Librarians and Code Year

Andromeda Yelton, Guest Columnist

*Correspondence concerning this column should be addressed to **Eric Phetteplace**, Emerging Technologies Librarian, Chesapeake College, 1000 College Circle, Wye Mills, MD 26179; e-mail: ephetteplace@chesapeake.edu.*

As soon as I became editor of the "Accidental Technologist," I thought of a column on Codecademy's "Code Year" program. Code Year is interesting not only for its unique online educational model but also because of the way librarians have organized a community around it. I met Andromeda Yelton at ALA's 2012 Midwinter conference, where she helped run an informal meeting of librarians participating in Code Year and organize an official ALA interest group around the event. She holds a BA in Mathematics from Harvey Mudd, an MA in Classics from Tufts, an MLS from Simmons, and works for Gluejar.inc, an innovative company set to "unglue" e-books by purchasing their rights and re-releasing them under Creative Commons licensing. As an active member of LITA and a co-chair of the Code Year Interest Group, she seemed like the perfect person to write a column explaining the program and its benefits.—*Editor*

What if you could write a program to make bulk edits to thousands of MARC records with a single click? What if you could change the way your whole catalog looks—even if it's proprietary software that doesn't let you change the HTML?[1] What if you could write your own mobile web site to display your historical photographs to people throughout the community, in the very locations those photos were taken?[2] What if you could explain to your systems folks exactly what you needed because you could talk their language—and what if you didn't even have to ask, because you could do the work yourself?

Questions like these—alternately practical, innovative, and empowering—are motivating librarians to learn to write software. There are many resources out there for people who want to learn to code: web sites, books, and classes (both free and paid). However, one that's been getting a lot of attention from librarians lately is Code Year, from codecademy.com. If you sign up at http://codeyear.com (it's free), Codecademy sends you a new programming lesson each week. The lessons are relatively short, build sequentially, and tie into projects for extra practice. Code Year is part of the "gamification" trend, so it awards badges for the successful completion of lessons and other milestones; it also makes it easy to share your progress on social networks. This free, bite-sized, rewarding approach—plus a splashy publicity campaign when it launched in January—got many dozens of librarians to sign up.

What kinds of things can you learn through Code Year? The first 12 weeks were an introduction to JavaScript, a programming language used to create dynamic web content. The lessons covered the syntax of the language and traditional programming concepts like loops, functions, and variables.

Extra projects, such as building a blackjack game, let people apply their knowledge. Code Year has now moved into an introduction to HTML and CSS. While it hasn't combined these languages yet, I expect it will down the line, enabling students to produce complete, interactive web pages.

Because Code Year is built on the Codecademy platform, though, it's possible to learn many more things. This platform supports interactive lessons in JavaScript, Python, and Ruby. In addition, it allows creators to build their own lessons. At least one librarian has been taking advantage of this; one of the community-submitted lessons teaches how to build a JavaScript program to manipulate MARC records (http://www.codecademy.com/courses/marc-viewer). In my experience Codecademy is quick to respond to feedback, so if you'd like to beta test lessons to make them more beginner-friendly, or create your own lessons to share what you know, they want to hear from you.

SO HOW DOES THIS CODE THING WORK, ANYWAY?

I've been talking in generalities about code so far; I'd like to give a specific example. I'm going to walk you through a bit of jQuery I wrote for my employer's web site, http://unglue.it. jQuery is a derivative of JavaScript that simplifies some of the syntax, especially for the functions web developers use most. This means it'll look familiar to you if you've been doing the Code Year lessons on JavaScript, but it's a lot more readable—so even if you have no code background, don't despair! This program uses some normal English vocabulary, and I'll walk you through it.

This is from a script named toggle.js. On our site, supporters can have wishlists of books, and site users can view those books in either a list or a panel layout. They can toggle between these views by clicking on list and panel icons. Here's part of the code that does that:

```
$j(document).ready(function() {
    $j('#toggle-list').click(toggleList);
    $j('#toggle-panel').click(togglePanel);
    });

function toggleList() {
    $j('div.panelview').addClass('listview').
removeClass("panelview");
    $j('#toggle-list').css({opacity: 1});
    $j('#toggle-panel').css({opacity: .2});
    }
```

The first few lines activate the function when the document is ready—that is, when the web page has been sufficiently loaded. In line 2, $j('#toggle-list') finds the element of the page whose ID is toggle-list and tells it, when someone clicks on this icon, execute the toggleList function. (If you've done any CSS, the #toggle-list ID selector will look familiar to you.) Can you guess what line 3 does?

Next, we have the function toggleList itself. $j('div.panelview')—again, the CSS syntax may be familiar—finds all the divs whose class is panelview. We then add the class "listview" to each of these div elements, and remove the class "panelview." At this point our stylesheet kicks in. Since the stylesheet has different rules for how to display listview and panelview elements, all these elements are immediately redisplayed with a different set of rules. You still see the same objects—a book cover image, a title, etc.—but they're laid out entirely differently. (Go ahead and try it! My wishlist is at https://unglue.it/supporter/andromeda.)

Finally, we need to grey out the button that switches to listview (since we're already in listview) and darken the button that switches to panelview, so users know it's the one to click on if they want to switch views. That's what the last two lines do.

As you can guess, there are a few lines I've left out here—the togglePanel function! It's quite similar to toggleList, but with a few necessary, logical changes. Can you guess how it works?

So, why do we care? Well, with just a handful of lines of jQuery, and a stylesheet, I can instantly change the layout of a page. Can you imagine any web pages at your library that would be better if they were laid out differently? As you recall, this was one of the examples in the first paragraph: a library catalog that didn't allow the librarians to edit HTML but did allow them to insert some jQuery and custom stylesheets. Using techniques like the ones above, Matthew Reidsma was able to dramatically improve the appearance and usability of the catalog even though he didn't have direct control over its proprietary software.[3]

RESPONSES FROM LIBRARIANS

In writing this article, I spoke with a half-dozen librarians who are working on Code Year. They work at both public and academic libraries, in both public and technical service positions.

Why are these diverse librarians interested in learning to code? Some had prior programming background and wanted to keep their skills sharp. Others knew that better knowledge of code would let them save time with everyday tasks. They spoke of being able to understand the online learning modules the library offered; communicate with IT; and get involved with library software. Abigail Goben, Assistant Information Services Librarian and Assistant Professor at UIC Health Sciences Library, spoke for several of her peers when she said learning to code "takes away a lot of the mystique and gives me more confidence."

That said, most librarians spoke of challenges, too. Learning to code requires stretching one's brain in new, maybe unfamiliar, ways; while many librarians have some background in HTML and CSS, this doesn't cover the concepts needed for a true programming language like JavaScript. Code Year (like

many programming courses) expects a certain comfort with mathematics, which has been a hurdle for some. It's time-consuming, and keeping up with the lessons consistently—as well as allowing oneself the time for the lessons to truly sink in—has been difficult.

Therefore, the librarians I spoke to suggested that anyone who wants to learn to code set aside consistent, sufficient time for it. They also recommended finding healthy ways to deal with frustration. Everyone gets frustrated while learning to code; it's important to remember that this is normal, that there are lots of people out there to help, and that you should be proud of yourself for putting in the effort. Setting manageable goals is important, too. "Learn to code" is big and unhelpful; aiming to take a specific course, learn a specific language, or accomplish a specific project will get you farther.

RESOURCES

So, what if I've convinced you? You've seen how much you can accomplish with just a few lines of code, you want to build your skills and self-confidence: you'd like to learn to code yourself. How can you get started? There are lots of beginner-friendly, often free, resources out there.

Free Online Resources

Code Year is the best known, and has a large community that can support you, but it's far from the only resource out there and it doesn't cover everything. If you'd like to tackle the Ruby programming language, there's an interactive, quirky, and charming introduction at http://tryruby.org/. If you need extra practice with Java or Python, there are lots of practice problems at http://codingbat.com/. Google has Python and C++ classes in the Programming Languages section of http://code.google.com/edu/. While these assume a little familiarity with the jargon of programming, they are well-organized, have useful practice problems, and cover surprisingly much ground; this is where I learned most of the Python I've ever needed. Finally, my Library Code Year co-chair Carli Spina has a huge collection of online resources at her Pearltrees page (http://bit.ly/IHpt4I).

O'Reilly Books

Sometimes you just need a reference book. In software, that book is almost always published by O'Reilly. The distinctive, pointillist animals on their paper editions make them easy to pick out on any bookshelf; their DRM-free e-books, available in a variety of formats, are easy to use on the device of your choice. O'Reilly books are known for their reliably high quality and their coverage of essentially any software topic you need. Check to see if your library has a subscription to their Safari service, which gives you access to the complete set online.

Beginner-friendly, in-person events

While code can have a reputation for being arcane and hostile to beginners, that isn't always deserved. Every programming language attracts its own community with its own culture, and two languages in particular are well-known for having beginner-friendly niches: Python and Ruby. Both languages have active organizations offering introductory workshops for "women and their friends." These workshops don't assume prior background in coding, have an explicit pro-diversity focus, and are free. If there's one in your area, you are welcome to attend (men, you may need to bring a female friend who also wants to learn to code). If there isn't, both groups have online resources for event organizers, including curriculum; you may be able to reach out to local users' groups and host your own event (libraries are great spaces for this!). For more information, see http://bostonpythonworkshop.com and http://railsbridge.org.

By the way, while beginner-friendly, these aren't "beginner" languages in any way: they're fully-featured languages that can be used to write nearly anything. Twitter was originally built on Ruby; much of Google is written in Python. They're friendly, not because they're watered down, but because they're cleaner and more readable than (for example) JavaScript or C++. If you'd like to learn to code but you want something that looks more like English and less like parenthesis and semicolons, you may find either Python or Ruby to your taste. I didn't really connect with my college C++ course, but Python feels beautiful and exciting to me—which makes me much more interested in learning it.

Communities of Librarians

Learning is easier when you have people to ask questions of, encourage you when you get stuck, and cheer you on when you make progress. And social learning is especially important in code, where tips, tools, and best practices are typically taught through osmosis, not in formal lessons. There are two online communities for librarians working through Code Year: a discussion forum in ALA Connect (http://connect.ala.org/codeyear), and a wiki especially geared toward catalogers (http://catcode.pbworks.com/). Each has over 120 members (with some overlap), with a wide variety of library roles and technical experience, so there's definitely a niche for you.

In addition, there's a newly formed LITA/ALCTS interest group, Library Code Year, for librarians who want to learn to code and apply code. (Disclosure: I'm a co-chair.) We had our first meeting, this past ALA Annual, at which we discussed what the interest group can do to best best support its members. Find out what we talked about and join (or start!) discussions in our Connect group, http://connect.ala.org/node/167971. We'd love to see you there.

Notes

1. A real project: http://matthew.reidsrow.com/articles/11. Matthew Reidsma, at Grand Valley State University, wanted to improve the

usability of his library's Serials Solutions interface. While he could add some custom code, he couldn't change the underlying HTML. He used jQuery to clean up the HTML and alter the page's design and layout to something that met his usability goals. The blog post linked above is very specific as to how he did this.

2. Also a real project—in fact, several: Scan Jose (http://scanjose.org/) and NCSU WolfWalk (http://m.lib.ncsu.edu/wolfwalk).
3. Hello. My Name is Matthew Reidsma, "jQuery for Customizing Hosted Library Services," http://matthew.reidsrow.com/articles/11.