P A R T N E R S H I P
The Canadian Journal of Library and Information Practice and Research
Revue canadienne de la pratique et de la recherche en bibliothéconomie et sciences de l'information

vol. 12, no. 2 (2018)
DOI: http://dx.doi.org/10.21083/partnership.v12i2.4121

Roundtable: Should Library Workers Learn To Code?

Alan Harnum
Brad Houston
Ruth Kitchin-Tillman
Danielle Robichaud
Anna-Sophia Zingarelli-Sweet

Compiled by Allana Mayer

For this issue, we asked readers responding to our recent pieces about technology, programming, and self-directed learning to answer this question: Should we expect LIS professionals to learn how to code? Responses came long, short, and even in haiku form. Enjoy!

Ruth Kitchin-Tillman

Linked Data Strategist, Penn State University Libraries, Penn State University

When answering the question “should library workers learn to code?” I need to provide some context to my response. I am a library worker who codes at work. More specifically, I am a library worker who loves coding, who loves teaching others coding basics or tricks, who serves as an editor of the *Code4Lib* journal, and who paid for her MLS in part by working late nights writing freelance code. For all that and perhaps because of it, my short answer is “no.” There is no overarching need for all library workers to learn to code.

In the rest of my response, I will not be addressing the case of library workers such as developers, systems support, and others whose job descriptions and daily labour require understandings of code. Their understandings may range from years of experience as developers to a familiarity with four or five scripts which are integral to their work, but they do need to know how to code.

Unfortunately, “learn to code” generally acts as a generic shorthand for acquiring some level of technical aptitude. Rarely does it include a concrete *what* or *why*. I suggest there are two situations in which library workers whose job does not require coding still

may have a compelling *what* and *why* to suggest they develop knowledge or skill in coding. The first is coding as a means of problem solving. The second is improved communication and partnership with other library workers who write code as a part of their job.

Despite the number of free classes and tutorials for various coding languages, a lack of application to their regular work makes it hard for those trying to "learn to code" to retain what they learn. A systems specialist at one of my former workplaces took classes and workshops to "learn Python," but until we sat down with the PyMARC library, she had difficulty finding a use for class material. Once she saw how she might apply her nuanced understanding MARC to Python and how she could manipulate records in Python in a way the ILS did not permit, the language began making sense to her. She then had a basis for targeted self-education and learning Python scripting as a way to solve real work problems. For other library workers, code may rarely or never be a relevant way to solve their work problems. At best, understanding more about code becomes a way about understanding what their colleagues do.

The "learn to code" model rarely applies to the second way in which understanding code may be helpful to library workers—knowing enough to know either how to ask better questions of or give better feedback to people who do code. As someone who has worked as a translator between library workers who write code and library workers who don't, I've often attempted to help each side understand what the other means or needs. Those who can do such translations generally have at least a little coding experience, but many more who could use the skills would have little need for "learn to code" in their regular work. Little has been done to make "learn to work with coders" an alternative to "learn to code."

Are there evident benefits for library workers in learning how to tackle certain tasks with scripts or in learning enough about how coding works to communicate better with coworkers? Yes. But a "learn to code" solution rarely addresses either of these situations. The world of library work has involved a variety of kinds of coding for decades. It will continue to do so. Not everyone will need to engage at all with coding. What it looks like will vary greatly by position and institution. What I would recommend to library workers is:

- that they remain open to coding as a possible means of problem-solving and experiment with it when it seems relevant,
- that they remain open to the idea that learning a *little* about coding may help them work better with others,
- and that they not spend too much time beyond that worrying that they should "learn to code" (unless this is being pushed on them at work, and then I'd recommend their supervisors spell out the *what* and *why*).

Alan Harnum
Senior Inclusive Developer, Inclusive Design Research Centre, OCAD University

I received my MLS in 2005, and worked in one of the world's largest public libraries for the next ten years, the last six as a member of the library's web and digital services team. In 2015, I left the library world, and now work as a researcher and software developer in an academic research centre focused on issues of digital inclusion and accessibility.

The simple answer to this question is "it depends." It depends on what work the library worker is doing or wants to do; it depends on what we mean by code. "It depends" is a usually correct answer when faced by a binary question about professional skills.

The longer answer is that this question hides the real dilemma before libraries, librarians and library workers. The "should we learn to code?" question was being asked in nearly the same form in my library school days (and I'm sure before that as well). I consider it a synecdoche, a part (code) standing for a whole (technical knowledge); the synecdoche is sometimes a valuable literary device, but I think it is dangerously reductive in other contexts. We miss, as the saying goes, the forest for the trees.

Whose interests does it serve that a question as complex as the role of technology in library work be reduced to this simplistic form and endlessly debated in the field, a question that can be emphatically answered "yes!" or "no!" depending on one's personal position until the heat death of the universe? Asking and answering the wrong questions is one way of avoiding change and averting the gaze from more substantive issues.

What are these more substantive issues? Everyone has their own list, and this is mine:

Multiple decades of functionally outsourcing technology expertise to vendors and consultants has stripped many libraries of sufficient internal capacity to think critically about technology in the context of the library's values and goals, to scan the horizon for change, and to separate value from hype, sales bullshitting, and bandwagon-jumping.

Heroic efforts by individuals to work technical miracles despite lack of professional and institutional support is not scalable, sustainable or humane; these challenges are structural and philosophical, not solvable merely with the coding skills of individuals.

The level of digital knowledge among much senior leadership in libraries is unacceptably low; leaders need not know how to code, but they must be able to articulate and reason intelligently about the digital realm as a means of achieving the library's purpose. Library leadership not caring or knowing much about digital issues sends the message throughout the organization that it's alright for everyone else not to care much as well, to "leave technology to the experts" (the vendors, consultants and often-harried internal staff with technical responsibilities).

These issues must be looked at with clear eyes not only by all library workers, but most importantly by those who manage and lead libraries and have the power to make decisions about resource allocation, staffing, purchasing, and technical governance.

I do not allege a deliberate conspiracy to avoid the discussion of these issues when we ask questions like "should library workers learn to code?" but I think it must be pointed out that this is simply the wrong question, asked in the wrong way.

Danielle Robichaud

Digital Archivist, University of Waterloo Library

Don’t worry about coding, focus on technical literacy

The anxiety around whether or not library workers should code stems predominantly from a moving-target definition of “coding”. Does it mean utilizing HTML tags to format a web page? Is it using markup to facilitate word processing? Are we talking cataloguing a MARC record? Crosswalking descriptive records to metadata schemas suitable for use in a digital repository? Or is all that out of scope because what it really means is developing an app and retiring at 26?

As a digital archivist who came to the field by way of an MLIS, my thoughts on the matter have evolved, ultimately leading me to this opinion: no, library workers do not need to learn how to code unless doing so is an immediate requirement of their (desired) job. That said, library workers *do* have a responsibility to learn enough about the tech they rely on to effectively communicate end user needs and service barriers to colleagues with the expertise to help.

Developing this skill, and yes, it is a skill, means taking the time to understand the underlying logic of software functionality and figuring out where staff or user expectations are failing to align with the capabilities of a specific platform. One can think of the skill as a type of code switching in which the ways we talk about end user requirements and expectations are situated and expressed in a technical framework. It means assessing both what is expected of software and why it is failing to meet those expectations, then working collaboratively to remedy that disconnect with those on staff who have the skills to develop (code by hand) additional functionality or implement new technical infrastructure to address identified service gaps. Further still, it means respecting that when it comes to digital initiatives (my preferred lens for understanding “coding”) we each have distinct skill sets that can only be of use to the extent that we are able to communicate across and between them.

As library workers, we don’t all need to be able to build tech from scratch, but we absolutely have to be able to talk about it and ask things of it in a way that recognizes technical literacy as an essential part of the library worker’s skill set. It is not our collective inability to code that risks stunting the potential of library workers, it is a chronic failure to recognize that a basic understanding of technical infrastructure is as important as project management, customer service, or thinking outside of the box. Without it, we are incapable of critically assessing whether new infrastructure

investments will address identified service requirements, or of knowing when a vendor is saying what we want to hear rather than telling us what we need to know. Most importantly, it prevents us from effectively advocating for the people we serve and risks wasting innumerable dollars and staff time on technical debt that could have been avoided in the first place.

Anna-Sophia Zingarelli-Sweet
Cataloging and Metadata Specialist, California State University, Northridge

Did you clear your cache?
Solve your database problem
Without asking me.

Code solves some problems
But not others. Use the tool
That's right for the job.

Brad Houston
Document Services Manager, City of Milwaukee

Why can you not code?
Should have learned in library school.
Old curriculum!

Coding is helpful
But it's not as useful as
Having people skills.