The Double Life Of Blogs

In the first two posts in this series, I discussed the origins of blogs[1] and how they led to certain elements in popular blog software that were in some cases good and in others bad[2] for my own purposes—to start a blog that consisted of short articles[3] on the intersection of digital technology, the humanities, and related topics (rather than my personal life or links with commentary). What I didn't realize as I set about writing my own blog software from scratch for this project was that in truth a blog leads two lives: one as a website and another as a feed, or syndicated digest of the more complete website. Understanding this double life and the role and details of RSS feeds led to further thoughts about how to design a blog, and how certain choices are encoded into blogging software. Those choices, as I'll explain in this post, determine to a large extent what kind of blog you are writing.

Creating a blog from scratch is a great first project for someone new to web programming and databases. It involves putting things into a database (writing a post), taking them out to insert into a web page, and perhaps coding some secondary scripts such as a search engine or a feed generator. It stretches all of the scripting muscles without pulling them. And in my case, I had already decided (as discussed in the prior post[4]) I didn't need (or want) a lot of bells and whistles, like a system for allowing comments or trackback/ping features. So I began to write my blogging application with the assumption that it would be a very straightforward affair.

Indeed, at first designing the database couldn't have been simpler. The columns (as database fields are called in MySQL) were naturally an ID number, a title, the body of the post (the text you're reading right now), and the time posted (while I disparaged time in my last post I thought it was important enough to have this field for reference). But then I began to consider two things that were critical and that I thought popular blogging software did well: automatically generating an RSS feed (or

feeds) and—for some blog software—creating URLs for each post that nicely contain keywords from the title. This latter feature is very important for visibility in search engines (the topic of my next post in this series).

I know a lot about search engines, but knew very little about RSS feeds, and when I started to think about what my blogging application would need to autogenerate its own feed, I realized the database had to be slightly more elaborate than my original schema. I had of course heard of RSS before, and the idea of syndication. But before I began to think about this blog and write the software that drives it I hadn't really understood its significance or its complexity. It's supposed to be "Really Simple Syndication" as one of the definitions for the acronym RSS asserts, and indeed the XML schemas for various RSS feeds are relatively simple.

But they also contain certain assumptions. For instance, all RSS feeds have a description of each blog post (in the Atom feed, it's called the "summary"), but these descriptions vary from feed to feed and among different RSS feed types. For some it is the first paragraph of the post, for others the first 100 characters, and for still others it's a specially crafted "teaser" (to use the TV news lingo for lines like "Coming up, what every parent needs to know to prevent a dingo from eating their baby"). Along with the title to the post this snippet is generally the only thing many people see—the people who don't visit your blog's website but only scan it in syndication.

So what kind of description did I want, and how to create it? I liked the idea of an autogenerated snippet—press "post" and you're done. On the other hand, choosing a random number of characters out of a hat that would work well for every post seemed silly. I'm used to writing abstracts for publications, so that was another possibility. But then I would have to do even more work after I finished writing a post. And I also needed to factor in that the description had to prod people to look at the whole post on my website, something the "teaser" people understood. So I decided

to compromise and leave part to the code and part to me: I would have the software take the first paragraph of the entire post and use that as the default summary for the feed, but I had the software put a copy of this paragraph in the database so I could edit it if I wanted to. Automated but alterable. And I also decided that I needed to change my normal academic writing style to have more of an enticing end to every first paragraph. The first paragraph would be somewhere between an abstract and a teaser.

Having made this decision and others like it for the other fields in the feed (should I have "channels"? why does there have to be an "author" field in a feed for a solo blog?), I had to choose which feed to create: RSS 1.0, RSS 2.0, Atom? Why there are so many feed types somewhat eludes me; I find it weird to go to some blogs that have little "chicklets" for every feed under the sun. I've now read a lot about the history of RSS but it seems like something that should have become a single international standard early on. Anyway, I wanted the maximum "subscribability" for my blog but didn't want to suffer writing a PHP script to generate every single kind of feed.

So, time for outsourcing. I created a single script to generate an Atom 1.0 feed (I think the best choice if you're just starting out), which is picked up by FeedBurner[5] and recast into all of those slightly incompatible other feed types depending on what the subscriber needs.

This would not be the first time I would get lazy and outsource. In the next part of this series, I discuss the different ways one can provide search for a blog, and why I'm currently letting Google do the heavy lifting.

Part 4: Searching for a Good Search[6]

This entry was posted on Thursday, December 22nd, 2005 at 2:44 pm and is filed under Blogs[7], Programming[8], RSS[9], Software[10]. You can follow any responses to this entry through the RSS 2.0[11] feed. You can leave a response[12], or trackback[13] from your own site.

References

1. ^ origins of blogs (www.dancohen.org)
2. ^ in some cases good and in others bad (www.dancohen.org)
3. ^ consisted of short articles (www.dancohen.org)
4. ^ as discussed in the prior post (www.dancohen.org)
5. ^ FeedBurner (www.feedburner.com)
6. ^ Part 4: Searching for a Good Search (www.dancohen.org)
7. ^ View all posts in Blogs (www.dancohen.org)
8. ^ View all posts in Programming (www.dancohen.org)
9. ^ View all posts in RSS (www.dancohen.org)
10. ^ View all posts in Software (www.dancohen.org)
11. ^ RSS 2.0 (www.dancohen.org)
12. ^ leave a response (www.dancohen.org)
13. ^ trackback (www.dancohen.org)