# TP12 – Sécurisation du site

Le menu « non administrateur » est bien adapté, mais personnaliser les menus n'est qu'une action cosmétique et superficielle. Même si ce n'est pas directement accessible par son menu, un non administrateur peut très bien par l'url accéder aux fonctionnalités de création, ce qui est anormal.

Pour s'en convaincre, il suffit de changer à la main dans l'url l'action du `controleurAdherent` en `afficherFormulaireCreationObjet`.

Ceci prouve que les contrôles doivent absolument être effectués au niveau des actions du contrôleur. Certaines actions seront réservées à l'administrateur. Pour sécuriser au mieux les actions, il convient de les factoriser au maximum (pour centraliser la sécurisation). Ceci a été fait au TP précédent.

Pour sécuriser le site, dans les divers contrôleurs, chaque action sera lancée A CONDITION QUE la personne connectée ait le statut nécessaire (administrateur, adhérent connecté).

Dupliquez votre code `TP11/ex2` en `TP12/ex1`.

## Exercice 1 – Sécurisation de l'affichage des liens dans la méthode lireObjets

1. Modifiez, dans `lireObjets`, la création de l'affichage selon l'algorithme suivant :

```
SI l'adhérent connecté est admin
      les liens « détails », « modifier » et « supprimer » seront affichés
      avec les liens éventuels supplémentaires pour les livres et auteurs
SINON
      SI la classe n'est pas « Adhérent »
            Seul le lien « détails » sera affiché
      SINON
            SI le login de l'adhérent à afficher est égal au login de session
                  Les liens « détails » et « modifier » seront affichés
            SINON
                  Seul le lien « détails » sera affiché
            FSI
      FSI
FSI
```

   En gros : un admin a droit à l'affichage total, alors qu'un adhérent connecté a droit à un affichage partiel (juste les détails), sauf dans le cas particulier où on affiche les adhérents : l'adhérent connecté peut se modifier lui et lui seul.

   Pour tout cela, vous utiliserez les méthodes de la classe `Session`.

2. Transférez et vérifiez l'affichage produit par `lireObjets` est bien adapté au statut de l'adhérent connecté (admin ou non). Vérifiez en particulier que pour un non admin, seul SON lien « modifier » est affiché.

3. Vérifiez que malgré tout, un adhérent peut, en modifiant directement l'url, arriver au formulaire de création d'un adhérent, en créer un, ou encore accéder au formulaire de modification d'un autre adhérent, et le modifier…

   CONCLUSION : sécuriser les liens est insuffisant… Le travail n'est pas terminé !

## Exercice 2 – Sécurisation de l'accès au formulaire de création d'un objet

Dupliquez votre code `TP12/ex1` en `TP12/ex2`.

1. Le formulaire de création d'un objet n'est destiné qu'à l'administrateur. Grâce à la fonction `Session::adminConnected` de la classe `Session`, sécurisez la méthode `afficherFormulaireCreationObjet` du `controleurObjet` :

   - Si l'utilisateur n'est pas admin, alors on affiche la page de tous les objets en question,
   - Sinon (cas d'un admin), l'adhérent admin a accès au formulaire.

2. Transférez et vérifiez que maintenant, un utilisateur non admin ne peut plus accéder au formulaire de création d'un objet en utilisant directement l'url.

## Exercice 3 – Sécurisation de la méthode de création d'un objet

Dupliquez votre code `TP12/ex2` en `TP12/ex3`.

Attention… Si vous pensez que la création d'objet est maintenant sécurisée, c'est raté…

En effet, nous avons seulement sécurisé

- le plus visible (les liens vers les formulaires),
- puis le moins visible (l'accès au formulaire via l'url).

Mais nous n'avons pas sécurisé l'invisible : en effet, qu'est-ce qui empêche un adhérent non admin d'accéder directement, par l'url, à la méthode `controleurObjet::creerObjet` en lui donnant dans l'url tous les renseignements nécessaires ?

1. Connectez-vous en tant qu'adhérent non admin, puis vérifiez, en agissant directement sur l'url, que vous pouvez créer un nouveau livre en appelant directement dans l'url l'action et le contrôleur adéquats, et en donnant les renseignements nécessaires à la création de ce livre. Refaites la même opération pour créer un nouveau genre.
   Il est ANORMAL qu'un non admin puisse faire ceci ! C'est possible car la méthode `creerObjet` n'est pas encore sécurisée.

2. Remédiez au problème en sécurisant la méthode `creerObjet` de façon que :

   - Si l'adhérent n'est pas admin, alors on affiche la page de tous les objets en question,
   - Sinon (cas d'un admin), l'adhérent admin crée l'objet en question et affiche la nouvelle liste des objets.

3. Connectez-vous en tant qu'adhérent non admin et retentez, via l'url, de créer un nouveau livre. Si vous aboutissez à la liste sans que votre tentative aboutisse, alors vous avez correctement sécurisé tous les accès à la création d'un objet.

## Exercice 4 – Sécurisation de l'action de suppression d'un objet

Dupliquez votre code `TP12/ex3` en `TP12/ex4`.

1. L'affichage de toutes les objets ne produit plus de lien supprimer, pour un adhérent non admin. Est-ce suffisant ? Connectez-vous en tant qu'adhérent non admin, puis choisissez un livre que vous n'aimez pas et ensuite, directement dans l'url, lancez la méthode `supprimerObjet` avec comme identifiant la valeur du `numLivre` le concernant.

   Est-ce que la disparition du lien « supprimer » sécurise suffisamment ?

2. Sécurisez la méthode `supprimerObjet` ainsi :

   - Si l'adhérent connecté n'est pas admin, alors on affiche la liste des objets,
   - S'il est admin, alors l'objet est supprimé et on affiche la nouvelle liste des objets.

3. Vérifiez que maintenant, un utilisateur non admin ne peut plus supprimer un objet en passant par l'url.

## Exercice 5 – Sécurisation de l'affichage du formulaire de modification d'un objet

Comme le formulaire de création, le formulaire de modification doit être sécurisé. Mais c'est plus compliqué. En effet, toutes les créations étaient réservées à l'administrateur (sauf la création de compte par l'internaute).

Maintenant, il faut tenir compte du fait qu'un adhérent non admin connecté pourra accéder à la modification de son profil (seulement le sien) alors que l'administrateur aura tous les accès.

Nous allons prendre les problèmes les uns après les autres…

Dupliquez votre code `TP12/ex4` en `TP12/ex5`.

1. Connectez-vous en tant qu'adhérent non admin. Ensuite, dans l'url, demandez l'action `afficherFormulaireModificationObjet` du `ControleurAdherent`, avec un identifiant différent de celui de connexion. Vérifiez que vous accédez à la page de modification, et qu'en plus vous pouvez modifier le profil en question.

2. Modifiez l'algorithme de la méthode `afficherFormulaireModificationObjet` du `controleurObjet` de la façon suivante :

```
SI     (l'utilisateur est admin) ou
       (la table est Adherent et le login de session = l'identifiant à modifier)
       Afficher le formulaire de modification
SINON
       Afficher le profil correspondant à l'identifiant passé en get (détails)
FSI
```

3. Vérifiez que maintenant, un utilisateur non admin ne peut pas accéder à un autre formulaire de modification que le sien, même s'il essaie de passer par une url changée à la main. Vérifiez aussi qu'il ne peut pas accéder à un formulaire de modification d'un livre par exemple.

## Exercice 6 – Sécurisation de la méthode modifierObjet

Dupliquez votre code `TP12/ex5` en `TP12/ex6`.

1. Connectez-vous en tant qu'adhérent non admin. Ensuite, dans l'url, demandez l'action `modifierObjet` du `ControleurLivre`, avec un `identifiant` connu dans la table des livres, et passez dans l'url un `titre`, une `anneeParution` et un `numGenre` changés. Vérifiez que la modification a bien été enregistrée dans la base de données…

2. Remédiez à cette anomalie en sécurisant la méthode `modifierObjet`. Les contrôles de sécurité sont les mêmes que pour l'affichage du formulaire de modification.

3. Retentez une modification d'un livre via l'url par un adhérent non admin. Normalement, ce n'est plus possible.

## Exercice 7 – Sécurisation des dernières méthodes

Dupliquez votre code `TP12/ex6` en `TP12/ex7`.

1. L'action `lireObjet` n'est pas critique. Nous allons néanmoins la réserver aux adhérents connectés (et donc aussi à l'admin qui est un adhérent particulier). Sécurisez cette action comme demandé, grâce aux fonctions de la classe `Session`.

2. Vérifiez qu'un adhérent connecté et non admin peut supprimer des auteurs d'un livre, en agissant via l'url. Sécurisez l'action `supprimerAuteurDuLivre` du `ControleurLivre` pour que seul l'administrateur puisse y accéder. Vérifiez ensuite que la suppression précédente n'est plus possible.

3. De la même façon, sécurisez `definirAuteurs` et `ajouteurAuteurDuLivre` pour que seul l'administrateur y ait accès.

4. Sécurisez de la même façon les trois méthodes spécifiques de `ControleurAuteur`.

5. Il reste les fonctions spécifiques du `ControleurAdherent`. Quelles sont celles qui doivent être sécurisées ? Si vous en trouvez, faites ce qu'il faut faire…