On poursuit avec un concept qui donne **structure et puissance** à tes architectures orientées objet :

**🧱 Concept #14 — L’Abstraction avec Classes Abstraites**

**🧠 1. Définition :**

L’**abstraction** en POO consiste à **définir un modèle commun** sans le concrétiser.  
Tu crées des **classes de base** que **d’autres classes devront compléter**.

**➕ En Python, ça se fait avec :**

python

CopyEdit

from abc import ABC, abstractmethod

* ABC = Abstract Base Class (classe abstraite)
* @abstractmethod = méthode **obligatoire à implémenter** dans les sous-classes

**📘 Exemple simple :**

python

CopyEdit

from abc import ABC, abstractmethod

class Animal(ABC):

@abstractmethod

def parler(self):

pass

class Chien(Animal):

def parler(self):

print("Wouf !")

rex = Chien()

rex.parler()

✅ Impossible de faire a = Animal() → car la méthode n’est **pas définie** !

**💡 Pourquoi l’utiliser ?**

* ✨ Forcer une **structure** commune à toutes les sous-classes
* 💡 Faciliter la **maintenance** du code
* 📐 Idéal pour les **grandes architectures** (ex : jeux, applications pro...)