# LibX 2.0 - an Open Source, Community Platform for Delivering Library Services

Code4Lib 2009 Godmar Back

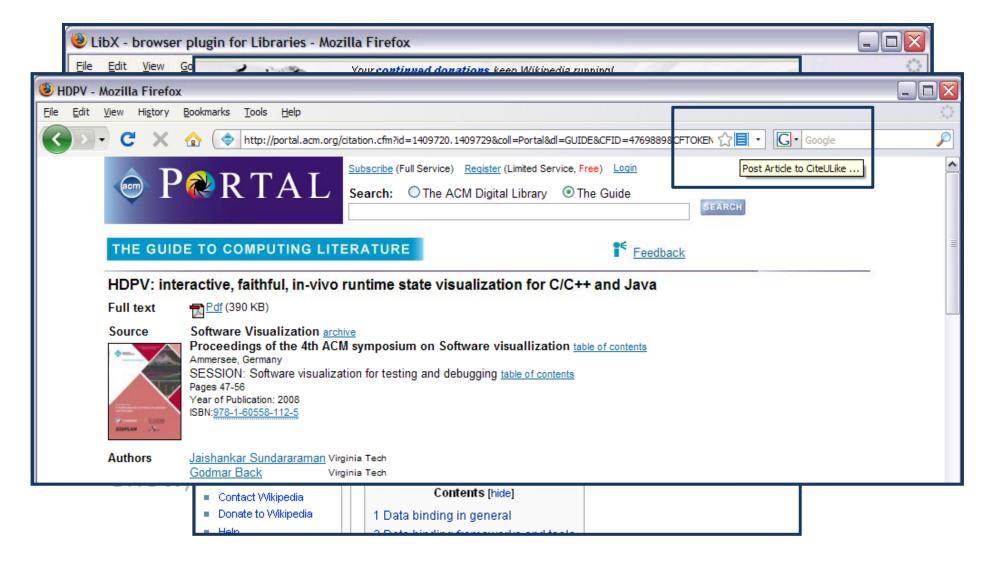
University Libraries and Dep. Of Computer Science Virginia Tech



#### LibX 1.0

- Toolbar and right-click context menu
- Adaptive and user-configurable context menus
- OpenURL support
- Magic Button (Google Scholar support)
- Web Localization via Embedded Cues
- Autolinking
- Off-campus access via EZProxy or WAM
- Support for CiteULike
- Support for COinS
- Support for xISBN
- Show/Hide Hotkey

#### LibX 1.0

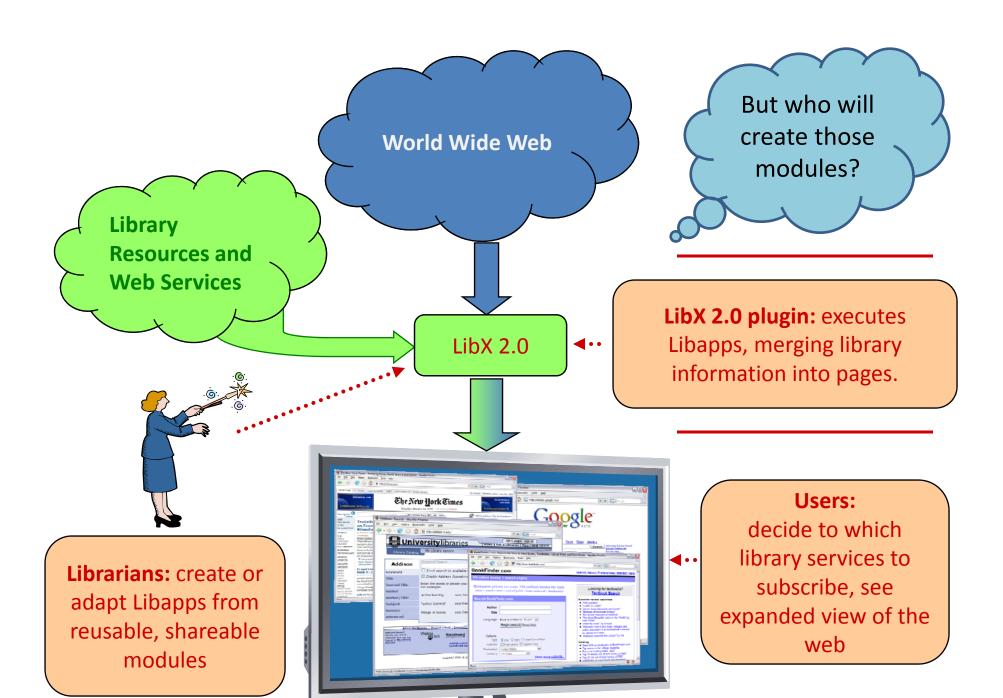


#### The LibX Edition Builder

- A configuration management tool for creating customized versions of LibX
  - Customized version of LibX = LibX edition
- Edition configuration includes descriptions of community-local resources:
  - OPACs, OpenURL, Proxy, Databases, Links, Branding, ...
- Edition Builder is easy to use
  - Makes heavy use of OCLC registries
  - Uses sophisticated auto-detection techniques
  - Usable by *librarians*, not just programmers
- Anybody can create, share, and manage editions
- Over 550 edition as of now, new ones created at a pace of 20/month
  - Huge human investment
  - 10's of thousands of end users

### Where to go from here?

- A toolbar is great, but...
- Emerging technology trends
  - Service-oriented architectures, web services interfaces – soon even to ILS!
  - Data mash-ups; HTML widgets
- Educational trends: librarians, educators, and users create
  - Online tutorials, subject guides, visualizations
  - Social OPACs: tagging, reviews, recommender services



## The LibApp Model

- How can the interaction of LibX with web content be modeled?
- Typical tasks involve
  - Examination of the page and extraction of information
  - Processing of information
  - Modification of the page
- A Module is a small piece of JavaScript code along with a metadata description of its input and/or output
- A *Libapp* is a group of modules
- A Package is a folder of libapps and packages

#### Modules

- Modules are named using a URL and published via REST/AtomPub
- Modules can reference JavaScript libraries (e.g., jQuery) and CSS stylesheets
  - Execute in a parallel environment to the client page
- Modules are trusted, with full access to LibX API, including edition configuration and user preferences
- Modules communicate with each other via tuple space

## LibX Tuple Spaces

#### tuple = TAKE(template)

- If a tuple matching template exists in the tuple space, remove it and return it
- If no tuple exists, wait until a matching tuple is written, remove it and return it

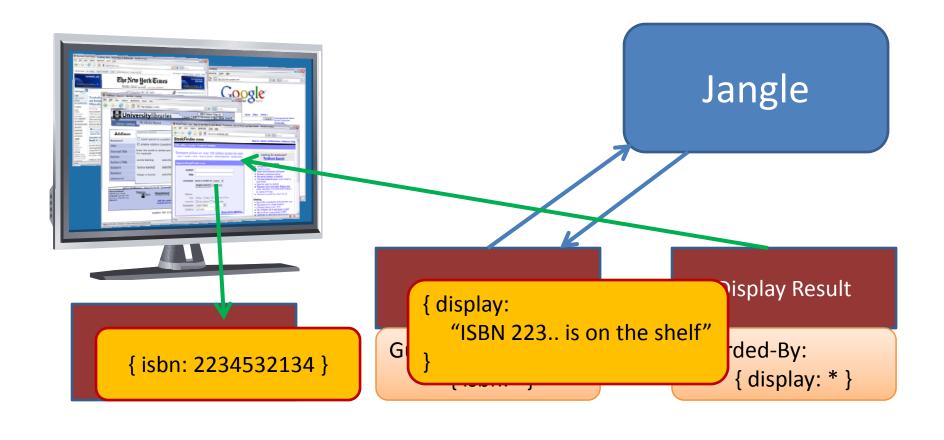
#### **WRITE** (tuple)

- Write a tuple into the space
- If a TAKE is pending with a matching template, complete the TAKE

Tuples and Templates are JavaScript objects in JSON notation.

Tuple example: { isbn: "0743226720" }

Template example: { isbn : \* } // any tuple with 'isbn' field



**Tuple Space** 

### Rationale for Tuple Spaces

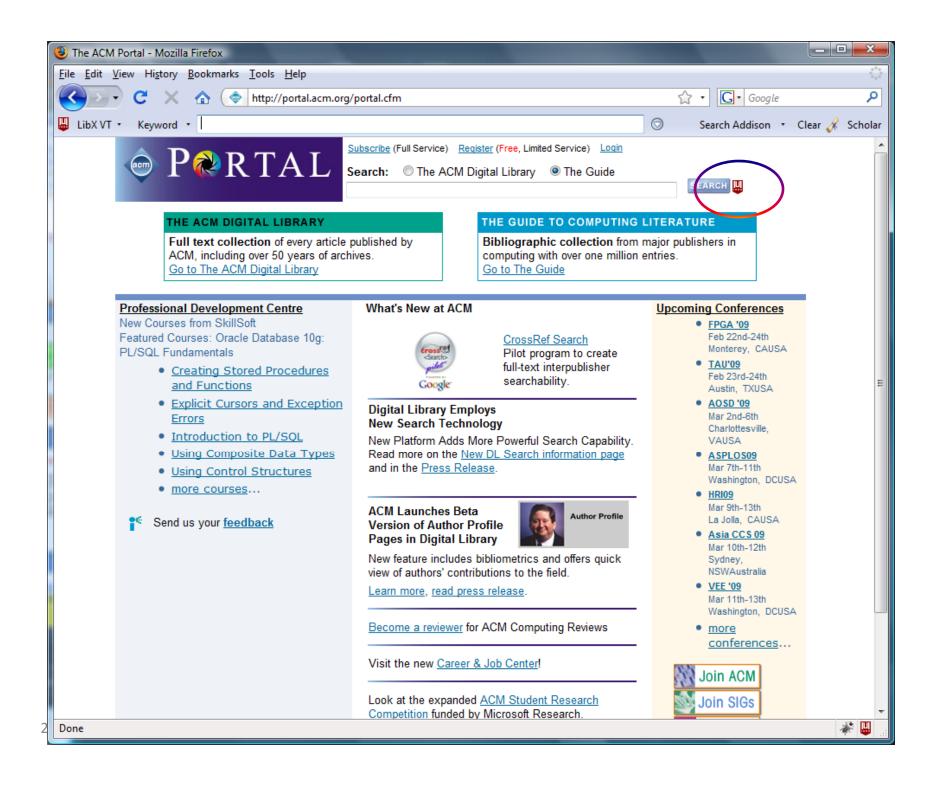
#### **Software Engineering**

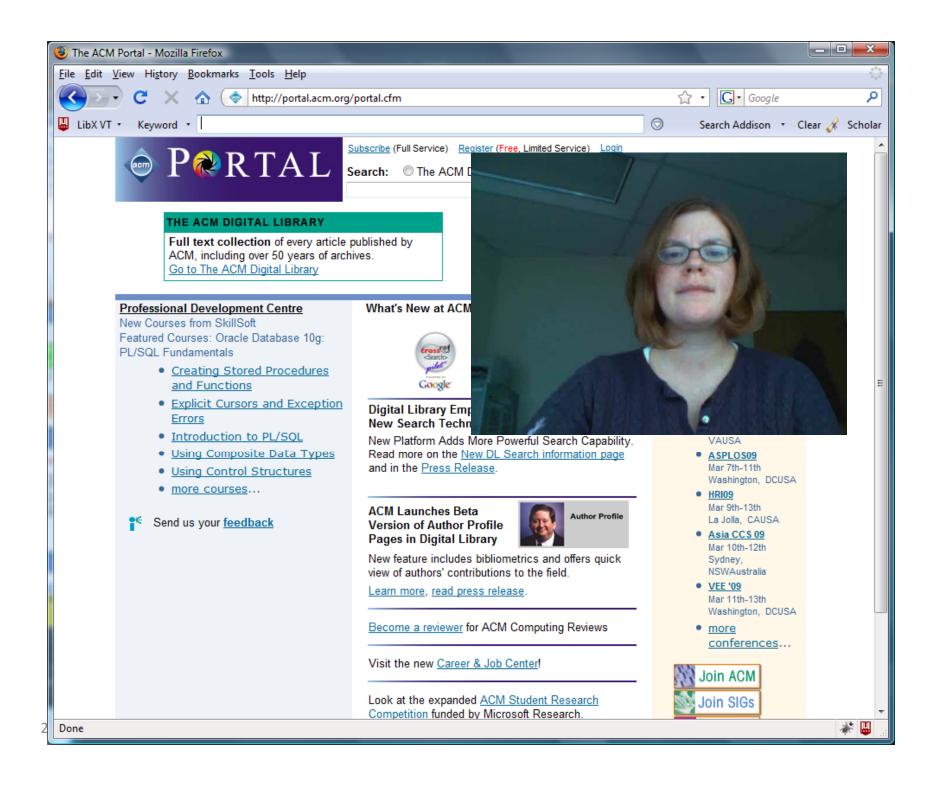
- Low coupling between modules
- Independent composition
- Simplicity
- Suitable for metaprogramming

#### **Handling Asynchrony**

- User actions happen asynchronously
- Information arrives asynchronously from external sources
- Execution order independence

# LIBAPP DEMO 1 CREATE A PRESENCE FOR THE LIBRARY





### Step 1: Arrangement

```
Description: Display a Help icon on the ACM page that plays a video when clicked.
Include: /portal\.acm\.org.*/
Require: jquery
// Place the current edition's icon into the ACM portal page, next to the
// current search button.
libx.space.write ( {
  needsuserbutton: $('input[src="http://portal.acm.org/images/search_small.jpg"]'),
  image: libx.edition.options.icon,
  text: "Click for a short tutorial",
  action: function () {
    // When user clicks, offer to show a YouTube clip
    libx.space.write ( { youtube: "ehkfEcQ5YnQ"} );
});
```

#### Step 2: Place UI Button

```
Description: Place a clickable image into a page
Include: /.*/
Guarded-By: { needsuserbutton: libx.space.WILDCARD, action: libx.space.WILDCARD }
Require: iquery
// Create a link with an embedded image
var a = $("<a href='javascript:void(0);'/>");
a.append('<img border="0" src="' + tuple.image + '"/>');
a.attr( 'title', tuple.text | | "");
// Insert link after element where a 'user button' is wanted
$(tuple.needsuserbutton).after(a);
// Associate onclick handler and animate
$(a).click(tuple.action).fadeOut("slow").fadeIn("slow");
```

Code4Lib 2009

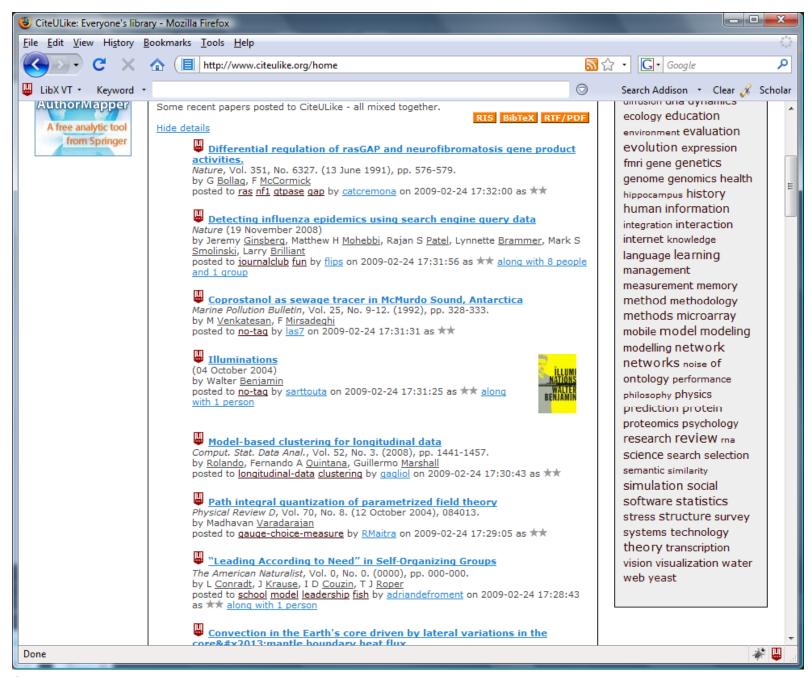
## Step 3: Create Youtube Clip

```
Description: Create a notification to play a YouTube video, based on Video ID
Include: /.*/
Guarded-By: { youtube: libx.space.WILDCARD }
// Create HTML based on tuple.youtube
libx.space.write ( {
  notify: '<object width="425" height="344"><param name="movie"
   value="http://www.youtube.com/v/'+tuple.youtube+'&hl=en&fs=1"></param><par
   am name="allowFullScreen" value="true"></param><param
   name="allowscriptaccess" value="always"></param><embed
   src="http://www.youtube.com/v/'+tuple.youtube+'&hl=en&fs=1"
   type="application/x-shockwave-flash" allowscriptaccess="always"
   allowfullscreen="true" width="425" height="344"></embed></object>',
  options: { width: '450px' }
} );
```

# Step 4: Display HTML Notification

```
Description: Display HTML notifications via an embedded panel using jGrowl
Include: /.*/
Guarded-By: { notify: libx.space.WILDCARD }
Require: jquery
Require: jgrowl
Require: jgrowl.css
// Set sticky:true unless provided in tuple.options
var jGrowlOptions = $.extend({}, {sticky:true}, tuple.options);
// Display notification
$.jGrowl(tuple.notify, jGrowlOptions);
```

# LIBAPP DEMO 2 "GET FULL TEXT LINKS FOR COINS"

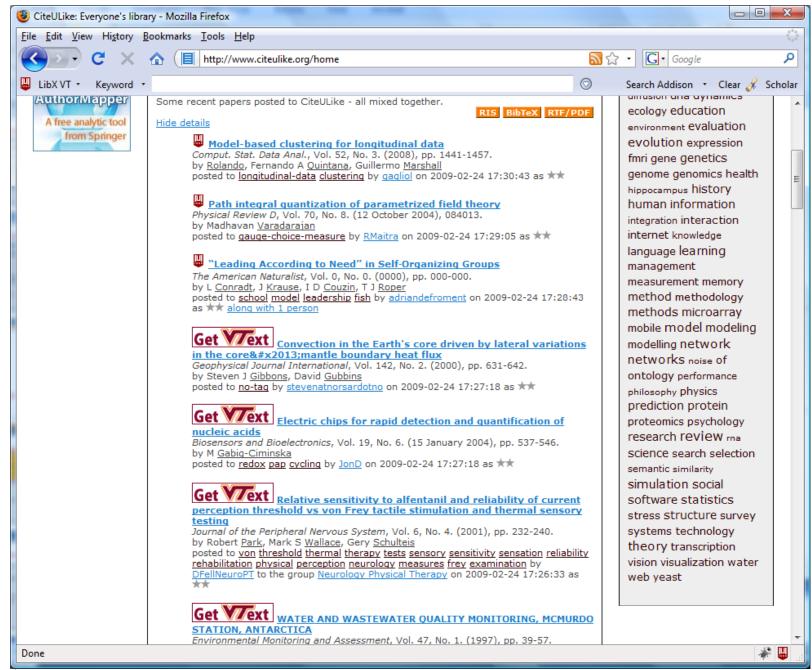


## COinS, the LibX 2.0 way

```
<span class="Z3988"</pre>
Include: /.*/
                           title="ctx ver=Z39.88-
Description: Find COinS
                           2004&rft val fmt=info:ofi/fmt:kev:mtx:j
                           ournal&rft id=info:pmid/16646082&rft.g
Require: jquery
                           enre=article">
$("span.Z3988").each(function () {
  libx.space.write({
    coins: this,
    contextobj: this.getAttribute('title')
  });
});
```

#### COinS, the LibX 2.0 way

```
Include: /.*/
Description: Link COinS
Guarded-By: { coins: libx.space.WILDCARD }
Require: legacy-cues
var cue = new libx.cues.StandardCoins(tuple.contextobj);
cue.insertBefore(tuple.coins);
```



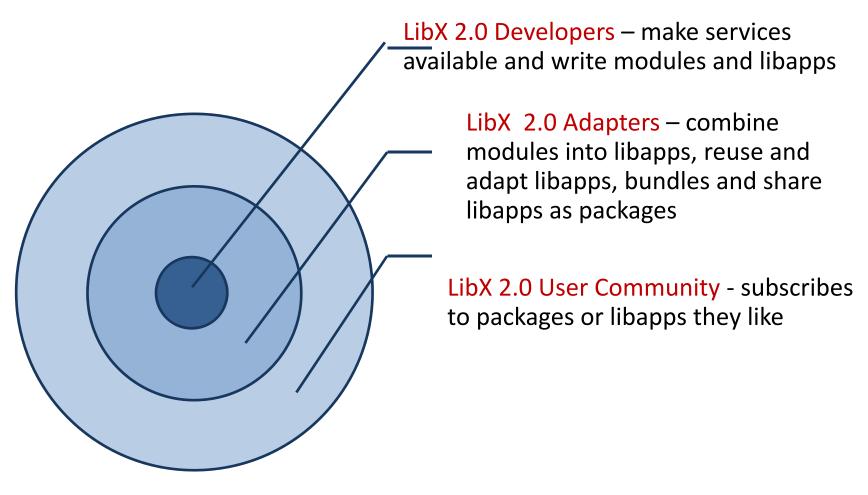
### Add Link/360 direct link

```
var link360 = libx.services.link360.getLink360(libx.edition);
if (link360)
 link360.getMetadata({
  query: tuple.contextobj,
  type: 'article',
  hasFullText: function (xmlDoc, url, databaseName) {
   cue.setAttribute('href', url);
   cue.setAttribute('title', "Users of " + libx.edition.links.primary.label
      + " click here for full text via " + databaseName);
   cue.setImageAttribute('src', 'http://www.lib.vt.edu/images/getvtext.gif');
   cue.animate();
  },
```

### Transitioning from LibX 1.0 to 2.0

- Complete redesign of LibX client code
- Browser-independent, strictly object-oriented JavaScript code with proper namespacing and encapsulation
- Provide full documentation (via jsdoc, accessible in about:libx)
- Provide built-in interactive reflection of data structures
- Include unit tests (run via Rhino)
- Hot updatable

#### LibX 2.0 – Roles



# LibX 2.0 Community Repository

- Based on tuple space model, create metadata descriptions for modules
- Create a repository of modules, libapps, and packages
- Editable like a Wiki
- Integrate repository into Edition Builder
  - Include consistency checking
- This will be done during the next 2 years (but libapps can already be created now)

# Call for Developers

- Launched Developer Website
  - http://developers.libx.org
- Launched mailing list
  - libx-dev@googlegroups.com
- Transition path: update from LibX 1.5
- Agile roll-out, no "releases"
- We invite you to try and play with it!

#### LibX Team

- Annette Bailey
- Godmar Back
- Kyrille Goldbeck
- Mike Doyle
- Arif Khokar
- Travis Webb
- Alumni
  - Nathan Baker
  - Tilottama Gaat
  - Tobias Wieschnowsky

