

TP7 – Modification et suppression d'un objet

Pour le moment, nous pouvons afficher les données de la base et nous pouvons créer de nouvelles entrées dans la base. Mais nous ne pouvons pas encore modifier ou supprimer des données par l'application. Nous ne pouvons le faire que par l'interface phpMyadmin.

Dans ce TP nous allons mettre en place un formulaire de modification générique (qui sera prérempli), accessible par un lien dans l'affichage des objets. Nous disposerons aussi d'un lien de suppression. Avec ces nouvelles possibilités, nous aurons à notre disposition un CRUD complet (Create, Read, Update, Delete), ce qui est classique pour une application web digne de ce nom.

Exercice 1 – Le formulaire de modification d'un objet

Dupliquez votre TP6/ex5 en TP7/ex1.

1. Créez dans TP7/ex1/vue un fichier `formulaireModificationObjet.php` dont le code est pour le moment le même que celui du formulaire de création. Nous allons le modifier un peu par la suite...
2. Dans la méthode `lireObjets` du `ControleurObjet`, créer pour chaque objet un lien « Modifier » qui appellera le routeur avec comme contrôleur le `Controleur` spécifique comme action `afficherFormulaireModificationObjet`, et en donnant comme identifiant la valeur déjà utilisée dans le lien « Détails ».
3. Créez ensuite la méthode `afficherFormulaireModificationObjet` qui :
 - Récupère les attributs `static` nécessaires de la classe `Controleur` appelante ;
 - Récupère la valeur de l'identifiant ;
 - Charge depuis la table de données l'objet à modifié (grâce à l'identifiant) ;
 - Donne le bon titre ;
 - Insère les bonnes vues.
4. La plupart des entités (toutes sauf Adhérent) sont créées sans préciser la valeur de la clé primaire, puisqu'elle est en `AUTO_INCREMENT` dans la table de données. Seul le champ `login` de `Adherent` est renseigné au moment de la création.

Si on reprend la structure du formulaire de création pour l'adapter à la modification d'un objet, il va falloir faire attention à ce cas particulier du `login` d'un adhérent qu'on ne pourra pas modifier.

Pour réaliser ceci, on va faire **une légère entorse à notre schéma MVC** : la vue va réfléchir, pour simplifier les choses. On pourrait rester strictement dans le MVC et laisser tout choix au contrôleur, mais parfois un peu de souplesse ne fait pas de tort.

Voici le choix que fera la vue `formulaireModificationObjet.php` :

Si le champ dont on affiche la valeur n'est pas l'identifiant, alors la valeur est bien affichée dans un `<input>`, sinon elle est affichée dans un `<label>`.

Réalisez ceci. Vous devriez avoir un résultat semblable à ceci :

formulaire modification Auteur

webdev.iut-orsay.fr/~sgagne/LP/TP/TP7/ex1/routeur.php?controleur=controleur...

auteur adhérent livre nationalité genre catégorie

nom EINSTEIN

prénom Albert

année de naissance 1879

modifier

Bibliothèque 2022

Modification d'un auteur

formulaire modification Adherent

webdev.iut-orsay.fr/~sgagne/LP/TP/TP7/ex1/routeur.php?controleur=contr...

auteur adhérent livre nationalité genre catégorie

login musclor

mot de passe ****

nom CHABAL

prénom Sébastien

email sebastien.chabal@yopmail.com

date d'adhésion 05/10/2022

catégorie 2

modifier

Bibliothèque 2022

Modification d'un adhérent

5. Il reste un point important : dans les classes autres que l'Adherent, le problème est que l'identifiant n'est pas dans le formulaire, alors que dans le cas de l'Adherent, on a le login dans le formulaire, il sera donc transmis. Pour parer à ce souci d'identifiant manquant, ajoutez dans le formulaire un input de type « hidden » précisant l'identifiant :

```
<input type="hidden" name="identifiant" value="<?php echo ...; ?>">
```

Bien entendu les ... sont à remplacer par l'echo d'une variable fournie par le contrôleur. Vous changerez l'action du formulaire en « modifier... »

```
<input type="hidden" name="action" value="modifier<?php echo $table; ?>">
```

Réalisez tout ceci pour que chaque lien « modifier » aboutisse au formulaire de modification correctement prérempli.

Exercice 2 – Les méthodes updateAuteur de Auteur et modifierAuteur de ControleurAuteur

Dupliquez TP7/ex1 en TP7/ex2.

1. En utilisant une requête préparée, codez dans la classe Auteur la méthode

```
public static function updateAuteur($i,$n,$p,$a) {  
  
}
```

Cette méthode retournera true si l'insertion s'est bien passée, et false sinon.

2. Créez, dans la classe ControleurAuteur, une méthode

```
public static function modifierAuteur() {  
  
}
```

Cette méthode :

- donne sa valeur à \$titre,
 - récupère du tableau \$_GET les identifiant, nom, prénom et année de naissance de l'auteur à créer,
 - appelle la méthode updateAuteur du modèle,
 - affiche la liste des auteurs.
3. Testez la modification, vérifiez que tout fonctionne.

Exercice 3 – La même démarche pour les adhérents et les autres entités

Dupliquez TP7/ex2 en TP7/ex3.

Reproduisez exactement les étapes des exercices 1 et 2 pour créer une fonctionnalité de modification d'un adhérent, d'un livre, ...

Vérifiez que tout est modifiable maintenant (sauf les dates d'emprunt)

Exercice 4 – La fonctionnalité de suppression

Dupliquez TP7/ex3 en TP7/ex4.

Dans cet exercice, on met en place les fonctionnalités liées à la suppression d'un objet.

1. Dans la méthode `lireObjets` de `ControleurObjet`, ajoutez un lien « supprimer » qui pointe vers l'action `supprimerObjet` du contrôleur de l'entité en question, en gardant dans l'url l'identifiant de l'objet.



The screenshot shows a web browser window with the address bar displaying `webdev.iut-orsay.fr/~sgagne/LP/TP/TP7/ex4/routeur.php?controleur=ControleurAdherent&action=lireObjets`. The page content features a table with columns for 'auteur', 'adhérent', 'livre', 'nationalité', 'genre', and 'catégorie'. Below these columns, there is a list of library members, each with three links: 'supprimer', 'modifier', and 'détails'. The footer of the page reads 'Bibliothèque 2022'.

auteur	adhérent	livre	nationalité	genre	catégorie
Adherent jcvd	supprimer	modifier	détails		
Adherent jupiter	supprimer	modifier	détails		
Adherent lafleche	supprimer	modifier	détails		
Adherent lala	supprimer	modifier	détails		
Adherent musclor	supprimer	modifier	détails		
Adherent rico	supprimer	modifier	détails		
Adherent speedy	supprimer	modifier	détails		
Adherent theboss	supprimer	modifier	détails		

2. L'action `supprimerObjet` est facilement générique, car elle n'a besoin que du nom de la table et de l'identifiant de l'entrée à supprimer. Nous allons donc directement en proposer une version générique, en codant la méthode `supprimerObjet` dans `ControleurObjet`:

```
public static function supprimerObjet() {  
  
}
```

Cette méthode :

- Récupère les attributs `static` correspondant au nom de la table et à la clé primaire pour la classe contrôleur appelante ;
- Récupère la valeur de l'identifiant en provenance de l'url du lien supprimer ;
- Appelle la méthode `deleteObjetById` du modèle correspondant à la table en question, en lui passant comme argument la valeur de l'identifiant. Cette méthode sera codée dans la question suivante ;
- Affiche la liste de tous les objets en question.

3. De la même façon, il est très simple de coder une méthode générique `deleteObjetById`, alors que par exemple la méthode `addObjet` est compliquée à écrire (c'est pour cette raison que nous avons laissé les versions spécifiques `addAuteur`, `addLivre`, etc). De même, la méthode `updateObjet` n'a pas été tentée... Mais ça viendra peut-être...

Occupons-nous donc de la méthode `deleteObjetById`.

Cette méthode :

- Récupère les attributs `static` correspondant au nom de la table et à la clé primaire pour la classe contrôleur appelante ;
- Prépare la requête de suppression sur la table récupérée, pour la valeur de l'identifiant passée dans l'url du lien ;
- Lance la requête.

4. Vérifiez, après transfert des fichiers, que tout fonctionne en supprimant divers objets par clic sur les liens « supprimer ».

A ce stade de la suite de TP, nous pouvons lire, créer, modifier ou supprimer des données par l'interface de notre site web. C'est un point essentiel. Dans le TP suivant nous ferons les liens entre les différentes classes, par les tables de données `estDeNationalite`, `estAuteurDe` et `emprunte`.

En effet, pour le moment, quand on enregistre un livre, on ne sait pas qui est l'auteur... C'est un peu frustrant...