IUT de Montpellier - Programmation Web - TD3
Architecture MVC

Rémi Coletta, Romain Lebreton, Bastien Vialla

Semaine du 7 Octobre 2014

Si vous programmez de manière monolithique, vos pages actuelles mélangent traitement (couche d'accès aux données) et présentation (balises HTML), *et c'est le mal !*

L'objectif de ce TD est donc de réorganiser le code du TD 2 pour finalement y rajouter plus facilement de nouvelles fonctionnalités.

Q 1. Récupérez sur `http://www.lirmm.fr/~lebreton/teaching.html` l'archive TD3.zip qui vous servira de base pour ce TD. Décompressez cette archive dans votre `public_html` et créez un nouveau projet 'Php à partir de sources existantes' si vous utilisez NetBeans. Rentrez vos informations dans `./config/Conf.php`.

1 M : Le modèle

Le modèle est chargé de la gestion des données, notamment des interactions avec la base de données. C'est, par exemple, la classe Voiture que vous avez créé lors des TD précédents. Nous avons renommé la classe en `ModelVoiture` et déplacé le fichier dans `./model/modelVoiture.php`. Dans notre cas, la nouvelle classe ModelVoiture gère la persistance, au travers des méthodes :

```
1  $mv->save():
2  $mv2 = ModelVoiture::getVoiture($immatriculation);
3  $arrayVoitures = ModelVoiture::getAllVoitures();
```

N.B. : Souvenez-vous que les deux dernières fonctions `getVoiture` et `getAllVoitures` sont `static`. Elles ne dépendent donc que de leur classe (et non pas des objets instanciés). D'où la syntaxe différente `Class::fonction_statique()` pour les appeler.

2 V : la vue

Ce avec quoi l'utilisateur interagit se nomme précisément la vue. Sa première tâche est d'afficher la page Web à l'utilisateur. La vue reçoit aussi toute les actions de l'utilisateur (remplissage de formulaire dans notre cas) et les envoient au contrôleur.

Les vues sont des fichiers qui ne contiennent quasiment exclusivement que du code HTML, à l'exception de quelques `echo` permettant d'afficher les variables pré-remplies par le contrôleur.

La vue n'effectue pas de traitement, elle se contente d'afficher les résultats des traitements effectués par le modèle et d'interagir avec l'utilisateur. Une boucle `for` est toutefois autorisée pour les vues qui affichent une liste d'éléments.

3 C : le contrôleur

Le contrôleur gère le tout ; il reçoit les actions de l'utilisateur, lit ou met à jour les données à travers le modèle, puis appelle la vue adéquate. Il existe une multitude d'implémentations du MVC :

— un gros contrôleur unique
— un contrôleur par modèle
— un contrôleur pour chaque action de chaque modèle

Nous choisissons ici la version intermédiaire. Vous trouverez dans `./controller/controllerVoiture.php` un squelette de contrôleur basé sur le contrôleur suivant :

```php
<?php
include ('Voiture.php');  //connait le modele
$action = $_GET['action'];    //recupere l'action  passee dans l'url

switch ($action) {
    case "get":
        $im = $_GET['immatriculation']; //recupere l'immatriculation passee dans l'url
        $v = Voiture::getVoiture($im);
        if ($v == null)
            require ('viewErrorVoiture.php'); //redirige vers une vue d'erreur
        else
            require ('viewVoiture.php');  //redirige vers la vue
        break;
    case "all":
        $tabVoitures = Voiture::getAllVoitures();
        require ('viewAllVoiture.php');
        break;
    case "save":
        // A vous de remplir cette partie
        break;
    default:
        require ('viewErrorVoiture.php'); //redirige vers une vue d'erreur
}

?>
```

4 À vous de jouer

4.1 Vue "détail d'une voiture"

Q 2. Remplissez la vue `./view/voiture/viewVoiture.php`

Q 3. Testez cette vue en appelant la page du contrôleur avec les bons paramètres dans l'URL.
(Souvenez-vous comment les formulaires utilisant la méthode "GET" écrivent les paramètres dans l'URL)

Q 4. Testez cette vue en appelant le contrôleur via un formulaire. Un squelette de formulaire vous est donné dans `./exempleFormulaire.html`

Q 5. Remplissez la vue `./view/voiture/viewErrorVoiture.php` pour gérer les immatriculations non reconnues

4.2 Vue "liste des voitures"

Q 6. Remplissez la vue `./view/voiture/viewAllVoiture.php`

Q 7. Testez cette vue en appelant la page du contrôleur avec les bons paramètres dans l'URL

Q 8. Dans la vue `viewAllVoiture.php` ajoutez un lien cliquable pour chaque voiture, qui renvoie vers la page de détail `viewVoiture.php`

4.3 Vue "ajout d'une voiture"

Q 9. Complétez l'action `save` de `./controller/controllerVoiture.php` pour que le contrôleur sauvegarde dans la base de donnée la voiture donnée en paramètre dans l'URL et appelle la future vue `viewCreateVoiture.php`

Q 10. Remplissez la vue `./view/voiture/viewCreateVoiture.php` pour signifier que la création a bien eu lieu

Q 11. Testez le contrôleur et la vue via un formulaire

5 Et si le temps le permet...

Q 12. Rajouter une fonctionnalité "Supprimer une voiture" à votre site. Ajouter un lien cliquable pour supprimer chaque voiture dans la liste des voitures.

Q 13. Gérer le cas particulier de l'ajout d'une voiture existant déjà dans la base de donnée.

Q 14. Rajoutons des comportements par défaut.

Nous voudrions qu'un utilisateur qui navigue à la racine du site arrive automatiquement sur la page du contrôleur. Créer pour cela un `index.php` à la racine qui charge seulement la page `./controller/controllerVoiture.php` par un require.

Nous voudrions aussi que le contrôleur exécute l'action "all" si aucune action n'est spécifiée dans les paramètres de l'URL. Utiliser la fonction `isset($_GET['action'])` pour déterminer si une action a été donnée.

Q 15. Gérer les options des voitures : Créer une table options dans votre base de données avec trois champs 'id_option', 'immatriculation' et 'option'.

Le champ 'id_option' sera un entier automatiquement incrémenté à chaque insertion d'option. On obtient ce comportement en cochant la case 'A_I' (auto_increment) lors de la création de la table et en insérant la valeur NULL à la colonne 'id_option' lors d'un ajout d'option.

Mettre à jour les fonctions `save()`, `getVoiture($im)` et `getAllVoitures()` pour qu'elles prennent en charge les options. Mettre aussi à jour la vue de détail pour qu'elle affiche les options.

Q 16. Créer dans le répertoire view deux fichiers `header.php` et `footer.php`. Le header correspond à l'en-tête de votre site qui ne varie pas selon la page. Vous pouvez par exemple le remplir avec une liste de lien vers les différentes actions (liste, ajout, recherche) sur vos voitures. Le footer pourrait être un simple bandeau avec votre nom, un copyright et un lien pour vous écrire.

Charger ces fichiers respectivement au début et à la fin du `<body>` de toutes vos vues.