UPMC- Master 2 Module Car **Reda Bendraou**
Tewfik Ziadi

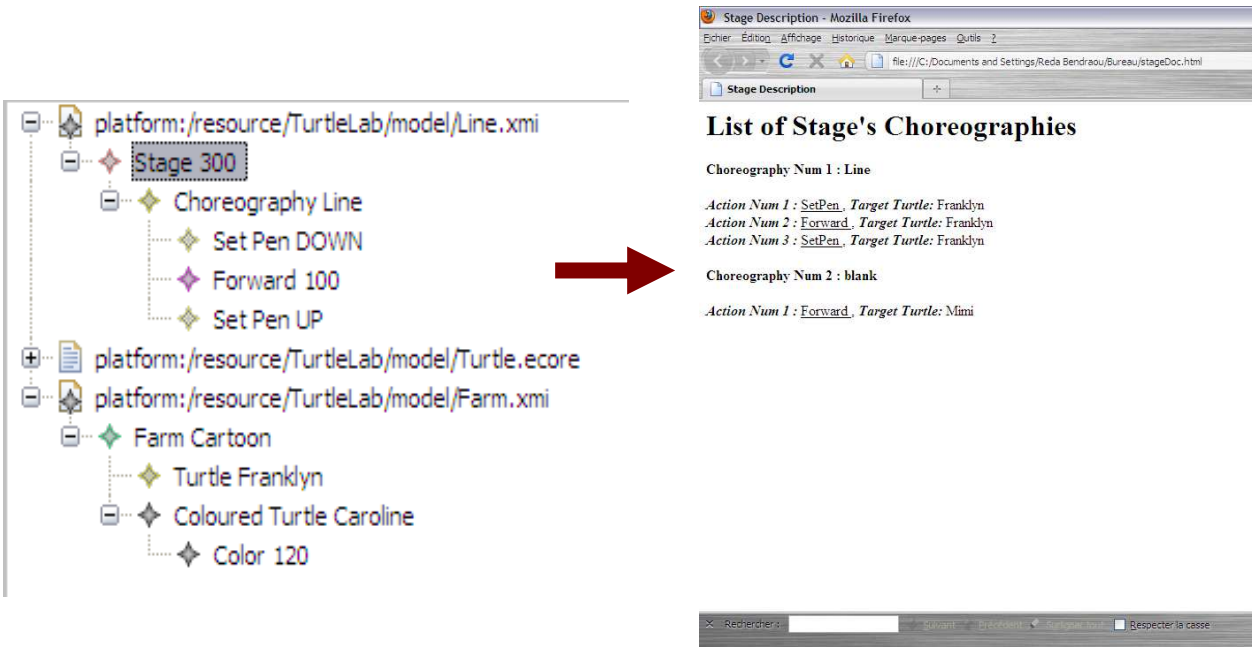
TP4 : M2T avec JET et M2M en Java

NB. : Ce TP est à commencer en salle et à finir chez vous au cas où vous manqueriez de temps. Il est important que vous fassiez l'effort de répondre à toutes les questions. Cela est primordial pour la suite des TPs. Ne comptez pas sur une solution de notre part, vous serez déçus ;-)

Ce TP suppose que vous avez réalisé l'étape dans le TP 3 où nous vous demandions de charger vos modèles instances de Stage (Turtle) en mémoire et d'afficher le contenu du modèle à l'aide de la méthode Display(). Le modèle devait contenir un stage, lui-même contenant une suite de Choegraphies avec des actions.

Génération de texte (M2T)

L'idée dans cette première étape est d'appliquer une transformation de modèle vers texte en utilisant JET (Java Emitter Template). Votre modèle sera le modèle instance du TP 2 et 3 qui contient un Stage avec plusieurs Choreographies et votre cible, i.e., le Texte, ça sera un fichier html pour documenter votre modèle (voir fig. ci-dessous).



1- Avant de commencer : lisez le tutorial suivant et arrêtez vous (absolument avant le paragraphe « under the hood » : http://www.eclipse.org/articles/Article-JET/jet_tutorial1.html
2- Ensuite au niveau de votre Projet Turtle, rajoutez la nature JET puis configurer l'emplacement des classes générées par JET (src par exemple).
3- Ensuite écrire le template qui prend un objet Stage et qui produit un fichier html en sortie conforme à la figure ci-dessus.

4- Aide : voici le début du fichier de template qui se nomme : stageHtmlDoc.htmljet

ATTENTION : dans le bout de code ci-dessous, dans la directive d'imports le **turtle.*** est à remplacer par le nom du package de plus haut niveau de votre ecore et qui correspond au nom de package généré par EMF.

```
<%@ jet package="generator.website"
class="SimpleToHTML" imports ="java.util.Iterator org.eclipse.emf.common.util.EList  turtle.*;"%>

<% Stage stage = (Stage) argument; %>
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head> <title> Stage Description</title>


<meta name="language" content="en">
<meta http-equiv="content-type" content="text/html; charset=UTF-8">

</head>
<body>
<h1> List of Stage's Choreographies</h2>
<%EList<Choreography> listChoreography = stage.getChoreographies();
……//A COMPLETER

%>
</body>

</html>
```

Votre Main devra appeler une méthode qui génére le fichier html et qui prend un entrée un objet Stage, elle doit ressembler à ça :

```
public void generateHtmlDoc(Stage stage){

		//SimpleToHTML est le nom de la classe générée par JET à partir du template
			SimpleToHTML htmlpage = new SimpleToHTML();
			FileWriter output;
			BufferedWriter writer;

				System.out.println("Creating Stage HTML Documentation");

				try {
					output = new FileWriter("stageDoc.html");
					writer = new BufferedWriter(output);

				//Appel de la méthode generate de la classe générée par JET
					writer.write(htmlpage.generate(stage));
					writer.close();
				} catch (IOException e) {
					e.printStackTrace();
				}
	}
```

Pour ceux qui veulent allez plus loin avec M2T, voici les autres projets Eclipse

- http://www.eclipse.org/modeling/m2t/

M2M

- Pour cette partie, il s'agit d'appliquer le design Pattern Visiteur pour réaliser une transformation de modèle. Votre modèle source c'est toujours le même i.e. Stage, votre cible, un fichier XML qui représente le stage et qui doit être valide par rapport à un XML Schéma que vous aurez définit.

- Comment faire si vous deviez changer le document XML, i.e., assurer la cohérence entre le modèle cible et source ?

- Pour ceux qui veulent allez plus en transformation de modèles, voir les langages ATL et QVT.

    - http://www.eclipse.org/atl/
    - http://wiki.eclipse.org/M2M/Operational_QVT_Language_%28QVTO%29

La suite au prochain TP :

- Souffler de la vie (du comportement) dans vos modèles avec Kermeta !