Tip: Organize content with Atom categories

Find out what categories are available and then add category elements to your entries

Nicholas Chase  
July 17, 2007

Atom lets you provide an easy way to manage lots of data, but there will come a point at which you need to add some sort of categorization to make that data manageable. This tip shows you how to use categories with Atom feeds and the Atom API. Although you will naturally add category information to information syndicated using Atom, this tip focuses on these categories from the perspective of the Atom Publishing Protocol.

Allowable categories

As you consider adding categories to your Atom entries, first consider what your choices are. The place to find that out is in the Atom service document, which defines the categories that exist for a particular collection. For example, consider the default service document for the Blogapps server (see Listing 1).

Listing 1. The service document

```xml
<?xml version="1.0" encoding="UTF-8"?>
  <app:workspace>
    <atom:title>AdminBlog</atom:title>
    <app:collection
      href="http://localhost:8080/roller/app/adminblog/entries">
      <atom:title>Weblog Entries</atom:title>
      <app:categories app:fixed="yes"
        app:scheme="http://localhost:8080/roller/adminblog/">
        <atom:category atom:term="/Java" atom:label="Java" />
      </app:categories>
    </app:collection>
  </app:workspace>
  <app:collection
    href="http://localhost:8080/roller/app/adminblog/resources">
    <atom:title>Media Files</atom:title>
  </app:collection>
</app:service>
```
In this case, both the adminblog and main weblog collections have sets of categories, but there is more to this data than might meet the eye at first glance.

First of all, notice that the category elements themselves have two important attributes. The first is the term, which is the actual category representation. Notice that in this case, each of the terms start with a slash (/). The Atom Syndication Format specification is silent on the matter of what the form that a term can take, so nothing stops you from creating a hierarchical structure of categories.

The second important attribute is the label, which consists of the human-readable version of the category.

Now, both sets of categories seem to have the same entries, but they don't. If you look at the categories elements, you'll notice that they carry a scheme attribute. The category scheme acts as a way to categorize your categories, so to speak. For example, the adminblog General category might mean something completely different from the main General category.

You can even mix schemes together in a single collection by adding a scheme element to the individual category element.

**Using categories**

Adding categories to your content is simply a matter of adding the category element to your entry. For example, you might add a new entry to the server (see Listing 2).

**Listing 2. Adding a category**

```
POST /roller/app/main/entries HTTP/1.1
Host: localhost
Authorization: Basic YWRtaW46YWRtaW4=
Content-Type: application/atom+xml
```
Here you use the established Java category, so the response shows that the entry was added to it (see Listing 3).

**Listing 3. The response**

HTTP/1.1 201 Created
Server: Apache-Coyote/1.1
Set-Cookie: JSESSIONID=7818CFD09D1A9A5F5EA389A283AA8BD0C; Path=/roller
Location: http://localhost:8080/roller/app/main/entry/40288182103d1af401103d61969a0068
Content-Type: application/atom+xml;charset=utf-8
Transfer-Encoding: chunked
Date: Sat, 20 Jan 2007 02:41:51 GMT

3fb
<entry xmlns="http://www.w3.org/2005/Atom">
  <title>Blogapps supports Atom (Java)</title>
  <link rel="alternate" href="http://localhost:8080/roller/main/entry/blogapps_supports_atom_java1" />
  <link rel="edit" href="http://localhost:8080/roller/app/main/entry/40288182103d1af401103d61969a0068" />
  <category term="/Java"/>
  <author>
    <name>admin</name>
  </author>
</entry>

However, if you try to add a new category, the response will depend on how the server is configured. For example, you can try to add an entry with a nonexistent category (see Listing 4).

**Listing 4. Using a nonexistent category**

...
If you try to send this request to the server as it stands, the result might not be what you expect (see Listing 5).

**Listing 5. The results**

```xml
...<link rel="edit" href="http://localhost:8080/roller/app/main/entry/40288182103d1af401103d62f0ea0069" />
  <category term="/General" />
  <author>
    <name>admin</name>
  </author>
...```

Notice that the system used one of the existing categories rather than adding the new one. If you go back in the service documenting **Listing 1**, you will see that the categories elements specify that the categories are *fixed*. That means that the categories shown are the only categories allowed.

**Category documents**

Categories can also be listed in separate category documents. In that case, the service document includes a pointer to that document. For example (see Listing 6).

**Listing 6. Pointing to a category document**

```xml
  <app:workspace>
    <atom:title>AdminBlog</atom:title>
    <app:collection href="http://localhost:8080/roller/app/adminblog/entries">
      <atom:title>Weblog Entries</atom:title>
      <app:categories href="http://localhost:8080/roller/adminCategories" />
      <app:accept>entry</app:accept>
    </app:collection>
    <app:collection href="http://localhost:8080/roller/app/adminblog/resources">
      <atom:title>Media Files</atom:title>
      <app:accept>image/*</app:accept>
    </app:collection>
    <app:workspace>
      ...
    </app:workspace>
  </app:workspace>
</app:service>
```

In this case, the fixed and scheme attributes might not be present, but you can, of course, put them in the category document itself.

**Summary**

Using categories in Atom is a matter of knowing where to look for them (the service document), and what’s permissible. Once you know what categories are available, you can simply add a category element to your entries.