



[Tech Notes are short articles discussing library-related technology.]

[Tech Note:] Only Connect: APIs

Some of you may have heard some buzz about application programming interfaces (more commonly referred to as “APIs”). They play an important role in mashups, the melding of two or more Internet services into a new tool. For example, through the API for Google Maps you could add a Google map to your library’s web page showing the location of your library and any branches.

APIs provide the means for one computer program or web application to “hook up” with another and exchange data. It is a means of communication that is intended for application-to-application transfer of data, so it is normally nothing that the end-user will actually see directly, but rather functions “under the hood”.

APIs are available for both conventional software programs and for web-based services and applications. The Windows operating system offers thousands of specific functions that are available to software running on a Windows PC. For example, the Windows API functions allow programs to control the behavior of the windows that a program displays; print, start or stop themselves; draw graphics; or, reboot the computer. Communication with the API can be done through the C and Basic programming languages, among others.

Some library vendors offer APIs within their systems. The API may offer the programmer direct access to searches and resultant records, allowing modification of the user interface to the OPAC or ILS. Or, depending on the features of the API, it might be possible to pre-process the search query in various ways for “better” search results. Thus, the availability of APIs can enable the expansion and augmentation of access to the catalog and the ILS. Internal processes, such as reports, might also be customized.

Besides the creation of custom maps on a web page, the Google Maps API offers the addition of markers and “infowindows”, the balloons that contain information about a map location. There are many additional functions available to create more elaborate maps with many more features. Connection to the API from a web page is accomplished through JavaScript coding embedded in that page.

How do APIs work? How does an application communicate with an API? Consider an example using the xISBN web service from OCLC. The service can perform a number of ISBN-related operations, one of which, given an ISBN number, returns the ISBNs of all editions of that work [1]. The service is accessed through URLs sent by the user (which is generally some kind of software rather than a person). Such a URL might be:

<http://xisbn.worldcat.org/webservices/xid/isbn/0596002815?method=getEditions&format=xml&fl=year,lang,ed>

If you click on the link above, you should see something like this (if you're using IE):

```
<?xml version="1.0" encoding="UTF-8" ?>
- <rsp xmlns="http://worldcat.org/xid/isbn/" stat="ok">
  <isbn year="2004" lang="eng" ed="2nd ed.">0596002815</isbn>
  <isbn year="1999" lang="eng">1565928938</isbn>
  <isbn year="1999" lang="eng" ed="1st ed.">1565924649</isbn>
  <isbn year="2008" lang="eng" ed="3rd ed.">0596513984</isbn>
  <isbn year="2000" lang="fre">2841770893</isbn>
  <isbn year="2006" lang="eng">1600330215</isbn>
  <isbn year="2002" lang="pol">8371975961</isbn>
</rsp>
```

What does all of this gobbledygook mean? Let's take a look at that URL first. The ISBN 0596002815 identifies a book called *Learning Python*. The request encoded in the URL is "Get Editions", which returns ISBNs and the other information requested for each item that is identifiable as the same work.

Now consider what is returned in response to this URL. Remember that the URL is actually a message to the xISBN software rather than a link to a web page. The software returns information in the XML (eXtensible Markup Language) format. Ignoring the formatting and housekeeping information, it is pretty easy to identify the ISBN number, year of publication, language, and edition for each item.

No doubt you noticed that the information returned is not the most useful in terms of human readability. That's because it's assumed it will be read by software. The format is actually quite easy for a computer program to "unpack" and use.

One possible application for xISBN is within a library's display of items. If ISBN is included in the item as a searchable field, it would be possible to display links from a found item to other items that are manifestations of the same work as identified by xISBN.

What's described here is a mashup. Mashups can enhance an existing application or make a brand-new application possible. And mashups would be impossible without a well-defined API for the two or more applications or web services that form the mashup. So, APIs will continue to be useful and creative parts of new and updated applications.

Takeaways From This Article

- Application Programming Interfaces (APIs) provide communication links between software or web services.
- APIs allow programmers to enhance or expand the capabilities of software and web services.
- Mashups require the web services that are joined to possess APIs.

- Some ILS vendors offer APIs so that customers can create customized user interfaces and other novel features or mashups.
- As new software and services are created, most of them will offer APIs, resulting in more people and companies creating mashups.

[1] The example is from the [xISBN API page](#) . (Link tested 10/22/08)

Many thanks to Jane Richard for her extraordinary editing skills on this and previous Tech Notes. Any errors or infelicities which remain are, of course, mine.

—Tom Zillner (tzillner@wils.wisc.edu)