dan tidak untuk object :

    def getJumlah1():

        return Hero.\_\_jumlah

ini terjadi karena tidak ada argument atau parameter yang kita kirimkan. Jadi anda harus memanggil classnya agar tidak error, berikut contoh perintah untuk memanggil classnya :

sniper = Hero("sniper")

print(Hero.getJumlah1())

di atas hanya menepel pada salah satu nya. Tapi jika anda ingin bisa membuatnya menempel pada keduanya ( **class dan object )** anda bisa menggunakan cara berikut :

    @staticmethod

    def getJumlah2():

        return Hero.\_\_jumlah

    @classmethod

    def getJumlah3(cls):

        return *cls*.\_\_jumlah

perbedaan antara keduanya hanya pada parameter atau argument saja. Jadi ketika anda ingin memanggilnya gunakan saja perintah berikut :

rikimaru = Hero("rikimaru")

print(sniper.getJumlah2())

drowranger = Hero("droranger")

print(Hero.getJumlah3())

anda bisa memanggilnya menggunakan class atau object, terserah anda.

def getJumlah3(cls):

        return *cls*.\_\_jumlah

**cls** di atas berarti saya mengganti nama classnya menjadi cls agar bisa menjadi lebih flexsibel