BugReport : Application de remontée d'incidents

- Equipes de Développement

Révisions

- technics.zip

```
class Groupe extends BaseObject{
    private $libelle;

    /**
     * @OneToMany(mappedBy="groupe",className="Utilisateur")
     */
    private $utilisateurs;
    /**
     * @ManyToMany(targetEntity="Module", inversedBy="groupes")
     * @JoinTable(name="droit")
     */
    private $modules;
}
```

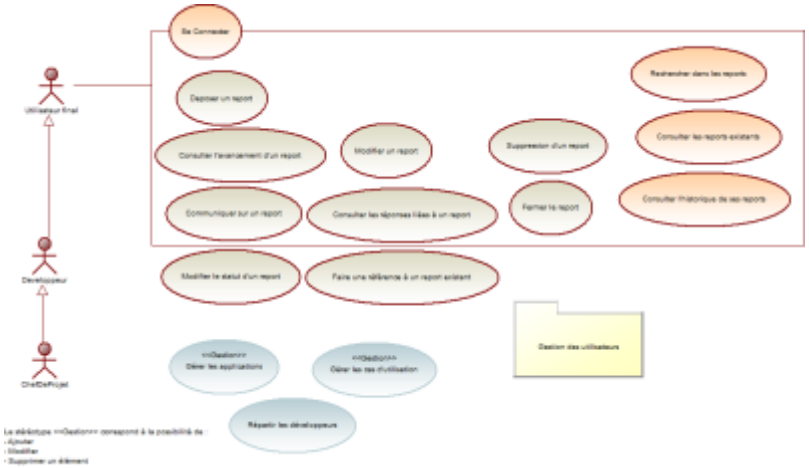
Travail à fournir

Pour le 2 décembre :

1. Documents d'analyse (diagramme de classes, diagramme de cas d'utilisation, de séquence, descriptifs textuels)
2. Archive zip de l'application Web + Script de la BDD
3. Mise en place de tests fonctionnels (pour les cas d'utilisation cités)
4. Documentation utilisateur
5. Documentation technique (sur le modèle ci-dessous)

Analyse fonctionnelle

Diagramme des cas d'utilisation



Analyse des données

- Règles de gestion
- Script de création de la Base de données bugReport (Version du Jeudi 7 novembre)

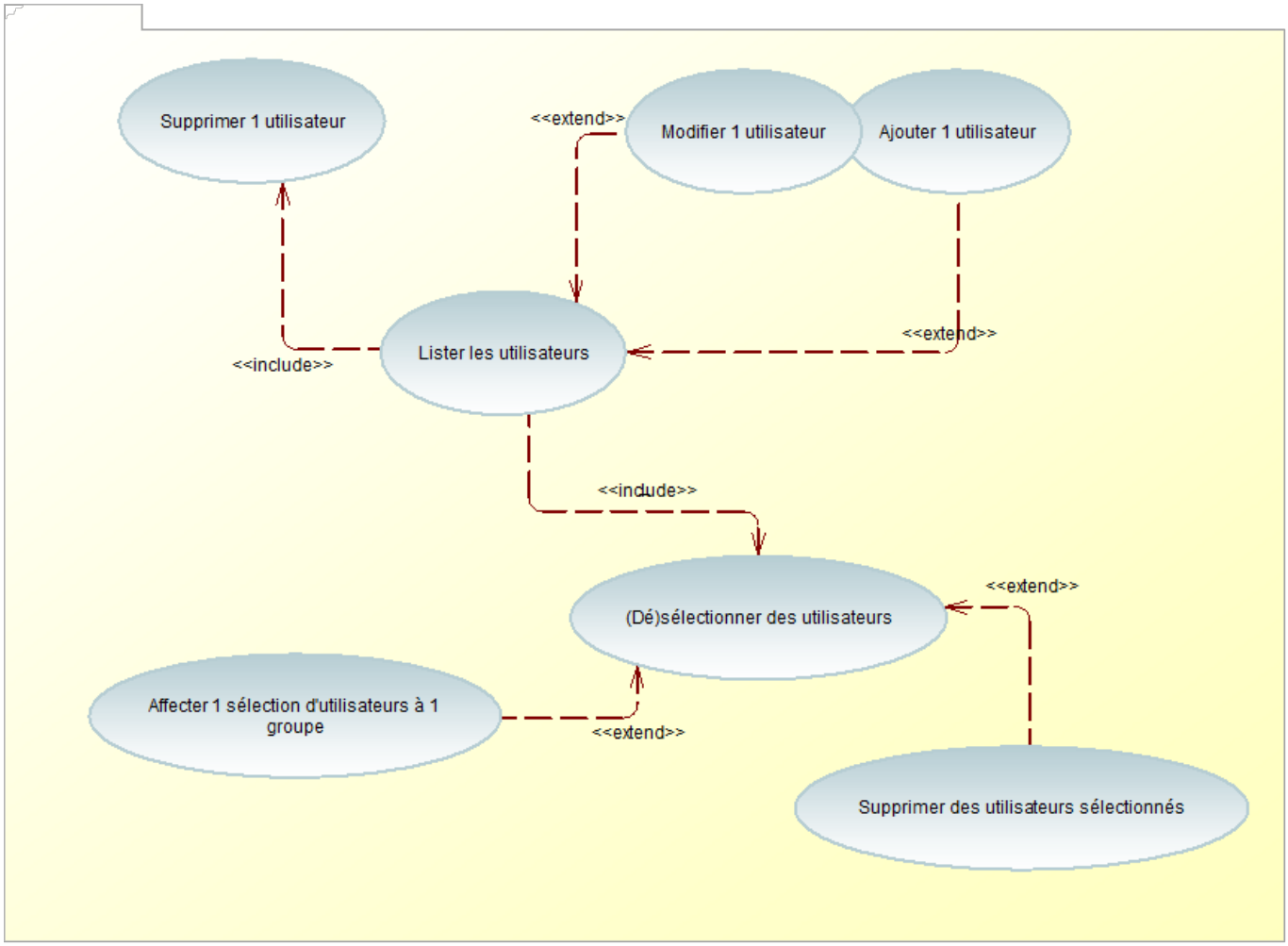
Gestion des utilisateurs : Exemple d'implémentation

-- Téléchargement des sources

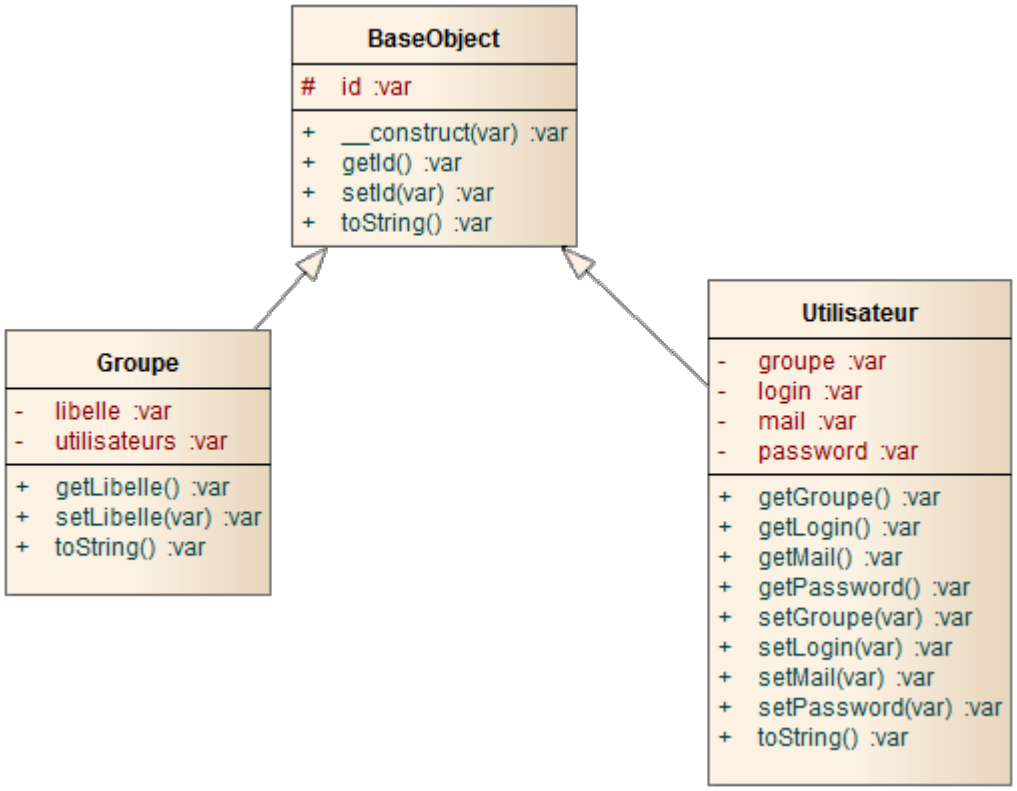
- Projet à intégrer dans Eclipse (File/import/existing project into workspace)

-- Fonctionnalités

Gestion des utilisateurs



-- Classes métier



-- Variables de session ($_SESSION)

Nom	Type	Descriptif
$_SESSION["ckAll"]	Booléen	Etat de la case à cocher permettant de sélectionner ou de désélectionner tous les utilisateurs
$_SESSION["selectedUsers"]	Tableau d'entiers	Ids des utilisateurs sélectionnés

-- Vues

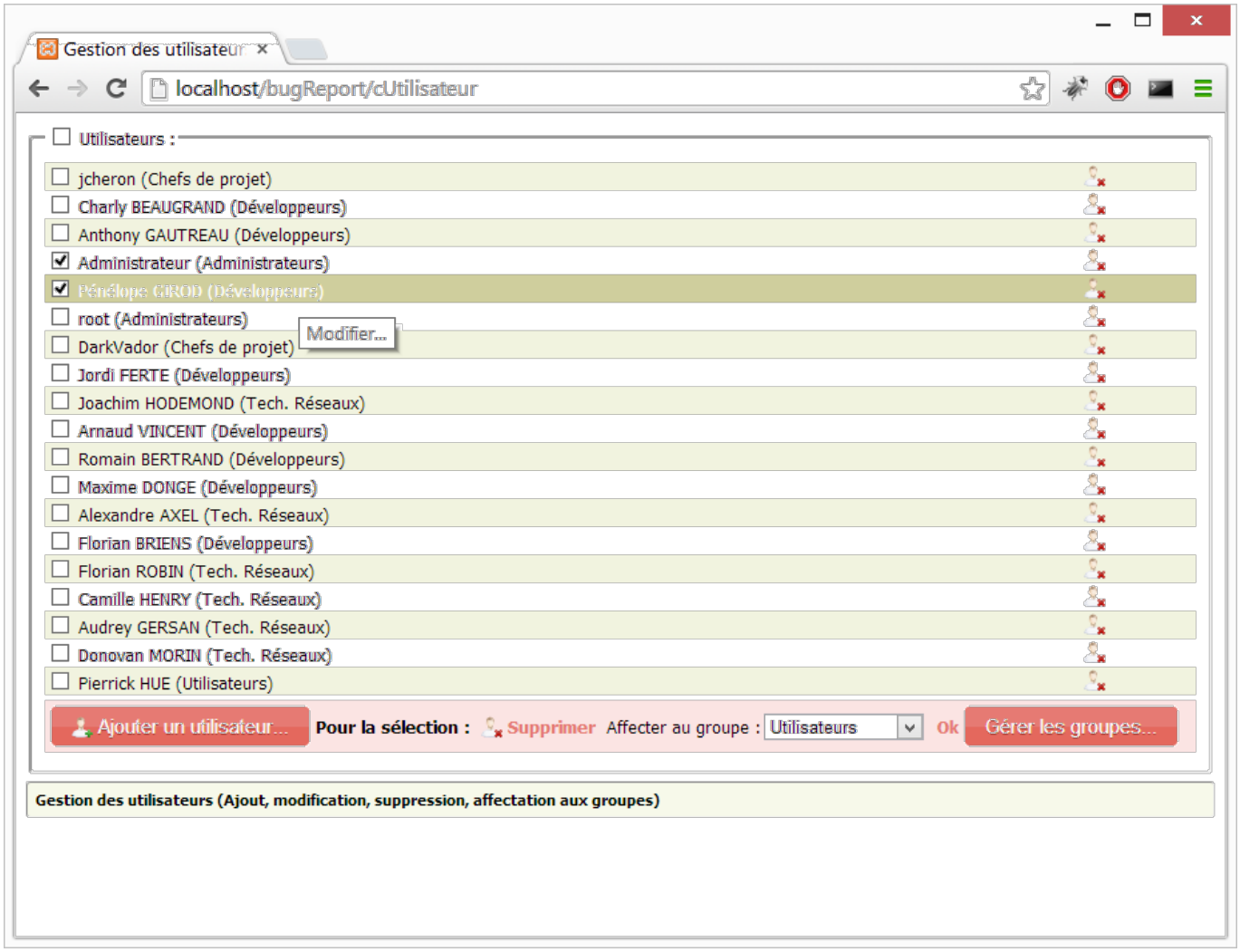
-- vHeader

Page en-tête

```
<?php
require_once 'technics/Gui.php';
?>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<script type="text/javascript" src="<?php echo
$GLOBALS["siteUrl"]?>js/jquery-2.0.3.js"></script>
<script type="text/javascript" src="<?php echo
$GLOBALS["siteUrl"]?>js/jquery.validate.js"></script>
<link rel="stylesheet" type="text/css" href="<?php echo
$GLOBALS["siteUrl"]?>css/main.css">
<title>Gestion des utilisateurs</title>
</head>
<body>
```

-- vUsers

Liste des utilisateurs (Affichage initial et rafraîchissement)




```
<div id="users">
    <fieldset class="cadre">
    <legend><input type="checkbox" id="ckAll" title="(Dé)sélectionner les
utilisateurs"> Utilisateurs :</legend>
        <table id="usersTable">
            <?php Gui::show($data["users"],"addDelete")?>
        </table>
        <div class="buttons">
            <div class="btn" id="btAdd" name="btAdd" title="Ajouter un nouvel
utilisateur"><span class="add">Ajouter un utilisateur...</span></div>
            <div id="multi"><b>Pour la sélection :</b> 
                <a href="#" class="delete" id="btMultiDelete" name="btMultiDelete"
title="Supprimer les utilisateurs sélectionnés">Supprimer</a> 
                Affecter au groupe :<select id="multiGroupe"><?php echo
Gui::show($data["groupes"],"select");?></select>
                 <a href="#" class="" id="btMultiToGroup"
name="btMultiToGroup" title="Affecter les utilisateurs sélectionnés au
groupe">Ok</a>
            </div>
            <div class="btn" id="btGroupes" name="btGroupes" title="Non
implémenté"><span class="">Gérer les groupes...</span></div>
        </div>
    </fieldset>
</div>
```

```
<?php echo $data["js"];?>
```

-- vFooter

Page pied-de-page

```
<div id="operation" style="display: none;"></div>
<div id="message">Gestion des utilisateurs (Ajout, modification, suppression,
affectation aux groupes)</div>
</body>
</html>
```

-- vAddUser

Ajout d'un nouvel utilisateur ou modification d'un utilisateur existant


```
<?php $user=$data["user"];?>
<form id="frmAddUser" name="frmAddUser" onsubmit="return false;">
<fieldset>
<legend><?php echo $data["title"];?></legend>
    <input type="hidden" id="id" name="id" value="<?php echo $user->getId();?>">
    <label class="label" for="login">Login* :</label><input title="Login
obligatoire" placeholder="Votre login" type="text" id="login" name="login"
value="<?php echo $user->getLogin();?>" maxlength="30" required><br>
    <label class="label" for="password">Password* :</label><input title="Password
obligatoire" type="password" id="password" name="password" value="<?php echo
$user->getPassword();?>" maxlength="10" required><br>
    <label class="label" for="mail">Email :</label><input title="Entrer un mail
valide" type="email" id="mail" name="mail" value="<?php echo
$user->getMail();?>"><br>
    <label class="label" for="groupe">Groupe :</label><select title="Sélectionner
un groupe" size="1" id="idGroupe" name="idGroupe"><?php echo
Gui::show($data["groupes"],"select");?>"></select><br>
    <div class="buttons">
        <div id="btValidAddUser" class="btn">Valider</div>
        <div id="btCancelAddUser" class="btn">Annuler</div>
    </div>
</fieldset>
</form>
<?php echo $data["js"];?>
```

-- Contrôleurs

-- cUtilisateur/index

Fonctionnalité	Descriptif
Gestion des utilisateurs	**Contrôleur de gestion des utilisateurs**
	Affichage de la vue vHeader
	Appel du contrôleur refresh
	Affichage de la vue vFooter

```
    public function index(){
        $this->loadView("vHeader");
        $this->refresh();
        $this->loadView("vFooter");
    }
```

-- cUtilisateur/refresh

Fonctionnalité	Descriptif
Liste des utilisateurs	**Utilisé pour l'affichage initial et le rafraîchissement après une modification**
	Chargement des utilisateurs $users
	Application de la sélection d'utilisateurs
	Mise en place logique client #btAdd -> add .delete -> confirmDelete .update -> update .ck -> select #ckAll -> selectAll #deleteMulti -> confirmDelete #btMultiToGroup -> addUsersToGroup #btGroupes -> non implémenté
	Passage de variables($data[]) et affichage de la vue vUsers

```
    public function refresh(){
        $users=DAO::getAll("Utilisateur");
        $selectedUsers=SessionUtils::getArray("selectedUsers");
        $js=JsUtils::getAndBindTo("#btAdd", "click",
"cUtilisateur/add","{}","#operation");
        $js.=JsUtils::getAndBindTo(".delete", "click",
"cUtilisateur/confirmDelete","{}","#operation");
        $js.=JsUtils::getAndBindTo(".update", "click",
"cUtilisateur/update","{}","#operation");
        $js.=JsUtils::getAndBindTo(".ck", "click",
"cUtilisateur/select","{}","#message");
        $js.=JsUtils::setChecked("ck", $selectedUsers);
        $js.=JsUtils::setChecked("ckAll", SessionUtils::getBoolean("ckAll"));
        $js.=JsUtils::getAndBindTo("#deleteMulti", "click",
"cUtilisateur/confirmDelete/multi","{}","#operation");
        $js.=JsUtils::postAndBindTo("#btMultiToGroup", "click",
"cUtilisateur/addUsersToGroup",'{idGroupe: $("#multiGroupe").val()}',"#message");
        $js.=JsUtils::postAndBindTo("#ckAll", "click",
```

```
"cUtilisateur/selectAll/",'{ids:
$("input:checkbox[class=ck]").map(function(){return
this.value;}).get().join(",")}',"#message");
        $js.=JsUtils::setHtmlAndBindTo("#btGroupes", "click",
"#operation","Fonctionnalité non implémentée",JsUtils::_doSomethingOn("#operation",
"show",400));
        $nb=sizeof($selectedUsers);
        if($nb>0)
            $js.=JsUtils::doSomethingOn("#multi", "show");
        else
            $js.=JsUtils::doSomethingOn("#multi", "hide");
        $data=array("users"=>$users,"js"=>$js,"groupes"=>DAO::getAll("Groupe"));
        $this->loadView("vUsers", $data);
    }
```

-- cUtilisateur/add

Remarque :

Le contrôleur **add** gère l'appel de l'affichage du formulaire d'ajout (GET) et la soumission de ce même formulaire (POST).

Fonctionnalité	**Descriptif**
Ajout	**Ajout d'un utilisateur**
	Si le formulaire est posté (POST)
	Instanciation d'un nouvel utilisateur Affectation des variables du POST aux membres de l'utilisateur Encryptage sha1 du password Ajout dans la base de données Appel du contrôleur refresh Affichage du message de mise à jour
	Sinon (GET)
	Mise en place logique client #btValidAddUser -> add #btCancelAddUser -> annulation Passage des variables et chargement de la vue vAddUser

```
    public function add(){
        if($_SERVER['REQUEST_METHOD']=='POST'){
            $user=new Utilisateur();
            RequestUtils::setValuesToObject($user,RequestUtils::getPost());
            $password=$_POST["password"];
            if(isset($password) && $password!="")
                $user->setPassword(sha1($password));
            $user->setGroupe(new Groupe($_POST["idGroupe"]));
            if(DAO::insert($user)){
                Gui::showOne($user);
                echo " ajouté";
                echo JsUtils::get("cUtilisateur/refresh","{}","#users");
            }else
                echo "ajout impossible";
            echo JsUtils::doSomethingOn("#operation", "hide",200);
        }else{
```

```
            $js=JsUtils::postFormAndBindTo("#btValidAddUser", "click",
"cUtilisateur/add", "frmAddUser","#message",true);
            $js.=JsUtils::doSomeThingAndBindTo("#btCancelAddUser", "click",
"#operation","hide",200);
            $js.=JsUtils::setHtmlAndBindTo("#btCancelAddUser", "click",
"#message","Opération d\'ajout annulée");
            $js.=JsUtils::setAttr("#password", "required",true);
            $js.=JsUtils::doSomethingOn("#operation", "show",400);
            $user=new Utilisateur();
            $groupes=DAO::getAll("Groupe");
            $this->loadView("vAddUser",
array("user"=>$user,"js"=>$js,"title"=>"Ajouter un utilisateur
:","groupes"=>$groupes));
        }
    }
```

-- cUtilisateur/update

Scénario pratiquement identique au contrôleur **cUtilisateur/add**

```
    public function update($id){
        if($_SERVER['REQUEST_METHOD']=='POST'){
            $user=new Utilisateur();
            RequestUtils::setValuesToObject($user,RequestUtils::getPost());
            $password=$_POST["password"];
            if(isset($password) && $password!="")
                $user->setPassword(sha1($password));
            $user->setGroupe(new Groupe($_POST["idGroupe"]));
            if(DAO::update($user)){
                Gui::showOne($user);
                echo " modifié";
                echo JsUtils::get("cUtilisateur/refresh","{}","#users");
            }else
                echo "modification impossible";
            echo JsUtils::doSomethingOn("#operation", "hide",200);
        }else{
            $id=str_replace("update", "", $id[0]);
            if(is_numeric($id)){
                $user=DAO::getOne("Utilisateur", "id=".$id);
                $groupes=DAO::getAll("Groupe");
                $js=JsUtils::setVal("#idGroupe", $user->getGroupe()->getId());
                $js.=JsUtils::postFormAndBindTo("#btValidAddUser", "click",
"cUtilisateur/update", "frmAddUser","#message",true);
                $js.=JsUtils::doSomeThingAndBindTo("#btCancelAddUser", "click",
"#operation","hide",200);
                $js.=JsUtils::setHtmlAndBindTo("#btCancelAddUser", "click",
"#message","Opération de mise à jour annulée");
                $js.=JsUtils::bindToElement("#password", "keypress",
"function(){".JsUtils::_setAttr("#password", "required",true)."}");
                $js.=JsUtils::doSomethingOn("#operation", "show",400);
                $this->loadView("vAddUser",
array("user"=>$user,"js"=>$js,"title"=>"Modifier l'utilisateur
:","groupes"=>$groupes));
```

```
            }
        }
    }
```

-- cUtilisateur/confirmDelete

Fonctionnalité	Descriptif
Confirmation	**Confirmation pour suppression d'utilisateur(s)**
	Test pour savoir si la sélection est unique ou multiple
	Affichage du message de confirmation et des options possibles
	Mise en place logique client #confirmDelete -> delete ou deleteMulti #cancelDelete -> annulation

```
    public function confirmDelete($id){
        $id=str_replace("delete", "", $id[0]);
        if(is_numeric($id)){
            echo "<div class='cadre'>Souhaitez vous supprimer l'utilisateur ? <a
id='confirmDelete".$id."' href='#' class='accept'>Confirmer la suppression</a>";
            echo "<a id='cancelDelete' href='#' class='cancel'>Annuler</a>";
            echo JsUtils::getAndBindTo("#confirmDelete".$id, "click",
"cUtilisateur/delete/".$id,"{}","#message");
        }else{
            echo "<div class='cadre'>Souhaitez vous supprimer les utilisateurs
sélectionnés ? <a id='confirmDelete".$id."' href='#' class='accept'>Confirmer la
suppression</a>";
            echo "<a id='cancelDelete' href='#' class='cancel'>Annuler</a>";
            echo JsUtils::getAndBindTo("#confirmDelete".$id, "click",
"cUtilisateur/deleteMulti/".$id,"{}","#message");
        }
        echo JsUtils::doSomethingOn("#operation", "show",400);
        echo JsUtils::doSomeThingAndBindTo("#cancelDelete", "click", "#operation",
"hide",200,JsUtils::_setHtml("#message","Supression annulée"));
    }
```

-- cUtilisateur/delete

Fonctionnalité	Descriptif
Suppression	**Supprimer 1 utilisateur**
	Récupération (GET) de l'$id de l'utilisateur à supprimer
	Suppression dans la BDD
	Mise à jour de la sélection (en session)
	Affichage du message post-suppression

```
    public function delete($id){
        $id=str_replace("delete", "", $id[0]);
        if(is_numeric($id)){
            $user=new Utilisateur();
            $user->setId($id);
            if(DAO::delete($user)){
                SessionUtils::removeValueFromArray("selectedUsers", $id);
```

```
                Gui::showOne($user);
                echo " supprimé";
                echo JsUtils::get("cUtilisateur/refresh","{}","#users");
            }else {
                echo "Suppression impossible";
            }
        }
        echo JsUtils::doSomethingOn("#operation", "hide",200);
    }
```

-- cUtilisateur/deleteMulti

Suppression	**Supprimer 1 sélection d'utilisateurs**
	Récupération (dans 1 tableau stocké en session) des ids de(s) utilisateur(s) à supprimer
	Parcours des utilisateurs et suppression dans la BDD
	Mise à jour de la sélection (en session)
	Affichage du message post-suppression

```
    public function deleteMulti(){
        $selectedUsersId=SessionUtils::getArray("selectedUsers");
        $nb=0;
        foreach ($selectedUsersId as $userId){
            $user=new Utilisateur();
            $user->setId($userId);
            if(DAO::delete($user)){
                SessionUtils::removeValueFromArray("selectedUsers", $userId);
                $nb++;
            }
        }
        echo JsUtils::get("cUtilisateur/refresh","{}","#users");
        echo Gui::pluriel("utilisateur supprimé", "utilisateurs supprimés", $nb);
        echo JsUtils::doSomethingOn("#operation", "hide",200);
    }
```

-- cUtilisateur/select

Fonctionnalité	**Descriptif**
Sélection	**(Dé)Sélectionner 1 utilisateur**
	Récupération (GET) de l'$id de l'utilisateur à (dé)sélectionner
	Ajout ou suppression de l'id de l'utilisateur dans le tableau $_SESSION["selectedUsers"]
	Affichage du message post-(dé)sélection

```
    public function select($id){
        $id=str_replace("ck", "", $id[0]);
        if(is_numeric($id)){
            SessionUtils::addOrRemoveValueFromArray("selectedUsers", $id);
            $nb=sizeof($_SESSION["selectedUsers"]);
            if($nb>0)
                echo JsUtils::doSomethingOn("#multi", "show");
            else
```

```
                echo JsUtils::doSomethingOn("#multi", "hide");
            Gui::pluriel("utilisateur sélectionné", "utilisateurs sélectionnés",
$nb);
        }
    }
```

-- cUtilisateur/selectAll

Fonctionnalité	Descriptif
Sélection	**(Dé)Sélectionner tous les utilisateurs**
	Récupération de l'opération à effectuer : sélection ou désélection
	Mise à jour dans le tableau $_SESSION["selectedUsers"]
	Affichage du message post-(dé)sélection

```
    public function selectAll(){
        $nb=0;
        $ckAll=SessionUtils::checkBoolean("ckAll");
        if(!$ckAll){
            $_SESSION["selectedUsers"]=array();
        }else{
            $_SESSION["selectedUsers"]=explode(",", $_POST["ids"]);
            $nb=sizeof($_SESSION["selectedUsers"]);
        }
        echo JsUtils::get("cUtilisateur/refresh/","{}","#users");
        echo Gui::pluriel("utilisateur sélectionné", "utilisateurs sélectionnés",
$nb);
    }
```

-- cUtilisateur/addUsersToGroup

Fonctionnalité	Descriptif
Affecter à un groupe	**Affecter la sélection d'utilisateurs à un groupe**
	Récupération (POST) de l'$id du groupe sélectionné
	Récupération (SESSION) des utilisateurs sélectionnés
	Mise à jour de la sélection dans la base de données
	Affichage du message post-affectation

```
    public function addUsersToGroup(){
        $idGroupe=$_POST["idGroupe"];
        $selectedUsersId=SessionUtils::getArray("selectedUsers");
        $nb=sizeof($selectedUsersId);
        $where=SqlUtils::getMultiWhere($selectedUsersId, "id");
        $statement=DAO::$db->prepareStatement("update Utilisateur set idGroupe=
:idGroupe where ".$where);
        DAO::$db->bindValueFromStatement($statement,"idGroupe",$idGroupe);
        $statement->execute();
        echo JsUtils::get("cUtilisateur/refresh/","{}","#users");
        echo Gui::pluriel("utilisateur a changé de groupe", "utilisateurs ont
changé de groupe", $nb);
        echo JsUtils::doSomethingOn("#operation", "hide",200);
```

```
    }
```

-- Classes techniques

Nom	Descriptif
DAO	Classe passerelle entre Base de données et objets
Database	Classe d'accès à une base de données, encapsule un objet PDO
GUI	Gestion de l'affichage (évite les excès de code dans les vues)
JsUtils	Génération de scripts côté client
OrmUtils	Récupération des annotations de mappage relationnel/objet
RequestUtils	Récupération des variables POST ou GET
SessionUtils	Méthodes utilitaires liées à la session
SqlUtils	Méthodes utilitaires liées à SQL

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

Permanent link:
http://slamwiki2.kobject.net/slam4/bugreport?rev=1385474222

Last update: **2019/08/31 14:38**