

TD n°6 : Conception de logique applicative

Objectifs

- Concevoir des interfaces (vues)
- Créer leur logique comportementale

Situation initiale

- Créer le projet **koTd6** en important le fichier war dans eclipse : [kotd6.war](#)
- Démarrer le projet sur le serveur web (**Run as...**)
- Démarrer éventuellement le serveur Mysql
- afficher ensuite l'adresse : <http://127.0.0.1/kotd6/>

Erratum :

Pour continuer le TD n°6 et éviter de rencontrer à nouveau les bugs de cette semaine, pensez à mettre à jour les fichiers de votre projet avec ceux contenus dans l'archive :

- [updateFilesTD6.zip](#)

Conception des vues

Villes

Objectifs

Il s'agit de donner à la liste des villes le comportement suivant :

- Ajout de la sélection au clavier et à la souris
- Édition par défaut supprimée et remplacée par une édition directe (dans la liste)
- Ajout d'un bouton détail pour visualiser la liste des entreprises de la ville
- Ouvrir le template de la classe KVille :

Ajout de la sélection au clavier

la méthode **addSelector** permet d'ajouter le contrôle clavier sur la liste : le code de touche **113** correspond à la touche F2, et permet l'édition d'une ville : voir <http://tutorial.kobject.net/java/ajaxinclude/keyboard>

- Modifier le template des villes :

```
{#func:this.addSelector(113)#}
{#func:this.setEditable(true)#}
{#mask:<td>{cp}</td><td>{ville}</td>#}
{#mask:<td>{cp}</td><td>{ville}</td>#}
{#set:this.ajaxIncludes=true#}
```

```

{#set:this.listContentUrl="villes.do"#}
{ _ajx}
{ _listContent}
  { _page}
  <div class="boxButtons">{ _pageCounter}{ _navBarre}</div>
{/ _listContent}

```

Tester la page **villes.do** et son comportement (clavier et souris)

Action	Effet
F2	Edition avec le formulaire de modification (il s'agit de la seule touche paramétrable)
Touches de direction Haut, bas, gauche, droite	Déplacement entre les villes
MAJ+Home	Atteindre la première page
MAJ+Fin	Atteindre la dernière page
MAJ+PageUp	Page précédente
MAJ+PageDown	Page suivante
Double clic	Déplacement sélection ligne

The screenshot shows a web application interface with a navigation bar at the top containing two tabs: 'Liste des entreprises' and 'Liste des villes'. Below the navigation bar is a table listing cities. Each row contains a city ID, the city name, and a 'Modifier' button. The row for 'Clermont-Ferrand' (ID 63000) is highlighted in blue. At the bottom of the table, there is a pagination control showing 'Afficher 10 enregistrement(s) sur 26' and navigation buttons for page 1, 2, and 3.

- Enlever l'édition de la liste

```

{#func:this.addSelector(113)#}
{#func:this.setEditable(false)#}
...

```

Activation de l'édition directe des membres

Pour qu'un membre soit éditable, il faut qu'il appartienne à un élément DOM de la classe **editable**.

Nous allons ajouter un **Display** associé à la classe **KVille** pour modifier l'affichage des éléments de la liste : La méthode **showInList** à surdéfinir gère l'affichage de chaque membre de la liste.

```
package net.display;
import net.ko.displays.KObjectDisplay;
import net.ko.kobject.KObject;

public class VilleDisplay extends KObjectDisplay {

    @Override
    public String showInList(KObject ko, String memberName) {
        String result=super.showInList(ko, memberName);
        //si le nom du membre est cp ou ville, on le met dans un span de classe css
        editable
        if("cp".equals(memberName) || "ville".equals(memberName)){
            result="<span class='editable'
            title='"+memberName+"'>" + result + "</span>";
        }
        return result;
    }
}
```

- Associer ensuite le **Display** créé à la classe **KVille** dans le fichier **/conf/kox.xml** :

```
...
<class name="KVille" display="net.display.VilleDisplay">
    <member max="5" name="cp" required="1" type="string" />
    <member max="100" name="ville" required="1" type="string"
transform="onlyFirstWordUpper"/>
</class>
...
```

Tester l'édition possible de la liste :



Action	Effet
Double clic	Mode édition du membre
ESCAPE	Sortie de l'édition
ENTREE	Sortie avec possible validation
Perte du focus	Sortie de l'édition

A noter que l'édition n'est pour l'instant que visuelle, et ne permet pas la validation des modifications

effectuées.

Ajout de la validation

- Editer le fichier **conf/mox.xml** et ajouter l'inclusion ajax suivante :
- Sur la page **villes.do**,
 - l'événement **updated** sur une zone de classe css **editable**,
 - provoque le post vers une url virtuelle de validation (**updateVille.do**) de l'instance de **KVille**,
 - et passe les informations contenues dans le champ édité (**target.innerHTML**) du champ (**target.title**)

target correspond à l'élément DOM qui a reçu l'événement :

```
<ajax-includes>
...
  <request requestURL="villes.do">
    <js triggerSelector=".editable" triggerEvent="updated">
      <updateOne virtualURL="updateVille.do" operation="update"
kobjectShortClassName="KVille" targetId="info" method="POST">
        <field name="{js:target.title}" value="{js:target.innerHTML}"/>
      </updateOne>
    </js>
  </request>
...
</ajax-includes>
```

- Ajouter dans le même fichier le mapping virtuel **updateVille.do**, associé à une action de type **updateOne** :

```
<mappings>
...
  <virtualMapping requestURL="updateVille.do" mappingFor="updateOne"/>
...
</mappings>
```

tester l'édition finalisée dans la page **villes.do**

Message après modification et validation par **ENTREE** :

14000	Authie
63000	Clermont-Ferrand
63000 Clermont-Ferrand mis à jour	

Ajout d'un champ supplémentaire dans la liste

Le champ supplémentaire devra afficher le nombre actuel d'entreprise(s) de la ville, dans un lien cliquable qui devra ensuite afficher la liste des entreprises de la ville.

- Modifier la classe **KVille** pour qu'elle charge automatiquement les entreprises depuis la base de données :

```
...
public KVille() {
    super();
    hasMany(KEntreprise.class);
}
...
```

- Ajouter l'élément **btDetail** dans le masque de la liste (il s'agit d'un membre inexistant de la classe)

```
{#func:this.setEditable(false)#}
{#func:this.addSelector(113)#}
{#mask:<td>{cp}</td><td>{ville}</td><td>{btDetail}</td>#}
{#mask:<td>{cp}</td><td>{ville}</td><td>{btDetail}</td>#}
{#set:this.ajaxIncludes=true#}
{#set:this.listContentUrl="villes.do"#}
{ _ajx }
{ _listContent }
  { _page }
  <div class="boxButtons">{ _pageCounter }{ _navBarre }</div>
{/ _listContent }
```

- * Modifier le Display de la classe **KVille**, **VilleDisplay**, pour qu'il gère l'affichage de **btDetail** :

```
public class VilleDisplay extends KObjectDisplay {

    @Override
    public String showInList(KObject ko, String memberName) {
        String result=super.showInList(ko, memberName);

        if("cp".equals(memberName) || "ville".equals(memberName)){
            result="<span class='editable'
title='"+memberName+"'>"+result+"</span>";
        }
        if("btDetail".equals(memberName)){
            KVille ville=(KVille) ko;
            int nb=ville.getEntreprises().count();
            if(nb>0)
                result="<div><a id='alink-"+ville.getId()+"' class='default'
title='Entreprises de
"+ville.getVille()+"'>"+KString.pluriel(ville.getEntreprises().count(),"entreprise"
)+"</a></div>";
            else
result="<div>"+KString.pluriel(ville.getEntreprises().count(),"entreprise")+"</div>";
        }
        return result;
    }
}
```

```
}
```

Affichage des entreprises de la ville

- Ajouter une zone **divEntreprises** dans le template des villes qui recevra l'affichage des entreprises de la ville :

```
{#func:this.setEditable(false)#}
{#func:this.addSelector(113)#}
{#mask:<td>{cp}</td><td>{ville}</td><td>{btDetail}</td>#}
{#mask:<td>{cp}</td><td>{ville}</td><td>{btDetail}</td>#}
{#set:this.ajaxIncludes=true#}
{#set:this.listContentUrl="villes.do"#}
{ _ajx }
{ _listContent }
    { _page }
        <div class="boxButtons">{ _pageCounter }{ _navBarre }</div>
{/ _listContent }
<div id="divEntreprises"></div>
```

- Créer un nouveau template de type **list** pour afficher les entreprises de la ville.
- Associer le template à un Display **net.display.EntrepriseVilleDisplay** (à créer par la suite)
- **%ville%** permettra de récupérer la variable ville passée dans la requête

```
{#koDisplay:net.display.EntrepriseVilleDisplay#}
{#mask:<td>{rs}</td><td>{adresse}</td><td>{tel}</td>#}
{#mask:<td>{rs}</td><td>{adresse}</td><td>{tel}</td>#}
<fieldset>
<legend>%ville%</legend>
{ _ajx }
{ _listContent }
    { _page }
{/ _listContent }
</fieldset>
```

Display

Le display va permettre de filtrer les entreprises de la liste, pour ne faire apparaître que les entreprises de la ville sélectionnée :

Surdéfinir la méthode **beforeLoading** de la façon suivante :

```
package net.display;

import javax.servlet.http.HttpServletRequest;

import net.ko.displays.KObjectDisplay;
import net.ko.http.objects.KRequest;
import net.ko.http.views.KPageList;
```

```
import net.ko.kobject.KObject;

public class EntrepriseVilleDisplay extends KObjectDisplay {

    @Override
    public void beforeLoading(Class<? extends KObject> clazz, KPageList list,
        HttpServletRequest request) {
        list.addWhere("idVille='"+KRequest.GET("idVille", request, "-1")+"'");
    }
}
```

Mapping

- ajouter le mapping associé à la liste dans `mox.xml`

```
<mappings>
...
    <mapping requestURL="entrepParVille.do" responseURL="/WEB-INF/list/ville/entreprise.list"/>
...
</mappings>
```

Inclusion ajax

- ajouter l'inclusion sur le clic d'un lien **a** de classe css **default**, pour que chaque lien de ce type affiche **entrepParVille.do**, et lui passe les paramètres souhaités

```
<ajax-includes>
...
    <request requestURL="villes.do">
        ...
        <js triggerSelector="a.default" triggerEvent="click">
            <include targetURL="entrepParVille.do"
targetParams="idVille={js:$vId(target.id)},ville={js:target.title}"
targetId="divEntreprises"></include>
        </js>
    </request>
</ajax-includes>
```

Tester le comportement de la vue **villes.do** (la touche **F2** ou un clic sur le lien d'une ville sélectionnée provoque l'affichage des entreprises de la ville):

Liste des entreprises | **Liste des villes**

14000	Caen	4 entreprises
14000	Authie	Aucun(e) entreprise
63000	Clermont-Ferrand	1 entreprise
5000	Gap	1 entreprise
73490	La Ravoire	1 entreprise
14400	Bayeux	3 entreprises
14100	Authie Nord	Aucun(e) entreprise
50000	Saint-Lô	Aucun(e) entreprise
35000	Rennes	Aucun(e) entreprise
33000	Bordeaux	Aucun(e) entreprise

Afficher 10 enregistrement(s) sur 26

Entreprises de Caen

Marie Automobiles	36, Boulevard André Detolle	02.31.22.16.36
Garage Crapart	26 Boulevard Maréchal Juin	02.31.95.55.15
S.A.R.L Lenrouilly-Vivier	35, Avenue Henry Chéron	08.99.18.75.66
Garage Colbert	6 Rue Jean-Baptiste Colbert	08.99.02.88.69

Juste pour le fun

- Ajouter des transitions sur l'apparition des éléments :

```

<ajax-includes>
...
  <request requestURL="villes.do">
    ...
    <js triggerSelector="a.default">
      <include targetURL="entrepParVille.do"
targetParams="idVille={js:$vId(target.id)},ville={js:target.title}"
targetId="divEntreprises" transition="skew">
        <transition targetId="divEntreprises">
          <oneTransition endValue="1" startValue="0"
property="opacity" duration="2" timing="ease"/>
        </transition>
      </include>
    </js>
  </request>
...
</ajax-includes>

```

Liste des entreprises | **Liste des villes**

14000	Caen	4 entreprises
14000	Authie	Aucun(e) entreprise
63000	Clermont-Ferrand	1 entreprise
5000	Gap	Aucun(e) entreprise
73490	La Ravoire	1 entreprise
14400	Bayeux	3 entreprises
14100	Authie Nord	Aucun(e) entreprise
50000	Saint-Lô	Aucun(e) entreprise
35000	Rennes	Aucun(e) entreprise
33000	Bordeaux	Aucun(e) entreprise

Afficher 10 enregistrement(s) sur 26
Entreprises de Caen

Mary Automobiles (Caen) Boulevard maréchal Juin
Garage Crapart 26 Boulevard Maréchal Juin
S.A.R.L Lenrouly-Vivier 35, Avenue Henry Chéron
Garage Colbert 6 Rue Jean-Baptiste Colbert

08.99.02.88.69

Afficher la liste des villes

La suite : Gestion des entreprises >>

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

Permanent link:
<http://slamwiki2.kobject.net/javaee/td6>

Last update: **2019/08/31 14:21**

