Guía Completa de Implementación de Hibernate en Java con IntelliJ IDEA

Introducción

Hibernate es una herramienta de **Mapeo Objeto-Relacional (ORM)** que permite mapear clases de Java a tablas de base de datos.
En esta guía aprenderás a configurar e implementar Hibernate en un proyecto Java en IntelliJ IDEA, usando anotaciones para definir las relaciones.

1. Requisitos previos

- Java JDK instalado
- IntelliJ IDEA
- Base de datos MySQL (o similar)
- Maven (o Gradle)
- Conocimientos básicos de SQL

2. Crear proyecto en IntelliJ

1. Abre IntelliJ IDEA.
2. Crea un proyecto Maven.
3. Agrega la dependencia de Hibernate en `pom.xml`:

```
<dependencies>
    <dependency>
        <groupId>org.hibernate.orm</groupId>
        <artifactId>hibernate-core</artifactId>
        <version>6.2.5.Final</version>
    </dependency>
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.33</version>
    </dependency>
</dependencies>
```

3. Configuración `hibernate.cfg.xml`

Crea el archivo en `src/main/resources`:

```
<?xml version='1.0' encoding='utf-8'?>
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN"
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
<hibernate-configuration>
    <session-factory>
        <property name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
        <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/escuela</property>
        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password">tu_password</property>
        <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
        <property name="hibernate.hbm2ddl.auto">update</property>
        <property name="hibernate.show_sql">true</property>
    </session-factory>
```

```
</hibernate-configuration>
```

4. Clases de modelo con anotaciones

Ejemplo de clase `Alumno`:

```
@Entity
@Table(name = "alumnos")
public class Alumno {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;
    @Column(nullable = false)
    private String nombre;
    @ManyToOne
    @JoinColumn(name = "curso_id")
    private Curso curso;
    @ManyToMany
    @JoinTable(
name = "alumno_materia",
joinColumns = @JoinColumn(name = "alumno_id"),
inverseJoinColumns = @JoinColumn(name = "materia_id")
)
    private List<Materia> materias = new ArrayList<>();
}
```

5. Anotaciones importantes

- **@Entity** → Marca la clase como entidad.
- **@Table(name="...")** → Define el nombre de la tabla.
- **@Id** → Marca el campo como clave primaria.
- **@GeneratedValue** → Estrategia de generación de IDs.
- **@Column** → Configura columnas.
- **@OneToMany / @ManyToOne / @ManyToMany** → Define relaciones.

6. Clase utilitaria `HibernateUtil`

```
public class HibernateUtil {
    private static final SessionFactory sessionFactory;
static {
try {
sessionFactory = new Configuration().configure().buildSessionFactory();
} catch (Throwable ex) {
throw new ExceptionInInitializerError(ex);
}
}
    public static SessionFactory getSessionFactory() {
return sessionFactory;
}
}
```

7. Ejemplo de uso en `MainApp`

```
public class MainApp {
    public static void main(String[] args) {
Session session = HibernateUtil.getSessionFactory().openSession();
Transaction tx = session.beginTransaction();
Curso curso = new Curso("Programación Java");
Materia m1 = new Materia("Hibernate");
Materia m2 = new Materia("Spring Boot");
session.persist(curso);
session.persist(m1);
session.persist(m2);
Alumno alumno = new Alumno("Juan Pérez");
alumno.setCurso(curso);
alumno.getMaterias().add(m1);
alumno.getMaterias().add(m2);
session.persist(alumno);
tx.commit();
session.close();
}
}
```

8. Ejecución

1. Ejecuta `MainApp` desde IntelliJ.
2. Observa las tablas creadas en la base de datos.
3. Verifica los registros insertados.

Conclusión

Hibernate simplifica enormemente el manejo de bases de datos en Java, permitiendo trabajar con objetos en lugar de SQL puro, y facilitando la gestión de relaciones complejas con simples anotaciones.