Tutorial 4: Introducing RDFS & OWL

Next: Querying Semantic Data

Having introduced the advantages of modeling vocabulary and semantics in data models, let's introduce the actual technology used to attribute RDF data models with semantics. RDF data can be encoded with semantic metadata using two syntaxes: RDFS and OWL.

After this tutorial, you should be able to:

- Understand how RDF data models are semantically encoded using RDFS and OWL
- Understand that OWL ontologies are RDF documents
- Understand OWL *classes*, *subclasses* and *individuals*
- Understand OWL *properties*
- Build your own basic ontology, step by step

Estimated time: 5 minutes

You should have already understood the following tutorial (and pre-requisites) before you begin:

- Tutorial 3: Semantic Modeling

In the last lesson, we compared some of the more popular traditional forms of modeling data with the semantic model, and then introduced a situation where data sharing was enhanced and made significantly easier by using a semantic web approach.

This example demonstrated a situation where two independent websites could mutually benefit by sharing data between them. In this lesson, we will explore the formal syntax used to annotate RDF data with semantic metadata. In the next lesson, we will then explore how this data is published and queried.

RDF data is annotated with semantic metadata using two principal syntaxes: **RDFS** and **OWL**. Both RDFS and OWL are W3C specifications.

4.1 A Starting Example

Just to show you what OWL looks like, here is a quick example:

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <!-- OWL Header Example -->
    <owl:Ontology rdf:about="http://www.linkeddatatools.com/plants">
        <dc:title>The LinkedDataTools.com Example Plant Ontology</dc:title>
        <dc:description>An example ontology written for the LinkedDataTools.com RDFS
        & OWL introduction tutorial</dc:description>
    </owl:Ontology>

    <!-- OWL Class Definition Example -->
    <owl:Class rdf:about="http://www.linkeddatatools.com/plants#planttype">
        <rdfs:label>The plant type</rdfs:label>
        <rdfs:comment>The class of plant types.</rdfs:comment>
    </owl:Class>

</rdf:RDF>
```

Do not concern yourself with the fine details for now, we will look at those later. Do notice however two new namespaces we've introduced in the header of our RDF document that we didn't have before: the RDFS (RDF Schema, **http://www.w3.org/2000/01/rdf-schema#**) and OWL (Web Ontology Language, **http://www.w3.org/2002/07/owl#**) namespaces on lines 03 and 04.

Notice also that we are defining our ontology in RDF - yes, an ontology is also an RDF document.

Let's break down a typical ontology document before building one up. For this example, we will look at a simple ontology that defines plant varieties.

Point Of Interest Why OWL, not WOL? When the acronym for Web Ontology Language (OWL) was first proposed by the working group, OWL was adopted instead of WOL as it is easily remembered, and suggested wisdom. But confusingly enough, it is still an acronym that should strictly speaking be WOL.

4.2 OWL Header

```xml
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/">

    <!-- OWL Header Example -->
    <owl:Ontology rdf:about="http://www.linkeddatatools.com/plants">
        <dc:title>The LinkedDataTools.com Example Plant Ontology</dc:title>
        <dc:description>An example ontology written for the LinkedDataTools.com RDFS & OWL introduction tutorial</dc:description>
    </owl:Ontology>

    <!-- Remainder Of Document Omitted For Brevity... -->

</rdf:RDF>
```

Although an ontology doesn't have to include a header, it is a good place to include information that will help others to understand what your ontology contains.

As above, we've included a title and description for the ontology. But, this is also the place where we could include version information (to make sure you can appropriately track and communicate updates to your ontology) and where you can state that your ontology imports other ontologies.

If your ontology does use elements from other ontologies, this will be absolutely necessary for tools or frameworks to know that your ontology is dependent on others.

Point Of Interest See line 05 where we've defined the namespace for the **dc:** prefix, **http://purl.org/dc/elements/1.1/**. This references the Dublin Core Metadata Initiative namespace and tells machine readers that elements such as **dc:title** and **dc:description** are defined in this ontology. Remember - we mentioned this often used ontology at the end of the previous tutorial. And, because it's an OWL ontology, it is also defined in RDF. Just point your browser to its namespace URI to download it.

4.3 OWL Classes, Subclasses & Individuals

The primary purpose of your ontology is to classify things in terms of semantics, or meaning. In OWL, this is achieved through the use of *classes* and *subclasses*, instances of which in OWL are called *individuals*. The individuals that are members of a given OWL class are called its *class extension*.

A *class* in OWL is a classification of individuals into groups which share common characteristics. If an individual is a member of a class, it tells a machine reader that it falls under the semantic classification given by the OWL class.

An Example

Here is our example OWL ontology again, this time with some added classes and subclasses. We define three plant classes: the **flowering plants** class and **shrubs** class. which are both subclasses of the **planttype** class.

The **planttype** class is the highest level class of all plant types.

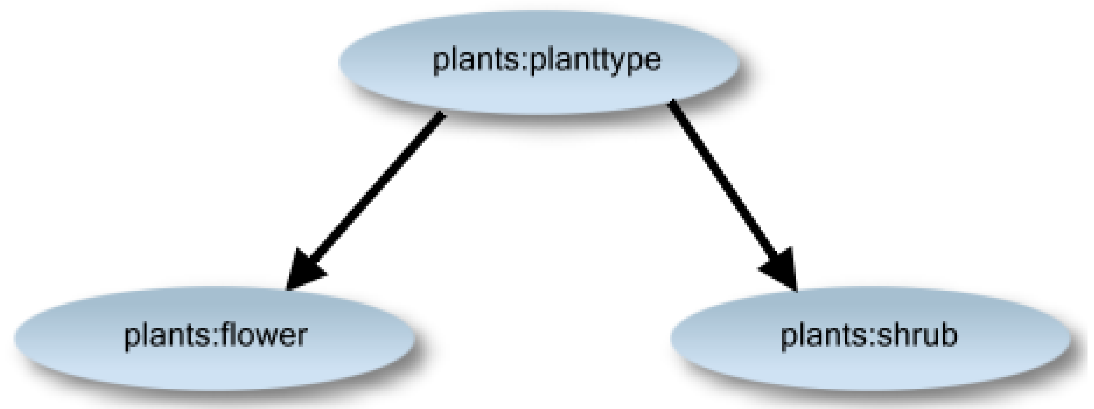
```
01. <rdf:RDF
02.     xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
03.     xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
04.     xmlns:owl="http://www.w3.org/2002/07/owl#"
05.     xmlns:dc="http://purl.org/dc/elements/1.1/"
06.     xmlns:plants="http://www.linkeddatatools.com/plants#">
```

```xml

    <!-- OWL Header Omitted For Brevity -->

    <!-- OWL Class Definition - Plant Type -->
    <owl:Class rdf:about="http://www.linkeddatatools.com/plants#planttype">

        <rdfs:label>The plant type</rdfs:label>
        <rdfs:comment>The class of all plant types.</rdfs:comment>

    </owl:Class>

    <!-- OWL Subclass Definition - Flower -->
    <owl:Class rdf:about="http://www.linkeddatatools.com/plants#flowers">

        <!-- Flowers is a subclassification of planttype -->
        <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/plants#planttype"/>

        <rdfs:label>Flowering plants</rdfs:label>
        <rdfs:comment>Flowering plants, also known as angiosperms.</rdfs:comment>

    </owl:Class>

    <!-- OWL Subclass Definition - Shrub -->
    <owl:Class rdf:about="http://www.linkeddatatools.com/plants#shrubs">

        <!-- Shrubs is a subclassification of planttype -->
        <rdfs:subClassOf rdf:resource="http://www.linkeddatatools.com/plants#planttype"/>

        <rdfs:label>Shrubbery</rdfs:label>
        <rdfs:comment>Shrubs, a type of plant which branches from the base.</rdfs:comment>

    </owl:Class>

    <!-- Individual (Instance) Example RDF Statement -->
    <rdf:Description rdf:about="http://www.linkeddatatools.com/plants#magnolia">

        <!-- Magnolia is a type (instance) of the flowers classification -->
        <rdf:type rdf:resource="http://www.linkeddatatools.com/plants#flowers"/>

    </rdf:Description>

</rdf:RDF>
```

Point Of Interest You can investigate this RDF document further by passing it through W3C's RDF validator, which automatically tabulates the document's RDF subjects, predicates and objects for you. This can help significantly improve your understanding of RDF. Just click the 'copy to clipboard' button on the top right of the code excerpt and paste into the validator.

Taxonomy - A Hierarchy Of Terms

What we've done is define our semantic terms, or classes, in a hierarchy. In the semantic web world, this hierarchy of terms is called a *taxonomy*. Here's a graphical illustration the taxonomy hierarchy we've defined:



An example of a taxonomy hierarchy

Note We haven't created another *subclass* of the **flower** class called **magnolia**. Rather, **magnolia** is an *individual* (instance) of class **flower**. Why is this? Magnolia is a member of the **flower** classification, but it is **not** a further flower **subclassification**. It makes sense from a semantic perspective for magnolia - and indeed other flowers - to be *individuals* (instances) of the class **flower** and not *subclassifications*.

4.4 OWL Properties

Individuals in OWL are related by *properties*. There are two types of property in OWL:

- **Object properties** (owl:ObjectProperty) relates individuals (instances) of two OWL classes.
- **Datatype properties** (owl:DatatypeProperty) relates individuals (instances) of OWL classes to literal values.

An Example

Let's first add a *data type* property (one which links an instance to a literal value) and add the name of the *species family* the Magnolia is part of.

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:plants="http://www.linkeddatatools.com/plants#">

    <!-- OWL Header Omitted For Brevity -->

    <!-- OWL Classes Omitted For Brevity -->

    <!-- Define the family property -->
    <owl:DatatypeProperty rdf:about="http://www.linkeddatatools.com/plants#family"/>

    <rdf:Description rdf:about="http://www.linkeddatatools.com/plants#magnolia">

        <!-- Magnolia is a type (instance) of the flowers class -->
        <rdf:type rdf:resource="http://www.linkeddatatools.com/plants#flowers"/>

        <!-- The magnolia is part of the 'Magnoliaceae' family -->
        <plants:family>Magnoliaceae</plants:family>

    </rdf:Description>

</rdf:RDF>
```

Important Point At this point, if you are an object oriented programmer, your mind may well be thinking of programmatic object classes and their associated properties and comparing them to what we've just learned out OWL classes. Don't - they're not quite the same. Note from the example above. The 'family' property was defined independent of any class type, and was assigned to the *instance* of class **flower** (magnolia). Another instance of the same class may not have this property. So in OWL, note that the properties that instances have are not described in their *class types*, but their *instances*. In this case, you may use the same 'family' property for an instance of a completely different class.

Finally, let's add an *object* property (one which links an instance to another instance). Let's say we're running a shop, and we want to link this plant (Magnolia) to another plant which we know as the shop owner is equally as popular. Let's add a property called "similarlyPopularTo":

```
<rdf:RDF
    xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
    xmlns:owl="http://www.w3.org/2002/07/owl#"
    xmlns:dc="http://purl.org/dc/elements/1.1/"
    xmlns:plants="http://www.linkeddatatools.com/plants#">

    <!-- OWL Header Omitted For Brevity -->

    <!-- OWL Classes Omitted For Brevity -->

    <!-- Define the family property -->
    <owl:DatatypeProperty rdf:about="http://www.linkeddatatools.com/plants#family"/>

    <!-- Define the similarlyPopularTo property -->
    <owl:ObjectProperty rdf:about="http://www.linkeddatatools.com/plants#similarlyPopularTo"/>

    <!-- Define the Orchid class instance -->
    <rdf:Description rdf:about="http://www.linkeddatatools.com/plants#orchid">

        <!-- Orchid is an individual (instance) of the flowers class -->
        <rdf:type rdf:resource="http://www.linkeddatatools.com/plants#flowers"/>

        <!-- The orchid is part of the 'Orchidaceae' family -->
        <plants:family>Orchidaceae</plants:family>

        <!-- The orchid is similarly popular to the magnolia -->
        <plants:similarlyPopularTo rdf:resource="http://www.linkeddatatools.com/plants#magnolia"/>

    </rdf:Description>

    <!-- Define the Magnolia class instance -->
    <rdf:Description rdf:about="http://www.linkeddatatools.com/plants#magnolia">

        <!-- Magnolia is an individual (instance) of the flowers class -->
        <rdf:type rdf:resource="http://www.linkeddatatools.com/plants#flowers"/>

        <!-- The magnolia is part of the 'Magnoliaceae' family -->
        <plants:family>Magnoliaceae</plants:family>

        <!-- The magnolia is similarly popular to the orchid -->
        <plants:similarlyPopularTo rdf:resource="http://www.linkeddatatools.com/plants#orchid"/>

    </rdf:Description>

</rdf:RDF>
```

To illustrate, we have defined a new individual (instance) of the flowers class representing the Orchid with the URI *http://www.linkeddatatools.com/plants#orchid*. See if you can understand this from how we have defined the Magnolia instance, which is an individual of the same class. Now, just like in our first two tutorials, see if you can draw a graph showing the Orchid and Magnolia class instances and their predicates, according to the RDF graph defined above.

Hint: You will need to show arrows for both the *family* and *similarlyPopularTo* properties. For the *family* property, they will point to the different family type literals for each plant. For the *similarlyPopularTo* property, the instances will point to each other.

Important Point Note in the example above, we have a two way link between two instances of the same OWL class via the *similarlyPopularTo* object property (both Orchid and Magnolia are individuals of the same class). However note that object properties need not be two way - they may be one way. And, just as importantly, need not be between instances of the same OWL class. They may be completely different OWL classes.

You have completed this lesson. You should now understand the following:

- That RDFS and OWL are W3C specifications with their own standard W3C namespaces.
- How OWL headers are constructed and some example uses.
- How to implement your own OWL classes, subclasses, individuals and properties.
- How to build your own basic ontology.

You should now be able to start the following tutorial:

- Tutorial 5: Querying Semantic Data

SHARE

Comments

#21 2016-05-13 07:40

The Semantic Web isn't just about putting data on the web. It is about making links, so that a person or machine can explore the web of data. With linked data, when you have some of it, you can find other, related, data. https://shampoosik.ru/catalog/shampoos/?brand=shu_uemura_art_of_hair_
Tj3eOI7yW

Quote

#20 2016-01-28 23:16

Its very clear and understandable awesome!!!

I appreciate your great effort..

Quote

#19 2015-05-08 07:01

thank you, I have tried to got the idea of (SW, OWL, RDF) , just now I understand them.

Quote

#18 2013-10-21 12:23

very useful info, thank you

Quote

#17 **lorein** 2013-08-21 12:40

thanx for this tutorial but i want to ask what is the tool that i should download to write all those codes ?

Quote

#16 2013-07-09 09:12

An excellent tutorial describing semantic technologies in a very understandable way

Quote

#15 2013-03-29 20:47

Big kudos!!!

I have been trying to piece together rdf, ontologies, semantic web, modeling for about 5 days, and it was tough. Reading W3C material is like being transplanted to planet abstraction.

Now I 'get it', and am well on my way to creating linked data models for SAP!

Thanks! john dot a dot conte at gmail dot com

Quote

#14 2013-01-14 17:57

thank you!!!

Quote

#13 2012-12-14 05:25

Even the more elaborate text books on Semantic Web and SPARQL was insufficient to clear my doubts on the topic. For a beginner like me this website seems to be giving crystal clear information. Thanks to the tutorial. Looking forward for more tutorials on SPARQL querying and RDF Schema architecture design. I cant think of going to any other place looking for tutorials.

Quote

#12 2012-10-29 09:05

hi sir i am not getting how to link uri for our own ontology can u please give some hint how and where i can access or else i should only create a new website or can i use http://www.w3.org for creating our own ontology

Quote

1 2 3

Refresh comments list

RSS feed for comments to this post.

Add comment

Name (required)

1000 symbols left

Refresh

Send

JComments

Community

Register to download software from our site and interact with other users as you learn semantic web.

A+ A- Reset