Let’s go pour le **Concept #11 : Les Méthodes de Classe et Méthodes Statique** — deux outils **hyper puissants** pour structurer ton code avec style et efficacité.

**🧠 Concept #11 — Méthodes de classe vs Méthodes statiques**

**🔹 1. Méthode d’instance (rappel)**

Tu connais déjà : une méthode d’**instance** utilise self pour agir **sur un objet précis**.

python

CopyEdit

class Chien:

def aboyer(self): # Méthode d’instance

print("Wouf !")

**🔹 2. Méthode de classe**

👉 Une **méthode de classe** agit sur **la classe elle-même**, pas sur une instance.  
Elle utilise @classmethod et prend **cls** comme premier paramètre.

python

CopyEdit

class Animal:

nb\_animaux = 0

def \_\_init\_\_(self):

Animal.nb\_animaux += 1

@classmethod

def combien(cls):

return f"Il y a {cls.nb\_animaux} animaux."

📌 cls = la **classe** → on peut lire ou modifier les attributs de **classe**.

**🔹 3. Méthode statique**

👉 Une **méthode statique** ne prend **ni self, ni cls**.  
Elle est liée **logiquement** à la classe, mais **n’utilise pas l’objet ni la classe**.

python

CopyEdit

class Calculatrice:

@staticmethod

def addition(a, b):

return a + b

🧠 C’est comme une **fonction utilitaire** regroupée dans la classe par logique.

**🧩 Tableau comparatif :**

| **Type** | **Décorateur** | **1er paramètre** | **Sert à…** |
| --- | --- | --- | --- |
| Méthode d’instance | *(aucun)* | self | Manipuler l’objet |
| Méthode de classe | @classmethod | cls | Accéder à la classe |
| Méthode statique | @staticmethod | aucun | Faire une action liée mais indépendante |