Séance 6 – Centraliser l’instanciation

Support apprenants (Factory, Singleton, Builder)

Lors des séances précédentes, vous avez appris à :

- séparer le métier du stockage,
- utiliser des repositories,
- rendre le métier testable et indépendant de la base de données.

Une nouvelle question se pose maintenant naturellement :

Qui crée les objets de l’application ?

Si chaque service crée lui-même ses dépendances :

- le code devient couplé,
- les changements techniques se propagent partout,
- l’architecture perd en lisibilité.

L’objectif de cette séance est donc de **centraliser l’instanciation des objets**.

1. Objectif de la séance

Objectif principal

Avoir un point unique responsable de la création des objets.

Cela permet :

- de simplifier l’architecture,
- de faciliter l’évolution technique,
- de rendre le code plus testable,
- de mieux maîtriser les dépendances.

2. La Factory : notion centrale

Rôle d’une Factory

Une **Factory** est une classe dont la responsabilité est de :

- créer les objets nécessaires à l'application,
- assembler leurs dépendances,
- masquer les détails de construction.

Le métier ne doit pas savoir *comment* les objets sont créés.
Il doit simplement utiliser des services **déjà prêts**.

3. Singleton : quand l'utiliser ?

Un **Singleton** permet de garantir qu'un composant n'existe qu'en **un seul exemplaire**.

Cas d'usage possibles :

- la Factory globale,
- un objet de configuration,
- un composant technique partagé.

Le Singleton doit être **justifié**. Il ne s'agit pas de l'utiliser systématiquement.

4. Builder : optionnel et ciblé

Le **Builder** est utile lorsque :

- un objet est complexe à construire,
- la création se fait en plusieurs étapes,
- certains paramètres sont optionnels.

Dans cette séance, le Builder est **facultatif** et ne doit être utilisé que s'il apporte une vraie valeur.

5. Ce que vous devez produire

À la fin de la séance, vous devez être capables de présenter :

- une **Factory globale** (même conceptuelle),
- un schéma ou pseudo-code montrant **qui crée quoi**,
- une justification écrite de vos choix (Factory, Singleton, Builder ou non).

Il n'est pas attendu de coder une application complète.

6. Livrable attendu

Le livrable doit montrer :

- un point unique d’instanciation,
- un métier qui ne crée plus directement ses dépendances,
- une architecture plus lisible et maîtrisée.

7. Message clé à retenir

Après avoir découplé le métier du stockage, vous découplez maintenant le métier de la création des objets.

Ce raisonnement est essentiel pour construire une architecture évolutive et propre.

8. En résumé

- Vous raisonnez en **architecture**, pas en implémentation.
- La Factory est l’élément central de la séance.
- Le code est secondaire, le raisonnement est prioritaire.
- Vos choix doivent être clairs et justifiés.