JDBC – Data Access Object

Guillaume Dufrêne – Lionel Seinturier

Université de Lille – Sciences et Technologies

Définition

Patron de conception (*design pattern*)
DAO : Data Access Object

Bénéfices

- Séparation des préocupations (SoC) :
  opération d'accès <-- --> mise en oeuvre

Illustration

- bas niveau : requête SQL JDBC

```
PreparedStatement pst =
  cx.prepareStatement("SELECT * FROM personnes WHERE age>?");
```

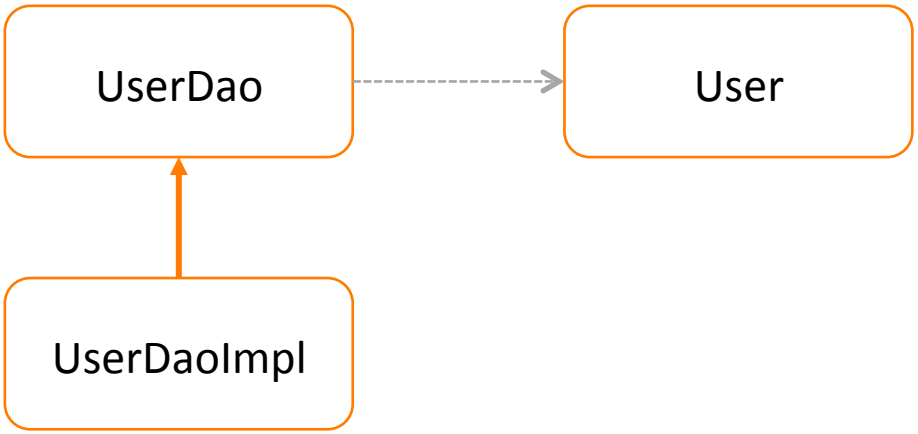
- haut niveau : interface Java pour l'accès aux **mêmes** données

```
interface PersonneDAO {
  List<Personne> findOlderThan( int age );
}
```

Mise en oeuvre

3 éléments

- une interface (eg. suffixe DAO)
- une classe d'implémentation (eg. suffixe DAOImpl)
- une classe contenant les données



Interface DAO

Principe

- Choix des méthodes à la charge du développeur
- Dépend des opérations nécessaires sur les données

Exemple

```
interface PersonneDAO {
  Personne find( int id );
  List<Personne> findOlderThan( int age );
  List<Personne> findAll();
}
```

Classe DAOImpl

Contient

- Implémentation de l'interface DAO
- avec une technologie particulière (eg. JDBC)

Exemple

```
class PersonneDAOImpl implements PersonneDAO {

  public Personne find( int id ) {
    // code JDBC pour la requête SELECT * FROM Personnes WHERE id = ?
  }

  public List<Personne> findOlderThan( int age ) {
    // code JDBC pour la requête SELECT * FROM Personnes WHERE age > ?
  }

  public List<Personne> findAll();
    // code JDBC pour la requête SELECT * FROM Personnes
  }
}
```

Classe de données

Exemple

```
public class Personne {

  private int id;
  private String nom;
  private int age;

  public Personne( int id, String nom, int age ) {
    this.id=id; this.nom=nom; this.age=age; }

  // les méthodes getId, getNom, getAge
  // les méthodes setId, setNom, setAge
}
```

Définition

- acronyme de **C** reate, **R** ead, **U** pdate, **D** elete

Exemple

```
interface PersonneDAO {
  void add( Personne p );
  Personne find( int id );
  List<Personne> findOlderThan( int age );
  List<Personne> findAll();
  void update( Personne p );
  void delete( int id );
}
```

En résumé

DAO

- Sépare logique métier et détails de mise en oeuvre
- Se concentre sur les opérations de haut niveau
- Décline des mises en oeuvre différentes
- Facilite l'optimisation des requêtes de bas niveau