Présentation Hibernate

Max Devulder

27 Slides

- **Framework** open source supporté par RedHat
- ORM : Object Relationnal Mapping.
- Couche Persistance
- Populaire

- **Indépendance physique :**
  - DB2, Oracle, MySQL, PostgreSQL, Sybase, SQL Server, Sap DB, Interbase, ...

- **Indépendance logique:** Abstraction du MPD relationnel

> Equivalence schéma Objet - Relationnel

Table
« relationnelle »

	Champ	Type	Interclassement	Attributs	Null	Défaut	Extra
☐	**events_id**	int(11)			Non	*Aucun*	auto_increment
☐	**events_allDay**	bit(1)			Oui	0	
☐	**events_beginDate**	date			Oui	*NULL*	
☐	**events_description**	varchar(255)	latin1_swedish_ci		Oui	*NULL*	
☐	**events_title**	varchar(255)	latin1_swedish_ci		Non	*Aucun*	

> Equivalence schéma Objet - Relationnel

Bean
« classique »

```
public class Event {


   private Integer id;
   private String title;
   private String description;
   private Calendar beginDate;
   private boolean allDay;

   public Event() {
   }

   // Getters et setters
}
```

> Equivalence schéma Objet - Relationnel

Bean
« Hibernate »

```
@Entity
@Table(name="events_01")
public class Event {

   @Id
   @GeneratedValue(strategy=GenerationType.AUTO)
   @Column(name="events_id")
   private Integer id;

   @Column(name="events_title", nullable=false)
   private String title;

   @Column(name="events_description")
   private String description;

   @Column(name="events_beginDate")
   private Calendar beginDate;

   @Column(name="events_allDay")
   private boolean allDay;

   public Event() {
   }

   // Getters et setters
}
```

> 1 fichier de conf : hibernate.cfg.xml

```
<!DOCTYPE hibernate-configuration PUBLIC
"-//Hibernate/Hibernate Configuration DTD 3.0//EN«
"http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">

<hibernate-configuration>
<session-factory>

<!--  Paramètres de connexion à la base de données -->
<property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
<property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
<property
name="hibernate.connection.url">jdbc:mysql://localhost/pge_jpa_v1</property>
<property name="hibernate.connection.username">root</property>
<property name="hibernate.connection.password"></property>

<!-- Comportement pour la conservation des tables -->
<property name="hbm2ddl.auto">create</property>

<!-- Fichiers à mapper -->
<mapping class="fr.mistra.pgejpav1.jpa.Event" />

<!-- Affiche les logs SQL -->
<!--     <property name="show_sql">true</property>-->
<!--     <property name="hibernate.format_sql">true</property>-->
<!--     <property name="use_sql_comments">true</property>-->

</session-factory>
</hibernate-configuration>
```

Connexion BDD

Mode

Entités

Logs

> Initialisation d'Hibernate

Mode	Action
validate	Valide le schéma, aucune modification n'est faite sur la structure de la base ;
update	met à jour le schéma existant
create	crée le schéma en supprimant les données préalablement existantes
create-drop	même comportement que create avec suppression du schéma en fin de session.
none	Rien

> Ordre INSERT

Classe java

```
Session s = HibernateUtils.getSession();

// Début de la transaction
Transaction t = s.beginTransaction();

// Création d'un objet Event
Event e = new Event();
e.setTitle("Titre de l'event 1");
e.setDescription("Description de l'évènement 1");
e.setBeginDate(new GregorianCalendar());
e.setAllDay(false);

// Enregistrement de l'event
s.save(e);

// Fin de la transaction
t.commit();

// Fermeture de la session Hibernate
s.close()
```

Console

```
2389 [main] INFO org.hibernate.tool.hbm2ddl.SchemaExport - Running hbm2ddl schema export
2390 [main] INFO org.hibernate.tool.hbm2ddl.SchemaExport - exporting generated schema to database
2413 [main] INFO org.hibernate.tool.hbm2ddl.SchemaExport - schema export complete
Hibernate:
/* insert fr.mistra.pgejpav1.jpa.Event
*/ insert
Into
pge_jpa_v1_events_01
(events_allDay, events_beginDate, events_description, events_title)
Values
(?, ?, ?, ?)
```

> Ordre delete

Classe java

```
// Début de la transaction
Transaction tx = s.beginTransaction();

// Création de la requête
Query q = s.createQuery("delete Event");
// Exécution de la requête
q.executeUpdate();

 // Fin de la transaction
tx.commit();
```

Console

```
Hibernate:
    Delete
    From
    events_01
```

> Ordre select (Langage HQL)

Classe java

```
  // Début de la transaction
  Transaction tx = s.beginTransaction();

  // Création de la requête
  Query q = s.createQuery("from Event");

  // Récupération de la liste des résultats
  ArrayList list = (ArrayList) q.list();

  // Affichage des résultats
  for (Event e: list) {
     System.out.println("Event : [id] = " + e.getId() + "\t" +
     "[title] = " + e.getTitle() + "\t" +
     "[desc] = " + e.getDescription() + "\n" +
     "[date] = " + e.getBeginDate().getTime().toLocaleString() + "\t" +
     "[allDay] = " + e.isAllDay());
}
```

Console

```
select
    event0_.events_id as events1_0_,
    event0_.events_allDay as events2_0_,
    event0_.events_beginDate as events3_0_,
    event0_.events_description as events4_0_,
    event0_.events_title as events5_0_
  From
    events_01 event0_
```

> Récupération d'un objet (HQL)

```
Query q = s.createQuery("from Event where title= :myTitle");
q.setString("myTitle", "Titre du premier évènement");
Event e = (Event) q.uniqueResult();
```

> Update

```
// Récupération de l'Event d'après son titre
Query q = s.createQuery("from Event where title= :myTitle");
q.setString("myTitle", "Titre du premier évènement");

Event e = (Event) q.uniqueResult();

// Modifications des attributs de l'objet
e.setDescription("Description modifiée");
e.setAllDay(false);

// Prise en compte de la modification
Transaction tx = s.beginTransaction();
s.update(e);
tx.commit();
```

HIBERNATE

> Relations (one-to-one)

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra
events_id	int(11)			Non	*Aucun*	auto_increment
events_allDay	bit(1)			Oui	0	
events_beginDate	date			Oui	*NULL*	
events_description	varchar(255)	latin1_swedish_ci		Oui	*NULL*	
events_title	varchar(255)	latin1_swedish_ci		Non	*Aucun*	
events_addressID	int(11)			Oui	*NULL*	

Ajout foreign key

Champ	Type	Interclassement	Attributs	Null	Défaut	Extra
adresses_id	int(11)			Non	*Aucun*	auto_increment
adresses_city	varchar(255)	latin1_swedish_ci		Oui	*NULL*	
adresses_comments	varchar(255)	latin1_swedish_ci		Oui	*NULL*	
adresses_name	varchar(255)	latin1_swedish_ci		Non	*Aucun*	
adresses_street	varchar(255)	latin1_swedish_ci		Oui	*NULL*	
adresses_zipcode	varchar(255)	latin1_swedish_ci		Oui	*NULL*	

> Relations (one-to-one)

Classe Address

```java
public class Address {
   @Id
   @GeneratedValue(strategy=GenerationType.AUTO)
   @Column(name="adresses_id")
   private Integer id;

   @Column(name="adresses_name" ,nullable=false)
   private String name;

   @Column(name="adresses_street")
   private String street;

   @Column(name="adresses_comments")
   private String comments;

   @Column(name="adresses_zipcode")
   private String zipCode;

   @Column(name="adresses_city")
   private String city;
}
```

Classe Event

```java
   @OneToOne(cascade={CascadeType.ALL})
   @JoinColumn(
   name="events_addressID",
   referencedColumnName="adresses_id")
   private Address address;
```

> Relations (one-to-one)

Java

```
Event e = new Event("Titre de l'évènement", "description", true);
Address a = new Address("Nom de l'adresse", "24 rue des cerisiers",
"75001", "Paris");
// Association entre Event et Address
e.setAddress(a);

// Enregistrements
Transaction tx = s.beginTransaction();
s.save(e);
tx.commit();
```

Console

```
Hibernate: insert into events (events_addressID, events_allDay,
events_beginDate, events_description, events_title) values (?,
?, ?, ?, ?)

Hibernate: insert into adresses (adresses_city,
adresses_comments, adresses_name, adresses_street,
adresses_zipcode) values (?, ?, ?, ?, ?)

Hibernate: update events set events_addressID=?,
events_allDay=?, events_beginDate=?, events_description=?,
events_title=? where events_id=?
```

> Relations (one-to-many)

Java

```
@ManyToOne(
cascade={CascadeType.PERSIST, CascadeType.MERGE, CascadeType.REFRESH })
@JoinColumn(name="events_userID")
private User user;
```

> Erreurs fréquentes

Message d'erreur	Explication
java.sql.SQLException: No suitable driver found for [...]	Dans le fichier hibernate.cfg.xml, l'url de la base de données comporte une erreur.
java.net.ConnectException: Connection refused: connect	base de données ne soit pas accessible ou ne soit pas démarré.
org.hibernate.MappingException:Unknown entity : [nom d'une entité]	Vous avez probablement oublié d'ajouter au fichier hibernate.cfg.xml la ligne indiquant à Hibernate les classes à mapper (ou celle-ci comporte une erreur)
org.hibernate.AnnotationException: No identifier specified for entity: [non d'une entité]	Il manque l'annotation @Id dans l'entité spécifiée par l'erreur.
org.hibernate.TransientObjectException: object references an unsaved transient instance - save the transient instance before flushing: [nom d'une entité]	Deux solutions envisageables: mettre en place une cascade entre l'entité citée et celle qui lui est associée et que l'on tentait de manipuler rendre persistante l'entité citée avant d'essayer de sauvegarder les instances d'entité qui y font référence