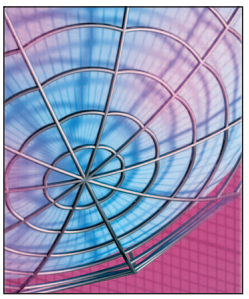

Evolving the Link

Danny Ayers • *Independent Consultant*

In this installment of Websense, I'm turning the column over to the Web's most basic — though arguably the most powerful — feature: the hyperlink.

When the Web appeared, many in the hypertext community criticized it because of its simplifications. For a start, links that break easily went against conventional wisdom. Broken links were, in fact, designed in as a Web feature (think HTTP 404 error). Somewhat counterintuitively, this has made the system as a whole significantly more robust. A Web that relied on everyone ensuring that something was always at the end of their links would be seriously brittle and hardly likely to survive long, given human nature.

Critics also pointed to the limitations of links that pointed in only one direction and were untyped. The Web's success has to a large extent overridden these criticisms without really proving them wrong. Ironically, it now seems that many of the early criticisms weren't exactly incorrect per se, but merely shortsighted.

The Hypertext Link

A typical link in the body of an HTML file looks something like this:

```
<a href="http://creativecommons.org/
licenses/by/2.0/">cc by 2.0</a>
```

When displayed in a typical browser, it will appear as the text "cc by 2.0" colored blue, and underlined. Clicking the mouse button while the pointer is over the text will cause the browser to replace the current display with a rendering of the page with the URL in the markup's `href` attribute. Hyperlinks' blue, underlined appearance happens to be a convention most browsers follow. Anyone who has written any HTML in recent years, however, will be familiar with CSS stylesheets, which let page authors customize the document display. One way to associate the stylesheet is via linking (in the `<head>` of the document); in this case, we might have

```
<link rel="stylesheet" type="text/css"
href="mystyle"/>
```

Like the `a` element, this contains a hypertext reference to a separate resource in the `href` attribute. Additionally, it suggests the media type of an available representation of that resource and, through the `rel` attribute, describes the relationship between the linked resource and this page.

Unlike the link in the body, no user interaction is expected before the browser obtains the remote CSS file. The conventional view of CSS with HTML is that it's a separation of document presentation from its content. This is true enough, but there's more to this than meets the eye. Although the HTML specification does talk a little about browser behavior, with links being *activated* and resources being *visited*, it also describes how the `rel` attribute (and its inverse, `rev`) can help express relationships between documents. As the spec puts it, "Even if they are not used for navigation, these links may be interpreted in interesting ways" (www.w3.org/TR/html401/struct/links.html#h-12.1.2).

An example of an "interesting way" to interpret a link is to associate a document with a license that applies to it. For the simple link just shown, a document's text might spell out licensing terms for human consumption, with a link provided to the legal document to benefit readers wishing to follow their noses to further information. However, although the prose might be clear about this, text without some kind of convention is difficult for a piece of software to interpret. Fortunately, you can make the relationship more explicit in the markup, like this:

continued on p. 94

 1089-7801/07/$25.00

continued from p. 96

```
<a href="http://creativecommons.
org/licenses/by/2.0/" rel=
"license">cc by 2.0</a>
```

Now a suitably equipped browser can look for that `rel` value and highlight or otherwise inform the reader of that particular relationship. A search engine could record this information and let users filter search results according to the license.

Although the HTML specification defines a handful of possible `rel` values, "license" isn't one of them. From one viewpoint, using a short string like this might lead to naming clashes, in which someone else uses the same string to mean something completely different. Less intuitive is the view that if someone asserts a specific interpretation of a particular string (for example, by supporting it in their software), they're denying anyone else the ability to use the string for *their* own purposes. Fortunately, the Web has a tried-and-tested system for disambiguation: the use of uniform resource identifiers (URIs) as global identifiers. It's possible to assert an interpretation for `rel` values (and those of various other extension points) by identifying what the HTML specification calls a metadata profile. The specification leaves open the question of what a profile might look like, which is fine because all that's really needed is the URI disambiguation. It's used like this:

```
<head profile="http://micro
formats.org/wiki/rel-license">
...
<a href="http://creativecommons.
org/licenses/by/2.0/"
rel="license">cc by 2.0</a>
```

Now the `rel` value, which gives the meaning of the link relationship, is associated with what microformats.org has defined for this term.

The Role of URIs

The notion of a URI is probably the most important part of the Web's architecture. When we type an `http://` address into a Web browser, we're using a URI to locate a Web page, but the URI itself names something; it doesn't locate it. One reason this distinction is important is that URIs can identify not just a Web page but anything. A *resource* is simply something that can be identified with a URI.

Thus, an object, person, place, or even an abstract concept can have a URI. Although putting someone's URI into a browser won't make them materialize, you could reasonably expect to get some representation or description of that person. URIs' primary use on the Web is with the HTTP protocol, to identify things that have a representation on the network — most commonly, things that we can view in a browser, such as an HTML page.

It's possible to assert an interpretation for `rel` values by identifying a metadata profile.

There's an obvious distinction here, and the W3C's Technical Architecture Group (TAG) came up with the term *information resource* (see www.w3.org/TR/webarch/) to describe things that can be expressed entirely as digital messages. So, documents about people, ideas, or photographs of places can be information resources, even though the people, ideas, and places themselves can't.

Tim Berners-Lee, in one of his *Design Issues* essays (www.w3.org/DesignIssues/LinkedData.html) listed four rules for maximizing links' benefits:

- Use URIs as names for things.
- Use HTTP URIs so that people can look up those names.
- When someone looks up a URI, provide useful information.
- Include links to other URIs, so that they can discover more things.

With the idea of an information resource in place, it's possible to apply these rules to URIs that identify things rather than documents. The TAG decided the appropriate action was for the server to redirect (an HTTP 303) to another resource that was a document.

Links Are Data

Working with a SQL database typically involves content-oriented aspects that deal with specific values in the table fields and structure-oriented aspects that deal with how the various fields interrelate. There's an analogy here with the Web, in which content-oriented aspects correspond to dealing with resource representations (Web pages), and structure-oriented aspects deal with the interrelation between resources (links). Although the correspondence isn't one-to-one, there's also some analogy when it comes to SQL commands (`SELECT`, `INSERT`, `UPDATE`, `DELETE`) and HTTP methods (`GET`, `POST`, `PUT`, `DELETE`).

To express things a little more formally, in a relational database, the data is expressed as sets of relations (tables), which in turn are sets of tuples (rows). The tuples in the relation are those for which some predicate holds true. So, a table containing personal contact data might have columns for name, email, and address. The rows are all those known valid (true) addresses. The predicate is what the table is expressing — in this case, it would probably be called "contacts."

You can express a link in exactly the same way. The source page containing an HTML link is one field; the link's target is another. The predicate is the link relation, which you might call

`href`. We could thus express the structure of the Web as one enormous two-column `href` table containing every source–target pair of every single link.

Abstracting the Link

We can express the information from the HTML example earlier, relating a document to its license, in RDF (here in N-Triples syntax) as

```
<> <http://microformats.org/
wiki/rel-license#license>
<http://creativecommons.org/
licenses/by/2.0/>.

<http://creativecommons.org/
licenses/by/2.0/>
<http://www.w3.org/2000/01/rdf
-schema#label> "cc by 2.0".
```

The first part is the association between the current document (<>) and the license resource. This relationship is expressed using the predicate `license` in the microformats.org namespace. The second statement simply takes care of labeling the linked resource, implicit in the HTML version.

One big difference between the RDF and HTML versions is that no built-in expectation exists for any particular kind of rendering or browser behavior. N-Triples syntax RDF is just one way to express the data in the abstract RDF model. This is a separation of concerns, not unlike that of content and style in HTML, but here it's more like *model* (RDF) and *view* (HTML or N-Triples).

Although in many cases a hyperlink in an HTML document might be optimal for expressing a relationship, if the RDF-modeled link can be abstracted from its format, there's no need to express that information in a specific document. In this case, we can pull out the URI of the document to which the license applies, like so:

```
<http://example.org/my_document>
<http://microformats.org/wiki/
rel-license#license>
<http://creativecommons.org/
licenses/by/2.0/>.
```

This is a typed link expressed as data in a representation-independent form. As abstract data, we can interpret it in either direction; there's an implicit inverse relationship: "document has license" or "license applies to document." It could appear as it does here in N-Triples, in RDF/XML, in HTML on the Web, in a SQL database, or associated with an object described in a programming language.

Open Data

Standing back from the technical details, here I present one route through which to use linking on the Web to further advantage. The *open data* movement aims to make data freely available to everyone, without limiting restrictions from copyright, patents, or other mechanisms of control. Like its cousin open source, no single organization is behind the movement — rather, it's more a philosophy shared by disparate individuals and groups. The one big thing these people have in common is the Internet as a communication medium. Although the Web allows the assertion of copyright and supports access control, its default mode tends to be wide open, making it a natural target for open data. Large data sets are already freely available on the Web, such as Wikipedia, Geonames, MusicBrainz, WordNet, the DBLP bibliography, and many more from the sciences.

The W3C's Semantic Web Education and Outreach group recently agreed to support a community project called Linking Open Data on the Semantic Web. The project's goal is to make various open data sources available on the Web as RDF and to set RDF links between data items from different data sources. Groups in the field made progress before the community project even began. Examples include the dbpedia.org project and the D2R Server publishing the DBLP bibliography.

To maximize utility, the linking between such data sets has to occur at the level of individual resources in the domain of interest. A critical part of this lies in determining when a resource in one data set has an identical occurrence in another. This obviously enables a particularly strong form of linking. On the one hand, this is a great challenge — the problems of vocabulary or ontology alignment have spawned many papers. But in many cases, all they need are simple algorithms. In the community project, initial efforts have attempted to interlink the data sources I just mentioned. The dbpedia.org RDF descriptions of cities include `owl:sameAs` links to the Geonames data about the city. Another example is the RDF Book Mashup, which links book authors to paper authors within the DBLP bibliography.

The Web's utility does depend on its level of deployment — the network effect — and it's doing rather well there. But it would be disappointing if, after all this time, the Web was only just catching up technically with its predecessors. Virtually all the hypertext features said to be lacking from the Web have been formalized within various specifications. Conceptually, the key is viewing the link as a unit of data. If this view is overlaid on the current Web, then not only are the shortfalls the critics describe illusory, but there's still a huge amount of untapped potential in the Web even in its current "simplistic" architecture. We're entering interesting times.

Danny Ayers is an independent developer, consultant, and author. His research interests are primarily around Semantic Web technologies. Ayers has coauthored 10 books on programming, generally covering Web-related topics. He is chair of the developers track for WWW 2007. His weblog is at www.dannyayers.com. Contact him at danny.ayers.ieee@gmail.com.