3.13.2 Déchiffrer les URLs directes pour positionner l'application dans un état donné

Bon pour relecture

Sommaire :

Du point de vue du programmeur, l'appréhension d'un lien direct consiste à :
Décoder les paramètres de l'URL,
Positionner les variables d'état (les contrôleurs de l'application) dans un état donné,
Envoyer sur la page _JSF_ voulue.

Comment ça marche

1. Décrire l'url grâce à l'objet UrlPatternDescriptor dans le fichier ***properties/deeplinking/deeplinking.xml***.

Par exemple pour une url : http://monurl/stylsheets/welcomes.faces?args=validation=true&login=cleprous

```
<bean id="validateSubscription" class="org.esupportail.commons.jsf.UrlPatternDescriptor">
  <property name="params">
    <list>
      <value>validation</value>
      <value>login</value>
    </list>
  </property>
  <property name="actionBinding" >
     <bean class="org.esupportail.commons.jsf.ActionBinding">
        <property name="action" value="inscriptionController.urlDecrypt"/>
        <property name="args">
           <list>
             <value>java.lang.Boolean</value>
             <value>java.lang.String</value>
           </list>
        </property>
      </bean>
  </property>
</bean>
```

* La propriété ***params*** définis les paramètres de l'url. L'ordre de paramètre doit être identique à celui dans l'url.
* La bean ***actionBinding*** permet de définir l'action qui va être exécutée. Dans l'exemple la méthode appelée est décrite ci-dessous:

```
Dans le bean inscriptionController, la méthode :


public String urlDecrypt(Boolean validation, String login) {

 ....

return "go_validation_page"; //cette chaîne décrite dans la navigation-rules et permet de rediriger vers une
page


}
```

* La propriété ***viewID*** permet de préciser une chaine de caractère qui va être utilisée dans la redirection.

⚠ Cette propriété est prioritaire. Cela signifie que si elle est renseignée, on utilise cette chaîne pour faire la redirection sinon on utilise la chaîne renvoyée par la méthode décrite dans l'ActionBinding

Pour en savoir plus...

Le cycle de vie JSF peut être décomposé en sept phases :
1- L'émission de la requête cliente demandant une ressource JSF
2- Reconstruction de l'arbre des contrôles
3- Application des valeurs de la requête
4- Validation des données saisies
5- Mise à jour des valeurs du modèle d'objets
6- L'invocation d'une application
7- L'affichage de la réponse

Pour déchiffrer une URL on doit alors être capable d'intercepter chaque requête utilisateur afin d'assurer l'affichage de la page demandée.

Techniquement l'interception dans le cadre JSF peut se faire en utilisant des écouteurs (listeners).
Les listeners peuvent être déclarés dans le fichier de configuration du composant : faces-config.xml.

Les **UrlPatternDescriptor** seront donc analysés par un listenener : **DeepLinkingPhaseListener** déclaré dans le cycle de vie JSF dans **faces-config.xml** :

```
	<lifecycle>
		<phase-listener>org.esupportail.commons.jsf.DeepLinkingPhaseListener</phase-listener>
		<phase-listener>org.esupportail.commons.jsf.ResourceBundlePhaseListener</phase-listener>
	</lifecycle>
```

Selon les possibilités techniques offertes par JSF, il est possible que le listener agisse à la première phase du cycle de vie JSF (beforePhase) ou à la phase d'affichage de réponse(afterPhase).

```
public class DeepLinkingPhaseListener implements PhaseListener {

	/**
	 * Constructor.
	 */
	public DeepLinkingPhaseListener() {
		super();
	}

	/**
	 * @see javax.faces.event.PhaseListener#afterPhase(javax.faces.event.PhaseEvent)
	 */
	public void afterPhase(final PhaseEvent arg0) {
		[...]
	}

	/**
	 * @see javax.faces.event.PhaseListener#beforePhase(javax.faces.event.PhaseEvent)
	 */
	public void beforePhase(final PhaseEvent event) {
		[...]
	}
[...]
}
```