



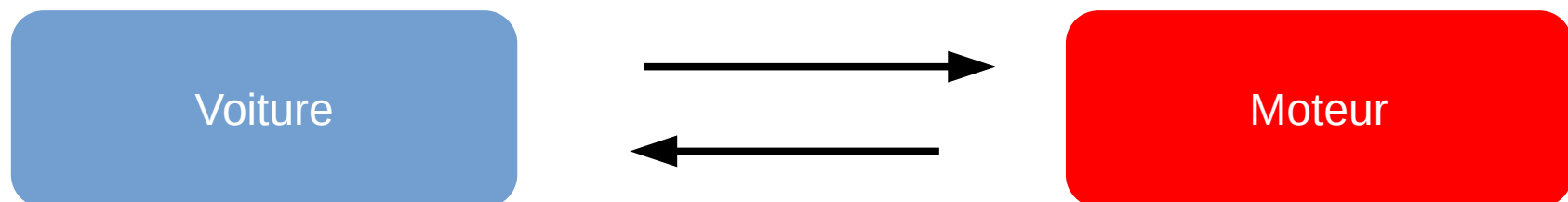
# **Hibernate relation OneToOne bidirectionnelle**

notion

CJD-Formation

# Nouveau use case

- Supposons maintenant que nous souhaitons à partir du moteur, pouvoir accéder à la voiture à laquelle il est relié.
- Actuellement il manque une information coté moteur pour faire cela.
- En ajoutant cette information coté moteur nous transformons la relation qui devient ainsi bidirectionnelle.



# Bi-directionnel

- Pour ajouter cette fonctionnalité , on peut conserver le schéma de bdd actuel.
- On va modifier le code java : la classe "aveugle" :
  - ajout d'un champ qui référence la classe "voyante"
  - Ajout des accesseurs et mutateurs
  - Ajout de @OneToOne
- On va créer une classe exécutable Main App

# Mettre à jour Moteur

```
@Entity
@Table(name="voiture")
public class Voiture {

    @Id
    @GeneratedValue(strategy= GenerationType.IDENTITY)
    @Column
    private Long id ;

    @Column
    private String modele;

    @Column
    private String immatriculation;

    @OneToOne(cascade=CascadeType.ALL)
    @JoinColumn(name="moteur_id")
    private Moteur moteur;

    //getters & setters
    . . .
}
```

```
@Entity
public class Moteur {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column
    private int id;

    @Column
    private int puissance;

    @Column
    private String carburant;

    @Column
    private int cylindree;

    @OneToOne(mappedBy="moteur", cascade=CascadeType.ALL)
    private Voiture voiture;

    //getters() & setters()

    public Voiture getVoiture() {
        return voiture;
    }

    public void setVoiture(Voiture voiture) {
        this.voiture = voiture;
    }

    . . .
}
```

# MappedBy, cascade

```
@Entity
@Table(name="voiture")
public class Voiture {

    @Id
    @GeneratedValue(strategy= GenerationType.IDENTITY)
    @Column
    private Long id ;

    @Column
    private String modele;

    @Column
    private String immatriculation;

    @OneToOne(cascade=CascadeType.ALL)
    @JoinColumn(name="moteur_id")
    private Moteur moteur;

    //getters & setters
    . . .
}
```

```
@Entity
public class Moteur {

    @Id
    @GeneratedValue(strategy=GenerationType.IDENTITY)
    @Column
    private int id;

    @Column
    private int puissance;

    @Column
    private String carburant;

    @Column
    private int cylindree;

    @OneToOne(mappedBy="moteur", cascade=CascadeType.ALL)
    private Voiture voiture;

    //getters() & setters()

    public Voiture getVoiture() {
        return voiture;
    }

    public void setVoiture(Voiture voiture) {
        this.voiture = voiture;
    }

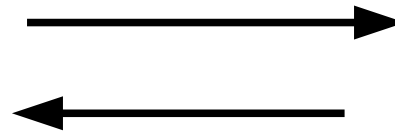
    . . .
}
```

# Main App : lire un moteur

```
public static void main(String[] args) {  
    . . .  
    // lire un objet moteur dans la bdd  
    Moteur m = session.get(Moteur.class, 2);  
    System.out.println("moteur lu : "+m);  
    System.out.println(" la voiture associée est : "+m.getVoiture());  
    . . .  
}
```

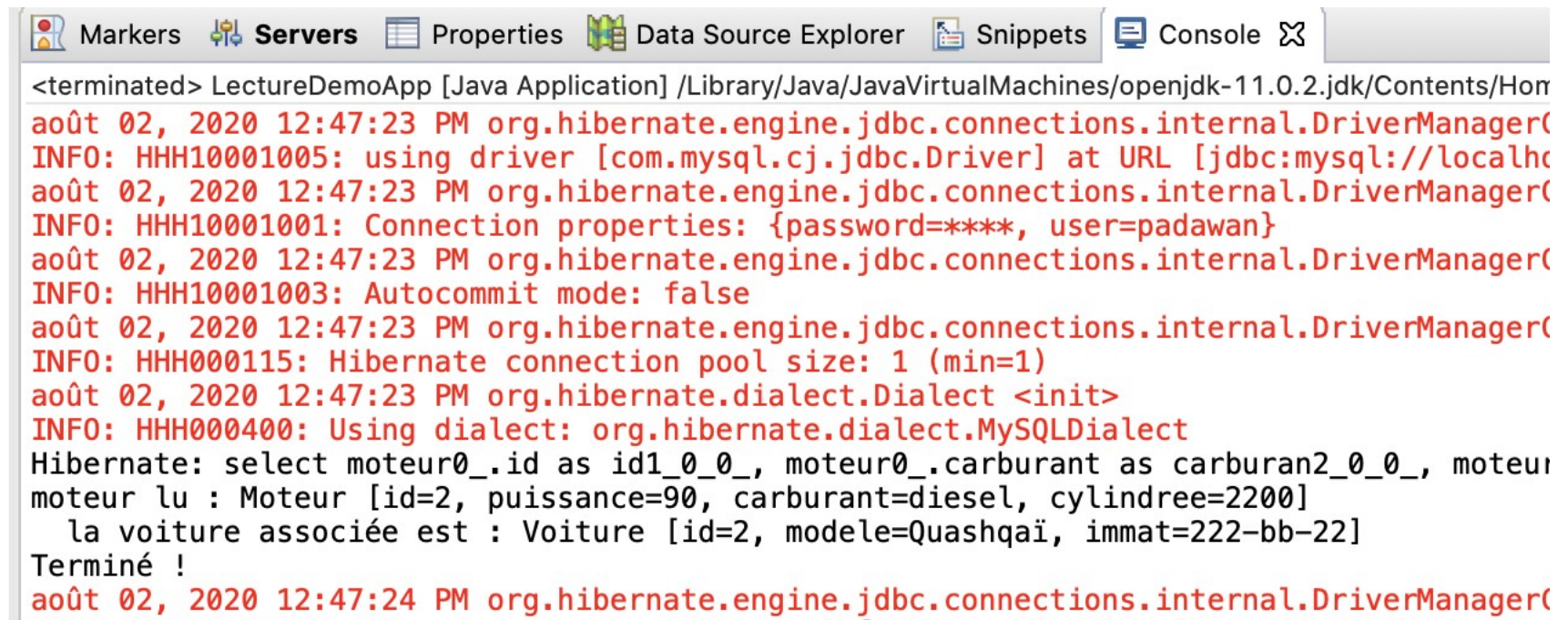
Choisissez un moteur  
qui existe déjà en bdd

Voiture



Moteur

# Résultat de l'exécution de ce code



```
<terminated> LectureDemoApp [Java Application] /Library/Java/JavaVirtualMachines/openjdk-11.0.2.jdk/Contents/Home
août 02, 2020 12:47:23 PM org.hibernate.engine.jdbc.connections.internal.DriverManager(
INFO: HHH10001005: using driver [com.mysql.cj.jdbc.Driver] at URL [jdbc:mysql://localhost:3306/lecturedemo]
août 02, 2020 12:47:23 PM org.hibernate.engine.jdbc.connections.internal.DriverManager(
INFO: HHH10001001: Connection properties: {password=****, user=padawan}
août 02, 2020 12:47:23 PM org.hibernate.engine.jdbc.connections.internal.DriverManager(
INFO: HHH10001003: Autocommit mode: false
août 02, 2020 12:47:23 PM org.hibernate.engine.jdbc.connections.internal.DriverManager(
INFO: HHH000115: Hibernate connection pool size: 1 (min=1)
août 02, 2020 12:47:23 PM org.hibernate.dialect.Dialect <init>
INFO: HHH000400: Using dialect: org.hibernate.dialect.MySQLDialect
Hibernate: select moteur0_.id as id1_0_0_, moteur0_.carburant as carburan2_0_0_, moteur0_.cylindree as cylindree3_0_0_ from moteur0_ where moteur0_.id=2
moteur lu : Moteur [id=2, puissance=90, carburant=diesel, cylindree=2200]
    la voiture associée est : Voiture [id=2, modele=Quashqaï, immat=222-bb-22]
Terminé !
août 02, 2020 12:47:24 PM org.hibernate.engine.jdbc.connections.internal.DriverManager(
```

Désormais on accède à l'entité liée des deux cotés de la relation (sans changer le modèle de données)