Voces

Ştefan Apostoaie, Alexandru Chica, Marina Ghiucă, Irina Roznovăţ

stefan.apostoaie, achica, mghiurca, irina.roznovat @infoiasi.ro

**Abstract.**

This paper is documentation for a project that creates an extension for the Eclipse IDE which allows editing of XML/XHTML documents with embedded meta data vocabularies. The editor shall support XML/XHTML syntax and will be able to dynamically import vocabularies like FOAF, DOAP. RDF parsing of vocabularies is done with Jena. The editor shall be able to consume web services which expose vocabulary syntax. All information about this project can be taken from its site: <https://code.google.com/p/voces/>.

Introduction

Plug-in

Web service

Voces Parser

In **Voces**, vocabularies are parsed using Jena. Jena is a Java API for semantic web applications. It was developed by Brian McBride into Hewlett-Packard laboratories and it is derived from SiRPAC. It can be used to create and manipulate RDF models. Jena provides some interfaces for RDF triplets’ manipulation.

The Jena "Statement" interface provides methods for getting and/or setting the subject, predicate and object of a statement. The object of a statement can be either a resource or a literal and the getObject() method returns an object typed as RDFNode, which is a common superclass of both Resource and Literal. An object type can be determine using instanceOf() method.

RDFNode interface provides a common base for all the elements that can be part from a RDF triple. Literal interface refers to literals and strings that are <object> from the RDF triple. The objects that implements Container, Alt, Bag or Seq interfaces can be also seen as <object> into RDF triples. Objects that implements Property interface can be RDF triples’ predicates.

To create a simple RDF/XML document, Model interface have to be used. ModelMem class creates a RDF model into memory. This class extends ModelCom class that contains all the methods for models’ usage. The ModelRDB class is used to manipulate those RDF models that are stored into relational databases as MySQL, Oracle or PostgreSQL. Against ModelMem models, the ModelsRDB models are persistent.

RDF models are directly accessed through iterators: NodeIterator (for generic nodes – also resources and literals), ResIterator and StmtIterator.

Another way to access information from a RDF model is using SPARQL – query language for RDF. Results are RDF triples or sets of RDF graphs.

Voces Parser uses Jena to parse vocabularies that are received through its WEB service. Only “namespace” and “subjects” properties are extracted from vocabularies and stored into xml files. These output xml files are provided by Voces Parser to web service and later processed by it.

RDF packages used by Voces Parser are:

**import** com.hp.hpl.jena.rdf.model.Model;

**import** com.hp.hpl.jena.rdf.model.ModelFactory;

**import** com.hp.hpl.jena.rdf.model.ResIterator;

**import** com.hp.hpl.jena.rdf.model.Resource;

The following code shows exactly how the RDF model is created with the information from the vocabularies, how its content is processed and only “subjects” properties are extracted from the model.

// create an empty model

Model model = ModelFactory.createDefaultModel();

…

// read the RDF/XML file

model.read(in, "");

…

//take all subjects from a RDF-triplet

ResIterator iter = model.listSubjects();

//process all the subjects

while (iter.hasNext())

{

Resource subject = iter.nextResource();

String resourceInfo = subject.toString();

…

}

For output file name, parser uses a template: File + id + xml extension. For example, being given as input the **FOAF** vocabulary (*http://www.foaf-project.org/*), the output will be File1.xml with the following values:

<?xml version="1.0" encoding="UTF-8" standalone="no" ?>

<Vocabulary>

  <Namespace>http://xmlns.com/foaf/0.1</Namespace>

  <Terminal>knows</Terminal>

  <Terminal>firstName</Terminal>

  <Terminal>icqChatID</Terminal>

  <Terminal>birthday</Terminal>

  <Terminal>givenname</Terminal>

…

</Vocabulary>

Conclusion

References

* <http://www.eclipsepluginsite.com/>
* <http://www.vogella.de/articles/EclipsePlugIn/article.html>
* <http://agile.csc.ncsu.edu/SEMaterials/tutorials/plugin_dev/>
* <http://jena.sourceforge.net/tutorial/RDF_API/index.html>
* <http://jena.sourceforge.net/IO/iohowto.html>
* <http://www.devx.com/semantic/Article/35906>
* <http://willdaniels.co.uk/reference/rdf-vocabulary>
* <http://www.foaf-project.org/>
* <https://trac.usefulinc.com/doap>
* <http://dublincore.org/specifications/>