

Fil rouge : corrigé séance 5  
Module FIP INF 211

# Gestion de la syndication des informations



## Les JSP

cabinet\_recrutement\_offres\_rss.jsp

<%@page import=*"eu.telecom\_bretagne.cabinet\_recrutement.front.rss.GenerationFluxRSS"*%><%@ page language=*"java"* contentType=*"text/xml; charset=UTF-8"* pageEncoding=*"ISO-8859-1"*%><%GenerationFluxRSS.offresEmploi(out, getServletContext().getInitParameter("URL\_BASE"));%>

Pour une lecture facilitée tout en sachant que les passages à la ligne empêchent le bon fonctionnement :

<%@page import=*"eu.telecom\_bretagne.cabinet\_recrutement.front.rss.GenerationFluxRSS"*%>

<%@ page language=*"java"* contentType=*"text/xml; charset=UTF-8"* pageEncoding=*"ISO-8859-1"*%>

<%GenerationFluxRSS.offresEmploi(out, getServletContext().getInitParameter("URL\_BASE"));%>

cabinet\_recrutement\_candidatures\_rss.jsp

<%@page import=*"eu.telecom\_bretagne.cabinet\_recrutement.front.rss.GenerationFluxRSS"*%><%@ page language=*"java"* contentType=*"text/xml; charset=UTF-8"* pageEncoding=*"ISO-8859-1"*%><%GenerationFluxRSS.candidatures(out, getServletContext().getInitParameter("URL\_BASE"));%>

## La classe GenerationFluxRSS

**package** eu.telecom\_bretagne.cabinet\_recrutement.front.rss;

**import** java.io.Writer;

**import** java.math.BigDecimal;

**import** java.util.Date;

**import** java.util.List;

**import** javax.servlet.jsp.JspWriter;

**import** javax.xml.bind.JAXBContext;

**import** javax.xml.bind.JAXBException;

**import** javax.xml.bind.Marshaller;

**import** eu.telecom\_bretagne.cabinet\_recrutement.data.model.Candidature;

**import** eu.telecom\_bretagne.cabinet\_recrutement.data.model.OffreEmploi;

**import** eu.telecom\_bretagne.cabinet\_recrutement.front.utils.ServicesLocator;

**import** eu.telecom\_bretagne.cabinet\_recrutement.front.utils.ServicesLocatorException;

**import** eu.telecom\_bretagne.cabinet\_recrutement.front.utils.Utils;

**import** eu.telecom\_bretagne.cabinet\_recrutement.service.IServiceCandidature;

**import** eu.telecom\_bretagne.cabinet\_recrutement.service.IServiceOffreEmploi;

/\*\*

\* Classe permettant la gestion des flux RSS publiant la liste des offres

\* d'emploi et la liste des candidatures. La classe contient deux méthodes

\* statiques utilisables au sein d'un JSP :

\* <ul>

\* <li>{@code GenerationFluxRSS.offresEmploi (Writer writer, String urlBase)}</li>

\* <li>{@code GenerationFluxRSS.candidatures(Writer writer, String urlBase)}</li>

\* </ul>

\* **@author** Philippe TANGUY

\*/

**public** **class** GenerationFluxRSS

{

//-----------------------------------------------------------------------------

/\*\*

\* Construction du flux RSS de la liste des offres d'emploi. Celles-ci sont obtenues

\* par l'appel de la méthode {@code listeDesOffres()}, voir : {@link IServiceOffreEmploi}.

\* **@param** writer l'instance du {@link Writer} sur lequel sera écrit le flux RSS.

\* La méthode étant appelée au sein d'un JSP, celui-ci est l'instance

\* de l'objet prédéfini {@code out}, instance de {@link JspWriter}.

\* **@param** urlBase l'URL de base (une chaîne de caractères) permettant la récupération

\* des éléments du flux.

\* **@throws** JAXBException

\* **@throws** ServicesLocatorException

\*/

**public** **static** **void** *offresEmploi*(Writer writer, String urlBase) **throws** JAXBException, ServicesLocatorException

{

// Récupération du service de gestion des offres d'emploi à l'aide de

// la classe ServiceLocator.

// A éventuellement adapter à votre projet.

IServiceOffreEmploi serviceOffreEmploi =

(IServiceOffreEmploi)ServicesLocator.*getInstance*().getRemoteInterface("ServiceOffreEmploi");

// Récupération des offres d'emploi.

List<OffreEmploi> offres = serviceOffreEmploi.listeDesOffres();

// Création du "contexte" JAXB. Celui-ci est paramétré avec le nom du package

// contenant les classes générées par l'outil xjc.

JAXBContext jc = JAXBContext.*newInstance*("eu.telecom\_bretagne.cabinet\_recrutement.front.rss");

// Le "marshaller" est la classe permettra de gérer la sérialisation :

// instances --> flux XML.

Marshaller marshaller = jc.createMarshaller();

marshaller.setProperty(Marshaller.*JAXB\_FORMATTED\_OUTPUT*, **true**); // Pour que le flux généré soit joli tout plein...

// Instance de l'objet ObjectFactory qui permettra de créer les instances qui

// permettront au marshaller de générer le flux XML.

ObjectFactory fabrique = **new** ObjectFactory();

// Création de l'objet racine (élément <rss>)

Rss rss = fabrique.createRss();

// Mise à jour du numéro de version RSS

rss.setVersion(**new** BigDecimal(2));

// Création du "channel" (élément <channel>)

Channel channel = fabrique.createChannel();

channel.setTitle("Cabinet de recrutement : les offres d'emploi");

channel.setLink(urlBase + "/CabinetRecrutement\_WEB/template.jsp");

channel.setDescription("Fil rouge : gestion d'un cabinet de recrutement. Flux RSS listant les offres d'emploi.");

// Normalement la date est celle du dépôt de la dernière offre. Ici, pour simplifier,

// c'est la date en cours.

channel.setPubDate(Utils.*date2StringRSS*(**new** Date()));

// Création de l'image (optionnel dans la spécif RSS), mise à jour des données

// de celle-ci et affectation de l'image au channel.

Image logo = fabrique.createImage();

logo.setTitle("Cabinet de recrutement : les offres d'emploi");

logo.setUrl(urlBase + "/CabinetRecrutement\_WEB/images/petite\_loupe.png");

logo.setLink(urlBase + "/CabinetRecrutement\_WEB/template.jsp");

channel.setImage(logo);

// On parcours la liste des offres d'emploi. Pour chacune de celles-ci,

// un item est créé, renseigné et ajouté au channel.

**for**(OffreEmploi offre : offres)

{

Item item = fabrique.createItem();

item.setTitle(offre.getTitre() + " (" + offre.getEntreprise().getNom() + ")");

item.setLink(urlBase + "/CabinetRecrutement\_WEB/template.jsp?action=infos\_offre&id=" + offre.getId());

item.setDescription(Utils.*text2HTML*(offre.getDescriptifMission()));

Enclosure enclosure = fabrique.createEnclosure();

enclosure.setUrl(urlBase + "/CabinetRecrutement\_WEB/images/icone\_offre\_emploi.png");

enclosure.setType("image/png");

item.setEnclosure(enclosure);

item.setPubDate(Utils.*date2StringRSS*(offre.getDateDepot()));

// Une petite subtilité : on pourrait penser pouvoir ajouter directement

// l'item au channel mais comme il y a potentiellement plusieurs items, il

// y a une List<Item> qui les référencie. Cette liste est accessible via la

// méthode getItem() qui renvoie la liste, sur cette liste est ajouté le nouvel

// item.

channel.getItem().add(item);

}

// Inclusion du channel dans le rss.

rss.setChannel(channel);

// ---------------------------------------

// A compléter...

//

// Principe, pour chaque offre :

// - créer un élement item

// - renseigner les infos

// - inclusion de l'item dans le channel

// ---------------------------------------

// A ce stade l'objet rss est complet (les données ont toutes été incluses), on

// procède à la sérialisation. La méthode prend en paramètres l'objet à sérialiser

// (rss), le flux sera écrit sur le writer.

// Le writer (instance de l'interface Writer) est en fait l'objet out provenant

// du JSP. ce qui sera écrit sur ce flux sera renvoyé par le serveur Web sur le

// navigateur (ou l'outil affichant le flux RSS).

marshaller.marshal(rss, writer);

}

//-----------------------------------------------------------------------------

/\*\*

\* Construction du flux RSS de la liste des candidature. Celles-ci sont obtenues par

\* l'appel de la méthode {@code listeDesCandidatures()}, voir : {@link IServiceCandidature}.

\* **@param** writer l'instance du {@link Writer} sur lequel sera écrit le flux RSS.

\* La méthode étant appelée au sein d'un JSP, celui-ci est l'instance

\* de l'objet prédéfini {@code out}, instance de {@link JspWriter}.

\* **@param** urlBase l'URL de base (une chaîne de caractères) permettant la récupération

\* des éléments du flux.

\* **@throws** JAXBException

\* **@throws** ServicesLocatorException

\*/

**public** **static** **void** *candidatures*(Writer writer, String urlBase) **throws** JAXBException, ServicesLocatorException

{

IServiceCandidature serviceCandidature =

(IServiceCandidature) ServicesLocator.*getInstance*().getRemoteInterface("ServiceCandidature");

List<Candidature> candidatures = serviceCandidature.listeDesCandidatures();

JAXBContext jc = JAXBContext.*newInstance*("eu.telecom\_bretagne.cabinet\_recrutement.front.rss");

Marshaller marshaller = jc.createMarshaller();

marshaller.setProperty(Marshaller.*JAXB\_FORMATTED\_OUTPUT*, **true**);

ObjectFactory fabrique = **new** ObjectFactory();

Rss rss = fabrique.createRss();

rss.setVersion(**new** BigDecimal(2));

Channel channel = fabrique.createChannel();

channel.setTitle("Cabinet de recrutement : les candidatures");

channel.setLink(urlBase + "/CabinetRecrutement\_WEB/template.jsp");

channel.setDescription("Fil rouge : gestion d'un cabinet de recrutement. Flux RSS listant les candidatures.");

channel.setPubDate(Utils.*date2StringRSS*(**new** Date()));

Image logo = fabrique.createImage();

logo.setTitle("Cabinet de recrutement : les candidatures");

logo.setUrl(urlBase + "/CabinetRecrutement\_WEB/images/petite\_loupe.png");

logo.setLink(urlBase + "/CabinetRecrutement\_WEB/template.jsp");

channel.setImage(logo);

**for**(Candidature candidature : candidatures)

{

Item item = fabrique.createItem();

item.setTitle(candidature.getNom() + " " + candidature.getPrenom());

item.setLink(urlBase + "/CabinetRecrutement\_WEB/template.jsp?action=infos\_candidature&id=" + candidature.getId());

item.setDescription(Utils.*text2HTML*(candidature.getAdresseEmail() + "\n" + candidature.getCv()));

Enclosure enclosure = fabrique.createEnclosure();

enclosure.setUrl(urlBase + "/CabinetRecrutement\_WEB/images/icone\_candidature.png");

enclosure.setType("image/png");

item.setEnclosure(enclosure);

item.setPubDate(Utils.*date2StringRSS*(candidature.getDateDepot()));

channel.getItem().add(item);

}

rss.setChannel(channel);

marshaller.marshal(rss, writer);

}

//-----------------------------------------------------------------------------

}