Inngéniérie du WEB sémantique
POLYTECH. 5ième année
UNIVERSITÉ D'AIX-MARSEILLE
Année universitaire 2017/2018

Pierre Drap
Odile Papini

Feuille de T. P. 2 : Raisonner avec une ontologie en java

Préliminaires

Se connecter sous Linux

Télécharger lles fichiers compréssés dans AMUBox,
dossier **POUR-TP-ING-WEBSEM** à partir de l'URL :
https ://amubox.univ-amu.fr/index.php/s/NydVUndV1deL8Uh

Celle-ci contient :
— un fichier pizza_1.owl
— un dossier **library-OWL-API** : qui contient de les librairies utiles pour le TP

Créer un nouveau projet java sous Eclipse ou NetBeans

Sauvegarder :
— toutes les librairies du dossier library-OWL-API
— 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 : $http://owlapi.sourceforge.net/owled2011\_tutorial.pdf$

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 : $file:///Users/\ chemin\ jusqu'au\ fichier\ /nom\_fichier.owl$

Reprendre la classe java $OWLAPI\_StepByStep$ du TP 1,.
Rappel : la classe $OWLAPI\_StepByStep$ contient une classe java $PIZZA$ qui contient les constantes : nom de l'ontologie, IRI, espace de noms.

Rappel : 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.

— Ecrire une méthode $validation(OWLOntologylocalOntology)$ qui teste si l'ontologie passée en paramètre est cohérente. Si elle elle est cohérente elle affiche un message de même si elle est incohérente de plus, elle contrôle l'expressivité de l'ontologie et la logique de description sous-jacente.

— Ecrire une méthode $generateInferredData(OntoData.ontology, OntoData.manager)$ qui génère l'ontologie inférrée. L'ontologie inférée est stockée dans le fichier $pizza\_1\_D.owl$

— Ecrire une méthode $getInstanceByProperty(OntoData.ontology, OntoData.manager)$ qui permet d'obtenir à partir d'un individu les valeurs des propriétés (Object properties et DataType properties) pour cet individu. Tester avec l'individu $MaRoyaleSansOlive$.

— Ecrire une méthode $printDataFromAnInstance(OntoData.ontology, OntoData.manager)$ qui permet d'obtenir à partir d'un individu, les classes auxquelles appartient cet individu et les valeurs des propriétés pour cet individu. Tester avec l'individu $MaRoyaleSansOlive$.

— Ecrire une méthode $printAllAxiomWithPellet(OntoData.ontology, OntoData.manager)$ qui affiche la hiérarchie de classes (avec indentation) de l'ontologie inférée.

— Ecrire une méthode $printAllIndividualsAndAllTheirProperties(OntoData.ontology)$ qui affiche tous les individus et les valeurs de leurs propriétés.

— Ecrire une fonction $main$ pour tester toutes ces méthodes.