SER – Laboratoire 2

HEIG-VD

Adriano Ruberto & Matthieu Villard

2016

Table des matières

[1. Introduction 2](#_Toc449628337)

[2. Modification de la Structure du document XML 2](#_Toc449628338)

[3. Grammaire DTD 3](#_Toc449628339)

[4. Génération du document xml 3](#_Toc449628340)

[ControleurXMLCreation 3](#_Toc449628341)

[5. Modification de la structure du document JSON 5](#_Toc449628342)

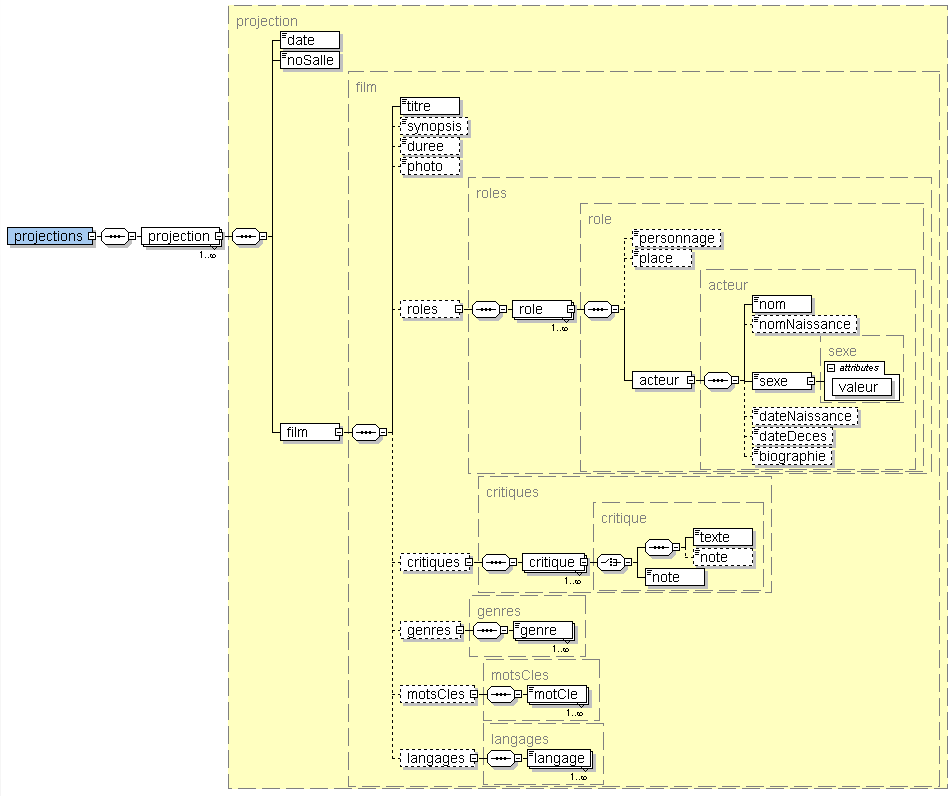
[6. Conclusion 6](#_Toc449628343)

# ****Introduction****

Ce laboratoire a pour but d’implémenter la sérialisation des données en documents XML et JSON. Nous avons donc repris l’application *SER\_PlexAdmin* et l’avons modifiée pour qu’elle permette de sérialiser les données en respectant le format que nous avions déjà défini durant le premier laboratoire. Nous verrons que le format a quelque peu été modifié, notamment pour ajouter certaines contraintes.

Durant ce laboratoire, il nous était demandé de compléter les contrôleurs *ControleurXMLCreation* et *ControleurMedia* et d’en modifier les méthodes *createXML et sendJSONToMedia.* Nous verrons que*,* au lieu de simplement implémenter ces deux méthodes et de décrire tout le processus dans celles-ci, nous avons opté pour l’utilisation de *Serializers*. Ainsi, nous avons créé deux nouvelles classes, à savoir la classe *XMLSerializer* pour générer le document xml et *JSONSerializer* pour générer le document json.

# ****Modification de la Structure du document XML****



Nous avons modifié la structure du document xml afin de vérifier les valeurs du sexe de l’acteur. Ainsi le sexe de l’acteur a un attribut valeur qui est un *enum* et ne peut donc contenir que des valeurs valables.

# Grammaire DTD

La figure ci-dessous illustre la nouvelle grammaire DTD contenue dans le fichier *validator.dtd*, située à la racine du projet *SER\_PlexAdmin*.

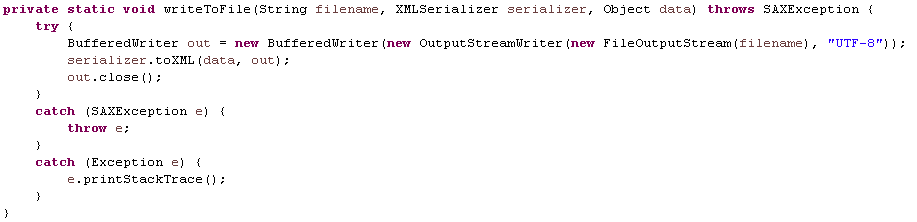


# Génération du document xml

## ControleurXMLCreation

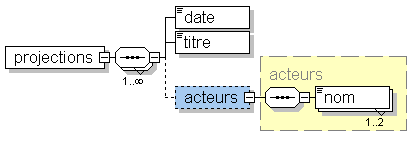


La méthode *createXml* ressemble beaucoup à la méthode *createXStreamXml* déjà existante. Elle récupère les données de la base de données et crée un document xml à partir de celles-ci. La spécificité de cette méthode est d’utiliser un objet XMLSerializer au lien d’un XStream. Cette classe que nous avons implémentée nous-même peut prendre en paramètre le chemin d’un fichier dtd permettant de valider le document xml généré. A noter que les exceptions levées, comme l’illustre la figure ci-dessus permettent de déterminer si le document générer est valide et le cas échéant d’afficher les messages d’erreur dans la fenêtre principale.



La méthode *writeToFile* déjà existante prenait en paramètre un *XStream* et nous avons donc dû la redéfinir. Nous noterons la nécessité de propager les exceptions *SAXException* pour indiquer que le document généré est invalide. Nous pouvons voir que c’est le *XMLSerializer* qui va réellement se charger de générer le document xml.

# Modification de la structure du document JSON



Nous avons modifié la structure du document json afin de supprimer certains éléments de la hiérarchie qui nous semblaient superflus. Ainsi, nous n’avons plus qu’une liste de projections et l’élément *projection* a disparu. De plus, les noms des acteurs ayant joué le premier et le deuxième rôle sont enregistrés dans un tableau, ce qui permet de garantir l’évolutivité de notre document.

# Conclusion