# **ORM Implementation with Hibernate**

## What is ORM?

* ORM (Object Relational Mapping) allows Java classes to map to database tables automatically.
* Hibernate is the most widely used ORM tool for Java.
* It lets developers work with Java objects — no need to write manual SQL CRUD.

## **XML Configuration**

## **Key Parts**

| Part | What it does |
| --- | --- |
| Persistence Class | Plain Java class to represent table |
| Mapping XML (.hbm.xml) | Maps class fields to table columns |
| Hibernate Config XML | Sets up DB connection & includes mappings |
| Main Java Code | Loads config, runs CRUD |

### Files Example

* Student.java
* Student.hbm.xml
* hibernate.cfg.xml
* MainApp.java

### Student.java

public class Student {  
 private int id;  
 private String firstName;  
 private String lastName;  
  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
 public String getFirstName() { return firstName; }  
 public void setFirstName(String firstName) { this.firstName = firstName; }  
 public String getLastName() { return lastName; }  
 public void setLastName(String lastName) { this.lastName = lastName; }  
}

### Student.hbm.xml

<hibernate-mapping>  
 <class name="Student" table="STUDENT">  
 <id name="id" type="int">  
 <generator class="increment"/>  
 </id>  
 <property name="firstName" column="first\_name"/>  
 <property name="lastName" column="last\_name"/>  
 </class>  
</hibernate-mapping>

### hibernate.cfg.xml

<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/testdb</property>  
 <property name="hibernate.connection.username">root</property>  
 <property name="hibernate.connection.password">password</property>  
 <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>  
  
 <mapping resource="Student.hbm.xml"/>  
 </session-factory>  
</hibernate-configuration>

### MainApp.java

import org.hibernate.\*;  
import org.hibernate.cfg.\*;  
  
public class MainApp {  
 public static void main(String[] args) {  
 Configuration cfg = new Configuration();  
 cfg.configure("hibernate.cfg.xml");  
 SessionFactory factory = cfg.buildSessionFactory();  
  
 Session session = factory.openSession();  
 Transaction t = session.beginTransaction();  
  
 Student s = new Student();  
 s.setFirstName("John");  
 s.setLastName("Doe");  
  
 session.save(s);  
 t.commit();  
  
 session.close();  
 factory.close();  
 }  
}

## **Annotation Configuration**

### **Key Parts**

| Part | What it does |
| --- | --- |
| Persistence Class | Has @Entity, @Table etc. inside the class |
| Annotations | @Id, @Column directly map fields |
| hibernate.cfg.xml | Configures DB + lists annotated class |
| Main Java Code | Loads config, runs CRUD |

### Files Example

* Student.java (annotated)
* hibernate.cfg.xml
* MainApp.java

### Student.java

import jakarta.persistence.\*;  
  
@Entity  
@Table(name = "STUDENT")  
public class Student {  
  
 @Id  
 @GeneratedValue(strategy = GenerationType.IDENTITY)  
 private int id;  
  
 @Column(name = "first\_name")  
 private String firstName;  
  
 @Column(name = "last\_name")  
 private String lastName;  
  
 public int getId() { return id; }  
 public void setId(int id) { this.id = id; }  
 public String getFirstName() { return firstName; }  
 public void setFirstName(String firstName) { this.firstName = firstName; }  
 public String getLastName() { return lastName; }  
 public void setLastName(String lastName) { this.lastName = lastName; }  
}

hibernate.cfg.xml

<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/testdb</property>  
 <property name="hibernate.connection.username">root</property>  
 <property name="hibernate.connection.password">password</property>  
 <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>  
  
 <mapping class="Student"/>  
 </session-factory>  
</hibernate-configuration>

### 

### MainApp.java

import org.hibernate.\*;  
import org.hibernate.cfg.\*;  
  
public class MainApp {  
 public static void main(String[] args) {  
 Configuration cfg = new Configuration();  
 cfg.configure("hibernate.cfg.xml");  
 SessionFactory factory = cfg.buildSessionFactory();  
  
 Session session = factory.openSession();  
 Transaction t = session.beginTransaction();  
  
 Student s = new Student();  
 s.setFirstName("Jane");  
 s.setLastName("Smith");  
  
 session.save(s);  
 t.commit();  
  
 session.close();  
 factory.close();  
 }  
}

## Common Hibernate Steps

1️ Load hibernate.cfg.xml with Configuration  
2️ Build SessionFactory  
3️ Open Session  
4️ Begin Transaction  
5️ Perform save, update, query  
6️ Commit Transaction  
7️ Close Session

Same flow for XML or Annotation style.

## **XML vs Annotation Summary**

| Feature | XML Config | Annotation Config |
| --- | --- | --- |
| Mapping | External .hbm.xml | Inside Java class with @Entity |
| Easy to change? | Flexible but more files | Cleaner, modern style |
| Common usage | Legacy, when code can’t change | Modern, preferred |