**ENCADREUR :**

M. BILONG

**MEMBRES :**

* ESSONO EFFA
* KANGNE Yvan
* NGOUO Franc
* DJEUDA Paul
* KENG TONG
* EDING Moise

**EXPOSE DE JEE**

**THEME** : JPA

2024

Table des matières

[Introduction 1](#_Toc160127389)

[I. Définition 1](#_Toc160127390)

[II. Historique et Contexte 1](#_Toc160127391)

[III. **Mise en œuvre (Démonstration)** 2](#_Toc160127392)

[**1.** **Entité Personne** 2](#_Toc160127393)

[2. Utilisation de l’EntityManager 2](#_Toc160127394)

[IV. **Recommandations et Mode d’emploi** 3](#_Toc160127395)

[V. RESULTAT DE DEMONSTRATION 4](#_Toc160127396)

[Conclusion 5](#_Toc160127397)

[BIBLIOGRAPHIE 6](#_Toc160127398)

# Introduction

La persistance des données est une composante essentielle dans le développement d’applications d’entreprise. Java EE (Java Platform, Enterprise Edition) fournit une API dédiée à la persistance des données, connue sous le nom de Java Persistence API (JPA). La gestion efficace de la persistance des données est un aspect crucial dans le développement d'applications robustes et évolutives. La Java Persistance API (JPA) représente une norme de facto (largement acceptée et utilisée dans la pratique) dans l'écosystème Java pour simplifier et standardiser le mapping objet-relationnel et la gestion des entités persistantes dans les applications. En offrant une approche orientée objet pour interagir avec les bases de données relationnelles, JPA facilite le développement d'applications Java en réduisant la complexité et en améliorant la productivité des développeurs. Dans cet exposé, nous explorerons la définition, l’historique, l’implémentation, les recommandations et le mode d’emploi de JPA.

# Définition

JPA est une API Java qui fournit un cadre pour la gestion de la persistance des données dans les applications Java EE et Java SE. Elle simplifie le stockage et la récupération des objets Java dans une base de données relationnelle, en offrant une couche d’abstraction entre l’application et la base de données.

# Historique et Contexte

Origine de JPA : Java Persistance API (JPA) a été introduite pour la première fois dans le cadre de la spécification EJB 3.0 (Enterprise JavaBeans) dans le cadre de Java EE 5 (Java Platform, Enterprise Edition 5). Cette spécification visait à simplifier le développement d'applications Java en offrant une approche plus orientée objet pour la gestion des données persistantes. JPA a été conçue pour faciliter le mapping objet-relationnel (ORM), permettant aux développeurs de manipuler les données de la base de données relationnelle en utilisant des objets Java. En standardisant cette approche, JPA a permis de réduire la dépendance des applications aux spécificités des bases de données et a simplifié le développement en offrant une couche d'abstraction cohérente pour la persistance des données. L’évolution des technologies de persistance en Java a connu plusieurs étapes clés avant l'émergence de la Java Persistance API (JPA). JPA a émergé comme une solution plus efficace par rapport aux anciennes approches basées sur JDBC.

# **Mise en œuvre (Démonstration)**

Pour illustrer l’implémentation de JPA, considérons un exemple simple d’une entité « Personne » stockée dans une base de données.

## Entité Personne

## Utilisation de l’EntityManager

Voici un exemple simple d'implémentation de JPA :

@Entity

public class Person {

@Id

private Long id;

private String name;

private int age;

// ...

}

public class Main {

public static void main(String[] args) {

EntityManagerFactory emf = Persistence.createEntityManagerFactory("my-persistence-unit");

EntityManager em = emf.createEntityManager();

Person person = new Person();

person.setName("John Doe");

person.setAge(30);

em.getTransaction().begin();

em.persist(person);

em.getTransaction().commit();

em.close();

emf.close();

}

}

?Cette démonstration illustre la définition d’une entité, l’utilisation de l’EntityManager pour la persistance et la récupération d’objets.

# **Recommandations et Mode d’emploi**

1. Écuries de versions

Il est recommandé d’utiliser les versions stables de JPA pour assurer la stabilité et la compatibilité. À la date de cette rédaction, JPA 2.2 est la version stable la plus récente.

1. Mode d’emploi

Pour utiliser JPA dans une application Java EE, suivez ces étapes :

* Définissez les entités avec des annotations JPA.

Configurez l’unité de persistence dans le fichier .**persistence.xml**

* Utilisez l’EntityManager pour effectuer des opérations de persistance et de récupération.

# RESULTAT DE DEMONSTRATION

La démonstration montre comment utiliser JPA pour persister un objet Java dans une base de données relationnelle. Voici les étapes en détail :

1. **Création d'une entité JPA:** La classe Person est une entité JPA. Elle est annotée avec @Entity et possède une propriété id annotée avec @Id.
2. **Création d'un EntityManagerFactory:** Un EntityManagerFactory est utilisé pour créer des EntityManager.
3. **Création d'un EntityManager:** Un EntityManager est utilisé pour gérer les entités JPA.
4. **Début d'une transaction:** Une transaction est nécessaire pour garantir la cohérence des données.
5. **Persistance d'une entité:** L'entité person est persistée dans la base de données en utilisant la méthode persist() de l'EntityManager.
6. **Validation de la transaction:** La transaction est validée pour commiter les modifications dans la base de données.
7. **Fermeture de l'EntityManager et de l'EntityManagerFactory:** L'EntityManager et l'EntityManagerFactory sont fermés pour libérer les ressources.

**Résultat:**

Après l'exécution de la démonstration, l'objet person sera enregistré dans la table Person de la base de données.

**Points importants:**

* La démonstration utilise une configuration JPA par défaut.
* Il est possible de configurer JPA plus finement en utilisant un fichier XML de configuration.
* La démonstration ne montre que la persistance d'une entité. JPA permet également de réaliser d'autres opérations CRUD (Create, Read, Update, Delete) sur les entités.

# Conclusion

JPA offre une solution standardisée et efficace pour la gestion de la persistance des données dans les applications Java EE. Son adoption a simplifié le développement d’applications en éliminant la fragmentation des solutions de persistance. En suivant les bonnes pratiques et en utilisant les versions stables, les développeurs peuvent bénéficier d’une persistance robuste et portable. JPA continue d’évoluer pour répondre aux besoins croissants des applications d’entreprise, assurant ainsi une place centrale dans le développement Java EE.

# BIBLIOGRAPHIE

**JPA**

https://www.jmdoudoux.fr/java/dej/chap-jpa.htm

**ECLIPSELINK** : <https://www.eclipse.org/downloads/download.php?file=/rt/eclipselink/releases/4.0.2/eclipselink-4.0.2.v20230613-3bfa6ac6dd.zip&mirror_id=1297>

**HIBERNATE**

<https://sourceforge.net/projects/hibernate/>