DokuMission

Présentation

Il s'agit d'adapter pour l'entreprise de créer un gestionnaire de documentation, qui permettra aux salariés de gérer/créer puis consulter/rechercher dans une documentation partagée.

Outils de développement

- PHP/Mysql
- Doctrine pour le mappage relationnel/Objet
- CodeIgniter pour la mise en place MVC

Documents

Script de la base de données

docu.sql

Diagramme de cas d'utilisation



Gérer Utilisateurs / Gérer Mondes

Travail fourni

Libelle	Document
Base de données	docu.sql
Documentation technique	docu.pdf
Application pour la classe + docu.sql	docu.zip

Modèle

Travail sur 3 classes métiers:

1. Utilisateur
2. Monde
3. Groupe

- Un Monde peut contenir zéro, un, ou plusieurs Utilisateurs
- Un Utilisateur appartient à un Monde
- Un Groupe peut contenir zéro, un, ou plusieurs Utilisateurs
- Un Utilisateur appartient à un groupe

Vue

Voici les vues utilisées:

Méthodes	Description
VAdmin	Page principal avec utilisateurs et mondes
VAddUser	Vue permettant l'ajout d'un nouvel utilisateur
VAddMonde	Vue permettant l'ajout d'un nouveau monde
VUpdateUser	Modification d'un utilisateur existant
VUpdateMonde	Modification d'un monde existant

Contrôleur

La gestion des Utilisateurs et Mondes est effectuée dans le contrôleur CAdmin . Le contrôleur possède plusieurs méthodes:

Méthodes	Description
index	Méthode appelée par défaut qui appelle la fonction resfresh
refresh	Appelée par l'index, cette méthode est utilisé pour l'affichage initial et le rafraîchissement après une modification
deleteUser	Supprimer un utilisateur
deleteMonde	Supprimer un monde
viewAddUser	Affichage vue VAddUser
addUser	Ajouter un utilisateur
viewAddMonde	Affichage vue VAddMonde
addMonde	Ajouter un monde

Méthodes	Description
viewUpdateUser	Affichage vue VUpdateUser
updateUser	Modification d'un utilisateur
viewUpdateMonde	Affichage vue VUpdateMonde
updateMonde	Modification d'un monde

Librairie

Appelée dans le constructeur : *$this→load→library('Modelutils');*

Utilisation de la librairie **ModelUtils**, plusieurs fonctions ont été créées.

- getAllUsers (Récupère tous les utilisateurs)
- getUserWithId (Récupère un utilisateur avec son Id)
- getAllMondes (Récupère tous les mondes)
- getMondeWithId (Récupère un monde avec son Id)
- getMondeWithLibelle (Récupère un monde avec son libellé)
- getAllGroupes (Récupère tous les groupes)
- getAllDomaines (Récupère tous les domaines)
- getGroupeWithId (Récupère un monde grâce à son Id)
- getGroupeWithLibelle (Récupère un groupe avec son libelle)

```
<?php

class ModelUtils{

    private $ci;
    public function __construct(){
        $this->ci =& get_instance();
    }

    public function getAllUsers(){
        $query = $this->ci->doctrine->em->createQuery("SELECT u FROM Utilisateur
u");
        return $query->getResult();
    }

    public function getUserWithId($param){
        $query = $this->ci->doctrine->em->createQuery("SELECT u FROM Utilisateur u
WHERE u.id='".$param."'");
        return $query->getSingleResult();
    }

    public function getAllMondes(){
        $query = $this->ci->doctrine->em->createQuery("SELECT m FROM Monde m");
        return $query->getResult();
    }

    public function getMondeWithId($param){
        $query = $this->ci->doctrine->em->createQuery("SELECT m FROM Monde m WHERE
m.id='".$param."'");
        return $query->getSingleResult();
    }
```

```
    public function getMondeWithLibelle($param){
        $query = $this->ci->doctrine->em->createQuery("SELECT m FROM Monde m WHERE
m.libelle='".$param."'");
        return $query->getSingleResult();
    }

    public function getAllGroupes(){
        $query = $this->ci->doctrine->em->createQuery("SELECT g FROM Groupe g");
        return $query->getResult();
    }

    public function getAllDomaines(){
        $query = $this->ci->doctrine->em->createQuery("SELECT d FROM Domaine d");
        return $query->getResult();
    }

    public function getGroupeWithId($param){
        $query = $this->ci->doctrine->em->createQuery("SELECT g FROM Groupe g WHERE
g.id='".$param."'");
        return $query->getSingleResult();
    }

    public function getGroupeWithLibelle($param){
        $query = $this->ci->doctrine->em->createQuery("SELECT g FROM Groupe g WHERE
g.libelle='".$param."'");
        return $query->getSingleResult();

    }
}
```

From:
http://slamwiki2.kobject.net/ - **SlamWiki 2.1**

Permanent link:
http://slamwiki2.kobject.net/etudiants/2014/matthias.lecomte/docu?rev=1419777566

Last update: **2019/08/31 14:32**