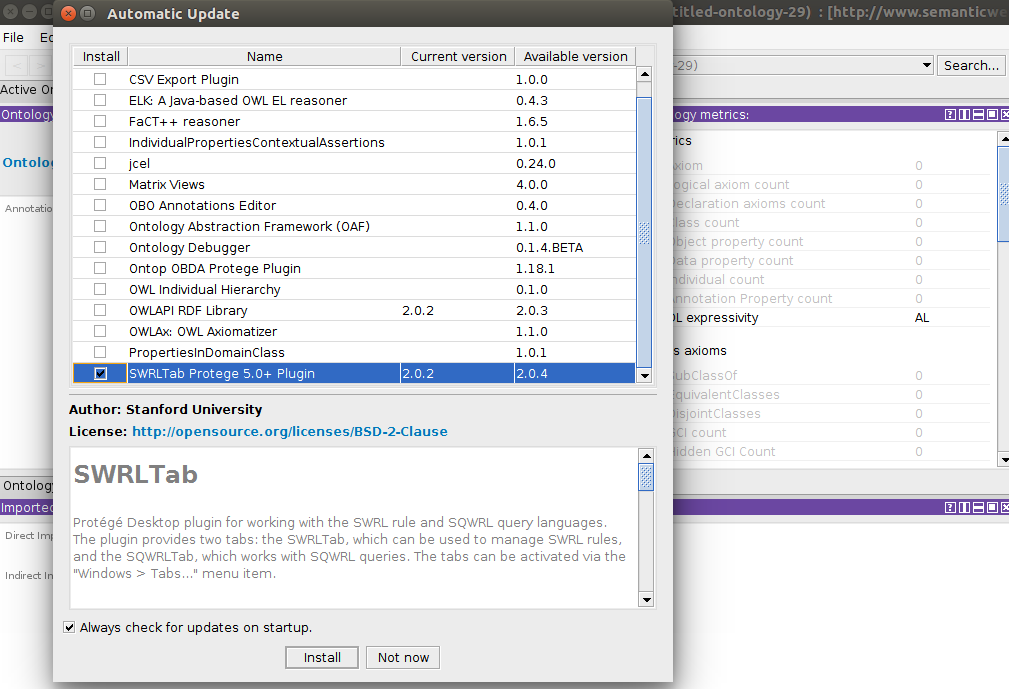
**Mode d’emploi**

**Environnement**

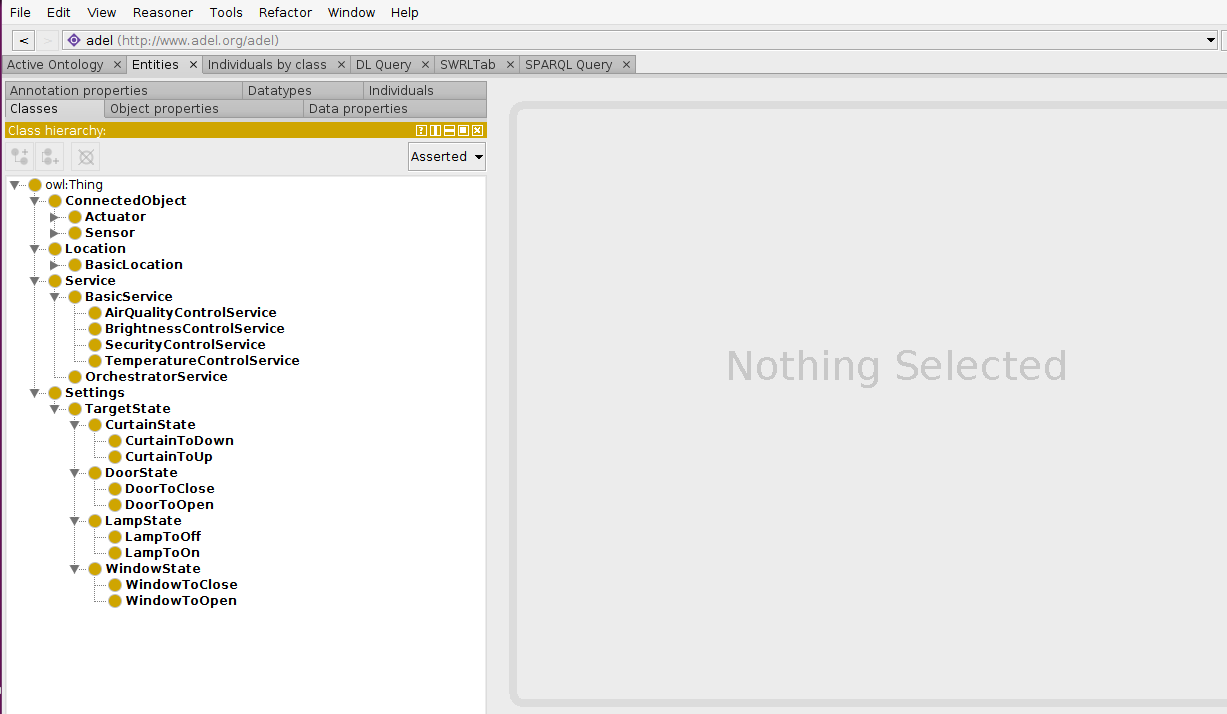
* **Protege 5,2**

-Téléchargement du plugin SWRLTAB

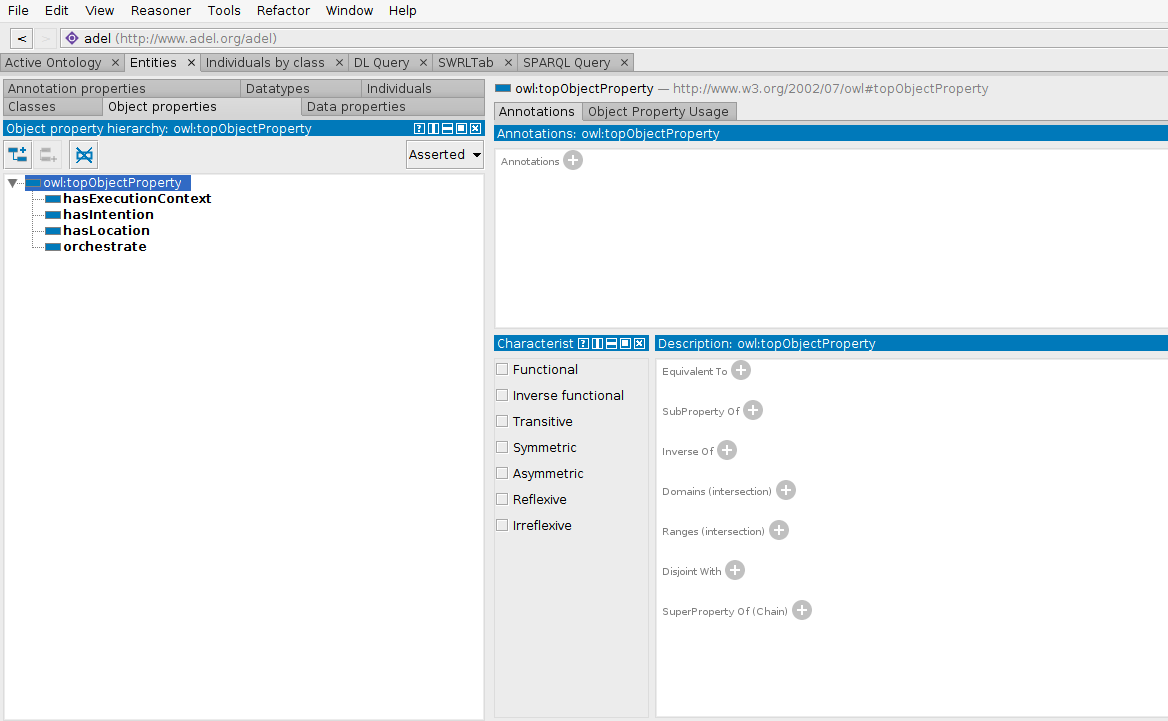


-chargement de l’ontologie : **Freedomotic/freedomotic-core/data/ontologie/freedo,owl**

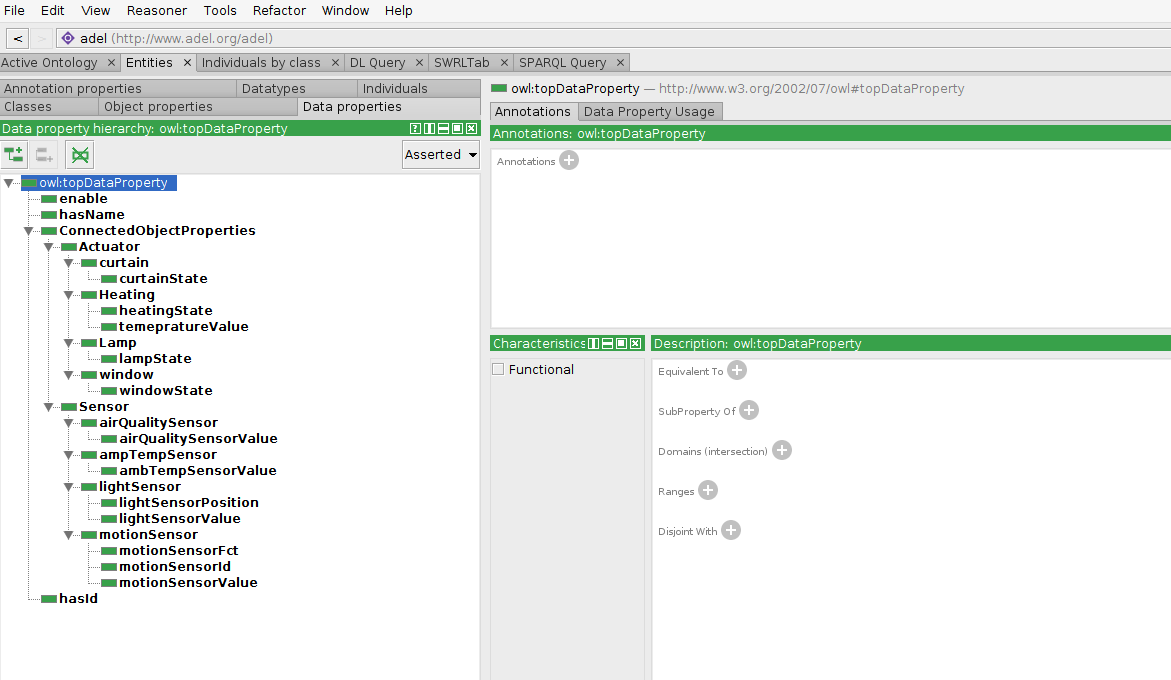
-l’onglet “**Entities => Classes**” pour voir les concepts de l’ontologie



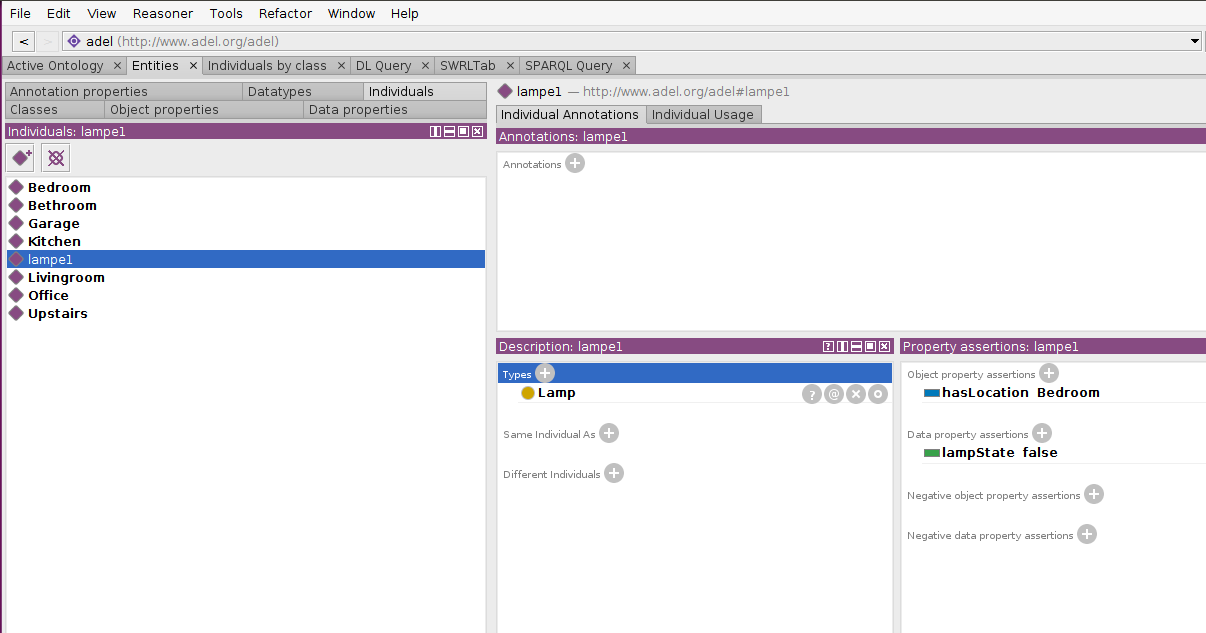
-L’onglet “**Entities => Object Properties**” pour voir les propriétés d’objets



-L’onglet “**Entities => data Properties**” pour voir les propriétés de données



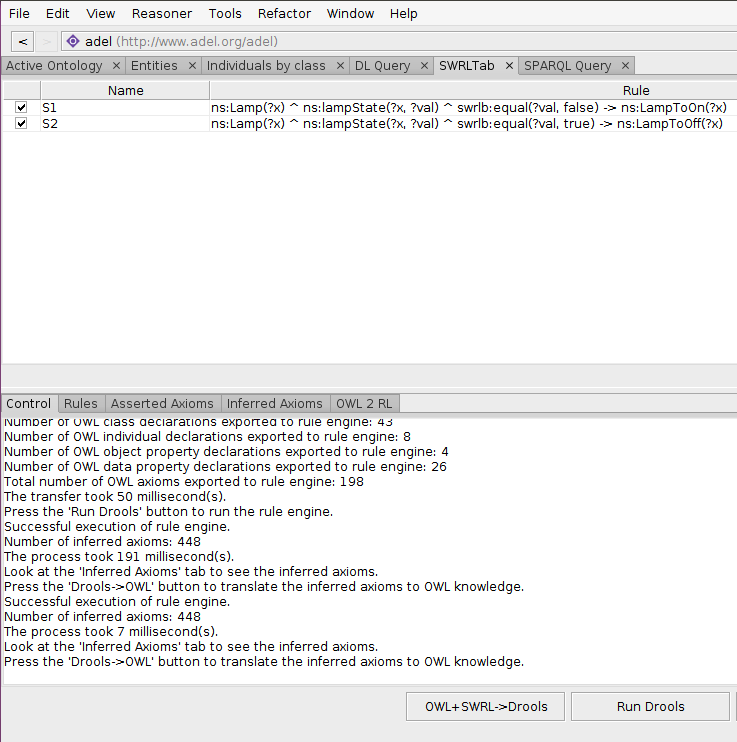
-l’onglet “**Entities => individual**” pour voir les individus ou les instances des concepts de l’ontologie



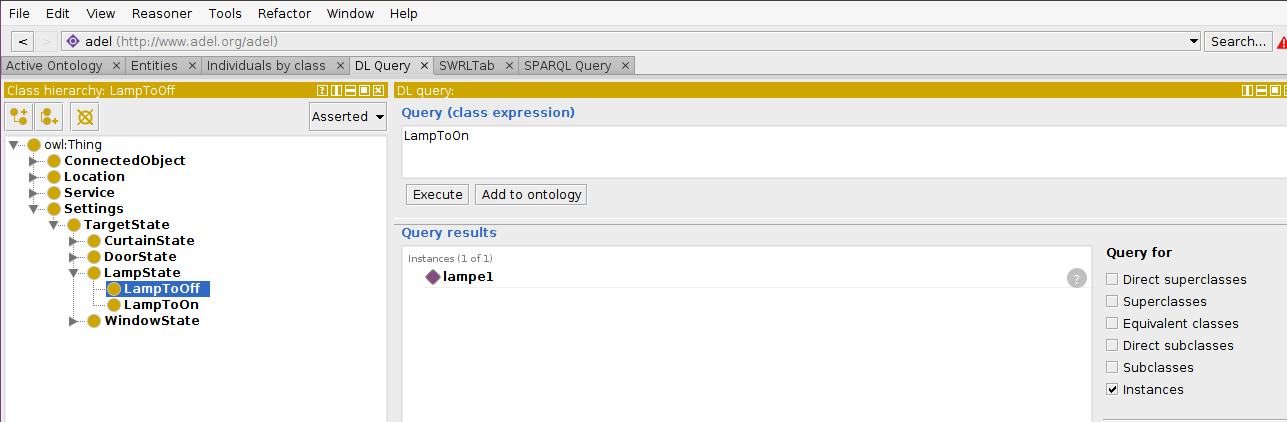
-l’onglet “**SWRLTAB**” pour écrire les règles SWRL

-sur l’onglet “**reasonner**” choisir “Pellet” et le lancer

-Appuyer sur “**run Drools**” pour lancer les règles



-l’onglet « **DL Query** »pour voir l’inférence



**Freedomotic**

**Environnement**

* Eclipse
* Maven
* java7
* Tomcat sur le port 8070

**Projets à importer :**

* freedomotic/plugins/objects/base-things
* freedomotic/plugins/devices/smartHouse
* *freedomotic/framework/freedomotic-core*
* *ProjetWeb (pour la simulation des services et des données des capteurs)*
* P**our les 3 premiers projet faire :**
  + Maven clean
  + Maven install
* **pour lancer freedomotic**
  + exécuter dans « freedomotic-core » la classe « **com. freedomotic.app.Freedomotic.java** »
* **Après chaque modification dans le code des projets freedomotic faire dans l’ordre :**
  + « Maven install » pour le projet SmartHouse
  + « Maven install » pour le projet freedomotic-core
  + lancer freedomotic
* **Pour lancer le projet web « ProjetWeb »**
  + lancer Tomcat sur le port :8070
  + «***http://localhost:8070/ProjetWeb/*** » pour la page du formulaire pour les capteurs
  + «***http://localhost:8070/ProjetWeb/listeService*** » pour la page du formulaire pour les priorités des services

**Mode d’emploi du projet « SmarHouse »**

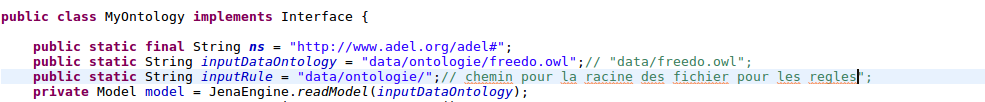
**-package : com.freedomotic.plugins.concepts :** regroupe les différents concepts de l’ontologie.

**-package :com.freedomotic.plugins.devices.smartHouse.application**: possède la classe « MyOntology » pour faire les différents traitement sur l’ontologie ,

Il faut bien s’assurer que les variables :

- « **ns »** représente bien le name space qui représente l’ontologie

-  « **inputDataOntology »** fait référence au chemin de l’ontologie



**-package : com.freedomotic.plugins.devices.smartHouse.tools**

Regroupent les classe nécessaires pour l’API Apache Jena

-**package :com.freedomotic.plugins.Service:**regroupe les concepts« service » de l’ontologie

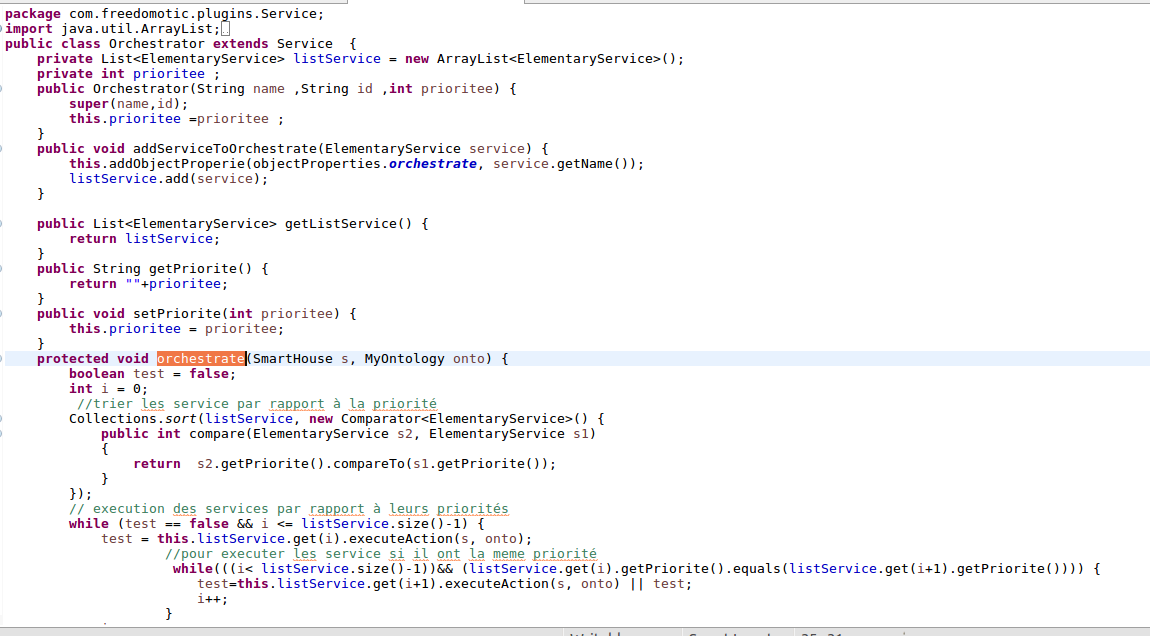
-la classe « **HomeOrchestrator.java** »

grâce a la méthode «**executeServices()**» on passe en revue tous les groupe objectifs et on les orchestres grâce a l’appel de la méthode «**orchestrate**() » et on exécute les service adéquat

****

-la classe «**Orchetrator.java** »

Grâce à la méthode «**orchestrate**()» on trie tous les services que possède l’orchestrateur (groupe objectif) et on exécute les services les plus prioritaire avec l’appel de la méthode «**executeAction**()»



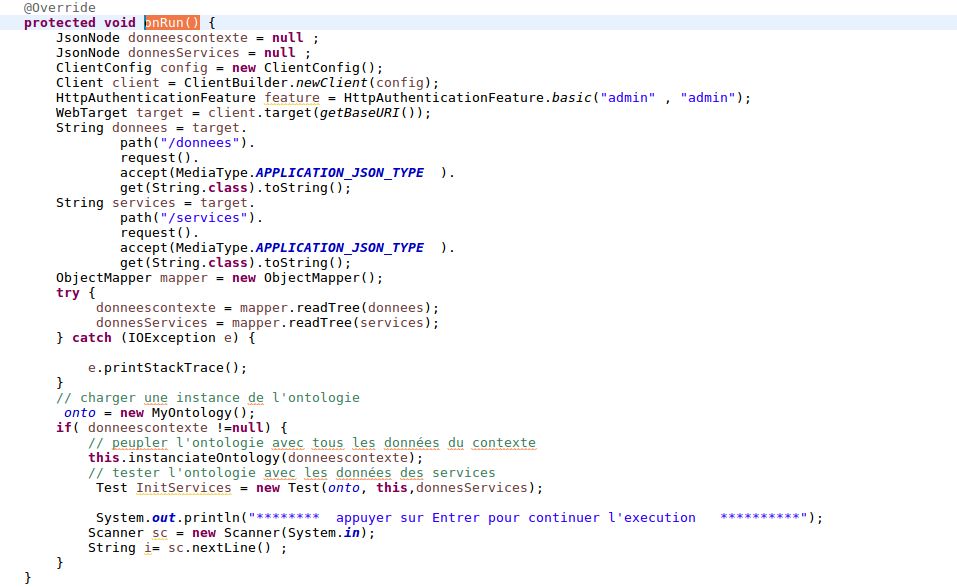
-la classe «**Service.java** »

grâce à la méthode «**executeAction**() » on agira sur les actionneurs pour leurs changer d’état

**-package :com.freedomotic.plugins.devices.smartHouse**

-la classe « **SmartHouse.java**» représente le cœur du plugin. C’est cette classe qu’appellera freedomotic

- la méthode «  **onRun**() » tournera en boucle au lancement du plugin sur freedomotic





-la methode **instanciateOntology() s**ert a récupérer tous les objets sur freedomotic et créer leur équivalents sur l’ontologie

- la classe «**Test.java »** instancie les service récupérer via l’url « http://localhost:8070/ProjetWeb/service/listeService/donnees» et exécute la méthode «**executeServices**() »