JavaServer Faces & Toplink : une application simple

Écrit par <u>Chris SCHALK</u>, Oracle Corporation Janvier 2005

Introduction

JDeveloper 10.1.3 Preview permet de développer simplement et efficacement des applications JavaServer Faces (JSF) grâce à un nouvel éditeur graphique. Il permet également les développements Toplink, ce qui rend simple la création d'applications JSF dont les composants d'accès aux données sont basés sur Toplink.

L'exemple qui suit présente comment développer une application JSF qui affiche un tableau de données géré par un mid-tiers Toplink

Démarrer

Pour démarrer, téléchargez <u>JDeveloper 10.1.3 Preview</u> sur OTN et installez-le sur votre machine. Vous devez également configurer une connexion à une base de données. Ce tutoriel utilise le schéma exemple SCOTT d'une base de données Oracle mais n'importe quelle connexion conviendra.

 Pour commencer, nous allons utiliser l'assistant de création d'applications qui générera le projet « Model » de gestion de la persistance et le projet « ViewController » pour nos composants d'affichage et la logique d'interactions. Sélectionnez le menu « File -> New -> General -> Application »

Options to specify the appl templates.	ication templa	pplication a te to use or	ina use to manage
<u>Application Name:</u>			
SimpleToplinkJSFExemple			
Directory Name:			
lou/jdevhome/mywork/Sin	npleToplinkJSF	Exemple	B <u>r</u> owse
Application Package Prefix: jsftoplink			
Application Template:			
Web Application [JSF, JSP, EJ	B] 👻	Manage 1	emplates
Template De <u>s</u> cription:			
This application template is data-bound web applicatio	s configured fo n. It consists c	or building and one proje	t or the

- 2. Appuyez sur OK pour générer le workspace
- 3. Après avoir généré nos 2 projets ViewController et Model, nous devons définir la dépendance du premier par rapport au second. Pour cela, double-cliquons sur le projet ViewController, sélectionnons « Dependencies » et activons la case à cocher devant Model. Sélectionnons le bouton OK pour continuer...

Project Content	Dependencies	
∃—Ant ∃—Compiler	🔿 Use <u>C</u> ustom Settings	Customige Settings
Debugger	O Use Project Settings	
Dependencies	Project Dependencies:	
Javadoc JSP Tag Libraries Libraries Offline Database	🕮 🖓 间 Modeljpr	
rechnology scope		
		- ALEXANDER
		The second secon

4. Nous pouvons créer le code du projet « Model » avec Toplink. Utilisons un assistant à partir du menu « File -> New -> Business Tiers -> Java Objects from Tables »

<u>C</u> ategories:	ltems:
⊞−General	🚱 Java Objects From Tables
🖨 - Business Tier	TopLink Mapping Descriptor
← EJB Web Services H-Client Tier H-Database Tier Templates H-Web Tier	TopLink Sessions Configuration
	Description: Launches the Create Java Objects from Tables wizard. Use this wizard to create Java Objects from existing database tables.
	To enable this option, you must select a project or TopLink package within a project in the Application Navigator.

5. Une fois l'assistant démarré, nous pouvons cliquer sur « Next » pour faire

apparaître la fenêtre de sélection de la connexion à la base de données. Choisissons parmi les connexions de bases de données prédéfinies, celle qui convient.

6. Dans l'étape 2/3 de l'assistant, nous sélectionnons la table EMP du schéma SCOTT.

	Nam <u>e</u> Filter:	Auto-Query
	Schema: SCOTT	Query
-	Object Types: 🔽 Tables 🔄 Views 📄	Synonyms
	BONUS DEPT	o (SCOTT.EMP)
	EMP SALGRADE	

- 7. Cliquons « Next » puis « Finish » pour générer les classes Java Toplink de la table EMP.
- Maintenant que nous avons généré notre classe Toplink et un mapping associé, nous devons générer le descripteur de déploiement Toplink. Sélectionnons « Toplink Mappings » puis « Generate toplink-deployment-descriptor.xml » depuis le bouton droit de la souris.



9. Nous pouvons maintenant créer un bean java qui accède à notre projet Toplink pour tester notre exemple et pour utiliser dans notre application JSF à venir. Sélectionnez la classe générée (Employees.java) et à l'aide du bouton droit de la souris « New sample java client »



- 10. Testons l'application cliente générée en l'exécutant (Menu Run)
- 11. Nous allons créer une méthode getEmployees() dans notre classe cliente. Pour cela, il faut :
 - Copier la méthode main() et son contenu et l'utiliser pour générer la méthode getEmployees() en la modifiant. La méthode doit retourner un type java.Util.Vector (nous utiliserons cette méthode dans l'application JavaServer Faces)
 - Au lieu d'afficher le Vector, comme dans la méthode main(), la méthode getEmployees() retournera la liste des employés
 - le code ajouté ressemble à celui qui suit :

12. Nous pouvons désormais créer la vue de notre application qui consistera en une page JavaServer Faces JSP qui utilise le composant JSF Datatable UI pour présenter les employés.

La « View »

 Dans le projet ViewController, nous allons construire une page JSP qui utilise JavaServer Faces. Pour cela, sélectionnons le menu « File -> New -> Web Tier -> JSF -> JSF JSP »

<u>C</u> ategories:	Įtems:
⊟…General	B JSF JSP
Appleations Deployment Descriptors Deployment Profiles Diagrams JavaBeans Projects Simple Files Web Tier Applet HTML ISF JSP Servlets	SF Configuration
Serviets	Description:
	Opens the Create JavaServer Faces (JSF) JSP wizard, in which you can specify information required to create a newJSP (jsp or jspx) file that includes a skeleton JSF template. To enable this option, you must select a project or a file within a project in the Application Navigator.

- 2. Utilisons l'assistant pour nommer la page emps.jsp. Le type de la page est « Page JSP » et l'option de gestion des exceptions pour la page est « Do Not Use an Error Page to Handle Uncaught Exceptions in This File ». Cliquons sur « Finish » pour générer directement la page sans passer par les étapes suivantes.
- 3. Une fois la page générée, nous pouvons simplement ajouter une balise <H2> pour générer une bannière comme ci-dessous. Nous pouvons également glisser/déposer une feuille de style (Cascading Style Sheet) comme celle de JDeveloper depuis la palette des feuilles de styles.

Oracle JDeveloper 10g Developer Preview - SimpleToplinkJSFExemple.jws :	ViewController.jpr
	w <u>H</u> elp
😤 🚽 - 22 🛍 🛍 🗞 - 🕒 - 🧭 🧃 🗐 🐰 🦈 🖉 🕞 🖼 🕙	
PApplicati 🗐 📮 🖸 emps.jsp	Component Palette
🐘 🛃 🔞 😽 Heading 2 👻 Default 🔹 None 🕶 🗛 🗞 🥔 [HTML Common 💌
	Pointer
E-1 jsftoplink.	🕹 Anchor 🧧
Empsj	Ta Applet
E Content	Email Link
⊕ 🔁 WEB-INF	🕅 H2 - Property Inspector
<u>⊡</u> emps.jsp	🧶 🛃 🧪 🦚 🚳 🧶 🧮 🖷 💻
	📧 General
emps.jsp - struc	E Core
	E International
1	∃Styles
head	
🚐 meta - text/html; charset	
B link - css/jdeveloper.css 🛞 Design Source History 4	
body	
圖 h:form	
A Liste des employé	
/Comment:auto-bi	
$/users/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/mywork/SimpleToplinkJSFExemple/ViewController/public_html/emps.jsplites/gguillou/jdevhome/ggu$	Inserting before H Web Editing.layout

- 4. Avant de continuer, nous devons ajouter la classe client toplink que nous avons créée dans le projet « Model » comme un Managed Bean dans le fichier facesconfig.xml. Double-cliquons sur ce fichier (Web Content -> WEB-INF -> facesconfig.xml) pour éditer son contenu
- 5. L'éditeur par défaut du fichier faces-config.xml est le modeleur des flux des pages JSF (Page Flow Modeler). Nous ne nous servons pas pour cet exemple des transitions entre les pages. Sélectionnons l'onglet « Overview » en bas de la page pour invoquer la console d'édition de faces-config.xml
- 6. Ajoutons un nouveau « Managed Bean » en cliquant sur la zone « Managed Bean », puis en sélectionnant le bouton « New …». Spécifions ce qui suit et cliquons sur OK
 - Name : EmpsBean
 - Class : jsftoplink.model.EmployeesClient (naviguer dans l'arborescence de votre projet pour sélectionner cette classe java)
 - Scope : Request

Applications	🖸 emps.jsp	Staces-config.xml			<u>1000</u>
🐘 🗄 🕅 🗸	Managed Beans	Managed Beans	i.		
ViewController	Navigation Rule	S Name	Class	Scope	New
🗄 📄 Application Sources	Converters	backing_emps	jsftoplink.view.ba	request	Delet
E Emps.java	Application Referer	C	reate Managed Be	an	×t
E web Content	Render Life Cyc <u>N</u> am	e: EmpsBean			
⊡⊡ WEB-INF ⊕⊡ lib	Factory Compo <u>C</u> las	s: jsftoplink.model.E	EmployeesClient	▼ <u>B</u> rowse	e
faces-config.	<u>S</u> col	oe: request			•
emps.jsp		Generate Java File Help	0	K Cance	

- 7. Ca y est presque ! Il nous reste à glisser/déposer le composant DataTable Ul sur la page JSP et mettre les données dans ce composant (DataBinding). En plus Jdeveloper 10.1.3 a un assistant qui permet d'effectuer ces 2 étapes.
- 8. Trouvons le composant JSF HTML « Data Table » dans la page JSF HTML de la palette de composants et faisons le glisser/déposer du composant sur la page JSP. Cette action démarre l'assistant.
- 9. Il faut cliquer sur « Next » pour aller sur la première page de l'assistant. Laissons le bouton radio sur la position : « Bind the Data Table Now » et cliquons sur « Next »
- 10. Saisissons les valeurs des champs comme suit:
 - Value : #{EmpsBean.employees}
 - Class : jsftoplink.model.Employees
 - Var : emps

Value:	#{EmpsBean.employees}	Bind
	jsftoplink.model.Employees	Bro <u>w</u> se
Var:	emps]

Note : pour le champ « valeur », l'expression builder permet de générer la syntaxe Expression Language (EL) pour accéder à la bonne méthode du Managed Bean.

	Expression Builder
Filter By: List, ResultSet, Result, Variables:	and Arrays 💌 <u>E</u> xpression:
JSF Managed Beans Def EmpsBean Def EmpsBean Def Employees JSP Objects	#{EmpsBean.employees}
	Logical Operators: && > < >= <= == != - Mathematical Operators: + - * // %
Help	OK Cancel

11. En cliquant sur « Next », la page de configuration des colonnes de la table apparaît. Utiliser cette page pour personnaliser l'ordre, les libellés et le type d'affichage de la DataTable.

Header Valu	e Component	Component Value	-
comm	Output Text	#{emps.comm}	Top
deptno	Output Text	#{emps.deptno}	Up
empno	Output Text	#{emps.empno}	Down
ename	Output Link 🔹	#{emps.ename}	
hiredate	Output Text	#{emps.hiredate}	Botton
job	Output Label	#{emps.job}	
mgr	Output Link	#{emps.mgr}	
sal	Command Button	#{emps.sal}	
	Custom		
		Add <u>R</u> emove	
Number of row	s to display:		

12. Le bouton « Finish » crée le code DataTable dans la page JSP. Vous verrez apparaître une DataTable avec plusieurs lignes et colonnes.



13. Maintenant exécutons la page ! Avec le bouton droit de la souris, il faut sélectionner emps.jsp et cliquer sur « Run »

				emps - N	Mozilla Firefox			X
<u>File E</u> d	it <u>V</u> iew	<u>G</u> o <u>E</u>	<u>3</u> ookmarks	<u>T</u> ools <u>H</u> elp				
•	. ک	B	6	Oracle 🔻	http://140.86	.88.25	1:: 🕶 🜔 Go 💽	
Liste	des ei	mploy	/és					
comm	deptno	empno	ename	hiredate	job	mgr	sal	
0.0	20	7369	SMITH	1980-12-17	CLERK	7902	800.0	
300.0	30	7499	ALLEN	1981-02-20	SALESMAN	7698	1600.0	
500.0	30	7521	WARD	1981-02-22	SALESMAN	7698	1250.0	
0.0	20	7566	JONES	1981-04-02	MANAGER	7839	2975.0	3
1400.0	30	7654	MARTIN	1981-09-28	SALESMAN	7698	1250.0	
0.0	30	7698	BLAKE	1981-05-01	MANAGER	7839	2850.0	
0.0	10	7782	CLARK	1981-06-09	MANAGER	7839	2450.0	
0.0	20	7788	SCOTT	1987-04-19	ANALYST	7566	3000.0	
0.0	10	7839	KING	1981-11-17	PRESIDENT	7839	5000.0	
0.0	30	7844	TURNER	1981-09-08	SALESMAN	7698	1500.0	
Done								

Synthèse

Cet exemple simple sert de point de départ pour commencer à créer des applications JavaServer Faces qui utilisent une couche de persistance sur un middletier. Cet exemple utilise Toplink, mais n'importe quelle autre technologie peut être utilisée. Il montre aussi l'intérêt des nouvelles fonctions de développement graphique d'Oracle JDeveloper 10.1.3, parmi lesquelles l'éditeur de fichiers de configuration faces-config.xml : « Overview », la boite de dialogue Expression Language (EL) Binder et l'assistant de création de DataTable. Parmi les nouveautés à venir sur la version de production de JDeveloper 10.1.3, plusieurs assistants permettront de créer des UIComponents complexes. N'hésitez pas à faire savoir ce que vous en pensez sur les forums à commencer par OTN !