Tu avances super bien — let’s keep the momentum!  
On passe maintenant au **Concept #6 : Les Méthodes Spéciales** – le secret pour rendre tes objets **intelligents**, **lisibles** et parfois même **magiques** ✨

**🧠 Concept #6 — Les Méthodes Spéciales**

**🔹 1. Définition :**

Les **méthodes spéciales** (ou "magic methods") sont des **fonctions prédéfinies** de Python, **entourées de double underscores** (\_\_), qui donnent des **pouvoirs spéciaux** à tes objets.

Elles permettent par exemple :

* D’afficher joliment un objet
* De le comparer (==, <, etc.)
* De le transformer en string
* D’agir comme une fonction, une liste…

**📌 Les plus courantes :**

| **Méthode** | **Rôle** |
| --- | --- |
| \_\_init\_\_() | Initialise un objet (vu précédemment) |
| \_\_str\_\_() | Représentation **humaine** de l'objet (print(obj)) |
| \_\_repr\_\_() | Représentation **technique** (debug / console) |
| \_\_eq\_\_() | Compare deux objets avec == |
| \_\_len\_\_() | Permet d’utiliser len(objet) |
| \_\_call\_\_() | Rendre un objet **appelable** comme une fonction |

**🧩 Exemple simple : \_\_str\_\_()**

python

CopyEdit

class Livre:

def \_\_init\_\_(self, titre, auteur):

self.titre = titre

self.auteur = auteur

def \_\_str\_\_(self):

return f"📖 {self.titre} par {self.auteur}"

livre = Livre("1984", "George Orwell")

print(livre) # Appelle automatiquement \_\_str\_\_()

✅ Résultat :

yaml

CopyEdit

📖 1984 par George Orwell