**Jena学习笔记（一） RDF**

官方描述：Apache Jena（或简称Jena）是一个用于构建语义Web和关联数据应用程序的自由和开源的Java框架。 该框架由不同的API组成，用于处理RDF数据。

Jena是一个用于Java语义Web应用程序的API（应用程序编程接口）。它不是一个程序或工具，如果这是你正在寻找，我建议或许TopBraid Composer作为一个好的选择。因此，Jena的主要用途是帮助您编写处理RDF和OWL文档和描述的Java代码。

官方网站：[http://jena.apache.org/index.html](https://link.jianshu.com/?t=http://jena.apache.org/index.html)  
RDF API教程：[http://jena.apache.org/tutorials/rdf\_api.html](https://link.jianshu.com/?t=http://jena.apache.org/tutorials/rdf_api.html" \t "_blank)

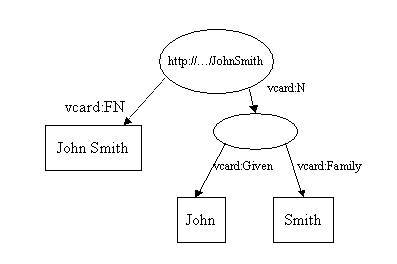
**RDF定义**

资源描述框架（RDF）是最初设计为元数据数据模型的万维网联盟（W3C）规范[1]的家族。 它已经被用作用于概念描述或者在web资源中实现的信息的建模的一般方法，使用各种语法符号和数据串行化格式。 它也用于知识管理应用程序中。  
RDF数据模型类似于诸如实体关系或类图的经典概念建模方法，因为它基于以主语谓词对象的形式对资源（特别是web资源）进行语句表。这些表达式在RDF术语中称为三元组。主语表示资源，谓词表示资源的特征或方面，并且表示主语和对象之间的关系。例如，在RDF中表示“天空具有蓝色”的概念的一种方式是作为三元组：表示“天空”的主题，表示“具有颜色”的谓词和表示“蓝色”的对象。因此，RDF将面向对象设计中的实体 - 属性 - 值模型的经典符号中使用的对象交换对象;实体（天空），属性（颜色）和值（蓝色）。 RDF是具有几种串行格式（即，文件格式）的抽象模型，因此资源或三元组被编码的特定方式随格式而变化。

**下载Jena并从Eclipse建立工程，导入jar**

* RDF API 教程 [http://jena.apache.org/tutorials/rdf\_api.html](https://link.jianshu.com/?t=http://jena.apache.org/tutorials/rdf_api.html)

**建立如下RDF图并打印**



package com.hdu.rdf;

import org.apache.jena.rdf.model.Model;

import org.apache.jena.rdf.model.ModelFactory;

import org.apache.jena.rdf.model.Property;

import org.apache.jena.rdf.model.RDFNode;

import org.apache.jena.rdf.model.Resource;

import org.apache.jena.rdf.model.Statement;

import org.apache.jena.rdf.model.StmtIterator;

import org.apache.jena.vocabulary.VCARD;

public class Tutorial2AddProperty {

public static void main(String[] args) {

// some definitions

String personURI = "http://somewhere/JohnSmith";

String givenName = "John";

String familyName = "Smith";

String fullName = givenName + " " + familyName;

// create an empty model

Model model = ModelFactory.createDefaultModel();

// create the resource

// and add the properties cascading style

Resource johnSmith = model.createResource(personURI).addProperty(VCARD.FN, fullName).addProperty(VCARD.N,

model.createResource().addProperty(VCARD.Given, givenName).addProperty(VCARD.Family, familyName));

// model.write(System.out);

// list the statements in the graph

StmtIterator iter = model.listStatements();

// print out the predicate, subject and object of each statement

while (iter.hasNext()) {

Statement stmt = iter.nextStatement();

Resource subject = stmt.getSubject(); // get the subject

Property predicate = stmt.getPredicate(); // get the predicate

RDFNode object = stmt.getObject(); // get the object

System.out.print(subject.toString());

System.out.print(" " + predicate.toString() + " ");

if (object instanceof Resource) {

System.out.print(object.toString());

} else {

// object is a literal

System.out.print(" \"" + object.toString() + "\"");

}

System.out.println(" .");

}

}

}

**说明**

他的RDF规范指定如何将RDF表示为XML。 RDF XML语法相当复杂。读者参考由RDFCore WG开发的引物以进行更详细的介绍。但是，让我们快速看看如何解释上面的。

RDF通常嵌入在<rdf：RDF>元素中。如果有其他方式知道某些XML是RDF，但是它通常存在，那么该元素是可选的。 RDF元素定义文档中使用的两个命名空间。然后是一个<rdf：Description>元素，描述其URI为“http：// somewhere / JohnSmith”的资源。如果缺少rdf：about属性，则此元素将表示空白节点。

<vcard：FN>元素描述了资源的属性。属性名称是vcard命名空间中的“FN”。 RDF通过连接命名空间前缀的URI引用和名称的本地名称部分的“FN”，将其转换为URI引用。这提供了一个URI引用"[http://www.w3.org/2001/vcard-rdf/3.0#FN](https://link.jianshu.com/?t=http://www.w3.org/2001/vcard-rdf/3.0#FN)"。属性的值是文字"John Smith"。

<vcard：N>元素是一个资源。在这种情况下，资源由相对URI引用表示。 RDF通过将其与当前文档的基本URI连接，将其转换为绝对URI引用。

此RDF XML中有错误;它不完全代表我们创建的模型。模型中的空白节点已经被赋予URI引用。它不再是空白。 RDF / XML语法不能表示所有RDF模型;例如它不能表示作为两个语句的对象的空白节点。我们用来编写这个RDF / XML的'dumb'作者没有尝试正确地写入可以正确写入的Models子集。它为每个空白节点提供一个URI，使其不再为空。

**读取RDF**

static final String inputFileName = "vc-db-1.rdf";

// create an empty model

Model model = ModelFactory.createDefaultModel();

// use the FileManager to find the input file

InputStream in = FileManager.get().open( inputFileName );

if (in == null) {

throw new IllegalArgumentException(

"File: " + inputFileName + " not found");

}

// read the RDF/XML file

model.read(in, null);

// write it to standard out

model.write(System.out);

**Jena RDF包**

Jena是用于语义Web应用程序的Java API。应用程序开发人员的关键RDF包是org.apache.jena.rdf.model。 API已经根据接口进行了定义，以便应用程序代码可以在不改变的情况下与不同的实现工作。此包包含用于表示模型，资源，属性，文字，语句和RDF的所有其他关键概念的接口，以及用于创建模型的ModelFactory。所以应用程序代码保持独立的实现，最好是尽可能使用接口，而不是特定的类实现。

org.apache.jena.tutorial包包含本教程中使用的所有示例的工作源代码。

org.apache.jena ... impl包包含可能对许多实现通用的实现类。例如，它们定义类ResourceImpl，PropertyImpl和LiteralImpl，它们可以被不同的实现直接使用或子类化。应用程序应该很少，如果有的话，直接使用这些类。例如，不是创建ResourceImpl的新实例，最好使用正在使用的任何模型的createResource方法。这样，如果模型实现使用了Resource的优化实现，则两种类型之间不需要转换。

**浏览模型**

给定资源的URI，可以使用Model.getResource（String uri）方法从模型中检索资源对象。 此方法定义为返回一个Resource对象（如果模型中存在），否则创建一个新对象。

static final String inputFileName = "vc-db-1.rdf";

static final String johnSmithURI = "http://somewhere/JohnSmith/";

public static void main (String args[]) {

// create an empty model

Model model = ModelFactory.createDefaultModel();

// use the FileManager to find the input file

InputStream in = FileManager.get().open(inputFileName);

if (in == null) {

throw new IllegalArgumentException( "File: " + inputFileName + " not found");

}

// read the RDF/XML file

model.read(new InputStreamReader(in), "");

// retrieve the Adam Smith vcard resource from the model

Resource vcard = model.getResource(johnSmithURI);

// retrieve the value of the N property

Resource name = (Resource) vcard.getRequiredProperty(VCARD.N)

.getObject();

// retrieve the given name property

String fullName = vcard.getRequiredProperty(VCARD.FN)

.getString();

// add two nick name properties to vcard

// 获取johnSmithURI资源后，对其添加两个nickname属性

vcard.addProperty(VCARD.NICKNAME, "Smithy")

.addProperty(VCARD.NICKNAME, "Adman");

// set up the output

System.out.println("The nicknames of \"" + fullName + "\" are:");

// list the nicknames

//返回声明迭代器，获取对象

StmtIterator iter = vcard.listProperties(VCARD.NICKNAME);

while (iter.hasNext()) {

System.out.println(" " + iter.nextStatement().getObject()

.toString());

}

}

The nicknames of "John Smith" are:

Adman

Smithy

**查询模型**

上一节讨论从具有已知URI的资源导航模型的情况。 本节介绍搜索模型。 核心Jena API仅支持有限的查询原语。 更强大的查询功能在SPARQL中。

Model.listStatements（）方法列出了模型中的所有语句，也许是查询模型的最粗糙的方法。 它的使用不推荐在非常大的模型。 Model.listSubjects（）是类似的，但返回一个迭代器在所有具有属性的资源，即一些语句的主题。

Model.listSubjectsWithProperty（Property p，RDFNode o）将返回所有资源的迭代器，这些资源具有值为o的属性p。 如果我们假设只有vcard资源会有vcard：FN属性，并且在我们的数据中，所有这些资源都有这样的属性，那么我们可以找到所有的vcards：

static final String inputFileName = "vc-db-1.rdf";

public static void main (String args[]) {

// create an empty model

Model model = ModelFactory.createDefaultModel();

// use the FileManager to find the input file

InputStream in = FileManager.get().open(inputFileName);

if (in == null) {

throw new IllegalArgumentException( "File: " + inputFileName + " not found");

}

// read the RDF/XML file

model.read( in, "");

// select all the resources with a VCARD.FN property

ResIterator iter = model.listResourcesWithProperty(VCARD.FN);

if (iter.hasNext()) {

System.out.println("The database contains vcards for:");

while (iter.hasNext()) {

System.out.println(" " + iter.nextResource()

.getRequiredProperty(VCARD.FN)

.getString() );

}

} else {

System.out.println("No vcards were found in the database");

}

}

The database contains vcards for:

Sarah Jones

John Smith

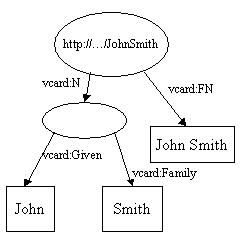
Becky Smith

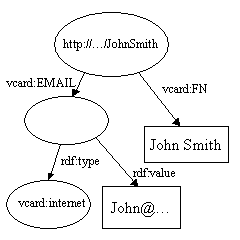
Matt Jones

**模型操作**

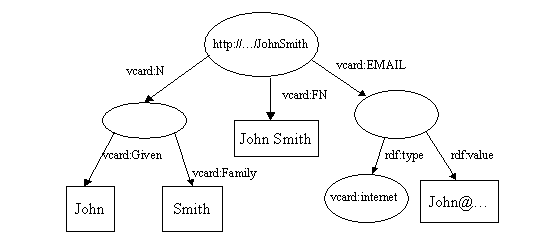
Jena提供了操作模型作为一个整体的三个操作,分別是并集，交集和差的运算。

两个模型的并集是表示每个模型的语句集合的合并。这是RDF设计支持的关键操作之一。它允许合并来自不同数据源的数据。 考虑以下两个模型：





当这些被合并时，两个http：//...JohnSmith节点被合并成一个，并且vcard：FN 弧被丢弃以产生：



static final String inputFileName1 = "vc-db-3.rdf";

static final String inputFileName2 = "vc-db-4.rdf";

public static void main (String args[]) {

// create an empty model

Model model1 = ModelFactory.createDefaultModel();

Model model2 = ModelFactory.createDefaultModel();

// use the class loader to find the input file

InputStream in1 = FileManager.get().open(inputFileName1);

if (in1 == null) {

throw new IllegalArgumentException( "File: " + inputFileName1 + " not found");

}

InputStream in2 = FileManager.get().open(inputFileName2);

if (in2 == null) {

throw new IllegalArgumentException( "File: " + inputFileName2 + " not found");

}

// read the RDF/XML files

model1.read( in1, "" );

model2.read( in2, "" );

// merge the graphs

Model model = model1.union(model2);

// print the graph as RDF/XML

model.write(System.out, "RDF/XML-ABBREV");

System.out.println();

**容器**

RDF定义了一种特殊的资源来表示事物的集合。 这些资源称为容器。 容器的成员可以是文字或资源。 有三种容器：

BAG是一个无序的集合  
ALT是旨在表示可选的无序集合  
SEQ是有序集合  
容器由资源表示。 该资源将有一个rdf：type属性，其值应为rdf：Bag，rdf：Alt或rdf：Seq之一，或其中一个的子类，具体取决于容器的类型。 容器的第一个成员是容器的rdf：\_1属性的值; 容器的第二个成员是容器的rdf：\_2属性的值，等等。 rdf：\_nnn属性被称为序数属性。

例如，包含Smith的vcards的简单包的模型可能如下所示：

static final String inputFileName = "vc-db-1.rdf";

public static void main (String args[]) {

// create an empty model

Model model = ModelFactory.createDefaultModel();

// use the class loader to find the input file

InputStream in = FileManager.get().open( inputFileName );

if (in == null) {

throw new IllegalArgumentException( "File: " + inputFileName + " not found");

}

// read the RDF/XML file

model.read(new InputStreamReader(in), "");

// create a bag

Bag smiths = model.createBag();

// select all the resources with a VCARD.FN property

// whose value ends with "Smith"

StmtIterator iter = model.listStatements(

new

SimpleSelector(null, VCARD.FN, (RDFNode) null) {

@Override

public boolean selects(Statement s) {

return s.getString().endsWith("Smith");

}

});

// add the Smith's to the bag

while (iter.hasNext()) {

smiths.add( iter.nextStatement().getSubject());

}

// print the graph as RDF/XML

model.write(new PrintWriter(System.out));

System.out.println();

// print out the members of the bag

NodeIterator iter2 = smiths.iterator();

if (iter2.hasNext()) {

System.out.println("The bag contains:");

while (iter2.hasNext()) {

System.out.println(" " +

((Resource) iter2.next())

.getRequiredProperty(VCARD.FN)

.getString());

}

} else {

System.out.println("The bag is empty");

}

}

**词汇表**

空白节点（Blank Node）  
表示资源，但不指示资源的URI。空白节点表现为一阶逻辑中存在的合格变量。  
Dublin Core  
关于Web资源的元数据标准。更多信息可以在Dublin Core网站上找到。  
文字（Literal）  
可以是属性值的字符串。  
对象（Object）  
三元组的一部分，即语句的值。  
谓词（Predicate）  
三元组的属性部分。  
属性（Property）  
属性是资源的属性。例如DC.title是一个属性，和RDF.type一样。  
资源（Resource）  
一些实体。它可以是诸如网页的web资源，或者它可以是具体的物理事物，例如树或汽车。它可以是一个抽象的想法，如象棋或足球。资源由URI命名。  
声明（Statement）  
RDF模型中的弧，通常被解释为事实。  
主语Subject）  
作为RDF模型中的弧源的资源  
三元组（Triple）  
包含主语，谓词和对象的结构。语句的另一个术语。

**脚注**

RDF资源的标识符可以包括片段标识符，例如， http：// hostname / rdf / tutorial /＃ch-简介，因此，严格地说，RDF资源由URI引用标识。  
除了作为一个字符串，字面量还有一个可选的语言编码来表示字符串的语言。例如，文字“two”对于英语可能具有“en”的语言编码，而对于法国，文字“deux”可能具有“fr”的语言编码。