Some notes on importing bookmarks from del.icio.us to FluidDB using `delicous2fluiddb.py`

Nicholas J. Radcliffe
njr@fluidinfo.com

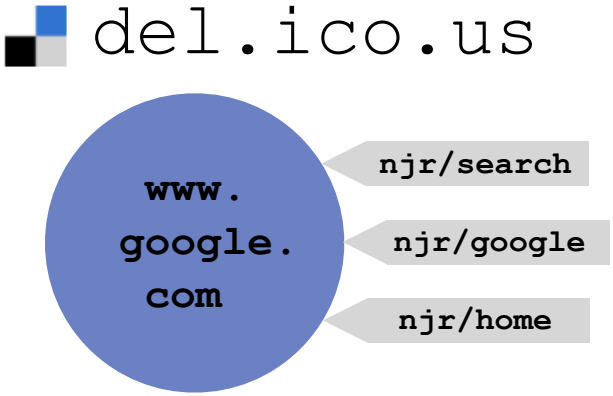
SOURCE: **http://github.com/njr0/fdb.py**
BLOG: **http://abouttag.blogspot.com/**
COMPANY: **http://fluidinfo.com/**

Some examples in this document use
the command line version of fdb.py.
fdb.py is a client library for FluidDB,
available from GitHub at

`http://github.com/njr0/fdb.py`

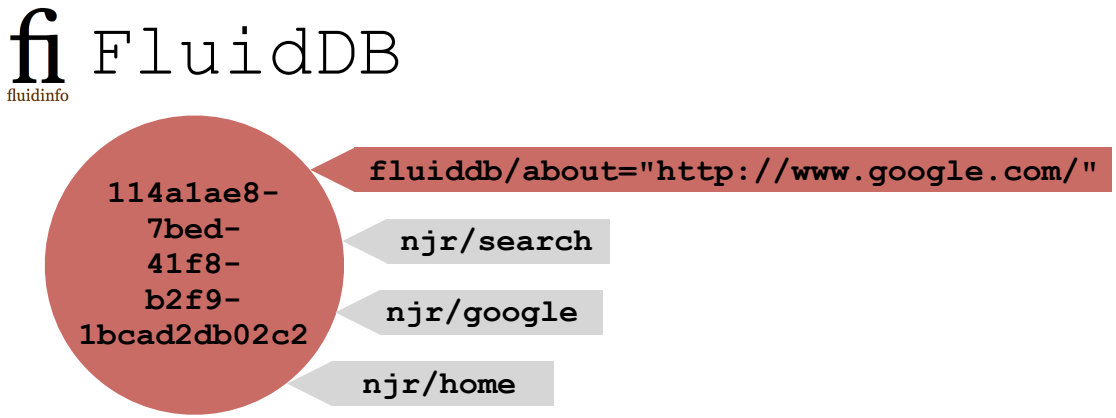
The delicious importer described here
is also distributed with fdb.py

A simple mental model of a bookmark in del.icio.us
looks something like this



This represents the site `www.google.com`, and I have tagged it with my `search`, `google` and `home` tags

Perhaps the simplest way to map this to FluidDB is as follows.
This is what `delicous2fluiddb.py` currently does.

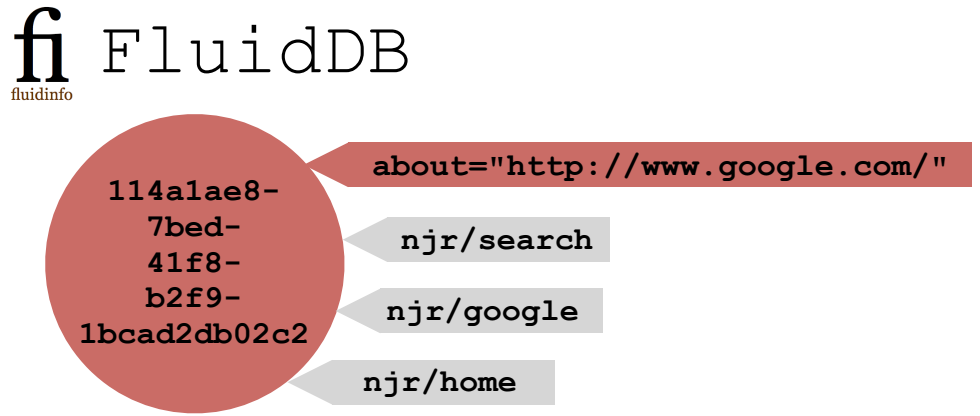


Everything looks very similar except that the address (ID) of the object we store the information on is more prominent and the subject of the bookmark (the URL) is placed on the special `about` tag. This address (id) is real (it exists in FluidDB and is permanent).

fi
fluidinfo

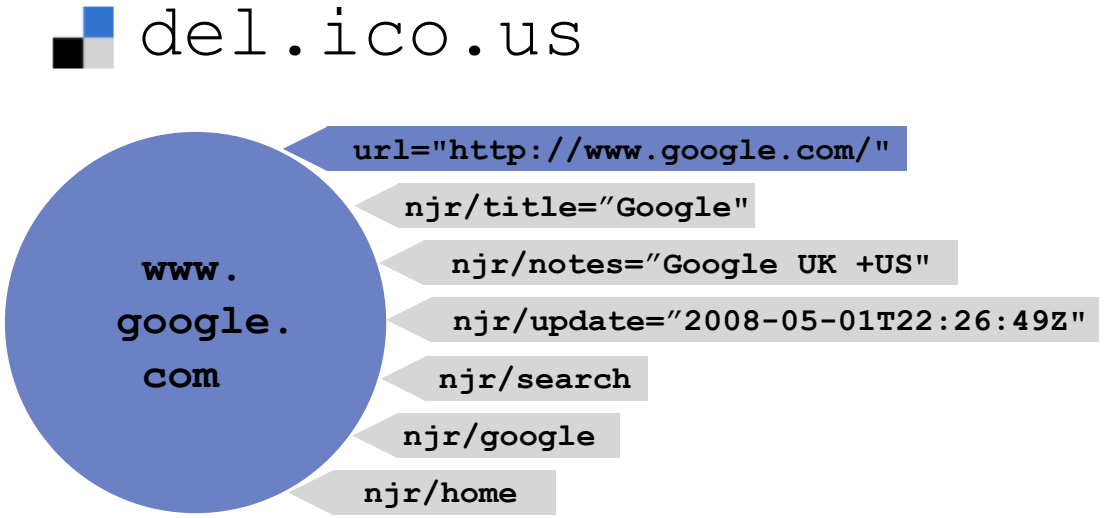
```
$ fdb show -a "http://www.google.com/" /id
Object with about="http://www.google.com/":
  /id = "114a1ae8-7bed-41f8-b2f9-1bcad2db02c2"
```

Unlike the other tags, the `about` tag is not owned by `njr`, but is system-wide. So we often abbreviate it to just `about`.



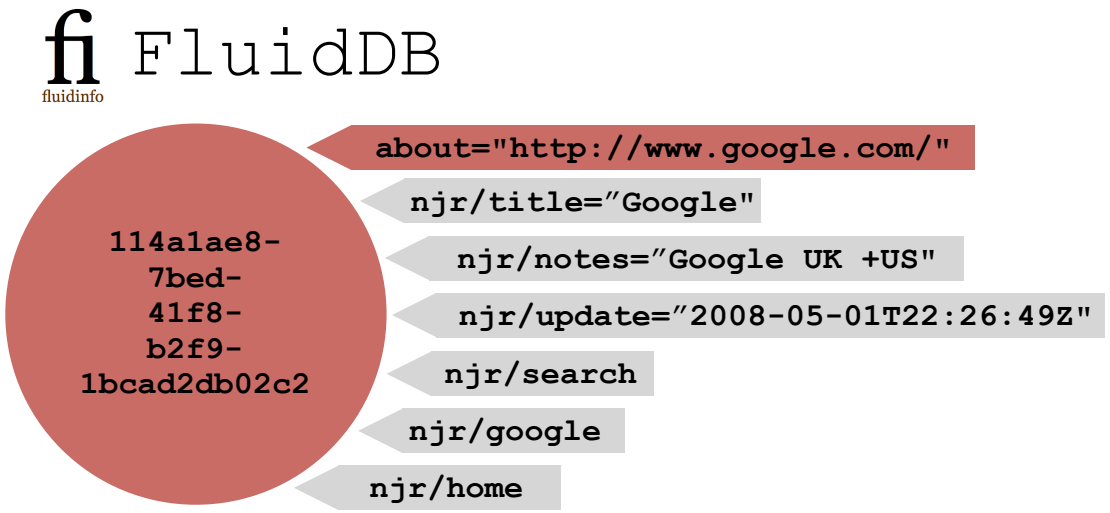
The `about` tag is special in that (a) its value can never be changed and (b) it is unique. This one is permanently associated with the object having the ID `114a1ae8-7bed-41f8-b2f9-1bcad2db02c2`. So by attaching these tags to this object, we can be entirely confident that they will always be associated with the URL `http://www.google.com/`

Del.icio.us actually allows us to store a little more information in a bookmark—a title, some notes and an update date. We can also think of the URL as a tag itself, but as in FluidDB, it's owned by the system, not by `njr`. So a more complete picture of a del.icio.us bookmark might be:



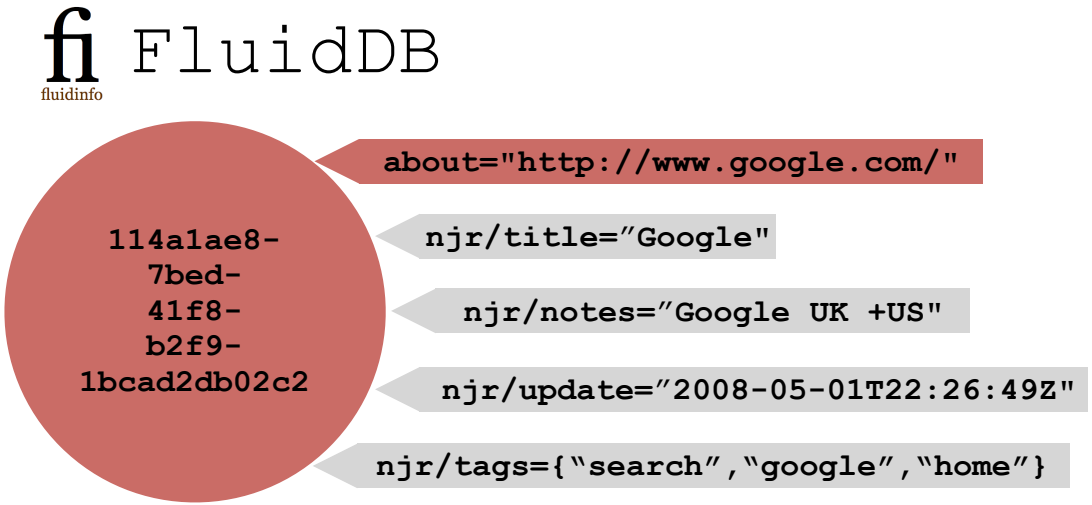
Note that in del.icio.us, the tags are just names, but the other four items of information on the bookmark have values.

Replicating this in FluidDB is trivial, because all tags can take values, which can have various types—strings, numbers, dates and much more.



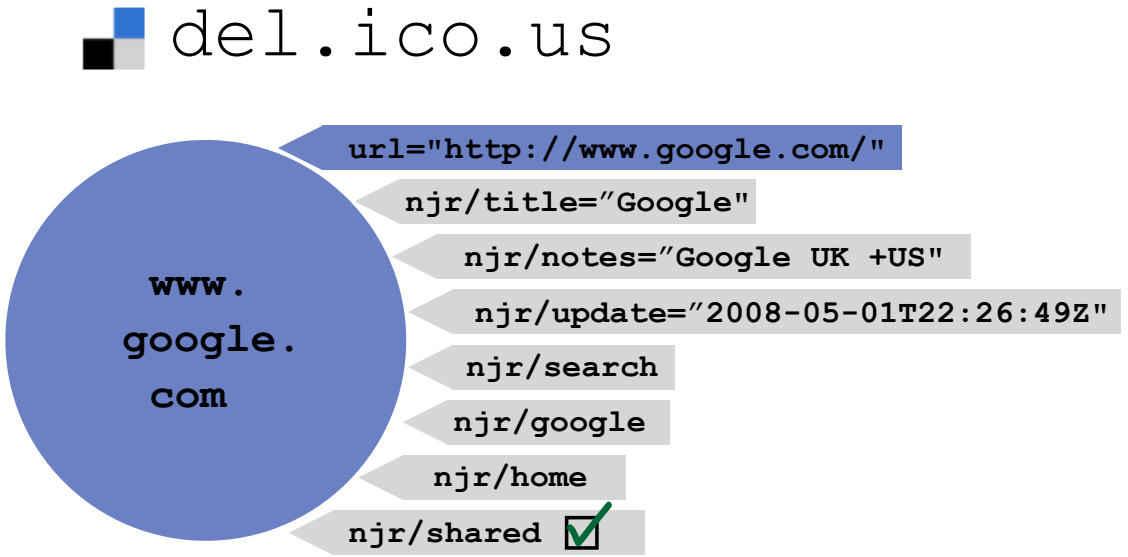
Although `delicous2fluiddb.py` doesn't do this yet, it would be trivial for it replicate the delicious title, notes and update fields, to produce the structure shown above; soon, it will.

There are many other ways we could represent information from del.icio.us in FluidDB. Since the value of a tag in FluidDB can be a set, another obvious way to represent the tags would be as a set of strings on a tag called `tags` (or perhaps `delicious-tags`)



Of course, once in FluidDB, we can also add as many other tags as we like to the object, and these tags may or may not have values.

But before we get onto that, there's one more crucial bit of information that del.icio.us stores about a bookmark—whether it is shared (meaning anyone can see that njr has the bookmark, and what tags I've put in it), or private (meaning that only I can see it).

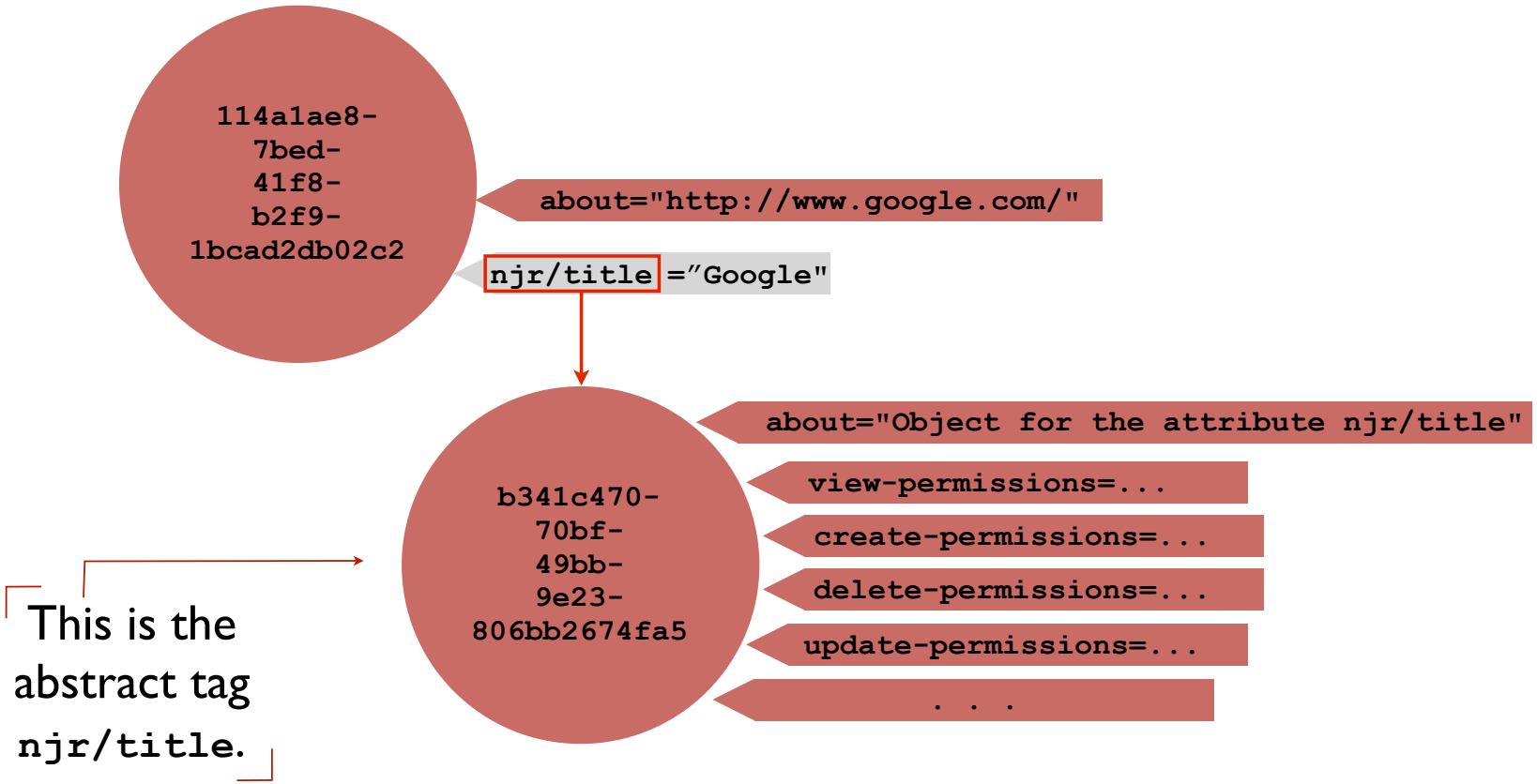


Right now, **`delicous2fluiddb.py`** only uploads the shared bookmarks to FluidDB. But how would it make information shared or private?

As well as allowing the arbitrary tagging of objects, and providing a query language, FluidDB also possesses a rich permissions structure. In fact, there are fine-grained permissions associated with every different tag name in the system (`njr/google`, `njr/title`, `terry/rating` etc.)

UNNECESSARY TECHNICAL DETAIL

Each tag name used in FluidDB actually has a corresponding object in the system; we sometimes call this the "abstract tag", to distinguish it from real tags attached to objects in the system. The permissions system operates on these abstract tags, and permissions set on an abstract tag apply to all the tags used with that name.



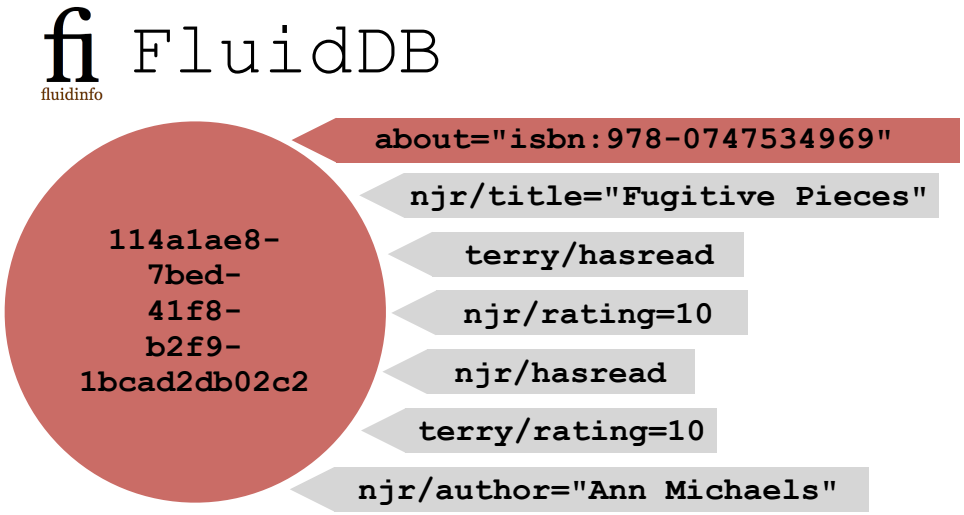
fi
fluidinfo

```
$ fdb show -a "Object for the attribute njr/title" /id
Object with about="Object for the attribute njr/title":
  /id = "b341c470-70bf-49bb-9e23-806bb2674fa5"
```

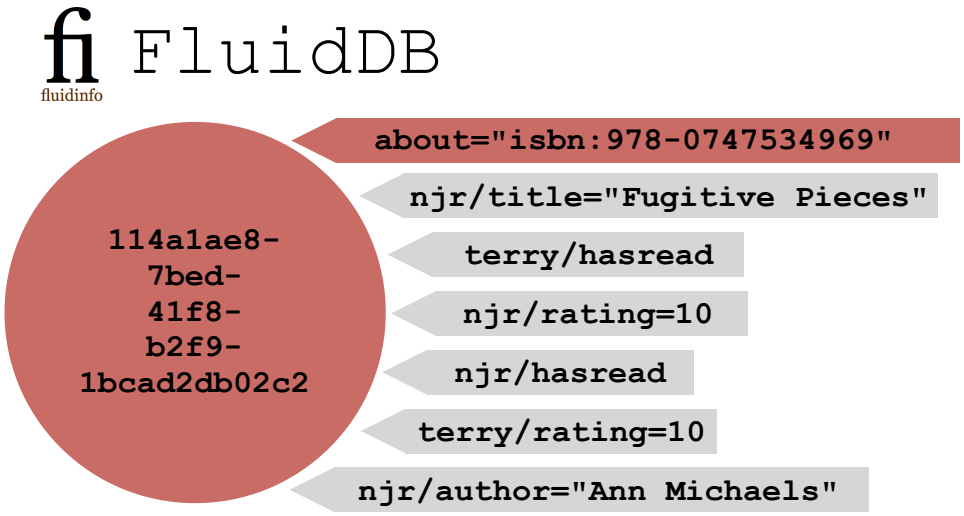
This isn't the place to go into all the detail,
but the key thing is that while all objects in FluidDB are shared, all data in FluidDB is subject to a strong, fine-grained permissions system.

A user can control various kinds of read and write access for each of their tags, with separate permissions for tags having each name (`njr/title`, `njr/google` etc.)

Of course, the whole point of FluidDB, is not to imitate del.icio.us, but to enable more powerful things. Some of the power comes when different people tag the same object, which doesn't have to be a URI.



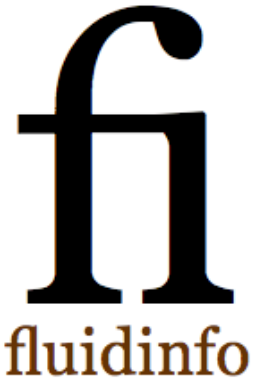
This object is guaranteed to be about `isbn:978-0747534969`, since the `about` tag can never change. So if we query FluidDB looking for books that both `njr` and `terry` have rated 10, we will find this and can retrieve any information on the object, subject to the permissions on the tags.



The FluidDB query `terry/rating=10 and njr/rating=10` would allow thise object to be retrieved, and title could be found with the FDB command

```
$ fdb show -q "terry/rating=10 and njr/rating=10" /njr/title
```

This won't actually work right now (because there is no user `terry` at the time of writing, still less one who has read Fugitive Pieces and rated it 10), but this shows the idea.



fluidinfo.com