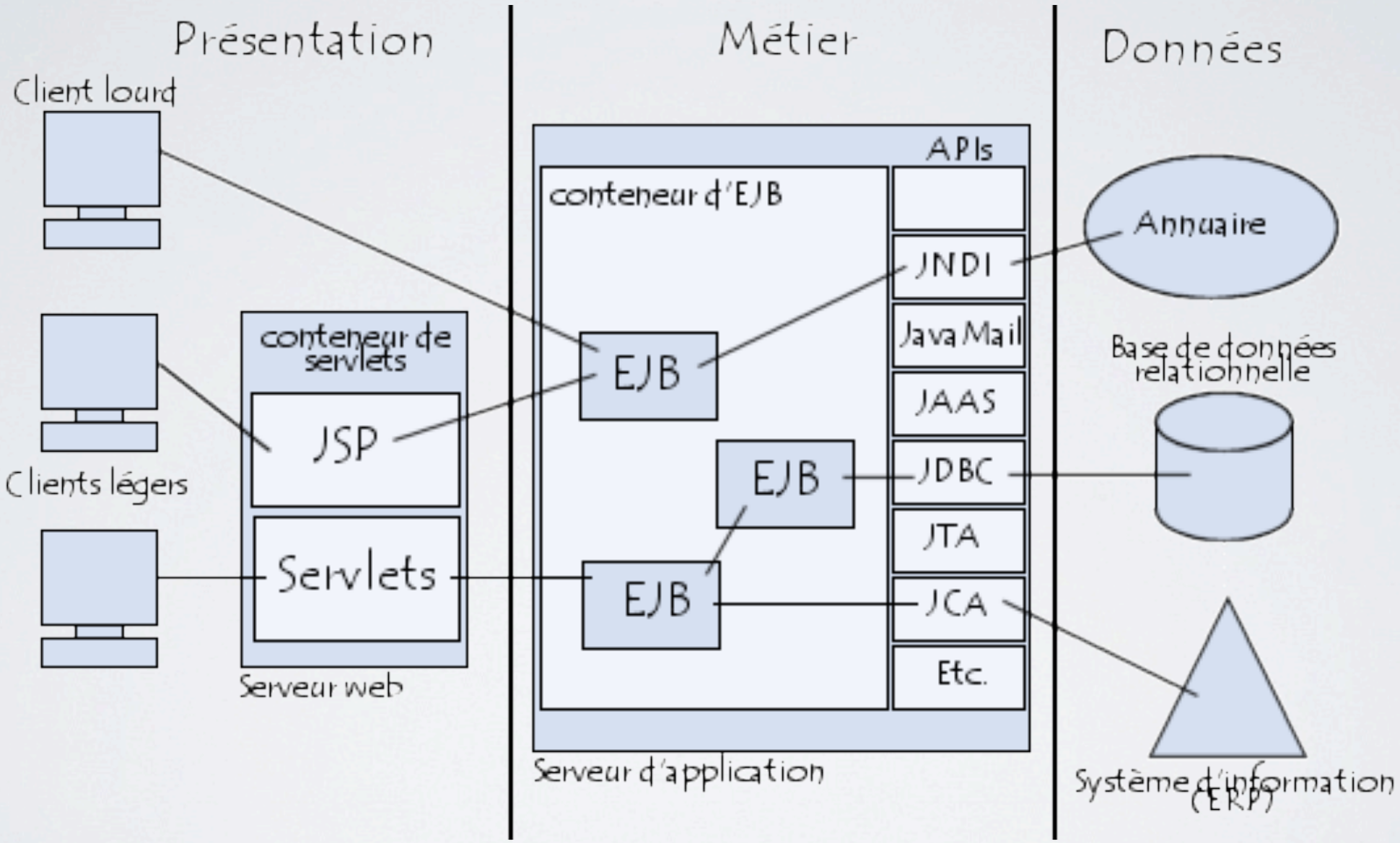
JAVA SERVER FACES

RAPPEL D'ARCHITECTURE



PROBLÉMATIQUE

Comment afficher des données en respectant le modèle MVC, simplement, rapidement et sûrement ?

JSF

- framework en Java
- respect du modèle MVC
- basé sur des composants

COMPOSANTS GRAPHIQUES

- classes Java qui codent le comportement et l'état
- classes de rendu qui traduisent la représentation graphique
- classes qui définissent les balises personnalisées
- classes qui modèlisent la gestion des événements
- classes qui prennent en comptent les *Converteurs*
- classes qui prennent en comptent les *Validators*

COMPOSANTS GRAPHIQUES

Pour utiliser les composants JSF on utilise des balises personalisées !

BALISES HTML

- description graphique d'un JSF
- ensemble fini de composants graphiques (d'autres librairies...)
- 25 composants proposés classifiés en quatre catégories :
  - Composants de saisies notés I
  - Composants de sorties notés O
  - Composants de commandes notés C
  - Composants de regroupement notés R

dimanche 7 novembre 2010

DÉTAILS DES BALISES

- Une balise est constituée de trois types d'attributs
  - des attributs de description
  - des attributs de description de présentation (HTML)
  - des attributs de description de présentation dynamique (DHTML)

Selon la balise le nombre d'attributs va varier !

ATTRIBUTS DE DESCRIPTION

- *id:* identifiant du composant
- *rendered:* masque ou non le composant
- *styleClass:* CSS à appliquer au composant
- *value:* valeur du composant
- *converter:* nom de la classe pour une conversion des données
- *validator:* nom de la classe pour une validation des données

ATTRIBUTS HTML

Modification de l'apparence du composant:

- *alt:* texte dans le cas où le composant ne s'affiche pas
- *border:* bordure du composant
- *disabled:* désactive un composant de saisie ou un bouton
- *maxlength:* maximum de caractères pour du texte
- *size:* taille d'un champ texte
- et bien d'autres encore...

ATTRIBUTS DHTML

Aspect dynamique aux composants JSF:

- *onblur:* le composant perd le focus
- *onClick:* clique de souris
- *onfocus:* focus sur le composant
- *onkeypress:* touche clavier pressée puis relâchée
- *onkeydown* & *onkeyup:* touche clavier enfoncée et relâchée
- *onselect:* texte est selectionné dans un champ texte

COMPOSANTS DE SAISIES

- inputHidden
- inputSecret
- inputText
- inputTextArea
- selectBooleanCheckbox
- selectManyCheckbox
- selectManyListbox
- selectManyMenu
- selectOneMenu
- selectOneRadio

INPUTHIDDEN: EXEMPLE

```
<h:inputHidden id="hidden">Hidden</h:inputHidden>
```

Voyons ce que l'on obtient...

INPUTTEXT: EXEMPLE

```
<h:inputText id="test" />
```

Voyons ce que l'on obtient...

INPUTTEXTAREA: EXEMPLE

```
<h:inputTextarea id="testAreaText" value="test deja
present" title="test text area" />
```

Voyons ce que l'on obtient...

LES SÉLECTIONS

- selectOneRadio & selectOneMenu gérent un ensemble d'éléments
- Ces éléments sont peuplés par :
  - *<f:selectItem>:* affiche un seul élement
  - *<f:selectItems>:* affiche plusieurs éléments

SELECTONERADIO: EXEMPLE

```
<h:outputText value="Fruit : " />
<h:selectOneRadio layout="pageDirection">
 <f:selectItem itemLabel="Banane" />
 <f:selectItem itemLabel="Orange" />
 <f:selectItem itemLabel="Clementine" />
 <f:selectItem itemLabel="Pomme" />
 <f:selectItem itemLabel="Framboise" />
</h:selectOneRadio>
```

Voyons ce que l'on obtient...

SELECTITEM

- spécification d'un seul élément d'une sélection
- les principaux attributs de cette balise sont:
  - *itemDisabled:* active ou non l'item
  - *itemLabel:* le texte à afficher
  - *itemValue:* la valeur retournée au serveur
  - *value:* la valeur qui pointe vers un objet *SelectItem*

CONSTRUCTEURS SUR SELECTITEM

- Deux constructeurs sont disponibles:
  - *SelectItem(Object value):* une valeur à afficher et la même valeur à retourner côté serveur
  - *SelectItem(Object value, String label):* une valeur à retourner côté serveur et un libellé à afficher

SELECTITEM

- Alourdit l'écriture des différents composantes d'une sélection
- Adaptée pour un nombre réduit d'éléments d'une sélection

<f:selectItems> permet d'alléger l'écriture puisqu'il n'y aura qu'une seule occurence à la place de toute les occurences de <f:selectItem> !

SELECTITEMS: EXEMPLE

```
<h:outputText>Civilites</h:outputText>
<h:selectOneRadio>
  <f:selectItems value="#{civilityBean.civilities}" />
</h:selectOneRadio>
```

Voyons ce que l'on obtient...

COMPOSANTS DE SORTIES

- column
- message et messages
- dataTable
- outputText
- outputFormat
- outputLink
- graphicImage

OUTPUTTEXT: EXEMPLE

```
<h:outputText value="Sympa?" />
```

Voyons ce que l'on obtient...

DATATABLE

- visualisation de données sur plusieurs lignes et colonnes
- utilisable quand le nombre d'éléments à afficher est inconnu
- column définit une nouvelle colonne

- attributs de la balise:
  - *value:* une collection de données
  - *var:* nom de la variable manipulée à chaque ligne
  - *rowClasses*, *headerClass:* CSS

dimanche 7 novembre 2010

EXEMPLE: DATATABLE

```
<h:dataTable value="#{flowScope.listAuthor}"
var="author" border="1" cellspacing="4"
width="60%" >
 <h:column>
  <h:outputText value="#{author.firstname}"/>
 </h:column>
 <h:column>
  <h:outputText value="#{author.lastname}"/>
 </h:column>
</h:dataTable>
```

Voyons ce que l'on obtient...

DATATABLE SUITE...

- *facet* permet de modifier les en-têtes et pieds de page
- *facet* permet d'effectuer une filiation entre deux composants
- association entre le conteneur du *facet* et le contenu du *facet*

- trois types de filiation possible spécifiée via l'attribut *name*:
  - *footer:* filiation entre la table et un nom
  - *header:* filiation entre un titre de colonne et un nom
  - *caption:* filiation entre le titre de la table est un nom

EXAMPLE: DATATABLE

```
<h:dataTable value="#{flowScope.listAuthor}"
var="author" border="1" cellspacing="4"
width="60%" >
 <h:column>
  <f:facet name="header">
   <f:verbatim>Prenom</f:verbatim>
  </f:facet>
  <h:outputText value="#{author.firstname}"/>
 </h:column>
 <h:column>
  <h:outputText value="#{author.lastname}"/>
 </h:column>
</h:dataTable>
```

Voyons ce que l'on obtient...

DATATABLE

Le composant de filiation à <*f:facet*> doit <u>absolument</u> être un composant JSF !

dimanche 7 novembre 2010

MAQUILLER UNE DATATABLE

- Les attributs à modifier:
  - *headerClass:* style CSS pour l'en-tête de la table
  - *footerClass:* style CSS pour le pied de la table
  - *rowClasses:* style CSS pour les lignes de la table
- rowClasses permet de définir un pattern de style CSS
  - Qu'affichera le pattern ? *row1CSS, row2CSS, row2CSS*

dimanche 7 novembre 2010

COMPOSANTS DE COMMANDES

- commandLink
- commandButton

COMMANDLINK: EXEMPLE

```
<h:commandLink id="hum" action="consulter" value="TEST"/>
```

Voyons ce que l'on obtient...

dimanche 7 novembre 2010

COMMANDBUTTON: EXEMPLE

```
<h:commandButton id="consulter" action="consulter"
                                    type="submit"
                                    value="Consultation"
                                    style="font:bold;"
                                    styleClass="none;" />
```

Voyons ce que l'on obtient...

dimanche 7 novembre 2010

COMPOSANTS DE REGROUPEMENTS

- form
- panelGrid
- panelGroup

dimanche 7 novembre 2010

PANELGRID

- définie l'agencement des composants dans une page
- conteneur dont l'organisation des enfants sous forme d'une grille
- les composants enfants sont ajoutés dans le corps

- Principaux attributs:
  - *columns:* nombre de colonnes de la grille
  - *cellpading*, *cellspacing:* espacement entre les cellules
  - *border:* bord autour de la grille

dimanche 7 novembre 2010

EXEMPLE: PANELGRID

```
<h:panelGrid cellspacing="25" columns="2" >
<h:outputText value="Nom : " />
<h:panelGroup>
<h:inputText size="15" required="true" />
<h:inputText />
</h:panelGroup>
<h:outputText value="Mot de passe : " />
<h:inputSecret />
</h:panelGrid>
```

Voyons ce que l'on obtient...

Mais... Comment avoir des données dynamiques et non pas statiques ?

BEAN: PRINCIPES

- Un Bean est une classe Java particulière:
  - un constructeur public sans argument
  - ses propriétés sont accessibles via des méthodes getXXX
  - ses propriétés sont modifiables via des méthodes setXXX

ET DANS JSF...

- L'utilisation des Beans dans JSF permettent:
  - l'affichage des données provenant de la couche métier
  - le stockage des valeurs d'un formulaire
  - la validation des valeurs

C'est l'attribut value dans les composants JSF...

Et dans notre contexte...

dimanche 7 novembre 2010

CONTEXTE

- navigation gérée via SpringWebflow
- action gérée via SpringWebflow
- cycle de vie des Beans géré via Spring

JSF nous fournie au final que des composants graphiques pour réaliser nos vues !

Utilisez le squelette fournie pour le TD3 !