



Nom i Cognoms: Victor Asensio Bermúdez

URL Repositori GitHub: https://github.com/victor06-git/Hibernate_PR2.3_AMS2

Objectius:

- Conèixer els arxius de configuració bàsics de Hibernate.
- Crear classes amb anotacions JPA que satisfacin el que es demana a l'enunciat.
- Afegir el codi que realitzi la gestió.

Instruccions:

- Es valorarà la presentació i els comentaris al codi

Criteris d'avaluació:

- L'exercici es puntua sobre 10 punts en funció de la seva complexió
- Es faran preguntes sobre el codi entregat durant una futura sessió de classe

Entrega:

- Enllaç a repositori GitHub public o compartit amb l'usuari **jpala4-ieti**

Materials:

- Necessiteu una eina per programar en JAVA
- Apunts de classe i repositoris d'exemple
- Cerca de tutorials alternatius

Repositoris de referència:

- <https://github.com/jpala4-ieti/DAM-JavaHibernateJPASOneToMany>
- <https://github.com/jpala4-ieti/DAM-JavaHibernateJPAManyToMany>

Punt de partida

- <https://github.com/jpala4-ieti/DAM-M0486-Tema2-PR23-Punt-Partida-25-26>



Exercici 0

L'objectiu és implementar la capa de dades completant les anotacions JPA a les entitats i la lògica de negoci al `Manager`.

1. Definició del Model de Dades (Entitats)

El projecte ja conté les classes amb els seus constructors. La teva tasca és afegir les anotacions de mapatge (`@Entity`, `@Id`, `@OneToMany`, `@ManyToOne`, `@ManyToMany`, etc.).

- **Biblioteca:** Conté la informació de la seu.
 - **Atributs:** `bibliotecaId` (PK), `nom`, `ciutat`, `adreca`, `telefon`, `email`.
 - **Relació:** Llista d'`exemplars` (OneToMany).
- **Llibre:** Defineix l'obra intel·lectual (no la còpia física).
 - **Atributs:** `llibreId` (PK), `isbn`, `titol`, `editorial`, `anyPublicacio`.
 - **Relacions:**
 - `autors`: Relació ManyToMany (és el costat `owner`, gestiona la `@JoinTable`).
 - `exemplars`: Llista de còpies físiques (OneToMany).
- **Autor:**
 - **Atributs:** `autorId` (PK), `nom`.
 - **Relació:** `llibres` (ManyToMany inversa, `mappedBy`).
- **Exemplar:** Representa una còpia física concreta d'un llibre.
 - **Atributs:** `exemplarId` (PK), `codiBarres` (únic), `disponible` (boolean).
 - **Relacions:** Vinculat a un `llibre` i a una `biblioteca` (ManyToOne). Té un `historialPrestecs` (OneToMany).
- **Persona:** Usuari de la biblioteca.
 - **Atributs:** `personaId` (PK), `dni`, `nom`, `telefon`, `email`.
 - **Relació:** Llista de `prestecs` (OneToMany).
- **Prestec:** Registre del moviment.
 - **Atributs:** `prestecId` (PK), `dataPrestec`, `dataRetornPrevista`, `dataRetornReal` (nullable), `actiu` (boolean).
 - **Relacions:** Vinculat a un `exemplar` i una `persona` (ManyToOne).

2. Lògica de Negoci (`Manager.java`)

Hauràs d'implementar els mètodes marcats amb `TODO` al fitxer `Manager.java` per gestionar:

1. **Persistència bàsica:** Creació d'autors, llibres, biblioteques i persones.
2. **Gestió de relacions M:N:** Vincular llibres amb autors recordant quin és el costat propietari de la relació per assegurar que es guardi a la base de dades.



3. **Cicle de vida del Préstec:**

- **Prestar:** Comprovar disponibilitat de l'exemplar, crear el préstec actiu i marcar l'exemplar com a no disponible .
 - **Retornar:** Tancar el préstec (data real i `actiu=false`) i alliberar l'exemplar .
4. **Consultes (HQL):** Implementar consultes per obtenir llibres amb els seus autors (optimitzat amb `FETCH`), llibres actualment prestats i llibres per biblioteca.

3. Execució i Proves (`Main.java`)

La classe `Main` ja està programada i estructurada per **Fases**. Hauràs de descomentar-les progressivament a mesura que implementis el codi:

- **Fase 1:** Creació d'entitats simples (Autors i Llibres).
- **Fase 2:** Vinculació Many-to-Many (assignar llibres a autors).
- **Fase 3:** Creació de la infraestructura (Biblioteca, Exemplars) i Usuaris.
- **Fase 4:** Prova de préstecs:
 - Intent de préstec vàlid.
 - Intent de préstec invàlid (mateix exemplar ja prestat).
 - Registre del retorn .
- **Fase 5:** Execució de les consultes HQL per visualitzar informes.

Ajuda addicional

En el repositori d'exemple s'ofereix un `Main` amb codi comentat.

I classes amb els mètodes `toString`, `equals` i `compare` implementats.



Exercicis ampliació

- Aconsegueix que el teu exemple pugui funcionar a la vegada amb SQLite i MySQL, permetent triar la BD usant un menú.