CHAPTER 18

Microformats and RDFa as Embeddable Data Formats

The central problem that we will study in this chapter is: how to send information in web pages that is easy to understand by both humans and computer programs. One solution that we will consider in depth is microformats. Microformats are little chunks of structured data that are seamlessly embeded in web pages. Hence, microformats can be easily parsed by computer programs so tha the data can be reused in other contexts – giving rise to plenty of mashup possibilities.

Let's look at a specific example to see how microformats can be used in a mashup. If you go to http://upcoming.yahoo.com to a specific event (e.g., Mashup Camp IV -- `http http://upcoming.yahoo.com/event/144855`), the interface of upcoming.yahoo.com gives you options to send the data for the event to a number of calendars. You can also download the event info in the iCalendar format. You can use the upcoming API (as explained in Chapter 15) to extract the event information from upcoming. If that weren't enough ways to get event data, the event information is embedded in the HTML as a microformat, specifically as an hCalendar microformat. You look at the source at the source and find a div element that begins with

```
<div id="calendarContainer" class="vcalendar"> <!-- Begin vCalendar -->
```

and ends lines later with

```
</div> <!-- End vCalendar -->
```

with pieces of HTML in between that is flagged to indicate the presence of attributes of event data. For instance

```
<abbr class="dtstart" title="20070718T130000">Wednesday, July 18, 2007
</abbr>
```

and

```
<abbr class="dtend" title="20070719T140000">
```

In the Javascript shell of Firefox, you can pull the node element with

```
divs = document.evaluate("//*[@class='vcalendar']",
document,null,XPathResult.UNORDERED_NODE_SNAPSHOT_TYPE,null)
```

As I will discuss in the hCalendar section, the presence of these specific class attributes indicates the presence of the hCalendar microformat. This pattern makes it easier to parse out details of the event.

With so many other ways to already get the event data, what does having the microformat contribute to the mix? At least two:

* For user-centered contexts, programs that have the HTML source already – such as a web brower, such as Firefox, and its associated machinery (browser extensions, Greasemonkey scripts), can reliably extract the event data and then present options to the user about what to do with that data. (The Operator Firefox add-on instantiates this idea.)
* Spiders and other screen-scrapers can also extract the information reliably – without the usual heuristical guessing -- to build aggregated databases of events across web pages. That's what Technorati, etc is doing.

In both cases, there is no need to make an extra call to the API to get this desired information about the event.

In addition to microformats, there are other ways to embed data in human-readable contexts. Specifically, I will refer to RDF and associated technology in a short discussion of RDFa .

In this chapter, we will:

* study specific examples of microformats
* look at how to use the Firefox add-on Operator to jumpstart your study of microformats
* look at programmatic approaches to consuming and creating microformats
* compare microformats to leading alternatives such as RDFa.

Definitions of Microformats

The curr[illegible] "offic[illegible]" definition of m[illegible]formats from http://m[illegible]formats.[illegible] is the followin[illegible]

[illegible]

Many oth[illegible] definit[illegible]sof m[illegible]crofor[illegible]ats h[illegible]ve been proposed, some which I find more illumina[illegible]g. The [illegible]efiniti[illegible]n th[illegible] highl[illegible]ghted at http://m[illegible]Definition is (note the verse-like layout :

[illegible]

Out of this definition, I would emphasize:

* human-readable (X)HTML/XML documents, Atom/RSS feeds, and "plain" XML
* using brief, descriptive class names

For more definitions of microformats, see

`http://microformats.org/wiki/what-are-microformats`

It's important to know about the intentional limitations built into microformats before criticizing them. From the list of things that microformats are not at http://microformats.org/about/, I will highlight a few....microformats are not:

* "infinitely extensible and open-ended"
* "a panacea for all taxonomies, ontologies, and other such abstractions"

Keep those limitations in mind when we compare microformats to specs like RDF, which are aimed to be highly generalizable.

Because microformats involve both design philoosophies and concrete formats (which you can use whether or not you subscribe to the microformats design philosophy), it's easy to just ignore the philosophy. So if just want to use the work of the microformats community, you can ask whether this philosophy is relevant. Probably – since there are a lot of design decisions that are hard to understand without knowing that philosophy

Using Operator to Learn about Microformats

Installing the Operator add-on in Firefox and seeing it in action is a good way to learn about microformats. You can download it from

`https://addons.mozilla.org/en-US/firefox/addon/4106`

Currently, the latest version is 0.7. There is 0.8a version, which I will use and describe here. See the blog entry announcing 0.8a (http://www.kaply.com/weblog/2007/06/04/operator-08a-is-available/) and download it at:

`http://www.kaply.com/operator/operator.xpi`

The formats that are recognized by default by Operator 0.8a gives one at least some hint at microformats that are popular or important, at least in the judgement of the developers of Operator:

* adr

* hCard
* hCalendar
* tag
* geo
* xFolk
* RDFa

Microformats Design Patterns

Microformats are meant to be embeddable in (X)HTML – so anywhere you can put (X)HTML, you should be able to stick in microformats – including RSS and Atom feeds.

The best way to start with microformats is to look at some specific examples while keeping an eye out for some general design patterns that are behind microformats:

`http://microformats.org/wiki/design-patterns`

rel-design-pattern

rel-design-pattern uses values in a rel attribute to indicate the meaning of a link.[1]

You can stick multiple values in an attribute by separating the individual values by whitespace. This is how you associate HTML element with multiple classes. Is this standard usage for other attributes such as rel and rev – yes:

http://www.w3.org/TR/html401/struct/links.html#adef-rel

and

http://www.w3.org/TR/html401/struct/links.html#adef-rev

The [illegible] is also [illegible]sed in thi[illegible] design [illegible]n. I[illegible] what "[illegible]" is supposed to [illegible] you [illegible] – [illegible] didn't [illegible] orm[illegible]ts), read the [illegible] ev [illegible]

[1] http://microformats.org/wiki/rel-design-pattern.

class-design-pattern

`http://microformats.org/wiki/class-design-pattern`

This involves selecting the most appropriate (X)HTML element and the class attribute to hold a whitespace-separated list of "semantic class names" (http://microformats.org/wiki/semantic-class-names)

When you come to selecting class names, you should note the ones that already in use to avoid reinventing the wheel and causing namespace collisions:

`http://microformats.org/wiki/existing-classes`

abbr-design-pattern

`http://microformats.org/wiki/abbr-design-pattern`

This pattern makes use of the abbr HTML tag to wrap text with a machine-readable version of that information. This pattern is most commonly used to encode datatime (in the datetime-design-pattern (`http://microformats.org/wiki/datetime-design-pattern`)

```
<abbr class="dtstart" title="20080310T1700-08">
 March 10, 2008 at 5 PM
</abbr>
```

There are concerns about the accessiblity of such constructions (http://microformats.org/wiki/datetime-design-pattern#Accessibility_issues). A recomended alternative is:

```
<span class="dtstart" title="20080310T1700-08">
 March 10, 2008 at 5 PM, Pacific Standard Time
</span>
```

include-pattern

`http://microformats.org/wiki/include-pattern`

This pattern is for including data from one microformat in the page into another microformat. It's used in the proposed hResume, hReview, hAtom microformats.

Examples of Microformats

In this section, I will show examples of a number of microformats. You can find a list of microformats at

`http://microformats.org/wiki/Main_Page`

at which you find a list of microformats for which there are specifications or drafts. I have gathered the examples in this section into one page:

`http://examples.mashupguide.net/ch18/sample_microformats.html`

rel-license

The official page for rel-licence is:

http://microformats.org/wiki/rel-license

rel-license is for specifying a license to be associated with the embedding page. For instance, when you can go to the Creative Commons site (http://creativecommons.org/license) to select a license, you will be given some HTML that uses the rel-license microformat to indicate a license. For instance, if you select the defaults, you will get something like:[2]

```
<a rel="license" href="http://creativecommons.org/licenses/by/3.0/">
<img alt="Creative Commons License" style="border-width:0"
src="http://i.creativecommons.org/l/by/3.0/88x31.png" />
</a>
<br />This work is licensed under a
<a rel="license" href="http://creativecommons.org/licenses/by/3.0/">Creative Commons
Attribution 3.0 License</a>.
```

Note the use of the rel-design-pattern.

rel-tag

The rel-tag micro, described at

http://microformats.org/wiki/rel-tag

is used to relate a tag (as in folksonomic tag – See Chapter 4) to a URL. You use it by adding using the rel-design-pattern, specifically, adding tag to the list of values in rel attribute. The resultant semantics is the last segment of the URL in the href attribute is considered the value of the tag, and the URL value of href points to a collection of items having the same tag. Consider the following example generated by Wordpress for the Google Maps category for http://blog.mashupguide.net:

```
<a rel="tag" title="View all posts in Google Maps"
href="http://blog.mashupguide.net/category/google-maps/">Google Maps</a>
```

In this case, google-map is the tag and http://blog.mashupguide.net/category/google-maps/ points to the blog entries that have been tagged with google-map.

xfn

http://www.gmpg.org/xfn/

xfn, which also uses the rel-design-pattern, is used to denote personal relationships between the author of a web page and the people associated to the linked page. The easiest way is to get started is to fill out the XFN creator,

http://gmpg.org/xfn/creator

Let me present two examples. The first is for my wife's weblog:

```
<a href="http://128.32.250.15:8080/yinzgandantananat/" rel="friend met colleague co-
resident spouse muse sweetheart">Laura Shefler</a>
```

[2] http://creativecommons.org/license/results-one?q_1=2&q_1=1&field_commercial=yes&field_derivatives=yes&field_jurisdiction=&field_format=&lang=en&language=en&n_questions=3

The second is for Tim Berners-Lee's web page. In this case, he is a "contact" since I don't know him personally:

```
<a href="http://www.w3.org/People/Berners-Lee/" rel="contact">Tim Berners-Lee</a>
```

When I learned about xfn, I wondered about its relationship to FOAF. My question was answered by:

```
http://gmpg.org/xfn/and/foaf
```

xfolk

xfolk, detailed at

```
http://microformats.org/wiki/xfolk
```

is used to publish a bookmark. xfolk uses the `class-design-pattern.` Consider the following example and note the use of the `xfolkentry`, `taggedlink`, and `description` class names:

```
<div class='xfolkentry'>
  <a class="taggedlink" href="http://www.w3.org/People/Berners-Lee/">Tim Berners-
Lee</a>: <span class='description'>The inventer of the WWW</span>
</div>
```

geo

```
http://microformats.org/wiki/geo
```

`geo` is used to denote the latitude and longitude of the resource tied to the embedding web page. Using the `http://microformats.org/wiki/geo-cheatsheet`, you can figure out how to make use of the class-design-pattern and use the `geo`, `latitude`, and `longitude` class names to write an example such as:

```
<div class="geo">Tim Berner-Lee's location is:
 <span class="latitude">42.3633690</span>,
 <span class="longitude">-71.091796</span>.
</div>
```

hCard and adr

```
http://microformats.org/wiki/hcard
```

`hcard` is used to represent "people, companies, organizations, and places, using a 1:1 representation of vCard". An easy way to get started with hCard is to use the hCard Creator at

```
http://microformats.org/code/hcard/creator
```

Let's create a `hCard` for Tim Berners-Lee, the inventor of the WWW, drawing on his web page at

```
http://www.w3.org/People/Berners-Lee/
```

to come up with

```
<div id="hcard-Tim-Berners-Lee" class="vcard">
 <a class="url fn" href="http://www.w3.org/People/Berners-Lee/">Tim Berners-Lee</a>
 <div class="org">World Wide Web Consortium</div>
 <a class="email" href="mailto:timbl@w3.org">timbl@w3.org</a>
 <div class="adr">
  <div class="street-address">77 Massachusetts Ave. (MIT Room 32-G524)</div>
  <span class="locality">Cambridge</span>
,
  <span class="region">MA</span>
,
  <span class="postal-code">02139</span>

  <span class="country-name">USA</span>

 </div>
 <div class="tel">+1 (617) 253 5702</div>
<p style="font-size:smaller;">This <a
href="http://microformats.org/wiki/hcard">hCard</a> created with the <a
href="http://microformats.org/code/hcard/creator">hCard creator</a>.</p>
</div>
```

You'll notice that inside of the `hCard` microformat is the `adr` microformat:

`http://microformats.org/wiki/adr`

`adr` is a mapping of `vCard`: "This specification introduces the adr microformat, which is a 1:1 representation of the aforementioned adr property from the vCard standard, by simply reusing the adr property and sub-properties as-is from the hCard microformat."

There is support in `adr` for the following properties:

* post-office-box
* extended-address
* street-address
* locality
* region
* postal-code
* country-name

`adr` uses the `class-design-pattern`.

hCalendar

hCalendar is a microformat based iCalendar used to represent calendar information:

`http://microformats.org/wiki/hcalendar`

To quickly create an instance, use the hCalendar creator

`http://microformats.org/code/hcalendar/creator` or consult the hCalendar cheatsheet (http://microformats.org/wiki/hcalendar-cheatsheet). Let's create you can create a hCalendar for the WWW 2008 conference (http://www2008.org/):

```
<div class="vevent" id="hcalendar-WWW-2008-17th-International-World-Wide-Web-
Conference">
  <a class="url" href="http://www2008.org/">
    <abbr class="dtstart" title="20080421">April 21th</abbr> &mdash;
    <abbr class="dtend" title="20080426">25th, 2008</abbr>
    <span class="summary">WWW 2008 (17th International World Wide Web
Conference)</span>&mdash; at
    <span class="location">Beijing International Convention Center, </span>
  </a>
  <div class="description">"The World Wide Web Conference is a global event bringing
together key researchers, innovators, decision-makers, technologists, businesses,
and standards bodies working to shape the Web. Since its inception in 1994, the WWW
conference has become the annual venue for international discussions and debate on
the future evolution of the Web."</div>
  <p style="font-size: smaller;">This
    <a href="http://microformats.org/wiki/hcalendar">hCalendar event</a> brought to
you by the
    <a href="http://microformats.org/code/hcalendar/creator">hCalendar Creator</a>.
  </p>
</div>
```

Other microformats

Some other noteworthy microformats that I will mention briefly:

* xoxo (http://microformats.org/wiki/xoxo) represents outlines
* vote-links (http://microformats.org/wiki/vote-links) is used to indicate whether a link represents a vote-for, vote-abstain, or vote-against the link.
* hReview (http://microformats.org/wiki/hreview) is used to represent reviews of URLs.
* hResume (http://microformats.org/wiki/hresume) is used to represent resumes.

Microformats in Practice

You can learn a lot about microformats by studying how they are actually being used on the Web. Some implementations include:

* the use of adr, hCard, hCalendar, tag, geo by upcoming.yahoo.com and eventful.com
* the use of adr, and hCard at Yahoo! Local.
* the use of hCard and adr on technorati

I suggest using the list of implementations of microformats "in the wild":

`http://microformats.org/wiki/examples-in-the-wild` which includes lists for geo, hCalendar, hCard, hReview, and Include-Pattern. Go to the listed sites and use Operator to pick out the microformats.

Programming with Microformats

For simple micformats, including the ones that depend on the `rel-design-pattern`, it should be simple enough to write your own code to parse data from and write data to the appropriate `rel` and `rev` attributes. It takes a lot more work to handcraft parsers for complicated microformats such as `hCard` and `hCalendar` because there are many possible properties.

There are no schemas for microformats, only specfications written for direct human interpretation, making difficult any auto-generation of high quality language-specific parsers from the specifications.[3]

A challenge in working with microformats is the lack of validators. Norm Walsh argues that W3C Schema and Relax-NG will not work for the purpose of expressing the syntax of microformats as schemas, though Schematron might be up for the task.[4] You can use XMDP, a schema (of sorts) geared to easy human consumption, to get part way to generating validators argues Brian Suda, at least for some simple formats.[5]

Hence, you will need to look for some hand-s) 62 ap ibrari-6(cofor)le syntax D 6 >>BDC -0.0007 Tc 0.0003 for micformats)[7]. Operator makes a great sandbox for experimenting with microformats. Some things to try:

* Download and install user-scripts to add new actions and new microformats (http://www.kaply.com/weblog/operator-user-scripts/)
* Try your hand at writing new actions or support for new microformats by studying exisiting scripts and the documentation.[8]
* Study the code for Operator to pick up on the subtleties that go into working code using microformats.[9]

RDFa: A Promising Complement to Microformats

There's a lot of hype around RDF and the Semantic Web , but the core concept of RDF, or the Resource Description Framework, is simple:

* A RDF document is just a series of statements about resources in the a subject-predicate-object (triplet) form. Another way to put it, they are statements of the form a Resource R has a Property P of Value V -- a triplet (R,P,V) ("Raymond Yee", "has age of ", 40)
* RDF Vocabularies give ways to talk such things as types of resources, terms for properties. For example, a geneological vocabulary would define properlies like "is mother of" , "is sister of"
* Once we have these types of RPVs around, we can to the mix various logical propositions. If V > 30 of a RPV with P="has age of", then (R, "has to trust status", No). In other words, a computer program should be able to deduce that Raymond Yee should not be trusted since he is older than 30 and since one must not trust anyone over 30.

Tim Bray's "What is RDF?" (http://www.xml.com/pub/a/2001/01/24/rdf.html) was the first essay that I read in my attempts to understand RDF. It's still very good. However, I think that the triples idea was still unclear to me after reading the essay. (And I don't blame Tim Bray for that since the idea is clearly in the essay). So I would say to readers that one should follow up Bray's essay with reading something like Aaron Schwartz's "RDF Primer Primer" (http://notabug.com/2002/rdfprimer/). The two complement each other.

You can express RDF triplets in many ways, including the standard RDF/XML syntax (http://[illegible]-syntax[illegible]/). S[illegible]ce [illegible]e h[illegible]ve be[illegible]dis[illegible]uss[illegible] microfor[illegible]ine-unde[illegible] in[illegible]M[illegible] we now [illegible]
(http://r[illegible]

7 http://www.readwriteweb.com/archives/mozilla_does_microformats_firefox3.php

8 http://www.kaply.com/weblog/2007/04/24/operator-action-architecture/ and http://www.kaply.com/weblog/2007/04/18/microformat-objects-in-javascript/

9 http://svn.mozilla.org/labs/operator/

[illegible] thcoming boo[illegible]n Apress).
(http://blo[illegible]hu[illegible]e.net

[illegible] ook with [illegible]
97[illegible]15[illegible]9[illegible] blin Core title) whose

```
<span xmlns:dc="http://purl.org/dc/elements/1.1/" about="isbn:9781590598580"
property="dc:title">Pro Web 2.0 Mashups: Remixing Data and Web Services</span>
```

I think that microformats and RDFa will both a place on the Web. Microformats already have some good uptake and groundedness in today's real world problems. They are focused on very specific usages. RDFa provides a mechanism for making more general assertions about pieces of data.

Reference for Further Study

* the microformat book: `http://microformatique.com/book/`
* Micah Dubinko. "What Are Microformats?"[10]
* Uche Ogbuji's "Microformats in Context"[11]

Summary

Microformats and RDFa can be used to embed data into the human readable contexts of (X)HTML. Microformats are geared to solving specific problems one at a time (such as embedding calendar information). RDFa is meant to express everything that RDF can express.

[10] http://www.xml.com/pub/a/2005/03/23/deviant.html

[11] http://www.xml.com/pub/a/2006/04/26/microformats-grddl-rdfa-nvdl.html