Fun With RSS

Mike Pirnat

Clepy Meeting: 12/05/2005

http://www.pirnat.com/geek/

RSS: Everybody's Doing It

- XML format for “Really Simple Syndication”
- Used for...
  - News
  - Blogs
  - E-Commerce
  - Wiki changes
  - Source control checkins
  - Software releases
  - Podcasts
  - And much more!

A Trivial Example

```
<?xml version="1.0"?>
<rss version="2.0">
  <channel>
    <title>ExileJedi</title>
    <link>http://www.livejournal.com/users/exilejedi/</link>
    <description>ExileJedi - LiveJournal.com</description>
    <lastBuildDate>Sun, 04 Dec 2005 15:34:38 GMT</lastBuildDate>
    <generator>LiveJournal / LiveJournal.com</generator>
    <image>
      <url>http://www.livejournal.com/userpic/35897486/652970</url>
      <title>ExileJedi</title>
      <link>http://www.livejournal.com/users/exilejedi/</link>
      <width>100</width>
      <height>100</height>
    </image>
    <item>
      <guid isPermaLink="true">http://www.livejournal.com/users/exilejedi/133715.html</guid>
      <pubDate>Sun, 04 Dec 2005 15:33:59 GMT</pubDate>
      <title>A Silly Example</title>
      <link>http://www.livejournal.com/users/exilejedi/133715.html</link>
      <description>
My cat did the cutest thing today, blah blah blah, yadda yadda yadda.
</description>
      <comments>http://www.livejournal.com/users/exilejedi/133715.html</comments>
      <category>cats</category>
      <lj:mood>exampleriffic</lj:mood>
    </item>
  </channel>
</rss>
```

Two Basic Exercises

- Parsing RSS feeds
- Generating RSS feeds

Parsing RSS

- Many options for writing your own
  - Fetch data:
    - urllib or urllib2
    - Read file on disk
  - Parse it:
    - Write your own parser (too much work)
    - Use your favorite Python XML parser (too many choices, too much work)
- But you're smarter than that, and too busy!

Feedparser to the Rescue!

- Fetches data for you
  - From a URL or a filename
- Transparently handles:
  - Atom
  - RSS 1.0
  - RSS 2.0
- Robust
  - Uses “loose” parser if strict XML parsing fails
- Delightfully Pythonic
  - Access elements by key or attribute

Using Feedparser

```
>>> import feedparser
>>> d = feedparser.parse("http://exilejedi.livejournal.com/data/rss")
>>> d['feed']['title']              # feed data is a dictionary
u'ExileJedi'
>>> d.feed.title                    # get values attr-style or dict-style
u'ExileJedi'
>>> d.channel.title                 # use RSS or Atom terminology anywhere
u'ExileJedi'
>>> d.feed.link                     # resolves relative links
u'http://www.livejournal.com/users/exilejedi/'
>>> len(d['entries'])               # entries are a list
25
>>> d['entries'][0]['title']        # each entry is a dictionary
u'Live From Westlake! Christmas Ale Tasting'
>>> d.entries[0].title              # attr-style works here too
u'Live From Westlake! Christmas Ale Tasting'
>>> d['items'][0].title             # RSS terminology works here too
u'Live From Westlake! Christmas Ale Tasting'
>>> e = d.entries[0]
>>> e.link                          # easy access to alternate link
u'http://www.livejournal.com/users/exilejedi/136919.html'
>>> e.modified_parsed               # parses all date formats
(2005, 11, 26, 23, 34, 38, 5, 330, 0)
```

Using Feedparser (Continued)

```
>>> e.summary                           # sanitizes dangerous HTML
u'<p>Liz, Karla, and I are embarking on a survey of this year\'s various
Christmas ales, and being the geek that I am, I am attempting to blog the
experience as it happens.  All the bottles are interesting--and there are quite
a number of them!--so for an added challenge I am photoing each bottle as we go
and posting it to my Flickr account.</p>\n\n<p>You can <a
href="http://www.flickr.com/photos/mikepirnat/sets/1451646/">follow along
here</a>!</p>\n\n<p>[Updated@21:01] Tasting is complete!  A few clear-cut
winners emerged from the fray:  Weyerbacher\'s Winter Ale, Delerium Noel, Lump
of Coal, and Holiday Spice Lager Beer.  Photos and tasting notes are over at the
Flickr photoset.  Cheers!</p>'
>>> d.version                           # reports feed type and version
u'rss20'
>>> d.encoding                          # auto-detects character encoding
u'utf-8'
>>> d.headers.get('content-type')  # full access to all HTTP headers
'text/xml; charset=utf-8'
```

Generating RSS

- Again many DIY options
  - Plain old string concatenation
  - Roll your own objects that str down to individual elements
  - Use your favorite Python XML generator (too many choices—it's like picking a web framework!)
- But you're still not getting any younger...

PyRSS2Gen to the Rescue!

- Emits RSS2, the current “recommended” version of RSS
- Escapes things properly
- Pythonic
  - Perhaps not as much as ElementTree, but...
  - Integers, dates, and lists are REAL integers, dates, and lists!

Using PyRSS2Gen

```
import datetime
import PyRSS2Gen

rss = PyRSS2Gen.RSS2(
    title = "ClePy r0x0rs!",
    link = "http://clepy.org",
    description = "The latest news from ClePy, "
                  "the Cleveland Python group",

    lastBuildDate = datetime.datetime.now(),

    items = [
       PyRSS2Gen.RSSItem(
         title = "Mike Rambles About PyRSS2Gen",
         link = "http://clepy.org/news/mike-rss2gen.html",
         description = "At tonight's meeting, Mike went on and on "
                       "about PyRSS2Gen, and you liked it.",
         guid = PyRSS2Gen.Guid("http://clepy.org/news/mike-rss2gen.html"),
    categories = ['presentations', 'Mike Pirnat'],
    pubDate = datetime.datetime(2005, 12, 5, 18, 30)),
    ])

xml = rss.to_xml()  # or do...
rss.write_xml(open("clepy.xml", "w"))
```

PyRSS2Gen Output (Slightly Reformatted)

```
<?xml version="1.0" encoding="iso-8859-1"?>
<rss version="2.0">
  <channel>
    <title>ClePy r0x0rs!</title>
    <link>http://clepy.org</link>
    <description>The latest news from ClePy, the Cleveland Python
group</description>
    <lastBuildDate>Sun, 04 Dec 2005 11:46:59 GMT</lastBuildDate>
    <generator>PyRSS2Gen-1.0.0</generator>
    <docs>http://blogs.law.harvard.edu/tech/rss</docs>
    <item><title>Mike Rambles About PyRSS2Gen</title>
      <link>http://clepy.org/news/mike-rss2gen.html</link>
      <description>At tonight's meeting, Mike went on and on about PyRSS2Gen,
and you liked it.</description>
      <category>presentations</category>
      <category>Mike Pirnat</category>
      <guid isPermaLink="true">http://clepy.org/news/mike-rss2gen.html</guid>
      <pubDate>Mon, 05 Dec 2005 18:30:00 GMT</pubDate>
    </item>
  </channel>
</rss>
```

Scratching My Itch

- I use LiveJournal to blog, but also have a personal site
- I want to republish my recent blog entries on the main page of my personal site
- I sometimes blog about TurboGears and want to be carried by planet.turbogears.org...
- But I don't want to pollute the planet site with vacation photos or stories about cute things my cats do (that would be rude)

Solution: Feedparser and PyRSS2Gen Combined!

- Feedparser downloads and parses my RSS
- Most recent posts written to HTML with a basic FeedRenderer class (DIY, very simple)
- Entry list is filtered based on category
- PyRSS2Gen builds a new feed from the filtered list

Making a Filtered RSS Feed

```
import feedparser
import datetime
import PyRSS2Gen

# get the data
d = feedparser.parse('http://exilejedi.livejournal.com/data/rss/')

# do the filtering & build a list of RSSItem objects
items = [PyRSS2Gen.RSSItem(
    title = x.title,
    link = x.link,
    description = x.summary,
    guid = x.link,
    pubDate = datetime.datetime(
        x.modified_parsed[0],
        x.modified_parsed[1],
        x.modified_parsed[2],
        x.modified_parsed[3],
        x.modified_parsed[4],
        x.modified_parsed[5]))
    for x in d.entries if some_criteria(x)]
```

Filtered Feed (Continued)

```
# make the RSS2 object
rss = PyRSS2Gen.RSS2(
    title = d.feed.title,
    link = d.feed.link,
    description = "ExileJedi's Filtered RSS Feed",
    lastBuildDate = datetime.datetime.now(),
    items = items)

#emit the feed
xml = rss.to_xml() # or perhaps do rss.write_xml(some_file)
```

Filtered Feed (Better)

```
def filter_entries(feed, func):
    # do the filtering & build a list of RSSItem objects
    entries = [PyRSS2Gen.RSSItem(...)
       for x in feed.entries if func(x)]
    return entries

def category_is(category):
    def f(d):
       if 'categories' in d:
          for (junk, cat) in d.categories:
             if cat.lower() == category:
                return True
          return False
    category = category.lower()
    return f

def find_in_title(title):
    def f(d):
       if 'title' in d:
          return title in d.title.lower()
       return False
    title = title.lower()
    return f
```

Filtered Feed (Better, Continued)

```
d = feedparser.parse('http://exilejedi.livejournal.com/data/rss')

e = filter_entries(d, category_is('turbogears'))

rss = PyRSS2Gen.RSS2(
    title = d.feed.title,
    link = d.feed.link,
    description = "ExileJedi Mutters About TurboGears",
    lastBuildDate = datetime.datetime.now(),
    items = e)

#emit the feed
xml = rss.to_xml())
```

Issues

- Feedparser is languishing
  - Author seems to have abandoned programming altogether
  - Patches aren't getting incorporated into CVS, releases
  - Check the SourceForge site for patches
  - Will it get a new maintainer?
  - Or will I just fork it? (May be an opportunity for Clepy to provide value to the community)
- Must subclass things in PyRSS2Gen to support non-standard tags (eg, <lj:mood>)

Links

- Wikipedia: http://en.wikipedia.org/wiki/Rss
- RSS2 spec: http://blogs.law.harvard.edu/tech/rss
- Feedparser: http://feedparser.org/
- PyRSS2Gen: http://www.dalkescientific.com/Python/PyRSS2Gen.html