TP N°3 – Manipulation de RDFS en utilisant les bibliothèques Jena

I. Objectif

Le but principal de ce travail pratique est d'approfondir vos connaissances en RDF/RDFS et se familiariser avec la création d'application pour le Web Sémantique en utilisant la bibliothèque Jena.

II. Le RDFS

Voir cours.

III. Intégration de Jena

Voir TP n°2.

IV. Utilisation de Jena

Voir TP n°2

V. Travail à faire (en présentiel)

Le code suivant présente un programme qui permet de créer une ontologie simple en RDFS

1. Créez un nouveau projet Java
2. Ajoutez la bibliothèque Jena à votre projet
3. Créez la Classe suivante dans votre projet (Page 2)
4. Instanciez cette classe et appelez la méthode **`createOnto()`** dans la méthode main de votre projet.

VI. Travail à rendre

1. Exécutez la méthode **`createOnto()`**, et dessinez le graphe RDF que représente le programme
2. Essayer la syntaxe abrégée "RDF/XML-ABBREV". Que remarquez-vous ?
3. Ajoutez à votre ontologie les concepts et les relations suivantes :
    - Concept : Etudiant en graduation « etudGrade » sous classe de la classe Etudiant
    - Concept : Etudiant en post graduation « etudPostGrade » sous classe de la classe Etudiant.
    - Concepts : « maitreAssist » (maitre assistant), « maitreConf » (maitre de conférences) : et « professeur » sous classe d'enseignant
    - La classe « cours »
    - La classe « Promo »
    - La relation « donne-cours », une relation entre Enseignant et cours
    - La relation « concerne », une relation entre cours et « Promo »
    - La relation « appartient », une relation entre « Etudiant » et « Promo »
    - Exprimer que le maitre de conférences « Mohamed » donne le cours « IA » pour la promotion « M2GL »
4. Réalisez un programme avec interface graphique qui permet d'ajouter des concepts et des relations a une ontologie et dessine son graphe (**le dessin est facultatif**).

```java
public class OntoTest {
    public void createOnto() throws FileNotFoundException {
        OntModel model = ModelFactory.createOntologyModel(OntModelSpec.RDFS_MEM);
        String exns = "http://www.exemple.com/vocabulary#";
        model.setNsPrefix("ns", exns);
        // creation des classes
        OntClass hum = model.createClass(exns + "Humain");
        OntClass homme = model.createClass(exns + "Homme");
        OntClass femme = model.createClass(exns + "Femme");
        OntClass etudiant = model.createClass(exns + "Etudiant");
        OntClass enseignant = model.createClass(exns + "Enseignant");
        // spécification des sous classes
        homme.addSuperClass(hum); // ou hum.addSubClass(homme);
        femme.addSuperClass(hum); // ou hum.addSubClass(femme);
        etudiant.addSuperClass(femme);
        etudiant.addSuperClass(homme);
        enseignant.addSuperClass(femme);
        enseignant.addSuperClass(homme);
        // des commentaires
        etudiant.addComment(model.createLiteral("homme ou femme et fait des études"));
        enseignant.addComment(model.createLiteral("homme ou femme qui enseigne"));
        // définition de la propriété "enseigne"
        OntProperty enseigne = model.createOntProperty(exns + "Enseigne");
        enseigne.addDomain(enseignant);
        enseigne.addRange(etudiant);
        // création des instances des classes
        Individual amine = etudiant.createIndividual(exns + "Amine");
        Individual youcef = enseignant.createIndividual(exns + "Youcef");
        youcef.addProperty(enseigne, amine);// ajout la propriété esseigne entre youcef et amine
        PrintWriter w = new PrintWriter(new File("D:\\onto.xml"));
        // Syntaxe complète "RDF/XML" ou abrégée "RDF/XML-ABBREV"
        RDFWriter wr = model.getWriter("RDF/XML");
        // RDFWriter wr=model.getWriter("RDF/XML-ABBREV");
        wr.setProperty("showXmlDeclaration", true);
        wr.write(model, w, null); // écrire dans le fichier
        wr.write(model, System.out, ""); // afficher dans l'écrans
    }
}
```