Inngéniérie du WEB sémantique
POLYTECH. 5ième année
UNIVERSITÉ D'AIX-MARSEILLE
Année universitaire 2020/2021

Mohamed Ben Ellefi
Pierre Drap
Odile Papini

Feuille de T. P. 3 : Manipuler une ontologie en java

Préliminaires

Se connecter sous Linux

S'assurer que la version de java est 1.8 (sinon l'installer)

Télécharger les fichiers compréssés dans AMeTICE :
Section **POUR-TP-ING-WEBSEM**
Celle-ci contient :
- un fichier pizza_1.owl
- un dossier compréssé **library-OWL-API-2020** : qui contient de les librairies utiles pour le TP
- un tutoriel OWL-API

Créer un nouveau projet java (java version 1.8) sous Eclipse
Sauvegarder :
- le tutoriel OWL-API
- toutes les librairies du dossier library-OWL-API -2020
- la documentation OWL-API : `http://owlapi.sourceforge.net/documentation.html` qui contient un tutoriel sor $OWL-API$ et des exemples de code java.
- le tutoriel : `http://owlapi.sourceforge.net/owled2011_tutorial.pdf`
- exemples de codes : `https://github.com/phillord/owl-api/blob/master/contract/src /test/java/org/coode/owlapi/examples/Examples.java`

ATTENTION : lorsqu'on travaille avec un fichier owl crée avec Protégé sauvegardé dans un répertoire de l'utilisateur, vérifier que l'IRI (sous Linux) du fichier est bien : $file : ///Users/\ chemin\ jusqu'au\ fichier\ /nom\_fichier.owl$

Création d'une classe java $OWLAPI\_StepByStep$, la classe $OWLAPI\_StepByStep$ contient une classe java $PIZZA$ qui contient les constantes : nom de l'ontologie, IRI, espace de noms.

La philosophie de $OWL-API$: une ontologie (ou une OWL ontologie) est une interface de type $OWLOntology$ qui modélise des axiomes ($OWLAxioms$). Une ontologie a un IRI et des méthodes qui lui sont associées. Dans la suite *ontology* désigne une variable de type $OWLOntology$. Lire attentivement les tutoriels avant de commencer à écrire les méthodes suivantes.

1 Ecrire une méthode $read(OWLAPI\_StepByStep.ontologyFileName)$ qui retourne une ontologie de type *ontology* à partir du fichier $pizza\_1.owl$. Cette méthode affichera le nombre d'axiomes ainsi que le format de la syntaxe de l'ontologie.

Réponses pour l'ontologie $pizza\_1.owl$ *:* 1000 *axiomes et format de l'ontologie : OWL/XML Syntax*

2 Ecrire une méthode une méthode $printClasses(ontology)$ qui affiche les classes d'une ontologie passée en paramètre.

Réponses pour l'ontologie $pizza\_1.owl$ *:* 103 *Classes*

3 Ecrire une méthode une méthode $printObjectProperty(ontology)$ qui affiche les propriétés d'une ontologie passée en paramètre.

Réponses pour l'ontologie $pizza\_1.owl$ *:* 9 *Object Properties*

4 Ecrire une méthode une méthode $printDataTypeProperty(ontology)$ qui affiche les propriétés de type de données d'une ontologie passée en paramètre.

Réponses pour l'ontologie $pizza\_1.owl$ *:* 1 *Data Type Properties*

5 Ecrire une méthode une méthode $printIndividuals(ontology)$ qui affiche les individus d'une ontologie passée en paramètre.

Réponses pour l'ontologie $pizza\_1.owl$ *:* 27 *Individuals*

6 Ecrire une méthode $save(OntoData.ontology, OntoData.manager, savedOntologyFileName\_A)$ qui sauvegarde l'ontologie dans le fichier $pizza\_1\_A.owl$

Vérifier que le fichier $pizza\_1\_A.owl$ *a bien été crée. Utiliser la méthode* $read(OWLAPI\_StepByStep.ontologyFileName)$ *pour vérifier que le nombre d'axiomes est bien le même et que le format est le même que celui de l'ontologie* $pizza\_1.owl$*.*

7 Ecrire une méthode $saveInOWLXML(OntoData.ontology, OntoData.manager, savedOntologyFileName\_B)$ qui sauvegarde l'ontologie au format OWL/XML dans le fichier $pizza\_1\_B.owl$

Vérifier que le fichier $pizza\_1\_B.owl$ *a bien été crée. Utiliser la méthode* $read(OWLAPI\_StepByStep.ontologyFileName)$ *pour vérifier que le nombre d'axiomes est bien le même et que le format est OWL/XML.*

8 Ecrire une méthode

$saveInRDFXML(OntoData.ontology, OntoData.manager, savedOntologyFileName\_B)$ qui sauvegarde l'ontologie $pizza\_1\_B.owl$ au format RDF/XML dans le fichier $pizza\_1\_AA.owl$

Vérifier que le fichier $pizza\_1\_AA.owl$ *a bien été crée. Utiliser la méthode* $read(OWLAPI\_StepByStep.ontologyFileName)$ *pour vérifier que le nombre d'axiomes est bien le même et que celui de l'ontologie* $pizza\_1\_B.owl$.

9 Ecrire une méthode $addingClassesAndComment(OntoData.ontology, OntoData.manager)$ qui ajoute une classe et un commentaire et sauvegarde la nouvelle ontology dans dans le fichier $pizza\_1\_C.owl$. Ajouter la classe $PizzaDuSamediSoir$ qui est une sous-classe de $PizzaDuSoir$ et $PizzaDuSoir$ est une sous-classe de la classe $LSISPizza$ dans l'ontologie contenue dans le fichier $pizza\_1.owl$

Vérifier que le fichier $pizza\_1\_C.owl$ *a bien été crée. Utiliser les méthodes* $read(OWLAPI\_StepByStep.ontologyFileName)$ *et* $printClasses(ontology)$ *pour vérifier que les deux nouvelles classes ont été créees* ou utiliser Protégé.

10 Ecrire une méthode $addingIndividuals(OntoData.ontology, OntoData.manager)$ qui ajoute un individu. Ajouter une instance $MaRoyaleSansOlive$ qui est une instance de $PizzaDuSamediSoir$ classe de l'ontologie contenue dans le fichier $pizza\_1\_C.owl$.
Réponses pour l'ontologie $pizza\_1_C.owl$ *:* 1005 *axiomes et* 105 *Classes*

Vérifier que avec la méthode $printIndividuals(ontology)$ *que l'instance instance* $MaRoyaleSansOlive$ *a bien té créee ou utiliser Protégé avec* $pizza\_1\_C.owl$.
Réponses pour l'ontologie $pizza\_1_C.owl$ *:* 28 *Individuals*

11 Ecrire une méthode $addingDataPropertyToIndividuals(OntoData.ontology, OntoData.manager)$ qui ajoute une propriété de donné (data property). Ajouter un diamètre de 33.3 cm à $MaRoyaleSansOlive$

Vérifier avec la méthode $read(OWLAPI\_StepByStep.ontologyFileName)$ *le nombre d'axiomes* 1008 *ou utiliser Protégé avec* $pizza\_1\_C.owl$.

12 Ecrire une méthode $walkThroughOntology(OntoData.ontology, OntoData.manager)$ qui parcours (visite) l'ontologie. (Voir la notion de visiteur d'ontologie dans les tutoriels)

— Ecrire une fonction *main* pour tester toutes ces méthodes. Tester ces méthodes une par une en suivant l'ordre de l'énoncé.