Rapport de TP2 JAVA

GHANDOUR Nouhaila

December 26, 2024

Lien GitHub

Vous pouvez acc'eder au projet complet sur GitHub via le lien suivant: NouhailaGhand/TP-java-

Interface G'en'erique

Dans cette section, nous avons expliqu'e le code d'une interface g'en'erique en Java. L'interface suivante est utilis'ee pour effectuer des op'erations de base de donn'ees (ajouter, supprimer, mettre `a jour, et afficher des 'el'ements g'en'eriques).

Code de l'Interface

Voici l'impl'ementation du code de l'interface g'en'erique :

```
1 package DAO;
2
3 import java.util.List;
4
5     public interface GenericDAOI <T> {
6     public void add(T e);
7     public void delete(int id);
8     public void update(T e);
9     public List<T> display();
10     }
```

Explication du Code

- Cette interface spécifie les méthodes fondamentales pour manipuler des objets de type générique T. Voici une présentation des différentes méthodes proposées :
- add(T e): Cette m'ethode permet d'ajouter un 'el'ement de type T dans une base de donn'ees.
- delete(int id): Cette m'ethode permet de supprimer un 'el'ement de type T de la base de donn'ees, en utilisant un identifiant unique.
- update(T e): Cette m'ethode permet de mettre `a jour un 'el'ement de type T dans la base de donn'ees.

• display() : Cette m'ethode retourne une liste d''el'ements de type T, permettant d'afficher les donn'ees de la base.

Impl'ementation de HolidayDAOimpl

La classe HolidayDAOimpl implémente l'interface GenericDAOI<Holiday>. Elle est responsable de la gestion des congés des employés dans la base de données. Ci-dessous, vous trouverez la structure de cette classe ainsi qu'une description des principales méthodes qu'elle contient.

Code de la Classe HolidayDAOimpl

```
1 package DAO;

3 public class HolidayDAOimpl implements GenericDAOI<Holiday>{

5     @Override
6     public void add(Holiday e) {....}
7

8     @Override
9     public void delete(int id) {....}

10     @Override
12     public void update(Holiday e) {....}
13

14     @Override
15     public List<Holiday> display() {....}
16
```

Explication de la Classe HolidayDAOimpl

La classe HolidayDAOimpl impl'emente l'interface g'en'erique GenericDAOI<Holiday> et fournit les m'ethodes n'ecessaires pour g'erer les cong'es des employ'es dans la base de donn'ees.

- add(Holiday e): Cette m'ethode permet d'ajouter un nouveau cong'e dans la base de donn'ees.
- delete(int id) : Cette m'ethode permet de supprimer un cong'e existant en utilisant son identifiant.
- update(Holiday e) : Cette m'ethode met `a jour un cong'e existant dans la base de donn'ees.
- display(): Cette m'ethode retourne une liste de tous les cong'es enregistr'es dans la base de donn'ees.

je cacher le code de chaque function par ce que comme le file EmployeDAOimpl.

Classe Holiday

La classe Holiday repr'esente un cong'e d'un employ'e dans le syst'eme. Elle contient des informations essentielles telles que l'identifiant du cong'e, l'identifiant de l'employ'e, la date de d'ebut et de fin du cong'e, ainsi que le type de cong'e. Cette classe sert de mod'ele pour g'erer les donn'ees li'ees aux cong'es dans la base de donn'ees.

Code de la Classe Holiday

Voici l'impl'ementation de la classe Holiday:

```
1 package Model;
3 import java.sql.Date;
5 public class Holiday{
          private
                      int id holiday;
          private
                      int id_employe;
 8
          private
                      Date startDate;
          private
                      Date endDate;
10
          private
                      Type_holiday type;
11
                public Holiday(int id_holiday, int id_employe, Date startDate, Date endDate, Type_holiday type){
                this.id_holiday = id_holiday;
13
                this.id employe = id employe;
                this.startDate = startDate;
                this.endDate = endDate; 17 this.type = type;
16
          }
18
         public int getId holiday() { 21
                                             return
id_holiday;
23
         public Date getStartDate() { 25
                                             return
startDate;
          }
27
                public Date getEndDate() {
                return endDate;
31
                public Type_holiday getType() {
                return type;
                }
34
         public int getId_employe() { 37
                                             return
id_employe;
         }
         }
```

Classe HolidayModel

La classe HolidayModel gère la logique métier des congés. Elle interagit avec la classe HolidayDAOimpl pour effectuer des opérations sur les congés et réaliser des vérifications logiques avant d'ajouter, de supprimer ou de mettre à jour des congés dans la base de données. La méthode principale de cette classe est addHoliday, qui comprend plusieurs vérifications avant l'ajout d'un congé.

Code de la Classe HolidayModel

Voici l'impl'ementation de la classe HolidayModel:

```
1 package Model;
        public class HolidayModel {
        private HolidayDAOimpl dao;
              public HolidayModel(HolidayDAOimpl dao) {
6
              this.dao = dao;
              }
              public boolean addHoliday(int id, int id employe, Date startdate, Date enddate, Type holiday type,
              Employe targetEmploye) {
11
                                        if(startdate.after(enddate)) return false;
                                        if(startdate.equals(enddate)) return false;
                                        if(startdate.before(new Date(System.currentTimeMillis()))) return false;
14
                                        if(enddate.before(new Date(System.currentTimeMillis()))) return false;
                                        for (Holiday holiday : new HolidayModel(new HolidayDAOimpl()).
                                        displayHoliday()) {
                                        if (holiday.getId employe() == id employe) {
17
                                        if(startdate.after(holiday.getStartDate()) &&
18
                                       startdate.before(holiday.getEndDate())
                                                                                                              &&
                                       enddate.before(holiday.getEndDate())
                                                                                         &&
                                                                                                        enddate.
                                       after(holiday.getStartDate())){
                                        System.out.println("le cong s est superpose a
                                                 un autre cong");
                                        return false;
                                        }
21
                                        if(startdate.after(holiday.getStartDate()) &&
23
                                       startdate.before(holiday.getEndDate())
                                                                                                               &&
                                       enddate.after(holiday.getEndDate())){
                                        startdate = holiday.getEndDate();
24
                                        continue;
25
26
                                        if(startdate.before(holiday.getStartDate()) &&
                                                               enddate.after(holiday.getStartDate()) && enddate
                                               .before(holiday.getEndDate())){
                                        enddate = holiday.getStartDate();
28
                                        continue;
29
```

```
}
30
                                        break;
31
                                        }
32
                                        }
33
               long daysBetween = (enddate.toLocalDate().toEpochDay() startdate.toLocalDate().toEpochDay());
               if(daysBetween > targetEmploye.getSolde()) return false;
36
               EmployeController.updateSolde(targetEmploye.getId(),
                                                                          targetEmploye.getSolde()
                                                                                                              (int)
               daysBetween);
               Holiday e = new Holiday(id, id employe, startdate, enddate,
38
               dao.add(e);
               return true;
               }
41
               }
```

Explication de la M'ethode addHoliday

La m'ethode addHoliday g`ere l'ajout d'un cong'e pour un employ'e. Elle effectue plusieurs v'erifications avant d'ajouter le cong'e dans la base de donn'ees. Voici les 'etapes de cette m'ethode :

V'erification des dates:

- Si la date de d'ebut est apr'es la date de fin, la m'ethode retourne false.
- Si la date de d'ebut est 'egale a` la date de fin, la m'ethode retourne false.
- Si la date de d'ebut ou la date de fin est ant'erieure a` la date actuelle, la m'ethode retourne false.
- **V'erification des chevauchements de cong'es**: La m'ethode v'erifie si le nouveau cong'e chevauche un cong'e existant de l'employ'e. Si un chevauchement est trouv'e, la m'ethode retourne false. Si le cong'e est partiellement chevauch'e, la m'ethode ajuste les dates de d'ebut et de fin.
- **V'erification du solde de cong'es**: La m'ethode compare le nombre de jours du cong'e demand'e avec le solde de cong'es de l'employ'e. Si le solde est insuffisant, la m'ethode retourne false.
- **Mise `a jour du solde de cong´es**: Si toutes les v´erifications sont r´eussies, le solde de cong´es de l'employ´e est mis a` jour en soustrayant le nombre de jours demand´es.
- **Ajout du cong'e**: Enfin, un objet Holiday est cr'e'e et ajout'e a` la base de donn'ees via le HolidayDAOimpl.

Classe HolidayController

La classe HolidayController est responsable de la gestion des interactions entre l'interface utilisateur, le mod`ele de donn'ees et la logique m'etier des cong'es. Elle g`ere les actions de l'utilisateur telles que l'ajout, la suppression, la mise a` jour et l'affichage des cong'es dans l'interface.

Explication du Code

Cette classe interagit avec les vues et le mod`ele pour effectuer les op'erations sur les cong'es des employ'es. Voici les diff'erentes m'ethodes et leurs ro^les :

• HolidayController : Constructeur qui initialise les composants de la vue et lie les actions de l'utilisateur (ajout, mise `a jour, suppression, affichage) aux m'ethodes correspondantes de la classe.

```
public HolidayController(Employe_HolidayView view, HolidayModel model) {
    this.View = view;
    this.model_holiday = model;

    View.getdeleteButton_holiday().addActionListener(e -> deleteHoliday());
    View.getupdateButton_holiday().addActionListener(e -> updateHoliday());
    View.getaddButton_holiday().addActionListener(e -> addHoliday());
}
```

• addHoliday: Cette m'ethode permet d'ajouter un nouveau cong'e. Elle r'ecup`ere les informations saisies par l'utilisateur, v'erifie si l'employ'e existe, puis ajoute le cong'e via la m'ethode addHoliday du mod`ele HolidayModel. Si l'ajout r'eussit, un message de succ`es est affich'e.

```
private void addHoliday(){
               int id_employe = View.getId_employe();
               Date startDate = Date.valueOf(View.getStartDate());
               Date endDate = Date.valueOf(View.getEndDate());
               Type holiday type = View.getType holiday();
               View.viderChamps_ho();
               boolean addreussi = model holiday.addHoliday(0, id employe, startDate, endDate, type,
                      targetEmploye);
               if(addreussi) {
10
                                                  View.afficherMessageSucces("Holiday a bien ete ajoutee.
                          <mark>"</mark>);
                      displayHoliday();
               } else {
13
                                                  View.afficherMessageErreur("Holiday n'a pas ete ajoutee
                          .");
```

```
14 }
15 }
```

• displayHoliday: Cette m'ethode affiche tous les cong'es dans la vue sous forme de tableau. Elle r'ecup'ere la liste des cong'es `a partir du mod'ele, puis les affiche dans la table de la vue. Si aucun cong'e n'est trouv'e, un message d'erreur est affich'e.

• deleteHoliday : Permet de supprimer un cong'e s'electionn'e. Elle r'ecup'ere l'ID du cong'e s'electionn'e dans la table et le supprime via la m'ethode deleteHoliday du mod'ele. Si la suppression r'eussit, le solde de l'employ'e est mis 'a jour et un message de succ'es est affich'e.

```
private void deleteHoliday(){
                            int selectedrow = Employe_HolidayView.Tableau1. getSelectedRow();
                            if (selectedrow == -1) {
                            View.afficherMessageErreur("Veuillez selectionner une ligne.");
                            } else {
                            int id = (int) Employe_HolidayView.Tableau1.getValueAt( selectedrow, 0);
                            boolean deletereussi = model_holiday.deleteHoliday(id);
                            if(deletereussi){
                            EmployeController.updateSolde(id employe,solde+ olddaysbetween);
                            View.afficherMessageSucces("Holiday a bien ete
                               supprimer.");
                            displayHoliday();
11
                            } else {
                            View.afficherMessageErreur("Holiday n'a pas ete
13
                               supprimer.");
                            }
                            }
                            }
```

• updateHolidaybyselect : Cette m'ethode permet de charger les informations d'un cong'e s'electionn'e dans les champs de saisie pour les modifier. Elle r'ecup'ere les informations du cong'e s'electionn'e et les affiche dans les champs correspondants de la vue.

```
private void updateHolidaybyselect(){
int selectedrow = Employe_HolidayView.Tableau1. getSelectedRow();
if (selectedrow != -1) {
```

• updateHoliday : Permet de mettre a` jour un cong'e existant. Cette m'ethode v'erifie que l'utilisateur a s'electionn'e un cong'e a` modifier, puis r'ecup`ere les nouvelles donn'ees saisies et les envoie au mod`ele pour mettre a` jour le cong'e. Un message de succ`es ou d'erreur est affich'e selon le r'esultat de la mise `a jour.

```
private void updateHoliday(){
                     if (!test) {
                     View.afficherMessageErreur("Veuillez d'abord
                                s lectionner une ligne
                                                                      modifier.");
                     return;
                     }
                               updateSuccessful = model_holiday.updateHoliday(id, id_employe,
                     boolean
                     startDate, endDate, type, targetEmploye);
                     if (updateSuccessful) {
                     View.afficherMessageSucces("Holiday a t
                                                                     modifi avec succ s.");
                     displayHoliday();
                     View.viderChamps ho();
                     } else {
                     View.afficherMessageErreur("Erreur lors de la mise jour de holiday.");
13
                     }
```

Vue Employe HolidayView

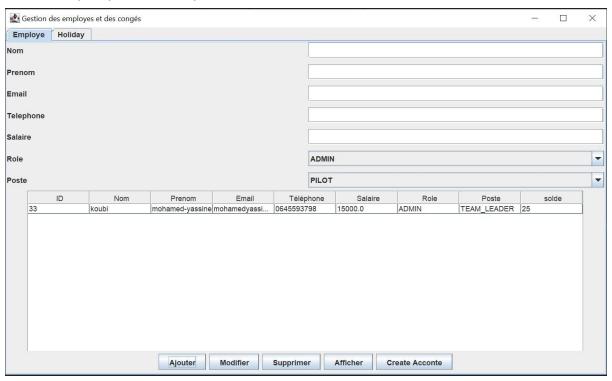


Figure 1: Ajouter un employe le solde de l'employe est 25.

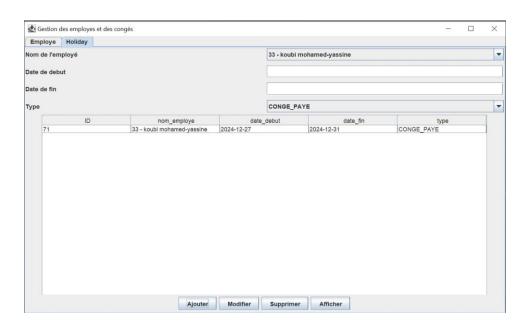


Figure 2: Ajouter un holiday le conge contient 4 days.

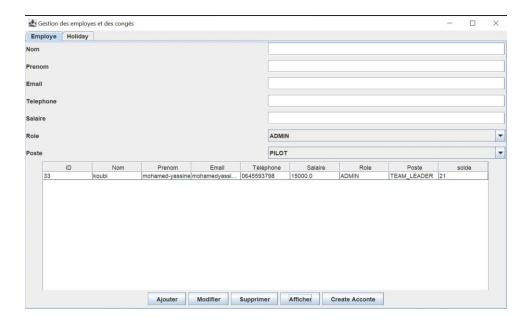
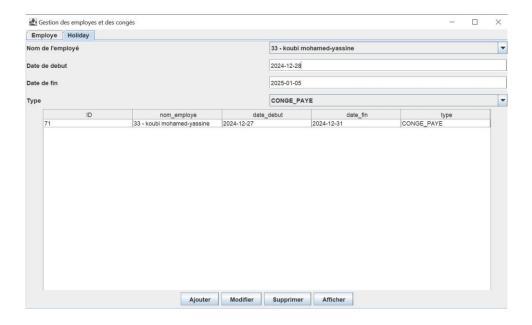


Figure 3: new solde



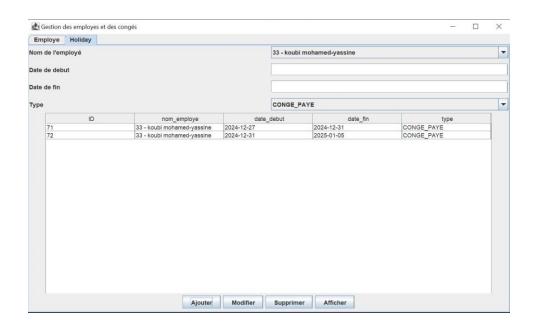


Figure 4: ajouter un conge qui debut au milieu de ancient conge et fini a une date depase le ancient fin date

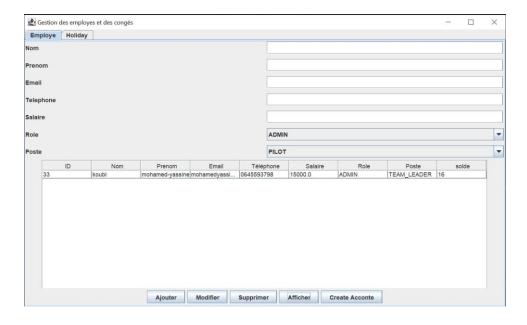


Figure 5: le solde devent 16 par ce que le conge debut au 31 jusqua le 5

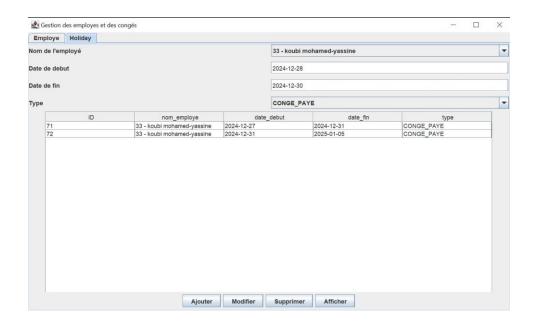


Figure 6: si le conge debut et fini en milieu d'un conge deja existe l'application donner un error

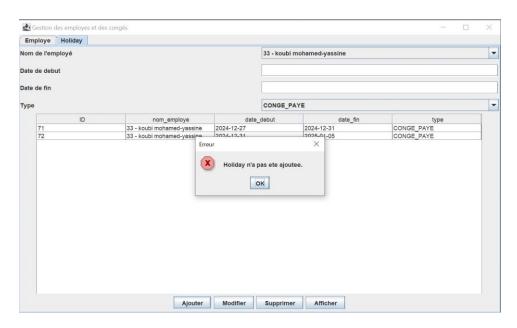


Figure 7: Error d'ajouter le conge'