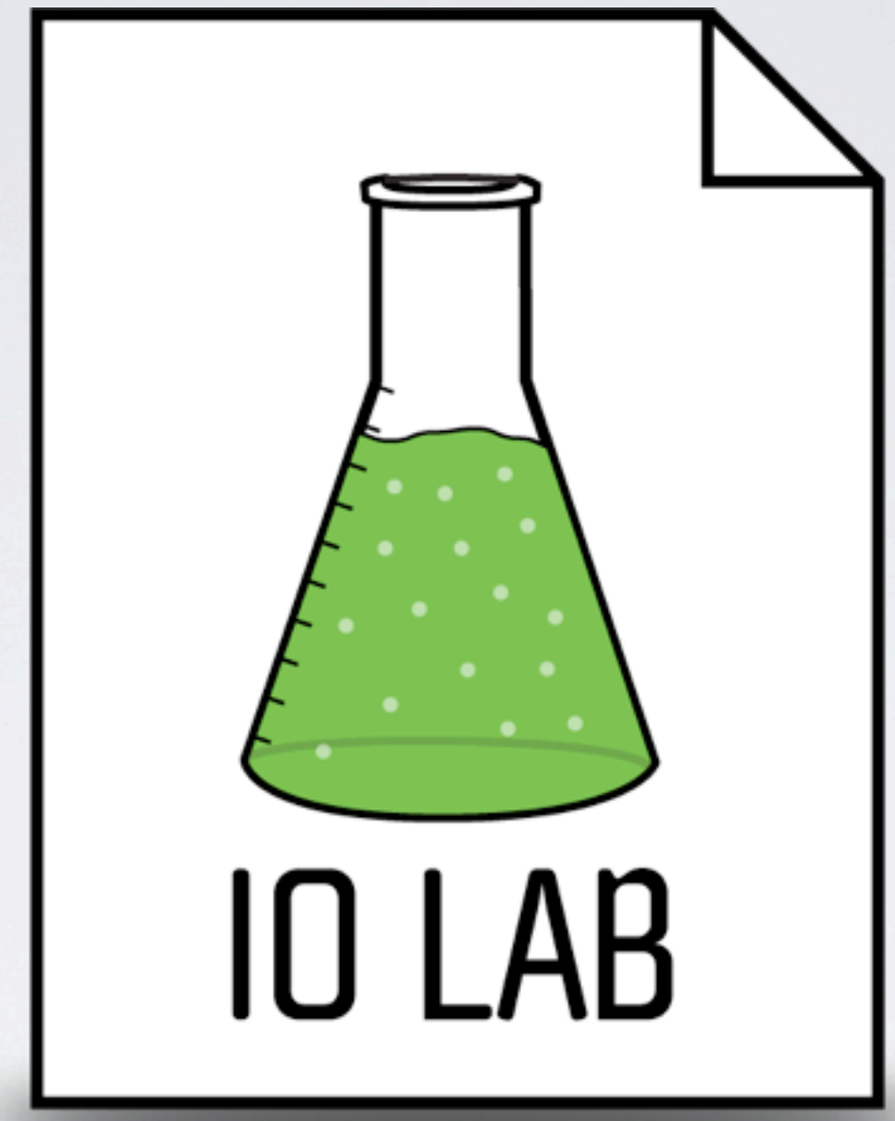
LAST WEEK ON IO LAB

Group for project 2.
Maximum 3 people.
Brainstorm controlled vocabulary ideas
and pick an idea tentatively.

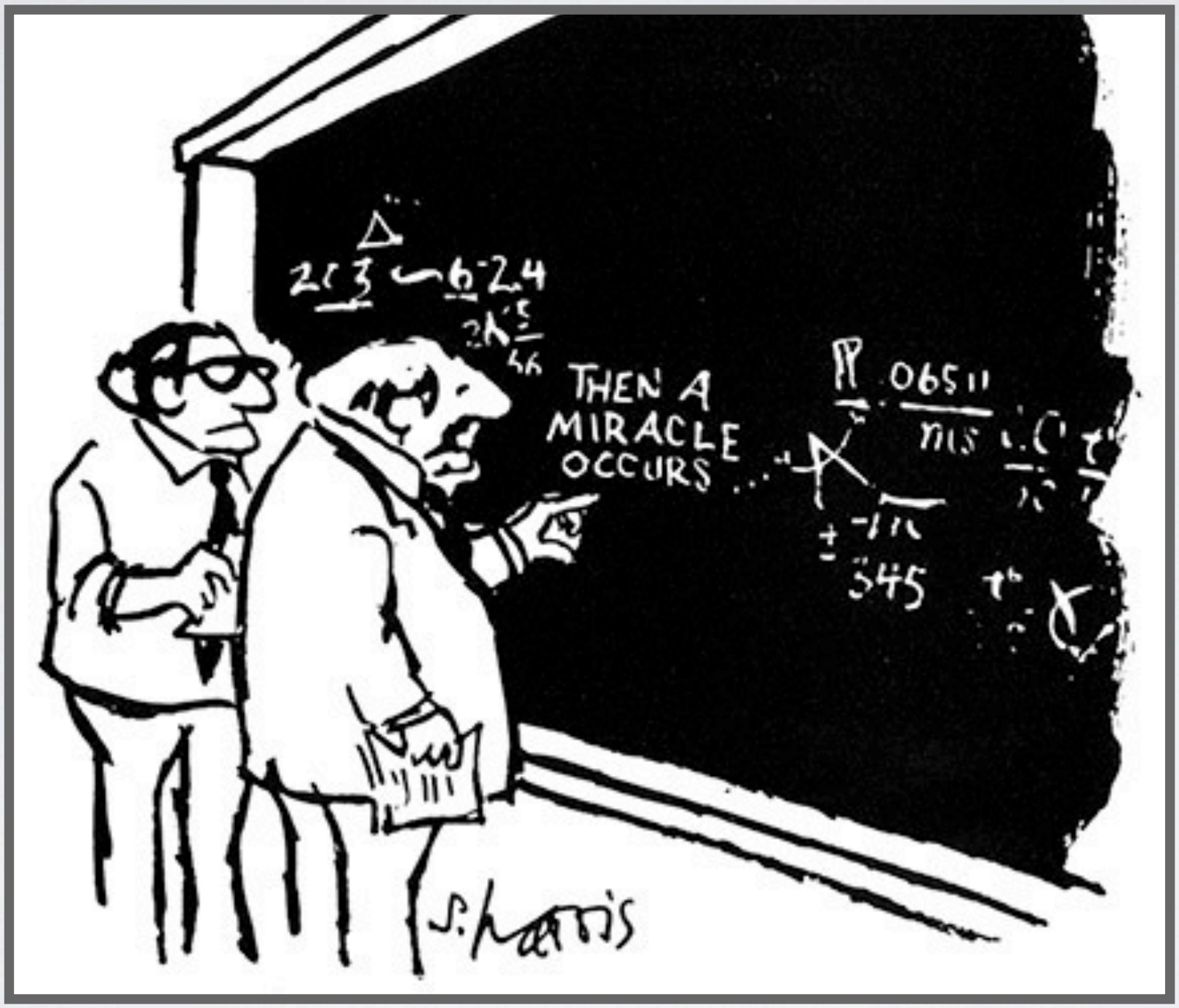
Install RapidSVN.

You should have received an email with
our **feedback on project 1.**



INFORMATION ORGANIZATION LAB

NOTES FROM PROJECT ONE



Cartoon courtesy of Sidney Harris

"I think you should be more explicit here in step two."

Comments in better description, more clarity. It's hard to follow what's going on. More inportant in group work. Also, describe what you are doing at a higher level, or why you are doing it.

MODULAR CODE

Draw example of writing the same function three times for different Delicious API calls.

CHAINABLE JQUERY

```
$('#myElement').text('Hello');
$('#myElement').css('color', 'red');
$('#myElement').fadeIn();
```

```
$('#myElement').text('Hello').css('color', 'red').fadeIn();
```

```
$('#myElement').text('Hello')
    .css('color', 'red');
    .fadeIn();
```

TIDBITS

Prefer literals to objects.

```
var s = new String();
var a = new Array();
var o = new Object();
var re = new RegExp();
```

```
var s = '';
var a = [];
var o = {};
var re = /…/
```

JavaScript: The Good Parts. "3.1 Object Literals", "6.1 Array Literals", "B.10 Typed Wrappers"
http://proquest.safaribooksonline.com/9780596517748/object_literals
http://proquest.safaribooksonline.com/9780596517748/array_literals
http://proquest.safaribooksonline.com/9780596517748/typed_wrappers

TIDBITS

Avoid for ... in to loop over arrays

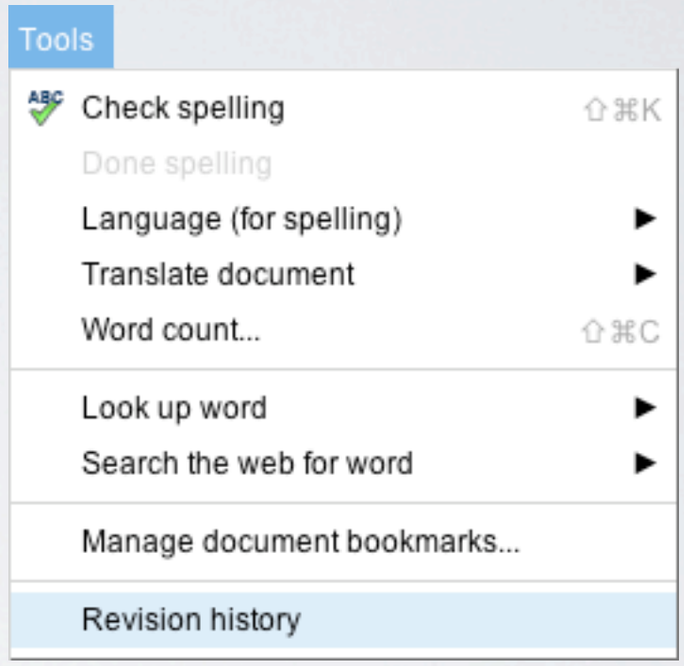
```
// Use a plain for loop or the jQuery .each method
for (var i=0; i < myArray.length; i++) {
    myArray[i];
};

$(myArray).each(function() {
    this;
});
```

http://proquest.safaribooksonline.com/9780596517748/enumeration-id1
http://www.prototypejs.org/api/array
http://dean.edwards.name/weblog/2006/07/enum/
http://yuiblog.com/blog/2006/09/26/for-in-intrigue/

The current version of Javascript implemented in Safari and Mozilla browsers has implement a forEach method: array.foreach(function(i){ console.log(i);}); This doesn't work in some versions of IE but you can extend Array.prototype to support it. See Dean Edwards article linked here.

VERSION CONTROL

Time Machine



Google Docs

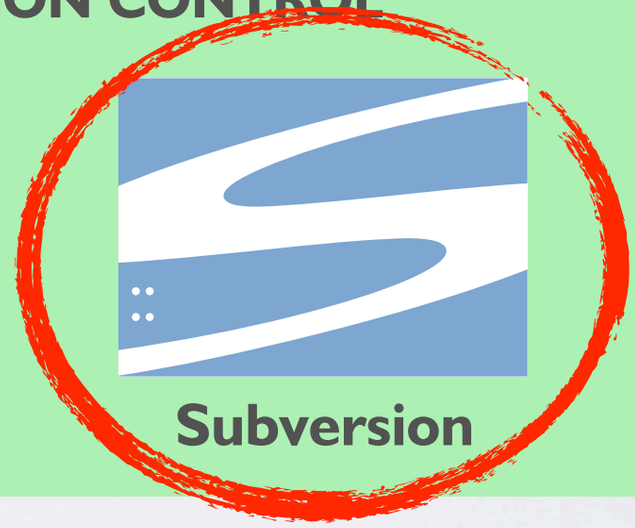
Alternately called revision control, source control. These are two common examples of automatic version control. In contrast, we'll be working with software where you track changes more explicitly. The central point to Subversion--and version control--is that files are stored in a repository which tracks changes to the files. Version control also largely solved the problem of "it's working here, but not here"

VERSION CONTROL

CENTRALIZED VERSION CONTROL

CVS

CVS



DISTRIBUTED VERSION CONTROL

git

Mercurial

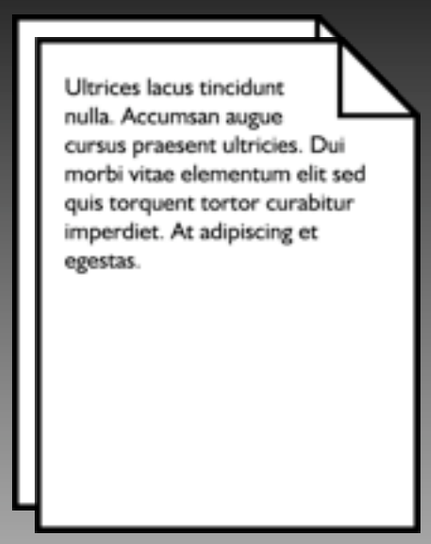
Bazaar

In centralized version control there is one main server. Distributed version control is becoming quite popular, but since it is somewhat more complicated to use, we will use centralized version control in this class. CVS has largely been replaced with Subversion, which is what we'll be using.

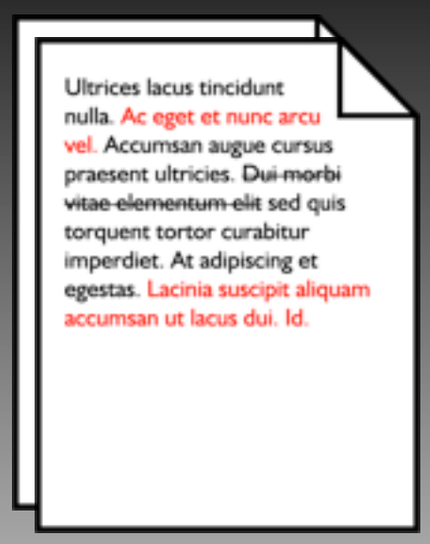
TYPICAL FIRST USAGE

REPOSITORY

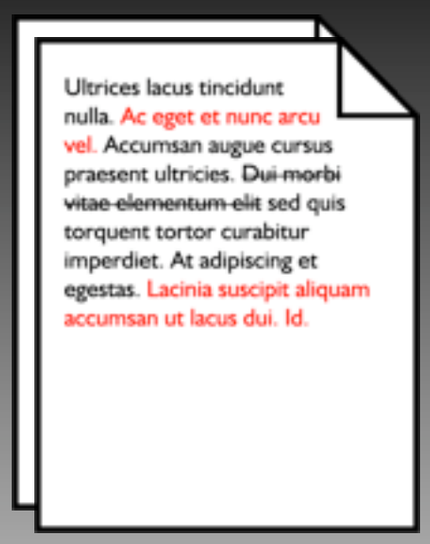
Check Out



Change



Commit



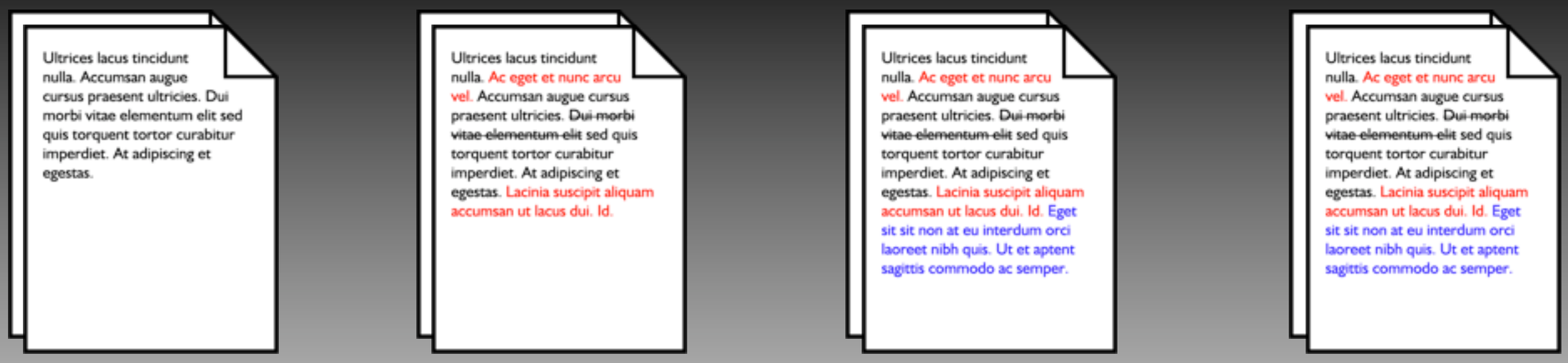
WORKING COPY

The central point to Subversion--and version control--is that files are stored in a repository which tracks changes to the files. The first thing you do is check out a **working copy**. A working copy is a local or personal copy of the repository (or a portion of it). Then you make changes which can include editing files, adding files, renaming, and deleting files. **Commit**. This is the process of copying changes from your working copy to the repository.

CONTINUED USAGE

REPOSITORY

Update Make Changes Update Commit



WORKING COPY

Update: Move files and changes from the repository to your working copy.
Why the second update: this is one of the first rules of version control: always update before you commit. Files may have changes in the meantime.

VERSION CONTROL NOUNS

- **repository** The central place where versioned files are stored.
- **working copy** A copy of the repository where changes are made.
- **revision** A set of changes to the repository, or all the files in the repository at a point in time.
- **changes** Modifying file content, adding files, renaming files, deleting files.

In svn, revisions have numbers. Revision 175 is the repository at a certain point in time. Two other concepts that we aren't talking abou today are tags and branches.

VERSION CONTROL VERBS

- **checkout** Create a working copy from a repository.
- **update** Move any changes in the repository to the working copy.
- **commit** Move changes in the working copy to the repository.
- **add, remove, rename** Schedule a file in the working copy to be added, removed, or renamed to the repository.
- **revert** Undo all the changes in the working copy and replace with the file in the repository.

Lots of information in The Red Book at http://svnbook.red-bean.com.

SUBVERSION TIPS

- Always update before you commit.
- Write something meaningful for your commit message.
- You must use the **add** command to add files in your working copy to the repository.

SUBVERSION CLIENTS

ALL PLATFORMS

Command Line

RapidSVN

MacOS X

Versions

svnX

WINDOWS

TortoiseSVN

More options at http://en.wikipedia.org/wiki/Comparison_of_Subversion_clients

There are also a number of Subversion clients that are integrated with your preferred editor, like Eclipse, NetBeans, or TextMate subversion plugins.

ALL TOGETHER NOW

RapidSVN

https://svn.ischool.berkeley.edu/iolab09/

Now let's do a demonstration. There are a lot of ways to access subversion. You don't have to use Subversion on the command line, but we'll be using it for clarity. Focus on the verbs we use, what's happening in the working copy, and what's happening in the repository.

When should you use version control? What kinds of files can you use version control with?

Vocabulary control is the **sine qua non** of information organization.

CONTROLLED VOCABULARIES

In the wild

Metadata data model

Subject-predicate-object triples

URIs for everything!

```
<foaf:Person rdf:nodeID="me">
   <foaf:title>Dr.</foaf:title>
   <foaf:givenName>Erik</foaf:givenName>
   <foaf:family_name>Wilde</foaf:family_name>
   <foaf:nick>dret</foaf:nick>
   <foaf:homepage rdf:resource="http://dret.net/netdret/"/>
   <foaf:weblog>http://dret.typepad.com/</foaf:weblog>
   <foaf:phone rdf:resource="tel:+1-510-6432253"/>
   <foaf:workplaceHomepage rdf:resource="http://ischool.berkeley.edu/"/>
   <foaf:knows rdf:nodeID="friend1"/>
   <foaf:knows rdf:nodeID="friend2"/>
   <foaf:holdsAccount rdf:resource="http://del.icio.us/dret"/>
</foaf:Person>

<foaf:Person rdf:nodeID="friend1">
   <foaf:name>seung-hyun rhee</foaf:name>
</foaf:Person>
```

FOAF: FRIEND OF A FRIEND

RDF in the wild

```
<fb:type.object.name xml:lang="ja">ハリソン・フォード</fb:type.object.name>
<fb:type.object.name xml:lang="id">Harrison Ford</fb:type.object.name>
<fb:type.object.name xml:lang="tr">Harrison Ford</fb:type.object.name>
<fb:type.object.name xml:lang="uk">Форд Гаррісон</fb:type.object.name>
<fb:type.object.name xml:lang="bg">Харисън Форд</fb:type.object.name>
<fb:type.object.name xml:lang="es">Harrison Ford</fb:type.object.name>
<fb:type.object.name xml:lang="fr">Harrison Ford</fb:type.object.name>
<fb:type.object.name xml:lang="zh">哈里森·福特</fb:type.object.name>

<fb:film.actor.film rdf:resource="http://rdf.freebase.com/ns/guid.9202a8c04000641f80000000011291a9"/>
<fb:film.actor.film rdf:resource="http://rdf.freebase.com/ns/guid.9202a8c04000641f8000000000111363d"/>
<fb:film.actor.film rdf:resource="http://rdf.freebase.com/ns/guid.9202a8c04000641f80000000010cd45c"/>
```

FREEBASE: HARRISON FORD

RDF in the wild

http://www.nytimes.com/2009/09/28/us/28safire.html

VS.

http://www.nytimes.com/2009/09/28/us/28safire.html?_r=1&hp

```
<link rel="canonical"/>
```

There are multiple URLs that point to the same resource. For example, each of these NY Times URLs are bookmarked separated on Delicious though they are the same article. rel=”canonical” lets us specify what the single canonical address is for this resource.

http://ischool.berkeley.edu

VS.

bit.ly tr.im TinyURL.com

```
<link rev="canonical"/>
```

URL shorteners create synonyms for longer URLs, which creates problems because we have even more URLs for a given resource. rev=”canonical” (note “rev” is different from “rel” on the previous slide) lets a site specify the canonical short URL for a given resource. For example, every Flickr photos has a corresponding short URL at http://flic.kr

JQUERY AUTOCOMPLETE

Demonstration of jQuery Autocomplete, professor demo, Virginia Joint Registry autocomplete.

SEMANTIC WEB APIS

The New York Times

TimesTags API

Get standardized terms that match your search query, and filter by Times dictionaries.

The NY Times has a large number of APIs where you can get all sorts of structured, controlled vocabulary information from their publication. One example: TimesTags, which return objects that link to TimesTopics pages. Note: bring your own proxy.

SEMANTIC WEB APIS

FREEBASE

Last week we looked at a demonstration of the Freebase API.

MEASURING LINK DILUTION

backtweets

Backtweets lets you search for a URL and find all the messages on Twitter that use that URL.

FOR NEXT WEEK

Project 2 is due.
Same as last time.
Email us by 12:00pm (noon) with your project, who was in your group and who did what, an explanation of why you did what you did, what challenges you faced and what worked and what didn't.

Install Google AppEngine